Modelowanie zawodów Formuły 1

Sabina Rydzek
Mateusz Kotlarz
Kacper Furmański

23 stycznia 2014

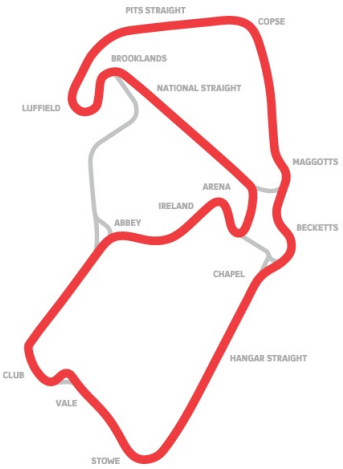
Streszczenie

Projekt ten obejmuje stworzenie modelu wyścigu Formuły 1 oraz jego symulację na wczytanym przez użytkownika torze. Wykorzystuje automaty komórkowe. Symulowane są różne czynniki i typy nawierzchni mające wpływ na trasę i sposób jazdy bolidów. Zamodelowane zostały umiejętności kierowców, które mają wpływ na styl ich jazdy oraz umiejętność przewidywania oraz osiągi. Statystyki umożliwiają odczyt prędkości poszczególnych bolidów oraz ich pozycję na mecie. Projekt jest otwarty i umożliwia tworzenie i wczytywanie innych tras w odpowiednim formacie.

1 Formuła 1

Formuła 1 (Grand Prix) to cykl międzynarodowych wyścigów samochodowych. Samochody, biorące udział w wyścigach nazywane są bolidami. Mają charakterystyczny kształt, który pozwala na uzyskanie bardzo dużej przyczepności. Samochody potrafią uzyskać bardzo dobre osiągi i prędkości powyżej 300 km/h w mniej niż 9 sekund.

Wykorzystany przez nas tor to Silverstone położony w Wielkiej Brytanii [6]. Ma długość $5,891$ km.



Rysunek 1: Tor Silverstone

2 Założenia projektu

- Użycie do modelowania automatów komórkowych
- Przeniesienie obrazu trasy wyścigu na dwuwymiarową siatkę
- Zamodelowanie ruchu samochodów po trasie na siatce
- Zamodelowanie zachowania się kierowców na drodze
- Wprowadzenie współczynników zmieniających zachowanie się samochodów i kierowców

3 State of the Art

Zagadnienie symulacji wyścigów F1 jest bardzo złożone. Niestety, do wielu studenckich prac dotyczących tego tematu nie ma pełnego dostępu.

Z symulacją F1 na University of Cambridge zmierzył się Robert Woolley [10], który dopasował istniejące programy do symulacji wyścigów na potrzeby F1 tak, by były one użyteczne dla FIA (Fédération Internationale de l'Automobile).

Jednym z programów pozwalających na symulację wyścigów jest RaceSim [7], który po wprowadzeniu danych samochodu oraz trasy 3D potrafi wyznaczać współczynniki przyczepności, drogę hamowania itp. oraz reprezentować dane graficznie. jest to jednak bardziej symulacja fizyki samochodu, niż jego zachowania się na torze.

Jednym z ciekawszych podejść do tematu symulowania wyścigów F1 za pomocą automatów komórkowych jest GCA - Global Cellular Automata, które rozszerzają definicję sąsiedztwa i określają je dynamicznie, a nie jak w definicji klasycznej statycznie [4]. Opiera się on również na znanym algorytmie symulacji samochodów.

(A1) **Acceleration.** Increment the current speed v if the maximum speed $v_{\max}$ of the car is not yet reached: $v' = \min(v + 1, v_{\max})$.

(A2) **Slowing down due to the car ahead.** If the gap (distance) d to the preceding car is less than the new speed, reduce it to the size of the gap: $v' = \min(v', d)$.

(A3) **Randomization.** Reduce the new speed by one with probability p, but not below zero: $v' = \max(0, v' - R)$.

(B) **Movement.** Move each car by v' sites (next position is $i + v'$). Set the next speed v (for the generation $t + 1$) to the new speed v' just computed in the current generation t through steps A1–3.

Rysunek 2: A cellular automaton model for freeway traffic

Podejście GCA upraszcza model, ponieważ w chwili t, automat wie już, gdzie będzie znajdował się w chwili $t+1$. Wie dokładnie, na których komórkach znajdują się samochody przed nim i za nim. Niestety przy próbie modelowania samochodów w trakcie wyścigu, sprawa się komplikuje, ponieważ listę z dołączeniami, na której oparty jest algorytm, trzeba rozszerzać i dopisywać kolejne warunki.

W pracy Hoffmanna i Margensterna [**?**] opisane jest podejście do algorytmu wyznaczania trasy przez jadący samochód. Na drodze ustawianie są punkty, które są dla pojazdu tymczasowymi celami, które stara się osiągnąć. Na każdym odcinku przyspiesza, a później hamuje. Pozwala to jednak na płynne przemieszczanie się samochodu po torze bez obijania się o barierki.

4 Automaty komórkowe

W projekcie zastosowany jest model automatów komórkowych. Jest to dyskretny, nieliniowy model, który składa się ze skończonej oraz uporządkowanej liczby *komórek*, które w każdej chwili posiadają swój z góry określony stan, który wyznaczany jest na podstawie komórek sąsiadujących (wg. wybranego sąsiedztwa).

Wg klasycznej definicji, automat komórkowy to trójka (L,S,f), gdzie:

- L - siatka komórek przestrzeni n-wymiarowej
- S - zbiór stanów pojedynczej komórki
- f - reguła określająca stan komórki w chwili $t + 1$ w zależności od stanu danej komórki i komórek sąsiednich w chwili t.

  $$s_i(t + 1) = f(s_j(t)), j \in O(i)$$

  gdzie $O(i)$ to otoczenie komórki

Takie podejście pozwala wyeliminować nadmiarowe obliczenia oraz potrzebę śledzenia pozycji. Dodatkowo, pozwala na dodatkowe optymalizacje przy wykorzystaniu programowania równoległego [3].

4.1 Czas

Automaty komórkowe nie działają w *czasie rzeczywistym*. Zmiana czasu to stworzenie nowej generacji komórek na podstawie określonych reguł ich zachowania [8].

4.2 Stany

Najprostsze automaty komórkowe posiadają tylko dwa stany - jak np. w Game of Life, gdzie komórka jest żywa (1) lub martwa (0) [5]. Zbiór stanów zazwyczaj jest taki sam dla każdej komórki.

5 Pojęcia podstawowe i opis rozwiązań

5.1 Reprezentacja bolidu

Bolid reprezentowany jest przez środek ciężkości znajdujący się w jednej komórce i nakładkę, która zajmuje pół komórki z przodu i pół komórki z tyłu bolidu.

5.2 Komórki

- SurfaceType - ROAD, GRASS, WORSE ROAD, BARRIER, START LINE, SAND - określają typ nawierzchni oraz przypisane im współczynniki, które wpływają m.in. na szybkość, przyczepność bolidów.
- Direction - NONE, TOP LEFT, TOP, TOP RIGHT, LEFT, RIGHT, BOTTOM LEFT, BOTTOM, BOTTOM RIGHT; - określają kierunki, w których może poruszać się bolid, z tych wybierane są te, na które pozwala orientacja bolidu, trasa oraz reguły. Określany jest tutaj również kąt położenia nakładki bolidu na siatce

Komórka w algorytmie może posiadać swój stan - jest/nie ma samochodu. Ograniczona jest również możliwa do rozwinięcia prędkość do 302 km/h.

Direction jest cechą pozwalającą kierowcy ocenić, kiedy należy hamować, przyspieszać, przygotowywać się do zakrętu. Model przyjmuje założenie, że kierowcy znają trasę, po której jadą, dlatego kierunki określane są na trasie tak samo dla każdego samochodu.

5.3 Sąsiedztwo

Podstawowym sąsiedztwem dla komórki jest sąsiedztwo Moore'a. Na decyzję kierowcy wpływają jednak również bolidy, które znajdują się w zasięgu DRIVER VISIBILITY

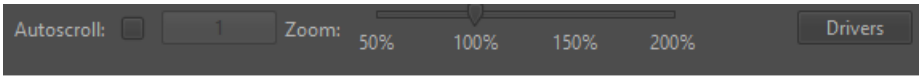
6 Modelowanie ruchu bolidów

Projektowanie modelu ruchu bolidów oparliśmy na rozwiązaniu przedstawionym w pracy *Social Distances* [9]. Samochód reprezentowany jest przez prostokąt o środku znajdującym się w środku jednej z komórek. Bok komórki to $2,6m$ w rzeczywistości, dlatego modelowane przez nas bolidy mają długość $5,2m$.
Samochód może poruszać się w 8 kierunkach, z których każdy jest odpowiednim kątem obrotu (wielokrotności $45°$) prostokąta będącego naszym bolidem. Przejazd bolidu na sąsiednią, odpowiednią komórkę zależny jest od orientacji bolidów w przestrzeni (czyli położenia reprezentujących je prostokątów). Bolidy starają się unikać kolizji, dlatego jeżeli konfiguracja uznana jest za niebezpieczną, bolid może zwolnić, przyspieszyć, bądź spróbować zmienić swoją orientację. Oczywiście, jeżeli nie będzie to możliwe, na co mogą wypływać odpowiednie współczynniki przyczepności albo umiejętności kierowców, nastąpi zderzenie i bolidy odpadną z wyścigu.

6.1 Skalowanie

Dużym problemem jest skalowanie, ponieważ tory są bardzo duże w stosunku do wielkości bolidu. Aplikacja posiada możliwość przybliżania i oddalania widoku

siatki oraz opcję podążania za wybranym samochodem.



Rysunek 3: Opcja przybliżania, autoscroll

6.2 Poruszanie się

Orientacja bolidu zmieniana jest w trakcie poruszania się do kolejnej komórki.

7 Dane wejściowe i wyjściowe

Wejścia systemu to:

- kierowcy (imię, nazwisko, poziom umiejętności)
- trasa przejazdu
- współczynniki (wprowadzone domyślne, bądź przez użytkownika)
- pozycja startowa bolidów

Wyjścia systemu to:

- wyniki (pozycje kierowców)
- wyliczone przyspieszenie, prędkość
- czas najlepszego okrążenie
- czas bieżącego okrążenia

8 Automaty komórkowe - zastosowanie w projekcie

8.1 Siatka komórek

Tor F1 oraz jego otoczenie jest tablicą komórek o różnych współczynnikach w zależności od ich typu (trawa, piasek, asfalt itp.). Głównym problemem w zastosowaniu modelu automatów komórkowych jest to, że bolid jako ciało stałe zajmuje więcej niż jedną komórkę. Oddziaływanie na siebie wielu *bloków* komórek kłóci się z podstawowym założeniem, że w obliczeniach pod uwagę brane są tylko komórki w określonym sąsiedztwie [1]. Aby poradzić sobie z tym problemem, bolid przedstawiony jest tylko jako środek ciężkości (jego obrzeża są

nakładką na torze).

Na przedstawienie toru jako tablicy złożonej z komórek odpowiedniego typu pozwala TrackEditor. Część potrzebnych skryptów wykonana została w środowisku Matlab.

Do toru ręcznie dodany jest punkt startowy oraz ustawione na starcie bolidy.

8.2 Stany komórek

W komórce może znajdować się środek ciężkości samochodu (1), bądź nie (0).

8.3 Reguły

8.3.1 Podstawowe kroki algorytmu

1. Sprawdzanie, kiedy nastąpi kolizja
2. Przeliczanie drogi oraz prędkości, ewentualna aktywacja KERS
3. Element losowości - dołączany element błędu kierowcy
4. Sprawdzanie możliwych komórek, na które samochód może przejechać oraz dystansu od przeciwników przed bolidem
5. Obliczanie nowej prędkości
6. Określenie przesunięcia samochodu wraz z nowymi współczynnikami

8.3.2 Sprawdzanie kolizji

Jeżeli w sąsiedztwie bolidu znajduje się inny bolid, którego kąt różni się od kąta wyliczanej komórki, wtedy następuje kolizja.

8.3.3 Przeliczanie drogi, prędkości, wyliczanie przyspieszenia

Wartości współczynników wyliczone zostały ze wzorów:

$$s = s_0 + v_0 t + \frac{at^2}{2}$$

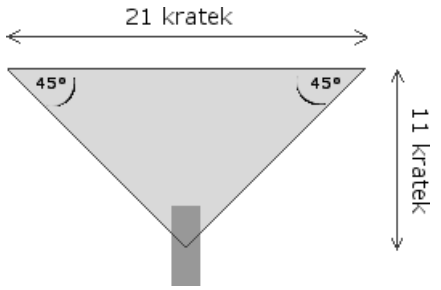
$$v = v_0 + at$$

8.3.4 Element losowy - błędy kierowców

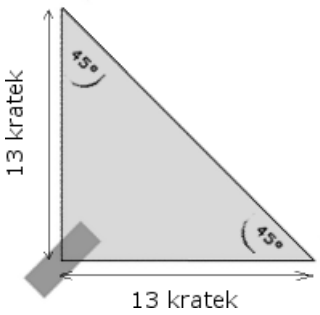
Każdy kierowca ma przypisany poziom umiejętności. Są to MONKEY, NOVICE, INTERMEDIATE, PRO, EXPERT, MASTER, w rosnącej kolejności. W zależności od poziomu, odejmowane są odpowiednie współczynniki do przyspieszenia (utrzymanie przyczepności na torze) oraz liczona możliwość popełnienia przez niego błędu w ocenie sytuacji, wyboru odpowiedniego kierunku.

8.3.5 Driver visibility

Driver visibility to przestrzeń, która ma wpływ na zachowanie kierowcy. Jest w postaci trójkątnej, rozszerzająca się od bolidu. Na jej podstawie wyliczany jest kierunek, do którego obrania próbuje dążyć kierowca. Wyliczana jest mediana z każdego rzędu komórek, a następnie z tego uzyskiwana średnia. Wygląd przestrzeni zmienia się w zależności od kąta położenia bolidu.



Rysunek 4: Driver visibility dla wielokrotności kąta 90°



Rysunek 5: Driver visibility dla kątów $k + 45°$

Jeżeli w zasięgu wzroku i na trasie kierowcy znajdzie się rywal, bolid stara się uniknąć kolizji i przyhamowuje. Wpływ na hamowanie ma poziom umiejętności kierowcy i wyliczony wcześniej współczynnik - jeżeli kierowca zahamuje za późno lub z zza dużej prędkości, kolizji nie uda się uniknąć.
Jeżeli w przestrzeni przed kierowcą nie znajduje się żaden jego rywal oraz zakręt, a naładowanie KERS osiąga 100%, kierowca włącza go na następne $6.7s$.

8.3.6 Obliczanie nowych współczynników

Wartości nowych współczynników wyliczane są za pomocą podanych wcześniej wzorów.

8.3.7 Przesunięcie samochodu do nowej komórki

W każdej iteracji wyliczany jest nowy przejechany dystans od ostatniej iteracji. Jeżeli jest on większy niż $2,6m$, nastąpi przesunięcie samochodu do nowej komórki, jeżeli nie, samochód pozostaje w tej samej komórce a dystans w następnej iteracji dodaje się. Jedna iteracja domyślnie trwa $20ms$.
Jeżeli prędkość samochodu jest za duża (zbliża się do $300km/h$), samochód może nie zdążyć wyhamować i nie może od razu zmienić kierunku jazdy, wtedy jest duża szansa, że wypadnie z toru.

8.3.8 Wyprzedzanie

Samochody wyprzedzają się z bardzo małym prawdopodobieństwem (podobnie jak w rzeczywistych wyścigach). Jeżeli w sąsiedztwie bolidu znajduje się inny samochód, kierowca stara się wyminąć go z prawej albo lewej strony, obierając właściwy kierunek.

8.3.9 Wypadki

Wypadkiem może być:

- kolizja - kiedy zderzają się dwa samochody (najczęściej spowodowane gwałtownym hamowaniem)
- wypadnięcie z trasy - bolid wypada z wyścigu, kiedy utknie na trawie albo piasku, nie ma już żadnego kierunku do obrania (NONE), a jego przyspieszenie jest mniejsze od zera
- uderzenie w barierę

9 Zakończenie symulacji

Symulacja nie ma z góry określonych warunków zakończenia. Zakończenie wywoływane jest na życzenie użytkownika.

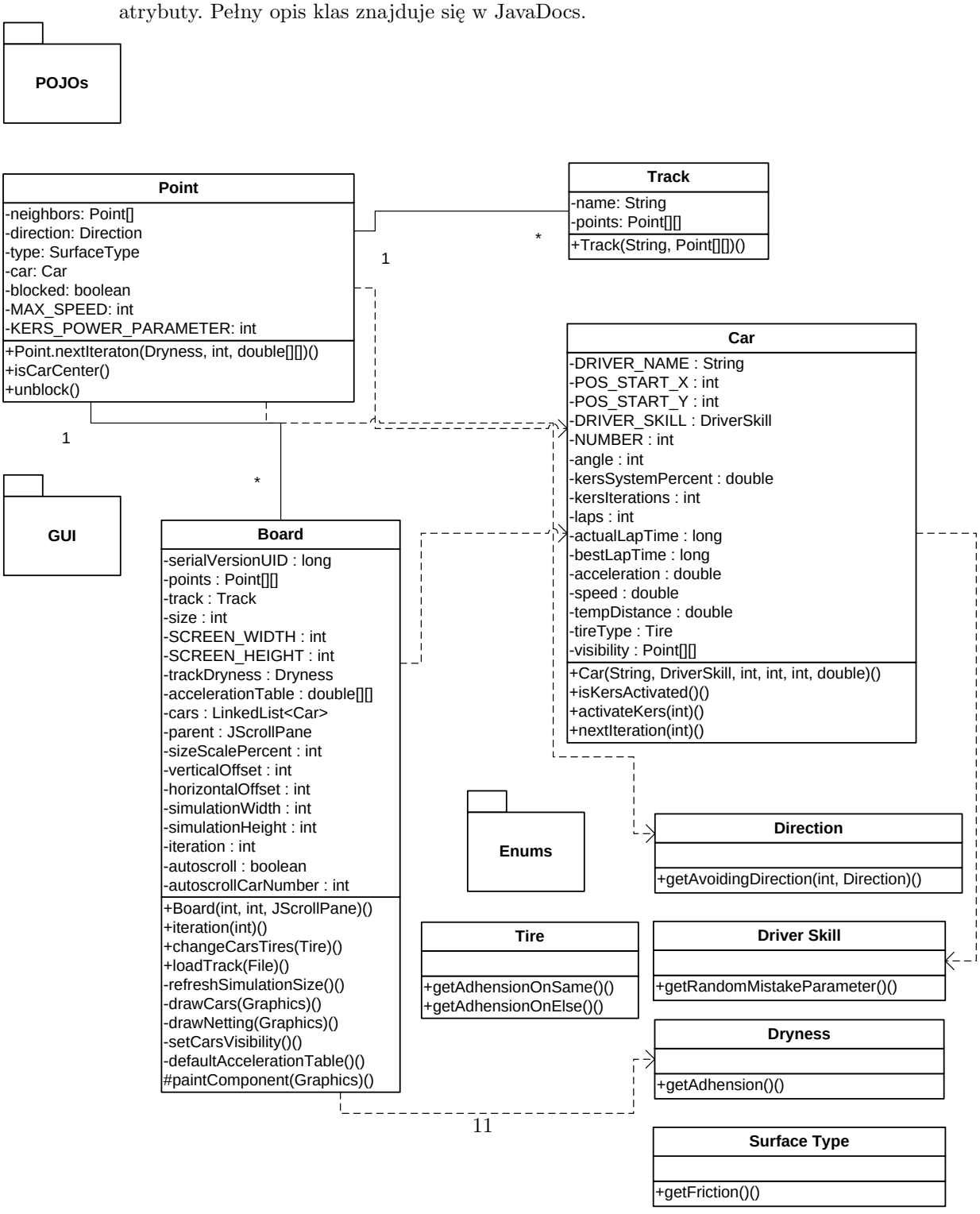
10 Implementacja

10.1 Wybór języka

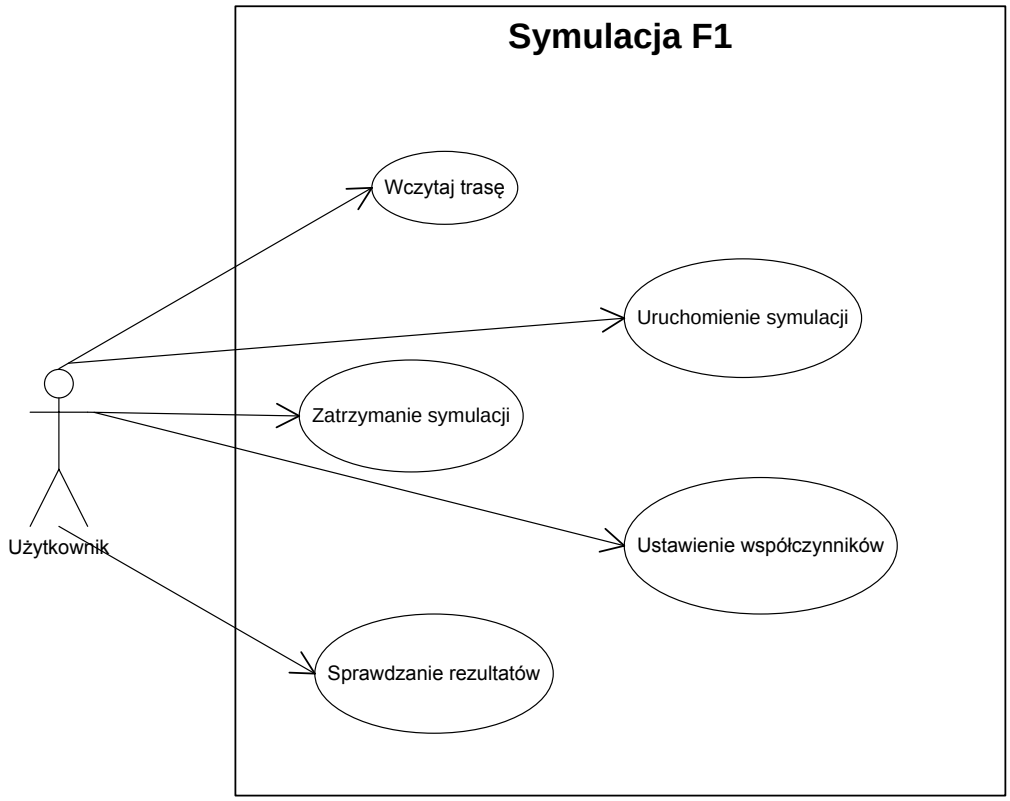
Algorytm zaimplementowany będzie przy użyciu języka programowania Java, który twórcy projektu znali najlepiej. Podejście obiektowe pozwala na jasne i przejrzyste przedstawienie algorytmu. Szybkość działania pozwala na stosunkowo płynną animację poruszania się samochodów na torze. Java pozwala również na korzystanie z aplikacji na innych platformach niż Windows (testowane na Windowsie 8.1 64-bit oraz na Fedora 19 64-bit). Siatka komórek jest zmodyfikowaną i zoptymalizowaną siatką używaną na laboratoriach z Symulacji i Sterowania Procesów Dyskretnych. Opis paczek oraz klas wraz z komentarzami dostępny jest jako *JavaDocs*. W projekcie wykorzystane są również dwie biblioteki zewnętrzne - Substance oraz Trident używane do poprawienia wizualnie wyglądu GUI.

10.2 Diagram UML

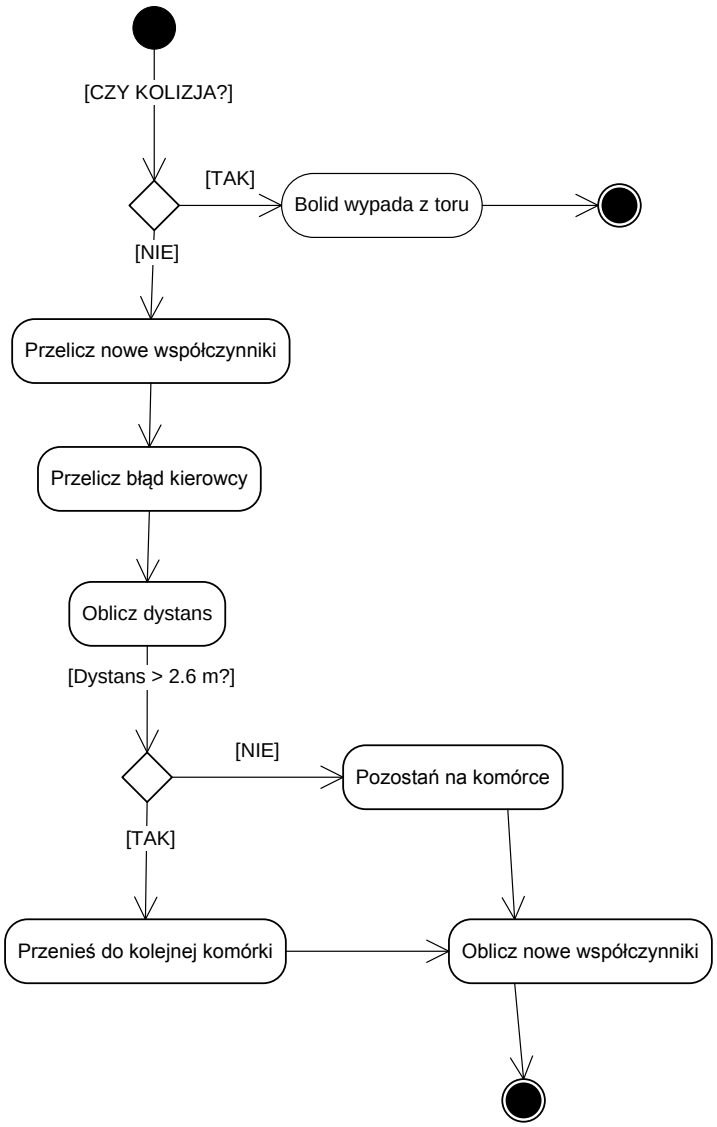
Diagram przedstawia wybrane klasy z paczek POJOs GUI oraz Enums. Dla czytelności nie są wyszczególnione settery oraz gettery atrybutów oraz niektóre atrybuty. Pełny opis klas znajduje się w JavaDocs.



10.3 Przypadki użycia dla użytkownika



10.4 Activity diagram dla pojednynczej komórki



10.5 GUI, opcje aplikacji, możliwości doboru współczynników

Po starcie programu uruchamia się okno do wyboru trasy. Trasy mogą być tworzone z wykorzystaniem Track Editor i mają rozszerzenie

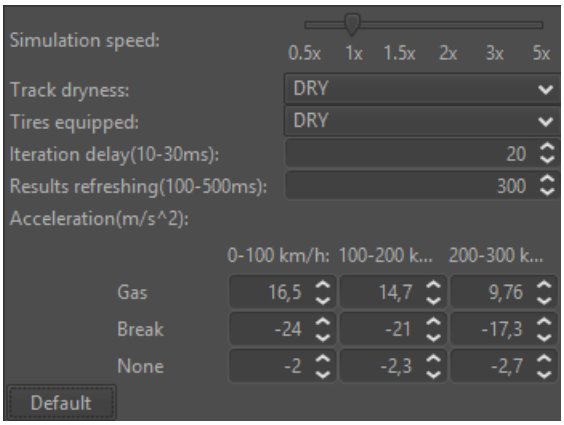
`.track`

Elementy GUI:

- Load track - pozwala na wczytanie trasy
- Simulation parameters - otwiera okno z opcjami
- Start - uruchamia symulację
- Clear - powraca do stanu początkowego
- About - o twórcach projektu
- Exit - wyjście z programu
- Autoscroll i Zoom - odpowiada za widok
- Drivers - otwiera okno ze statystykami
- Results - otwiera okno z wynikami

Simulation speed wpływa na przedstawienie na symulacji jednej sekundy (np. $2x$ oznacza, że jedna sekunda symulacji to $0.5s$ w rzeczywistości.
Iteration delay odpowiada za to, jak wiele iteracji przypada na jedną sekundę.
Refreshing odpowiada za tempo odświeżania tabeli wyników i statystyk.

Współczynniki Gas, Brake i None określają wartość przyspieszenia (hamowania) w różnych zakresach prędkości samochodu.



Rysunek 6: Możliwości wpływu na symulację

11 Wyniki symulacji

11.1 Przykładowe uzyskane wyniki i statystyki

Pos.	Name	Skill	Lap	Actual Lap Time	Best Lap Time
1	Vettel	MASTER	2	0:20.940	1:33.660
2	Webber	EXPERT	2	0:16.860	1:37.300
3	Alonso	MASTER	2	0:15.280	1:38.340
4	Massa	INTERMEDIATE	2	0:06.120	1:46.940
5	Button	NOVICE	2	0:00.580	1:51.920
6	Grosje	INTERMEDIATE	1	1:50.780	-
7	Rosberg	PRO	1	1:50.200	-
8	Hamilton	EXPERT	1	1:49.680	-
9	Hulkenberg	MONKEY	1	1:49.060	-
OUT	Perez	EXPERT	1	1:24.720	-
OUT	Raikkonen	PRO	1	1:24.160	-

Rysunek 7: Przykładowe wyniki po pierwszym okrążeniu na nawierzchni suchej (opony DRY)

Pos.	Name	Skill	Lap	Actual Lap Time	Best Lap Time
1	Vettel	MASTER	2	0:29.240	1:35.880
2	Webber	EXPERT	2	0:23.620	1:40.980
3	Alonso	MASTER	2	0:20.860	1:43.280
4	Massa	INTERMEDIATE	2	0:17.620	1:45.940
5	Button	NOVICE	2	0:14.560	1:48.380
6	Perez	EXPERT	2	0:13.780	1:48.660
7	Raikkonen	PRO	2	0:11.400	1:50.420
8	Grosje	INTERMEDIATE	2	0:09.780	1:51.500
9	Rosberg	PRO	2	0:08.620	1:52.080
10	Hamilton	EXPERT	2	0:07.100	1:53.060
11	Hulkenberg	MONKEY	2	0:04.600	1:54.920

Rysunek 8: Przykładowe wyniki po pierwszym okrążeniu na nawierzchni mokrej (opony WET)

Pos.	Name	Skill	Lap	Actual Lap Time	Best Lap Time
1	Vettel	MASTER	2	0:39.580	1:44.020
2	Webber	EXPERT	2	0:35.800	1:47.300
3	Alonso	MASTER	2	0:34.560	1:48.060
4	Massa	INTERMEDIATE	2	0:28.320	1:53.740
5	Button	NOVICE	2	0:24.640	1:56.800
6	Perez	EXPERT	2	0:24.100	1:56.800
7	Raikkonen	PRO	2	0:22.800	1:57.540
8	Grosje	INTERMEDIATE	2	0:21.860	1:57.900
9	Rosberg	PRO	2	0:20.720	1:58.460
10	Hamilton	EXPERT	2	0:19.440	1:59.200
11	Hulkenberg	MONKEY	2	0:12.780	2:05.260

Rysunek 9: Przykładowe wyniki po pierwszym okrążeniu na nawierzchni mokrej (opony DRY)

Jak widać, wyniki są podobne dla opon suchych na torze suchym i opon WET na torze mokrym. Znacznie się pogarszają, gdy opony nie są przystosowane do nawierzchni.

Statystyki na bieżąco pokazują prędkość bolidu, przyspieszenie (hamowanie) oraz poziom naładowania systemu KERS.

11.2 Kalibracja i walidacja modelu

Początkowo problemem było takie dopasowanie współczynników, by można było wiernie odwzorować rzeczywisty wyścig na torze. Znając prędkości na zakrętach oraz średnią prędkość na torze, mogliśmy modyfikować współczynniki hamowania, przyspieszenia [6]. Model bierze pod uwagę również przyczepność opon

No.	Name	Skill	Lap	Speed	Accelerate	KERS %
1	Vettel	MASTER	1	156 km/h	-2.3 m/s2	56 %
2	Webber	EXPERT	1	194 km/h	-2.8 m/s2	49 %
3	Alonso	MASTER	1	214 km/h	8.284 m/s2	49 %
4	Massa	INTERMEDIATE	1	201 km/h	11.859 m/s2	39 %
5	Button	NOVICE	1	202 km/h	7.374 m/s2	34 %
6	Perez	EXPERT	1	201 km/h	13.168 m/s2	31 %
7	Raikkonen	PRO	1	198 km/h	-2.3 m/s2	29 %
8	Grosje	INTERMEDIATE	1	199 km/h	-2.3 m/s2	27 %
9	Rosberg	PRO	1	199 km/h	11.446 m/s2	25 %
10	Hamilton	EXPERT	1	185 km/h	12.632 m/s2	24 %
11	Hulkenberg	MONKEY	1	121 km/h	13.203 m/s2	22 %

Rysunek 10: Przykładowe statystyki

różnego rodzaju na mokrej i suchej nawierzchni. Kalibracja współczynników przyczepności polegała na takim ich dostosowaniu, aby widać było ich wpływ na czas okrążenia oraz umiejętność pokonywania zakrętów.

Współczynniki dobieraliśmy sprawdzając czas i ilość kolizji kolejnych okrążeń, na przykład dla suchej nawierzchni i dobranych do niej opon:

Test	Kolizje	Najlepszy czas	Najgorszy czas
1	1	1 : 35	1 : 58
2	2	1 : 30	1 : 48
3	-	1 : 29	1 : 38

Kalibracja współczynników przyspieszania i hamowania jest również dostępna dla użytkownika. Wartości domyślne dobrane zostały korzystając z dostępnych szybkości na zakrętach. Prędkość maksymalna samochodów została w projekcie ograniczona z góry do 302 km/h, by prędkości mierzone na zakrętach przypominały prędkości rzeczywiste [2].
Aktywacja KERS trwa tyle samo, ile w rzeczywistości, a jej współczynnik można przeliczyć na dodatkowych 80 koni mechanicznych.

Rekord toru należy do Webbera i wynosi 1 : 33.401. Prędkości samochodów dopasowane są tak, aby wynik najlepszego kierowcy był zbliżony do czasu Webbera. System KERS (Kinetic energy recovery system) wykorzystywany jest najczęściej na najdłuższej prostej toru - Wellington(1034m).

Walidację rozpoczął test komponentów programu. Zostało sprawdzone, czy samochody rysują się poprawnie i czy zmiany współczynników faktycznie mają wpływ na zachowanie się bolidów. Testowanie funkcjonalności było sprawdzeniem poprawności algorytmu - m.in. czy samochody prawidłowo odczytują zbliżający się zakręt, czy nie przekraczają dozwolonych prędkości i czy starają się unikać kolizji. Weryfikacja ilościowa projektu potwierdziła, że prędkości osiągane przez kierowców zgadzają się z prędkościami osiąganymi rzeczywiście, a czasy przejazdu odzwierciedlają umiejętności kierowców, co szczególnie widać na ostrych zakrętach.
Weryfikacja ilościowa ukazuje ważny błąd symulacji, którym jest za duża długość bolidu w stosunku do danych rzeczywistych - $5,6m$ w stosunku do $4,58m$. Jest to spowodowane za małą wydajnością komputera, który ma problem z wyliczaniem tak małych oczek siatki.

12 Wnioski

12.1 Problemy powstałe przy tworzeniu modelu

Pierwsze problemy pojawiły się już na etapie wybierania podejścia do modelowania tematu za pomocą automatów komórkowych. Zarówno prosta siatka, jak i siatka, której komórki mają inny kształt na zakrętach, miały swoje zalety i wady. Wyzwaniem było również takie zamodelowanie samochodów, które umożliwiało traktowanie go jak pojedynczej komórki.

Największym problemem była jednak mocno ograniczona przestrzeń, na podstawie której kierowca może podejmować decyzję. Rozszerzanie sąsiedztwa, albo powiększanie driver visibility znacznie wydłużało czas obliczeń i animacja samochodów przestawała być płynna.

Ostatnim z problemów była ograniczona możliwość walidacji modelu. Informacje o ważnych dla symulacji zmiennych są trudnodostępne, a wyliczanie dokładnych współczynników tarcia, przyczepności albo oporu powietrza jest już dziedziną fizyki, z którą twórcy projektu mają małą styczność.

12.2 Usytuowanie modelu na tle innych rozwiązań

Bardzo dużym utrudnieniem był fakt, że podobna symulacja nie jest dostępna, a literatura w dużej mierze dotyczy samego poruszania się samochodów, bądź wyścigów, które nie są tak specyficzne, jak F1.

Pomimo dostępnej i omówionej wcześniej literatury ([4] [7] [9]), twórcy projektu nie mogli skorzystać z przedstawionych tam rozwiązań i technik. Poziom skomplikowania symulacji wymuszał rozszerzenie sąsiedztwa, dodawanie kolejnych instrukcji algorytmu. Dodatkowo ciągłą uwagę należało zwracać na płynność symulacji.

W przeciwieństwie do projektu Roberta Woolleya [10], ten model nie wykorzystuje tak skomplikowanej fizyki, dlatego nie może być używany w celach badawczych, choć takie jego przystosowanie jest ważnym kierunkiem jego rozwoju.

12.3 Sukcesy

Twórcom udało się osiągnąć równowagę pomiędzy dokładnością obliczeń i szybkością ich wykonywania. Zoptymalizowana siatka pozwoliła na szybsze jej rysowanie i też znacząco zredukowała czas wykonywania kroku symulacji.

Jednym z ważnych elementów projektu jest także przyjazne dla użytkownika GUI, które pozwala na ustawianie współczynników, podgląd wyników i statystyk.

12.4 Future works - kierunki rozwoju symulacji

- możliwość wyboru opon dla konkretnego kierowcy
- poprawa umiejętności wyboru idealnej trasy przez kierowców
- zamodelowanie pitstopu
- prosta możliwość przystosowywania nowych tras do wczytywania
- zaplecze mechaniczne i fizyczne - podobne do statystyk RaceSim [7]
- poprawa wyglądu siatki, ewentualna zmiana kształtu komórek
- poprawa współczynników
- zrównoleglenie obliczeń, umożliwiające zwiększenie ich dokładności oraz wielkości siatki

Literatura

[1] Carter Chamberlain. Particle simulation with cellular automaton. 2009.

[2] Christian Danner. Silverstone circuit tracj map. `http://www.f1fanatic.co.uk/f1-information/going-to-a-race/silverstone-circuit-information/`. Accessed: 23-01-2013.

[3] Jacopo Tagliabue Francesco Berto. Cellular automata. `http://plato.stanford.edu/entries/cellular-automata`. Accessed: 04-12-2013.

[4] Rolf Hoffmann. Gca-w algorithms for traffic simulation. 2011.

[5] Krzysztof Kułakowski. *Automaty komórkowe*. Akademia Górniczo-Hutnicza im. St. Staszica w Krakowie, 2000.

[6] Formula One World Championship Limited. Silverstone. `http://www.formula1.com/races/in_detail/great_britain_924/`. Accessed: 10-01-2013.

[7] DATAS Ltd. Racesim. `http://www.datas-ltd.com/RaceSim.htm`. Accessed: 10-01-2013.

[8] Daniel Shiffman. Nature of code. `http://natureofcode.com/book/chapter-7-cellular-automata/`. Accessed: 04-12-2013.

[9] Matuszyk PJ Was J., Gudowski B. Social distances model of pedestrian dynamics. 2006.

[10] Robert Woolley. Simulation in f1. `http://www-formula1.eng.cam.ac.uk/student-proj/simulation`. Accessed: 10-01-2013.