JCSI 3 (2017) 23-27
Journal of Computer Sciences Institute

Wysłane: 2016-11-23
Przyjęte: 2016-12-04

Symulacja zachowań typu BOIDs w środowisku Unity

Taras Lypovyi, Jerzy Montusiewicz*
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W pracy opisano i scharakteryzowano zachowania typu BOID (bird-oid object) polegające na wspólnym przemieszczaniu się gromady obiektów o tych samych właściwościach. Zaprezentowano model Reynolds'a uwzględniający 3 reguły: separacja, wyrównanie i spójność oraz procedury sterowania gromadą obiektów zaproponowaną przez Parkera uwzględniająca takie wielkości, jak: wiatr, cel, prędkości, kolejność oraz występujące siły. Metoda badawcza polegała na przeprowadzeniu eksperymentów symulacyjnych przy różnych konfiguracjach współczynników sił sterowania modelem. Dla każdej symulacji był wymierzony czas ruchu od punktu początkowego do punktu końcowego. W pracy przeprowadzono sto symulacji dla każdej poszczególnej grupy współczynników, a następnie wykorzystując opisane metody statystyki wyznaczono uogólnione wartości czasu, które umożliwiały porównywanie uzyskanych wyników. Przeprowadzone symulacje numeryczne zrealizowano w środowisku Unity. Obliczanie czasu potrzebnego na przebycie identycznej drogi przeprowadzano przy zmianie wartości siły separacji, spójności, wyrównywania oraz unikania. Z uzyskanych wartości wynika, że największy wpływ na wzrost czasu przemieszczania obiektów typu BOID-s ma zwiększanie wartości współczynnika sił separacji oraz wyrównywania. Środowisko Unity dobrze nadaje się do prowadzenia takich symulacji, ponieważ umożliwia otrzymanie zarówno wartości liczbowych jak i wizualizacji procesu w postaci obrazu 3D. Oprócz tego Unity zezwala na tworzenie własnych skryptów do zarządzania symulacją we własnym IDE, a także przedstawia dokumentację, co upraszcza ich pisanie.
Słowa kluczowe: BOID; symulacja zachowań BOID-s; środowisko Unity

*Autor do korespondencji.

Adres e-mail: j.montusiewicz@pollub.pl

Simulation of BOID type behaviours in Unity environment

Taras Lypovyi, Jerzy Montusiewicz*
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The study describes and characterises BOID (bird-oid object) type behaviours, consisting of joint movement of a cluster of objects with the same properties. Authors presented Reynolds' model, which takes into account 3 rules: separation, alignment and consistency, as well as the control procedures of a cluster of objects suggested by Parker, considering such variables as wind, aim, speed, order and the occurring forces. The test method was to conduct simulation experiments with different configurations of coefficients of the forces controlling the model. For each simulation the time of moving from the start point to the end point was measured. A hundred simulations were carried out for each individual group of coefficients, and then, using the described statistics methods, generalised time values were determined. This allowed a comparison of the results and made a conclusions. The numerical simulations carried out were implemented in Unity environment. Calculating the time required for travelling the same route was done by changing the value of the separation force, cohesion, alignment and avoidance. From the values obtained, it can be seen that the biggest influence on the increase of the time of moving BOID objects, is increasing value of the coefficient of separation and levelling forces. Unity environment is well suited to conduct such simulations, since it allows to obtain both numerical values and process visualization as a 3D image. In addition, Unity allows to create individual scripts to manage simulation in individual IDEs, and consists reliable documentation, which simplifies their writing.

Keywords: BOID; simulation of BOID behaviour; Unity environment

*Corresponding author.

E-mail address: j.montusiewicz@pollub.pl

1. Wstęp

Skrót "BOID" to skrócona wersja terminu "bird-oid object" („bird-like object") czyli obiekty *ptakopodobne*. W świecie przyrody w ten sposób poruszają się ławice ryb lub stada małych ptaków (nie dotyczy to jednak ptaków wędrownych przemierzających duże dystanse tworzące w powietrzu tzw. klucze).

Pierwsza praca, w której były opisane zachowania takich obiektów ujrzała światło dzienne w 1986 r. i był to program sztucznego życia Craiga Reynoldsa [1]. Autor symulował w niej zachowania grupy ptaków na potrzeby fotorealistycznie wyglądających obiektów realizowanych przy zastosowaniu grafiki komputerowej oraz przy tworzeniu naturalnie przemieszczających się grup ptaków, ryb i innych zwierząt. Obecnie algorytm zaproponowany przez Reynoldsa używany jest nie tylko w grafice czy grach komputerowych, ale także może być adaptowany do potrzeb kontroli i stabilizacji ruchu zespołów prostych bezzałogowych pojazdów naziemnych, powietrznych czy podwodnych oraz wizualizacji danych i rozwiązywanych zadań optymalizacji.

Metodę ciągłej optymalizacji funkcji nieliniowych nazwaną algorytmem roju cząstek podano w pracy [2] w 1995 r. Koncepcja algorytmu polegała na symulacji systemu wielokrotnego agenta, w którym cząstki poruszają się do optymalnego rozwiązania wymieniając przy tym informacje ze swoimi sąsiadami. W 1998 r. w pracy [3] zaproponowano zmianę polegającą na wprowadzeniu równowagi między dokładnością badania przestrzeni poszukiwania i szybkością zbieżności algorytmu. Z kolei w 2002 r. w pracy [4] wprowadzono modyfikacje algorytmu roju cząstek (algorytm ten uważany jest obecnie za kanoniczny). Zastosowany sposób

wyliczania wektorów prędkości cząstek wzbogacono o nowy czynnik, zapewniający zbieżność algorytmu bez konieczności bezpośredniego kontrolowania prędkości cząstek. W pozycji [5] autorzy wprowadzili do modelu wszystkie cząstki nazwane jako „w pełni poinformowane", czyli takie, które otrzymywały informację od wszystkich cząstek sąsiednich. W pracy [6] zaproponowano proste rozszerzenie idei BOID-ów poprzez możliwość zmiany lidera. Zmiana przywództwa jest oceniana w widocznym obszarze każdego BOID-a, czyli tylko lokalne cząstki brane są pod uwagę. Użytkownik posiada parametr, który jest atrybutem przywództwa. Szansę zostania liderem mają BOID-y znajduje się na granicy stada. W pracy [7] pokazano metodę do rozwinięcia parametrów modelów opartych na agentach, która prowadzi do powstania tzw. krytycznego zachowania. Właściwości takich modelów były badane przy użyciu algorytmów ewolucyjnych stosując pięć parametrów dla każdego BOID-a: unikanie (separacja), spójność, jednolitość, pęd i sąsiedztwo. Wykorzystanie algorytmów genetycznych do optymalizacji wektora ruchu poprzez znalezienie optymalnych współczynników opisano w pozycji [8]. Autorzy w pracy [9] opisywali jednorodny rój z uwzględnieniem komunikacji wszyscy-do-wszystkich, stosując mieszaną konfiguracja rzeczywistości, w której mała liczba robotów fizycznych współdziała z większym liczebnie rojem wirtualnym. Takie podejście pozwalało na obserwowanie powstawania wzorca ruchu dla dużych rojów w ograniczonej przestrzeni laboratoryjnej.

Celem pracy było wykonanie symulacji zachowań roju BOID-ów przy różnych wartościach sił sterowania opracowanym modelem. Poszukiwano tych sił i wartości, które najbardziej wpływają na czas przemieszczania się roju oraz jego wygląd.

2. Opis modelu BOIDs

Grupa BOID-ów to grupa obiektów wykonujących płynny ruch. Zgodnie z badaniami ruch grupy jest wynikiem działania poszczególnych obiektów, działających wyłącznie na podstawie własnego lokalnego spostrzegania świata. Stado jest więc formacją powstałą na skutek oddziaływania pomiędzy zachowaniami pojedynczych ptaków. Opracowanie symulowania ruchu stada może być zrealizowanie przez stworzenie modeli kilku ptaków i wprowadzenie dodatkowych interakcji między nimi [10]. Przykładowa struktura programu symulacji zachowań grupy obiektów może być opisana pseudokodem zaprezentowanym w Listingu 1.

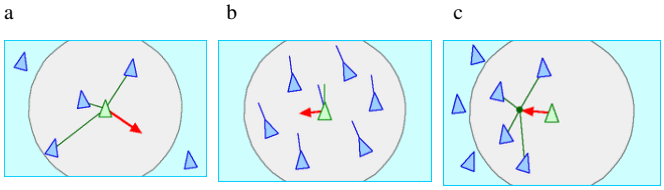
Listing 1. Struktura programu symulacji [11]

```
initialise_positions()
LOOP
    draw_boids()
    move_all_boids_to_new_positions()
END LOOP
```

Procedura `initialise_positions()` umieszcza każdy BOID na początkowej pozycji, zwykłe jest to losowa lokalizacja na ekranie. Na początku symulacji obiekty ruszają prosto przed siebie [11]. Funkcja `draw_boids()` odpowiada za rysowanie wszystkich BOIDów na swoich pozycjach [11]. Procedura `move_all_boids_to_new_positions()` zawiera właściwy algorytm przemieszczania się BOID-sów.

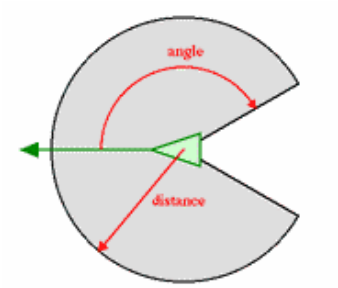
Podstawowy model stada obiektów tworzących rój składa się z trzech prostych zachowań rządzących ich ruchem. Opisują one w jaki sposób poszczególne manewry BOID-a są realizowane na podstawie położenia i prędkości innych obiektów [1]:

- Separacja (rys. 1a) – odpowiada za unikanie tłoku lokalnych członków stada, dzięki czemu nie występuje kolizja z pobliskimi obiektami;
- Wyrównanie (rys. 1b) – żąda aby przemieszczanie następowało w odniesieniu do średniej pozycji lokalnych członków stada;
- Spójność (rys. 1c) – żąda aby obiekty poruszały się w kierunku średniej pozycji lokalnych członków stada, co prowadzi do próby utrzymania się blisko sąsiednich członków stada.



Rys. 1. Podstawowe siły modely grupy BOID-ów [1]

Każdy BOID ma bezpośredni dostęp do całego opisu geometrycznej sceny, ale udział w stadzie powoduje, że obiekt reaguje tylko na członków z pewnego ograniczonego obszaru wokół siebie (rys. 1.). Sąsiedztwo opisane jest przez odległość (mierzona od środka pojedynczego BOID-a) oraz kąt kierunku ruchu BOID-a. Członkowie stada spoza tego sąsiedztwa są ignorowani [1].



Rys. 2. Obszar sąsiadów [1]

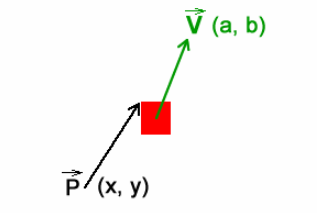
W symulacji istnieje również możliwość wprowadzenia procedury ograniczającej prędkości obiektów. Bez takiego ograniczenia rozpatrywane obiekty musiałyby latać bardzo szybko, co byłoby w sprzeczności z rzeczywistymi ptakami, które przemieszczają się z prędkościami zależnymi od danego gatunku, Listing 2.

Listing 2. Kolejność wywoływania procedur [12]

```
b.velocity = b.velocity + v1 + v2 + v3 + ...
```

```
limit_velocity(b)
b.position = b.position + b.velocity
```

Implementację wszystkich sił, które wpływają na ruch obiektu, można osiągnąć wykorzystując zapis wektorowy. Na rysunku 3 pokazano obiekt w punkcie $\vec{P}(x,y)$ o prędkości $\vec{V}(a,b)$.



Rys. 3. Obiekt, jego pozycja i prędkość w postaci wektorowej [12]

Następna pozycja obiektu P_1 jest wyliczana jako *następna_pozycja = pozycja + prędkość*, wzór (1).

$$P_1(x_1,y_1) = \begin{Bmatrix} x_1 = x + a, \\ y_1 = y + b \end{Bmatrix} \quad (1)$$

Zwrot wektora wskazuje kierunek ruchu BOID-a, długość – wartość o ile ma się przesunąć obiekt względem poprzedniej pozycji. Im większą wartość osiąga drugi parametr tym szybszy jest ruch obiektu. Do tego, aby ruch modelowanego BOID-a był bardziej podobny do ruchu realnych obiektów wykorzystywane są siły sterowania, czyli wektory, które są dodawane do wektora prędkości. W zależności od tych sił, obiekt BOID-a będzie poruszać się w okreslonym kierunku. Uzyskane zachowanie BOID-a wynika więc z żądanej prędkości oraz występujących sił sterowania.

Żądana prędkość jest to siła, która wskazuje obiektowi cel osiągany po najkrótszej możliwej trajektorii. Tym samym czas siły sterowania, to różnica między żądaną prędkością i aktualną prędkością, która także wskazuje na cel. Siły są wyliczane przez procedury jak w Listingu 3.

Listing 3. Zastosowane procedury [11]

```
desired_velocity = normalize(target - position) *
max_velocity
steering = desired_velocity - velocity
```

Po wyliczeniu siły sterowania niezbędne jest dodanie jej do składowej prędkości obiektu. Dodawanie siły sterowania do prędkości w każdej ramce symulacji spowoduje stopniową zmianę kierunku ruchu w stronę celu, co można w łatwy sposób wykreślić na rysunku przedstawiającym tą sytuację, Listing 4.

Listing 4. Zastosowane procedury [11]

```
position = position + velocity
steering = truncate (steering, max_force)
steering = steering / mass
velocity = truncate (velocity + steering , max_speed)
```

3. Metodyka badawcza

3.1. Opis badań i ich cel

Dla badania zachowań typu BOIDs wybrano metodę eksperymentalną, która polegała na przeprowadzeniu symulacji i porównaniu czasów pokonywania zadanej drogi przez grupę obiektów. Badano wpływ poszczególnych sił na zachowanie grupy BOID-ów, oraz wpływ jednoczesnej zmiany wartości wszystkich sił na zachowanie całej grupy obiektów. W symulacji uwzględniono 3 siły podstawowego modelu Reynolds'a: *separacja*, *spójność*, *wyrównanie* oraz dodatkowo siłę *unikanie* przeszkód. Dla każdego z analizowanych przypadków wykonano po sto symulacji zapisując czas ruchu przejścia od pozycji start do pozycji cel. Na podstawie wartości uzyskanych czasów, a także ogólnego wyglądu grupy BOID-ów, tzn. wzajemnego położenia poszczególnych członków grupy, przeprowadzono analizę dotyczącą wpływu każdej z badanej siły na szybkość ruchu grupy obiektów podążającej do celu.

Celem symulacji było więc zbadanie jak zmiany wartości poszczególnych sił wpływają na ruch grupy obiektów. A także czy dobór odpowiedniej konfiguracji i wartości badanych sił może zdecydowanie skrócić czas przemieszczania się grupy obiektów oraz czy ma to wpływ na ogólny wygląd grupy.

Badania były dwuetapowe. W pierwszym etapie przeprowadzono wstępne symulacje zakładając, że wszystkie siły przyjmują takie same wartości współczynników wpływu (wartość 1). W drugim etapie wykonano badania, w których dokonywano zmian wartości poszczególnych sił – wzrost dwukrotny, przy jednoczesnym zachowaniu wartości pozostałych. Wykonano również symulacje, w których wyłączano kolejne siły oddziaływujące na grupę BOID-ów (wartość takiej siły wynosiła 0).

3.2. Środowisko Unity

Do badań został opracowany model w środowisku Unity służącym do tworzenia trójwymiarowych i dwuwymiarowych gier komputerowych, wizualizacji oraz animacji tzw. bytów interaktywnych. Zaletą środowiska Unity jest to, że jest ono bezpłatne oraz może działać w różnych systemach operacyjnych, np. Windows, OS X, i zezwala na tworzenie zarówno aplikacji internetowych, komputerowych, jak i komórkowych. Oprócz tego możliwe jest pisanie skryptów do zarządzania obiektami. Do tego celu można wykorzystać jeden z trzech języków programowania wspieranych przez środowisko Unity: C#, Unity Script i Boo. Silnik Unity posiada również możliwość importu bibliotek dynamicznych (DLL). Ponadto w tym środowisku można stosować kompatybilne hełmy rzeczywistości wirtualnej, np. Oculus Rift i Gear VR.

3.3. Model badawczy

Utworzony wirtualny model badawczy składał się z miejsca generowania BOID-ów, to jest miejsca gdzie zaczynały one swój ruch, przeszkód stojących na ich drodze

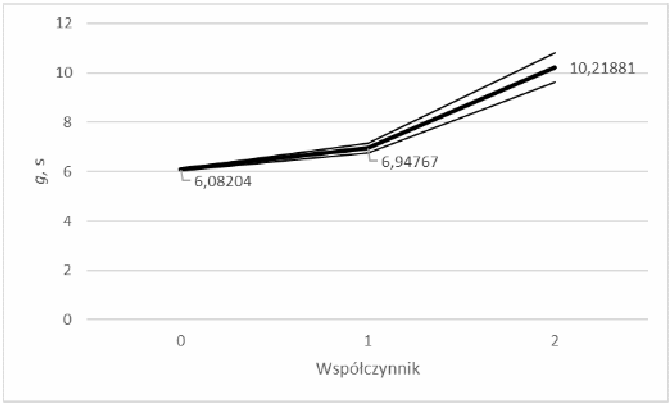
oraz obiektu będącego celem – czyli miejsca, gdzie BOID-y kończą ruch. Do zbudowanego modelu należy również zaliczyć same BOID-y.

Na podstawie opisów algorytmów przedstawionych w pozycjach [1] i [12] oraz własnych modyfikacji zostały napisane skrypty w języku C#. Zmiany polegały na tym, że do procedur dotyczących sił separacji i unikania dodano nowe współczynniki c1 i c2. Współczynnik c1 przyjmował wartości odwrotnie proporcjonalne do odległości między BOID-em a jego sąsiadem. Został wykorzystywany do tego, aby największy wpływ na siłę separacji mieli najbliżsi sąsiedzi. Do wyznaczenia przeszkody najbardziej niebezpiecznej z punktu widzenia ruchu całej grupy wykorzystano współczynnik c2. Współczynnik ten przyjmował wartości odwrotnie proporcjonalne do odległości między centrem przeszkody, a końcem wektora ahead. Wektor ten jest tzw. linią „wzroku" BOID-a. Powstaje jako kopia wektora prędkości, ale ma inną wartość.

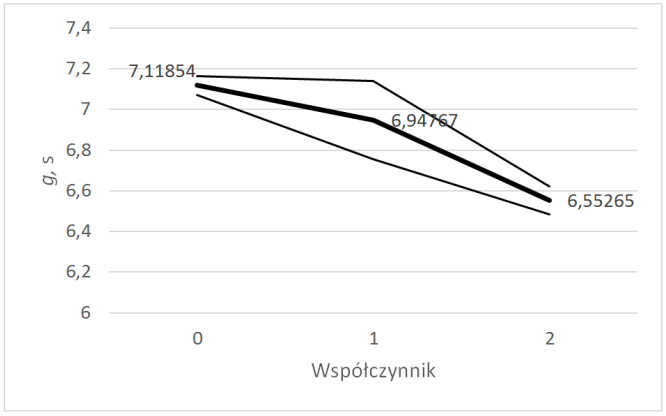
W zastosowanym skrypcie występują również tzw. publiczne zmienne: maksymalna prędkość, promień sfery sąsiadów, odległość separacji, współczynnik separacji, współczynnik spójności, współczynnik wyrównania, współczynnik unikania, odległość widoczności. W badanym modelu te wielkości były potraktowane jako parametry, tzn., że ich wartości nie podlegały zmianom w procesie symulacji. Wielkości te posłużyły do skonfigurowania badanego modelu (rysunek 3), czyli określały ogólne warunki względem których wyliczane były badane siły wpływu na zachowanie BOID-a.

4. Wyniki badań

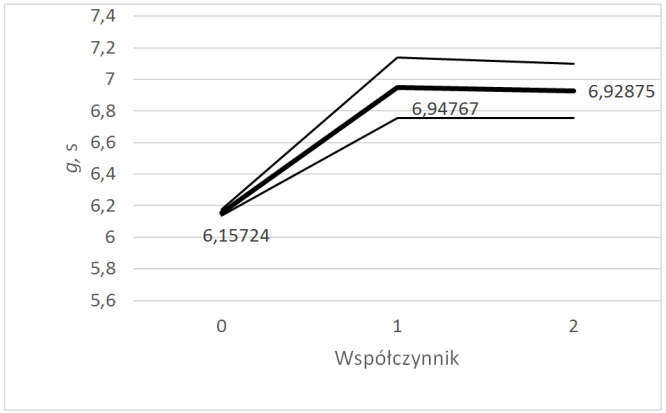
Wyniki wykonanych symulacji w postaci wykresów średnich wartości czasu potrzebnych na pokonanie zadanego dystansu oraz ich średnich odchyleń przy zmianie wartości poszczególnych współczynników sił oddziaływujących na grupę pokazano na rysunkach 4-8.



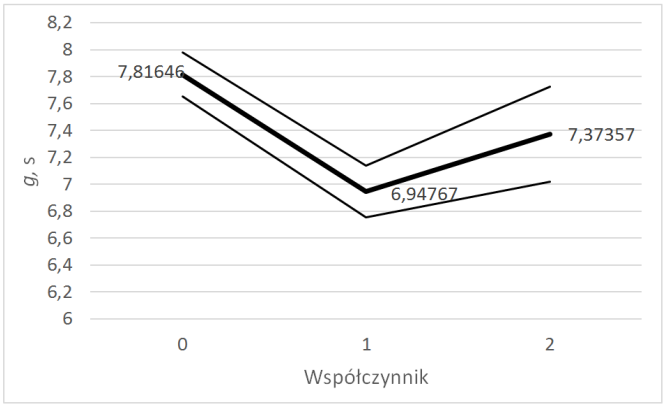
Rys. 4. Średnia geometryczna i średnie odchylenie przy zmianach współczynnika siły separacji



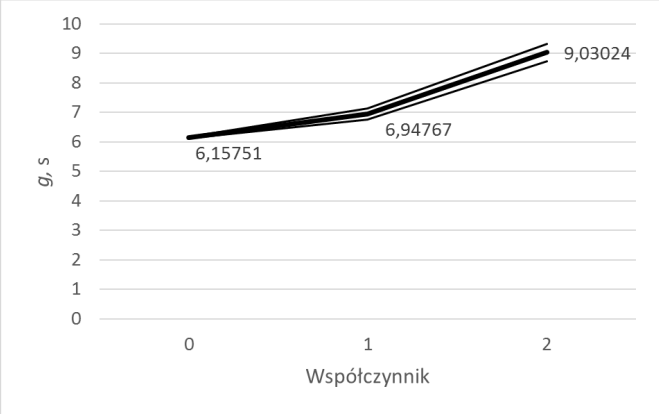
Rys. 5. Średnia geometryczna i średnie odchylenie przy zmianach współczynnika siły spójności



Rys. 6. Średnia geometryczna i odchylenie średnie przy zmianach współczynnika siły wyrównywania

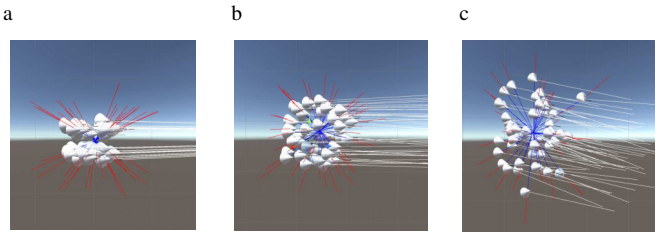


Rys. 7. Średnia geometryczna i odchylenie średnie przy zmianach współczynnika siły unikania



Rys. 8. Średnia geometryczna i odchylenie średnie przy zmianach współczynnika przy każdej sile

Na rysunku 9 pokazano wizualizację położenia grupy BOID-ów zrealizowaną w środowisku Unity.



Rys. 9. Widok grupy BOID-ów w środowisku Unity przy wartościach współczynnika każdej z sił a) – 0, b) – 1, c) – 2

5. Dyskusja i wnioski

Przeprowadzone badania i uzyskane wyniki pozwalają wyciągnąć następujące wnioski.

1) Zwiększenie współczynnika siły separacji z wartości 1 na 2 powoduje ponad 50% wzrost średniego czasu pokonania zadanej drogi przez grupę BOID-ów. Wartość 0 powoduje, że obiekty poruszają się w szyku „gęsiego".
2) Zwiększenie współczynnika siły spójności z wartości 1 na 2 powoduje, że czas pokonania drogi przez grupę obiektów zmniejsza się o około 6%.
3) Zwiększenie współczynnika siły wyrównania z wartości 1 na 2 w zasadzie nie powoduje zmiany czasu średniego na pokonania drogi przez grupę BOID-ów. Natomiast wyłączenie działania tej siły (wartość = 0) zmniejsza średni czas o około 13%.
4) Zmiana wartości współczynnika siły unikania powoduje, że przebieg wykresu jest niemonotoniczny. Najdłuższy średni czas pokonania drogi przez grupę obiektów uzyskujemy przy braku tej siły (wartość 0), przy wartości 1 średni czas zmniejsza się o około 11%, zaś przy wartości 2 zwiększa się o około 6%.
5) Gdy analizujemy jednocześnie identyczny wzrost wszystkich sił (od wartości 1 do 2) to okazuje się, że wzrost średniej wartości czasu na przebycie przez grupę BOID-ów zaplanowanej drogi wzrasta z około 6,95 s do 9,03 s, co stanowi około 30%. Przy pominięciu tych wszystkich sił (wartości = 0) średni czas jest krótszy o około 11%, ale w takiej sytuacji poszczególne obiekty BOID wpadają na siebie, co jest niedopuszczalne.
6) Środowisko Unity dobrze nadaje się do przeprowadzenia symulacji zachowań typu BOIDs oferując przy tym możliwość wykonania wizualizacji, co znacząco ułatwia obserwację poszczególnych etapów przemieszczania się grupy obiektów i zrozumienia działania poszczególnych sił na ich ruch.

Literatura

[1] C. W. Reynolds: Boids. Background and Update: www.red3d.com/cwr/boids, dostęp: 2016.10.10.

[2] Clerc M., Kennedy J.: The Particle Swarm – Explosion, Stability, and Convergence in a Multidimensional Complex Space [W]: IEEE Transactions on Evolutionary Computation. Volume 6, 1, Feb 2002

[3] R. Mendes, J. Kennedy, J. Neves: The Fully Informed Particle Swarm: Simpler, Maybe Better [W]: IEEE Transactions on Evolutionary Computation, Volume 8, 3 June 2004

[4] C. Hartman, B. Benes: Autonomous boids: Computer Animation and Virtual Worlds, 2006, Volume 17

[5] M. Wagner, W. Cai, M. H. Lees: Emergence by Strategy: Flocking Boids and Their Fitness in Relation to Model Complexity: Simulation Conference (WSC), 8-11 Dec 2013

[6] S. Alaliyat, H. Yndestad, F. Sanfilippo: Optimisation of Boids Swarm Model Based on Genetic Algorithm and Particle Swarm Optimisation Algorithm (Comparative Study): files.matlabsite.com/docs/papers/sp/pso-paper-126.pdf, dostęp: 2016.10.10.

[7] K. Szwaykowska and I. B. Schwartz, L. Mier-y-Teran Romero, C. R. Heckman, D. Mox and M. Ani Hsieh: Collective Motion Patterns of Swarms with Delay Coupling: Theory And Experiment: arxiv.org/pdf/1601.08134.pdf, dostęp: 2016.10.10.

[8] C. W. Reynolds: Steering Behaviors for Autonomous Characters: www.red3d.com/ cwr/steer/gdc99/, dostęp: 2016.10.10.

[9] Reynolds C. W.: Flocks, Herds, and Schools:A Distributed Behavioral Model. Computer Graphics, 1987, Nr 21(4)

[10] C. Parker: Boids Pseudocode: www.kfish.org/boids/ pseudocode.Html, dostęp: 2016.10.10.