JCSI 2 (2016) 85-88
Journal of Computer Sciences Institute

Wysłane: 2016-10-15
Przyjęte: 2016-10-17

Analiza możliwości zastosowania środowiska Unity 3D w tworzeniu symulacji postaci

Aleksandra Agnieszka Woźniak*, Małgorzata Plechawska-Wójcik
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Niniejsza praca przedstawia analizę możliwości zastosowania środowiska Unity 3D w kontekście symulacji postaci. Jej celem jest analiza porównawcza procesu animacji postaci w wyżej wymienionym środowisku. Głównym aspektem badawczym jest proces symulacji ruchu, który zrealizowano w projekcie na dwa sposoby. Dostrzeżono przewagę procesu symulacji ruchu opierającego się na blend tree, który cechował się większą płynnością i jak się okazało, także większą wydajnością niż animacja stanowa.

Słowa kluczowe: Unity 3D, symulacja postaci; blend tree; animacja stanowa

*Autor do korespondencji.

Adres e-mail: wozniak.aleksandra92@gmail.com

Performance analysis of Unity 3D environment in development of a character simulation

Aleksandra Agnieszka Woźniak*, Małgorzata Plechawska-Wójcik
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. This work presents the performance analysis of Unity 3D environment character simulation development. Ihe purpose of this work is a comparative analysis of character animation processes in the aforementioned environment. The analysis is based on motion simulation process, which has been shown in the two different ways. The results showed advantage of motion simulation process based on the blend tree, which was characterized by greater fluency and as it also turned out higher efficiency than the state animation.

Keywords: Unity 3D; character simulation; blend tree; state animation

*Corresponding author.

E-mail address: wozniak.aleksandra92@gmail.com

1. Wstęp

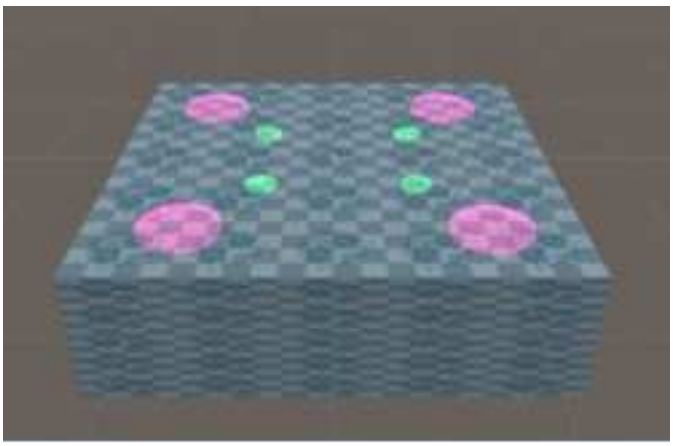
Trójwymiarowe postacie humanoidalne są powszechnie zauważalne w grach, filmach, aplikacjach mobilnych jak i webowych. Zastosowanie modeli 3D w grze komputerowej lub symulacji środowiska (tzw. terrarium) wymaga odpowiedniej interwencji w celu skojarzenia takiego modelu z odpowiednim zestawem ruchu i mechanizmów kontrolnych. Włączenie takiego modelu w system animacji jest procesem złożonym, dającym wiele możliwości poczynając od lokomocji, manipulacji obiektem a kończąc na syntezie mowy i synchronizacji ruchu warg. Wytwarzanie trójwymiarowej postaci humanoidalnej składa się z wielu etapów, w tym wykonania modelu postaci, określenia szkieletu dla tego modelu, manipulacji modelem w zależności od ruchu szkieletu, dołączenia algorytmów ruchu oraz kontroli, a ostatecznie instruowania postaci do wykonywania ruchów. Ze względu na popularność i szeroki zakres zastosowania modeli 3D powstaje coraz więcej rozwiązań przeznaczonych do ich obsługi. Takim rozwiązaniem jest również zintegrowanie środowisko oparte o silnik do tworzenia gier Unity 3D.[1,2,3,5]

1.1. Założenia projektu

Celem pracy jest analiza możliwości zintegrowanego środowiska Unity 3D w kontekście symulacji postaci. Analizę oparto na projekcie symulacji postaci w wyżej wymienionym środowisku. Głównym aspektem badawczym jest proces symulacji ruchu, który zrealizowano w projekcie na dwa sposoby: animacja stanowa oraz blend tree. Następnie typy animacji poddano wnikliwej analizie oraz rzetelnemu porównaniu, na których oparto wnioski niniejszej pracy. Założono, że animacja opierająca się o blend tree, ze względu na znacznie większe możliwości tzw. miksowania ruchu będzie w sposób bardziej realistyczny oraz płynniejszy ukazywała ruch postaci humanoidalnej, a niżeli animacja stanowa. Kolejną hipotezą wynikającą ze złożoności animacji blend tree stała się jej mniejsza wydajność w stosunku do animacji stanowej. Na potrzeby wyżej postawionych tez skorzystano z gotowego modelu postaci typu rigged and skinning, posiadającego gotowy zestaw animacji.

2. Projekt symulacji postaci oraz jej animacji

Badanie oparte zostało na dwóch oddzielnie stworzonych w Unity 3D projektach symulacji ruchu. Obie symulacje zostały przeprowadzone w takich samych warunkach. Na potrzeby projektu została stworzona plansza utworzona z obiektu typu cube, po której poruszać się będzie analizowana postać. Plansza zawiera również obiekty, które wymuszają na postaci ściśle określone zachowania w momencie kolizji. Kolizja z obiektem goal (różowa kapsuła) – symulacja animacji zwycięstwa. Obiekt checkpoint (zielona kapsuła) – po spadnięciu postaci z planszy powraca ona w miejsce ostatniej kolizji z obiektem tego typu przed upadkiem.



Rys. 1. Widok planszy po której poruszają się postacie.

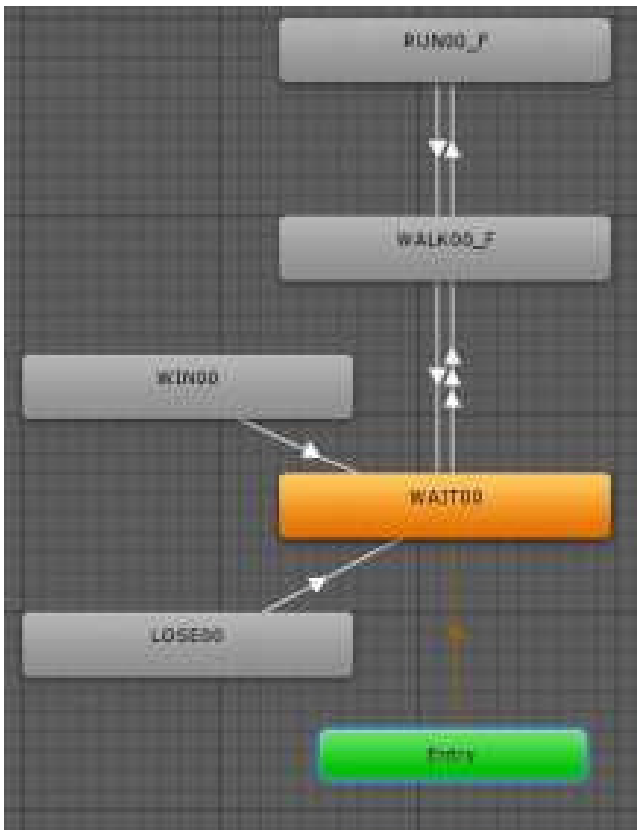
Dla pełnego wykorzystania symulacji ruchu, wybrano model humadoidalny (model postaci Unity Chan dostępny pod licencją Unity Technologies Japan G.K)[9], posiadający rigged oraz skinned typ siatki (ang. mesh). Termin rigged oznacza proces tworzenia hierarchii połączeń inaczej zwanego szkieletem. Definiuje on położenie oraz przemieszczanie się względem siebie kości znajdujących się w siatce reprezentującej obiekt 3D. Proces ten pozwala na pełną kontrolę ruchu postaci. Jeżeli chodzi natomiast o termin skinned jest to proces łączenia szkieletu z siatką postaci. Wybrana postać zawiera również zestaw gotowych animacji, które wykorzystano na potrzeby opisanego w niniejszej pracy projektu.[1,4,6].

.

Rys. 2. Wykorzystany model postaci Unity Chan dostępny pod licencją Unity Technologies Japan G.K

2.1. Animacja stanowa (ang. state)

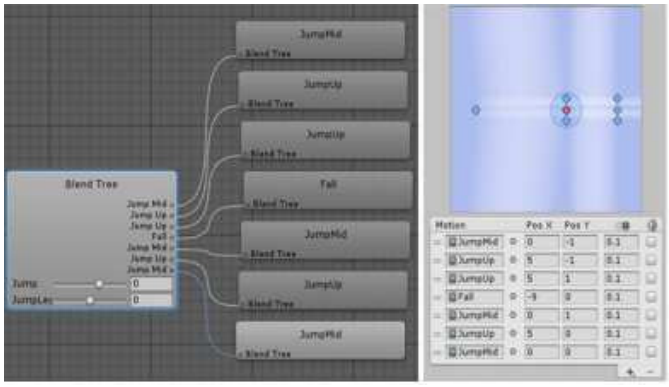
Przedmiotem badania jest analiza porównawcza dwóch rozwiązań symulacji ruchu postaci. W pierwszym projekcie animacja stworzona została poprzez animacje stanową. Główną ideą animacji stanowej jest określona hierarchia wykonywanych ruchów. Postaci można nadać animacje, które będzie wykonywała w zadanej kolejności oraz pod zdefiniowanymi warunkami. Mówiąc ogólnie postać posiada restrykcje przejścia z jednego stanu w drugi i nie jest możliwe jej bezpośrednie przejście z dowolnego stanu w inny. Dla przykładu, jeżeli chcemy uzyskać animację skoku podczas biegu to możliwość przejścia do niej występuje tylko w przypadku, gdy postać jest już w stanie biegu, a nie w stanie spoczynku.



Rys. 3. Schemat animacji stanowej stworzonej na potrzeby projektu symulacji

2.2. Blend tree

W drugim projekcie animacja stworzona została poprzez blend tree. Blend tree pozwala na tzw. miksowanie animacji. Umożliwia płynne mieszanie wielu animacji poprzez wcielanie ich części między siebie w różnym stopniu. Udział poszczególnych części animacji w efekcie końcowym kontrolowany jest przez parametry mieszające (ang. blending parameter).[7]



Rys. 4. Schemat blend tree animacji skoku (po lewej) oraz jego parametry (po prawej) stworzony na potrzeby projektu symulacji

2.3. Analiza porównawcza względem parametrów wizualnych

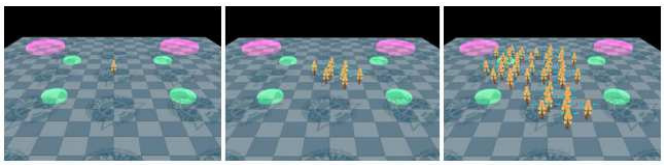
Pierwszy etap analizy miał na celu porównanie obu typów animacji pod względem wizualnym. W obu

przypadkach przedmiotem był obiekt postaci poruszający się po planszy. Porównania dokonano dzięki stworzonym na potrzeby analizy parametrom wizualnym. Wśród wyróżnionych kryteriów znalazły się: płynność ruchu, łączenie animacji, manipulacja zachowania obiektu.

Płynność ruchu jest parametrem opisującym płynność przejść animacji z jednej w drugą oraz stopień, w jakim ruch przypomina realistyczne zachowanie postaci humanoidalnej. Łączenie animacji opisuje w jaki sposób animacje zostały połączone ze sobą, natomiast manipulacja zachowania obiektu świadczy jakie elementy miały wpływ na kontrolę ruchu postaci

2.4. Analiza porównawcza względem parametrów wydajnościowych

Kolejny etap analizy miał na celu porównanie obu typów animacji pod względem wydajnościowym, przy pomocy wbudowanego w środowisko Unity Profilera. Jako główny parametr wzięto pod uwagę zużycie procesora. Analiza wydajnościowa składała się z trzech etapów. W pierwszym etapie wyświetlono jeden obiekt, w następnym 10, a w ostatnim 50. Podczas każdego z nich zebrane zostały dane dotyczące zużycia procesora w trakcie działania poszczególnych animacji.



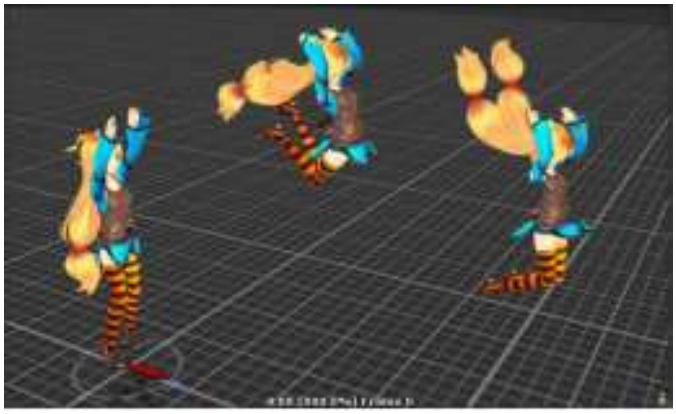
Rys. 5. Etapy analizy wydajnościowej (kolejno): etap I, etap II, etap III

3. Otrzymane wyniki analizy

3.1. Analiza względem parametrów wizualnych

Dla animacji stanowej przejścia okazały się być ostre. Na pierwszy rzut oka ruch jest zbliżony do realistycznego i wydaje się być całkiem naturalny, jednak przy dłuższej obserwacji zauważono wyraźne przejścia z jednego stanu w drugi. Jako przykład warto podać animacje chodu oraz biegu. Postać w pewnym momencie zaczyna bieg i nie wykazuje dla tego elementu przyspieszenia, jakie pojawia się w ruchu naturalnym podczas przechodzenia ze stanu chodu do biegu. Co za tym idzie zauważono, że animacje następują jedna po drugiej nie przenikając się w żadnym stopniu. Jeżeli chodzi o manipulacje zachowania obiektu stwierdzono, że na postać działa tylko fizyka. Dla animacji skoku działająca na postać siła grawitacji sprawia, że postać porusza się w górę i w dół, ale nie wykonuje animacji skoku tylko ciągle biegnie.

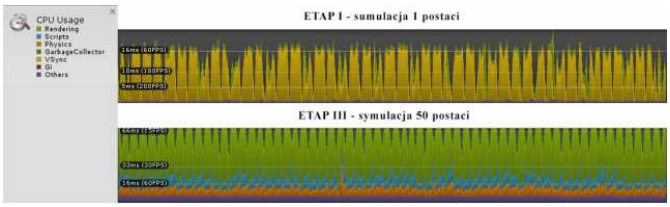
Dokonując analizy animacji blend tree stwierdzono, że symulacja w pełni przypominała poruszanie się obiektu w sposób realistyczny. Postać płynnie przechodziła z chodu w bieg co powodowało naturalność ruchu z jakim się poruszała. Zauważono również, że animacje miksują się między sobą. Postać z chodu poprzez przyspieszanie zaczyna biec. Na podstawie obserwacji skoku stwierdzono, że postać nie tylko porusza się w górę i w dół (co sprawia siła grawitacji), lecz również wykonuje animację skoku. Głównym efektem, na którego podstawie wysunięto wniosek był ruch włosów oraz rąk. Podczas wznoszenia się postaci w górę włosy oraz ręce płynnie się unosiły, natomiast w końcowej fazie skoku płynnie opadały.



Rys. 6. Przebieg ruchu podczas animacji skoku dla blend tree

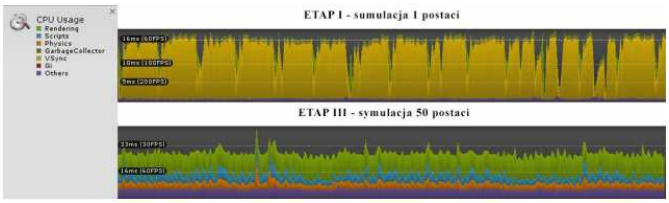
3.2. Analiza względem parametrów wydajnościowych

Na potrzeby niniejszej pracy zostaną przedstawione wykresy zużycia procesora dla obu animacji, dla etapu I oraz etapu III.



Rys. 7. Zużycie procesora dla animacji stanowej

Po przeanalizowaniu wykresów animacji stanowej, dla pierwszego etapu ilość oscyluje w granicy 60 FPS natomiast dla trzeciego 15 FPS.



Rys. 8. Zużycie procesora dla animacji blend tree

Jeżeli chodzi o animacje blend tree to podczas symulacji jednej postaci ilość oscyluje w granicy 60 FPS, natomiast dla 50 postaci 30 FPS..

4. Wnioski

Jak założono we wstępie animacja opierająca się o blend tree ukazywała ruch postaci humanoidalnej w sposób bardziej realistyczny oraz płynniejszy, aniżeli animacja stanowa.

Spowodowane jest to złożonością symulacji ruchu opartej o blend tree, której główną zaletą okazało się być miksowanie animacji. Jak już wspomniano w rozdziale 3.1. niniejszej pracy postać płynnie przechodziła z jednej animacji w drugą. Taki efekt próbowano również uzyskać poprzez animację stanową. Niestety, jak się spodziewano, w każdej sekwencji ruchu widać wyraźnie moment przejścia z jednego stanu w drugi, za co odpowiada tylko jeden parametr lub kilka warunków. Powoduje to, że ruch jest ostrzejszy niż w przypadku blend tree. Główną zaletą blend tree jest sposób manipulacji zachowania obiektu, na który wpływa miksowanie oraz fizyka, co nie występuje w animacji stanowej. Animacja stanowa opiera się bowiem wyłącznie na fizyce, za którą odpowiedzialny jest Rigidbody

Zaskoczeniem okazały się natomiast wyniki analizy wydajnościowej obu typów animacji. Założono, że ze względu na większą złożoność, blend tree będzie mniej wydajny niż animacja stanowa. Po analizie wykresów przedstawionych w rozdziale 3.2., blend tree okazało się być nie tylko wydajniejszym, ale również stabilniejszym rozwiązaniem niż animacja stanowa. O ile przy symulacji jednego obiektu wyniki w postaci liczby FPS są zbliżone (60 FPS) dla obu badanych rozwiązań, to podczas symulacji 50 obiektów wyniki diametralnie się różnią. W obu przypadkach zauważono spadek wydajności. Porównując jednak liczby dla blend tree zauważono spadek o 30 FPS natomiast w animacji stanowej o 45 FPS, co stanowiło aż 75%. Różnice zaobserwowano także wizualnie. Aplikacja przestała reagować w odpowiedni sposób na przesyłane sygnały sterujące, a co za tym idzie obraz zaczął klatkować. Ponieważ w wykresach animacji opartej o blend tree nie zauważono skoków wydajności jak w przypadku animacji opartej o stany, stwierdzono kolejną przewagę blend tree jaką jest stabilność rozwiązania

Na podstawie przeprowadzonych analiz i testów, należy stwierdzić, że Unity 3D okazało się doskonałym rozwiązaniem w zakresie symulacji postaci. W pełni pozwala na obróbkę modelu 3D dzięki rigged and skinned system. Środowisko umożliwia nie tylko symulacje modelu 3D, ale również daje wiele możliwości i rozwiązań w kontekście symulacji ruchu

Literatura

[1] Miller, Christian, Okan Arikan, and Don Fussell. "Frankenrigs: building character rigs from multiple sources." Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games. ACM, 2010.

[2] Feng, Andrew, et al. "Fast, automatic character animation pipelines." Computer Animation and Virtual Worlds 25.1 (2014): 3-16

[3] Haas, John. A History of the Unity Game Engine. Diss. WORCESTER POLYTECHNIC INSTITUTE.

[4] Unity 5.4 documentation. "Preparing your own character". https://docs.unity3d.com/Manual/Preparingacharacterfromscratch.html. [14.10.2016]

[5] Murdoch, Steven. "Agent-oriented modelling in the production of 3D character animation." Studies in Australasian Cinema 10.1 (2016): 35-52.

[6] Suma, Evan A., et al. "Rapid generation of personalized avatars." 2013 IEEE Virtual Reality (VR). IEEE, 2013.

[7] Unity 5.4 documentation. "Blend Trees". https://docs.unity3d.com/Manual/AnimationStateMachines.html. [14.10.2016]

[8] Unity 5.4 documentation. "Animation State Machines" https://docs.unity3d.com/Manual/class-BlendTree.html [14.10.2016]

[9] Unity Chan – official website. "Unity-Chan License Terms – Summarized Version 2.00" http://unity-chan.com/contents/guideline_en/ [14.10.2016]

[10] Unity – Game Engine. https://unity3d.com/ [14.10.2016]