Modelowanie zawodów Formuły 1

Sabina Rydzek
Mateusz Kotlarz
Kacper Furmański

13 grudnia 2013

Streszczenie

Projekt ten obejmuje stworzenie modelu wyścigu Formuły 1 oraz jego symulację na wczytanym przez użytkownika torze (używanym przez nas torem jest Hungaroring Circuit). Wykorzystuje automaty komórkowe. Symulowane są różne czynniki i typy nawierzchni mające wpływ na trasę i sposób jazdy bolidów. Próbujemy zamodelować umiejętności kierowców, które mają wpływ na styl ich jazdy oraz umiejętność przewidywania. W końcowej fazie planowane jest również zamodelowanie pitstopu. Dodane będą również czynniki atmosferyczne. 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 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).

Takie podejście pozwala wyeliminować nadmiarowe obliczenia oraz potrzebę śledzenia pozycji. Dodatkowo, pozwala na dodatkowe optymalizacje przy wykorzystaniu programowania równoległego [2].

Tor F1 oraz jego otoczenie jest tablicą komórek o różnych współczynnikach w zależności od ich typu (trawa, krawężnik, 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).

1.1 Grid

Wybraliśmy Hungaroring Circut, znajdujący się w pobliżu Budapesztu. Ma on 4.381km długości oraz 14 zakrętów [3]. Na przedstawienie toru jako tablicy złożonej z komórek odpowiedniego typu pozwala TrackEditor. Część potrzebnych nam skryptów wykonaliśmy w środowisku Matlab.

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

1.2 Czas

Automaty komórkowe nie działają w *czasie rzeczywistym*. Zmiana czasu to stworzenie nowej generacji komórek na podstawie określonych reguł ich zachowania [4]. Z powodu ograniczenia przez wielkość komórek, staramy się uzyskać średnio cztery do pięciu generacji w ciągu trzech sekund, tak, aby możliwe było zamodelowanie rzeczywistej prędkości bolidów na torze.

1.3 Typy komórek

- SurfaceType - Road, Grass, Worse Road, Barrier, None - określają typ nawierzchni oraz przypisane im współczynniki, które wpływają na m.in. na szybkość, przyczepność bolidów. Bolidy, które znajdą się na trawie lub barierkach, wypadają z wyścigu. Worse Road ma gorsze współczynniki niż Road, można z niej wrócić jednak jeszcze do wyścigu.

- Direction - 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. Kierunki wykorzystywane są również przy określaniu idealnej trasy.
- Angle - wykorzystywany przy modelowaniu zakrętów oraz ustalaniu orientacji bolidu w przestrzeni.

1.4 Stany

Komórki mogą zawierać środki ciężkości bolidów, które mogą czekać na start, mogą poruszać się w określonym kierunku, bądź *czekać*, oceniać obecną sytuację i ewentualnie zmieniać swój cel.

1.5 Reprezentacja bolidu

Docelowo bolid będzie reprezentowany przez środek ciężkości znajdujący się w jednej komórce i nakładkę, która będzie zajmowała pół komórki z przodu i pół komórki z tyłu bolidu.

1.6 Sąsiedztwo

Dobrym sąsiedztwem do zastosowania jest rozszerzone sąsiedztwo Moore'a, które pozwoli kierowcom na szerszy zakres widzenia.

2 Modelowanie ruchu bolidów

Projektowanie modelu ruchu bolidów oparliśmy na rozwiązaniu przedstawionym w pracy *Social Distances* [5]. Samochód reprezentowany jest przez prostokąt o środku znajdującym się w środku jednej z komórek, której bok reprezentuje długość 42m. Przyjmując średnią prędkość samochodu jako $200km/h$ możemy wyrysowywać nową pozycję bolidów co niecałą sekundę.

Samochód może poruszać się w 8 kierunkach, z których każdy jest odpowiednim kątem obrotu 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 wypadną z toru.

2.1 Skalowanie

Dużym problemem jest skalowanie, ponieważ tory są bardzo duże w stosunku do wielkości bolidu. Jednym z planowanych rozwiązań jest możliwość przybliżania i oddalania widoku.

2.2 Poruszanie się

Orientacja bolidu zmieniana jest w trakcie poruszania się do kolejnej komórki. Bardziej doświadczeni kierowcy będą próbowali orientować swoje bolidy tak, jak prowadzi trasa idealna. Będą mieć również priorytet przejazdu do tej komórki nad kierowcami mniej doświadczonymi.

Brana jest pod uwagę pozycja, którą samochód chce osiągnąć. Jeżeli jest to niemożliwe, samochód stara się wyznaczyć swój nowy cel. Odpowiada to stanowi *Wait* [5].

Jednym z naszych rozwiązań modelowania poruszania się samochodów na zakrętach, jest sygnalizowanie zakrętów wcześniej, tak, żeby kierowcy mieli czas, by odpowiednio zareagować na informację o ostrym zmianie kierunku. Odpowiednie współczynniki pozwolą określić, czy bolid wypadnie z zakrętu, czy będzie musiał zwolnić, bądź przyspieszyć.

3 Implementacja

Algorytm zaimplementowany będzie przy użyciu języka programowania Java. GUI jest zmodyfikowanym GUI używanym na laboratoriach z Symulacji i Sterowania Procesów Dyskretnych. Opis paczek oraz klas wraz z komentarzami dostępny jest jako *JavaDocs*.

Literatura

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

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

[3] Formula One World Championship Limited. Hungaroring circuit. `http://www.formula1.com/races/in_detail/hungary_903/circuit_diagram.html`. Accessed: 07-12-2013.

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

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