Skinning

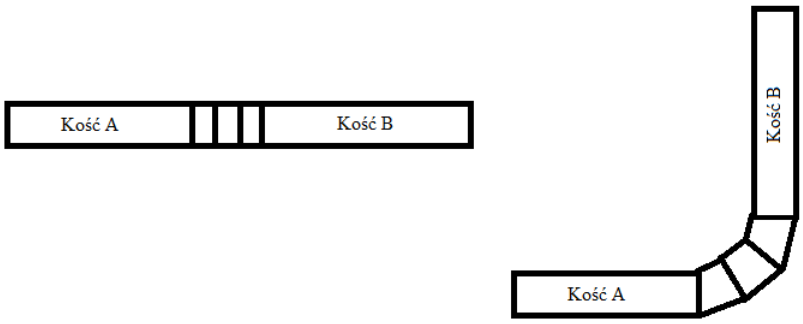
Bartosz Głowacki

Wrzesień 2019

Spis treści

1 Wprowadzenie

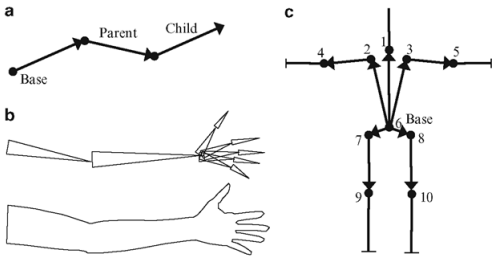
Skinning jest techniką dzięki, której w prostu sposób możemy modyfikować kształt siatki. Polega ona na tym, że tworzymy szkielet dla jednolitej siatki (nie rozbijamy jej na osobno animowane części odpowiadające poszczególnym kościom). Na wierzchołek skóry może oddziaływać więcej niż jedna kość. Jego ruch jest więc wyznaczany na podstawie średniej ważonej przekształceń finalnych, które oddziaływują na wierzchołek. Dzięki temu możliwe jest uśrednienie wierzchołków w stawach dające wrażenie łagodnego przejścia i elastyczności skóry.



Rysunek 1: Modyfikowanie siatki przy pomocy szkieletu. Widać, że na wierzchołki znajdujące się w stawach mają wpływ obie kości.

1.1 Hierarchie układów odniesienia

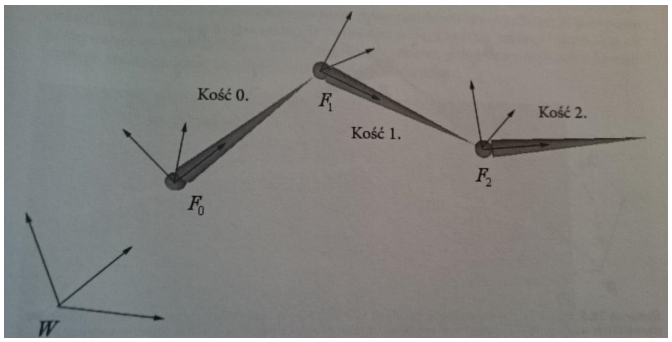
Zanim jeszcze będziemy liczyć przekształcenie siatki, należy skupić się na szkielecie. Zauważmy, że jeśli będziemy wykonywać ruchy ręką to między innymi położenie dłoni będzię zależeć od pozycji przedramienia, a ruch przedramienia będzie zależeć od ruchu ramienia. Widzimy tu zależność rodzic - dziecko. Zatem tak jak w przypadku projektu *Puma* przekształcenia kolejnych kości będą zależeć od przekształceń kości im nadrzędnej.



Rysunek 2: Przykładowy szkielet

1.1.1 Ujęcie matematyczne

Dla zachowania zwięzłości będziemy pracować z hierarchią obwjmującą ramię (kość główna), przedramię i dłoń, które oznaczymy odpowiednio jako Kość 0, Kość 1 i Kość 2. Znając zależnosci między elementami hierarchii, możemy je uogólnić do obsługi bardziej skomplikowanych przypadków. Każdy obiekt hierarchii ma swój lokalny układ współrzędnych. W środku tego układu znajduje się staw obrotowy, który umożliwia obracanie (Rys. 3).

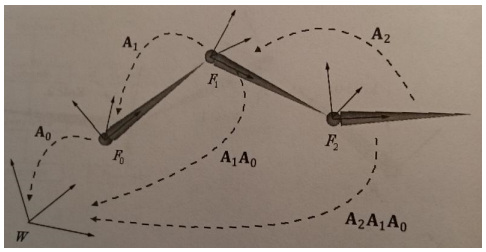


Rysunek 3: Geometria każdej kości jest opisana w jej własnym lokalnym układzie współrzędnych.

Ponieważ wszystkie układy współrzędnych współistnieją w tym samym wszechświecie, możemy określich ich wzajemne relacje. A dokładniej, opisujemy każdy układ współrzędnych względem układu współrzędnych rodzica. Rodzicem układu współrzędnych kości głównej F_0 jest układ współrzędnych przestrzeni świata W; inaczej mówiąc, układ współrzędnych F_0 jest opisany w układzie współrzędnych świata. Mając określoną zależność miedzy układami współrzędnych dziecka i rodzica, możemy przekształcić układ dziecka do przestrzeni rodzica za pomocą macierzy przekształcenia. Będzie to ten sam rodzaj przekształcenia co zamiana współrzędnych lokalnych na współrzędne świata. Z tą różnicą, że zamiast przekształcać z przestrzeni lokalnej do przestrzeni świata, przekształcamy z przestrzeni lokalnej do przestrzeni rodzica. Niech A_2 będzie macierzą przekształcającą geometrię z układu F_2 do F_1, A_1 macierzą przekształcającą geometrię z układu F_1 do F_0, a A_0 macierzą przekształcającą geometrię z układu F_0 do W. Wtedy i-ty obiekt w hierarchii ręki możemy przekształcić do przestrzeni świata przy użyciu następującej macierzy M_i:

$$M_i = A_i A_{i-1} ... A_1 A_0 \tag{1}$$

Rysunek 4 jest graficznym przedstawieniem wzoru 1. Ogólnie rzecz biorąc, aby przekształcić obiekt w hierarchii ręki, stosujemy do niego i wszystkich jego przodków przekształcenie dziecko - rodzic, przechodząc przez hierarchię kolejnych układów współrzędnych do momentu, aż obiekt znajdzie się w przestrzeni świata.



Rysunek 4: Zależności między poszczególnymi kośćmi

1.2 Siatki - Skóry

1.2.1 Przkeształcenie kość - szkielet

W celu usprawnienia naszego programu zmodyfikujemy nieco sposób przekształcenia kości opisany w poprzednim rozdziale. Tym razem będziemy przekształcać z układu współrzędnych kości głównej do układu współrzędnych świata w oddzielnym kroku. Tak więc zamiast szukać macierzy świata, użyjemy dla każdej kości macierzy kość - szkielet (tj. przekształcenia z lokalnego układu współrzędnych kości do układu współrzędnych kości głównej). W poprzednim rozdziale poruszaliśmy się w górę drzewa szkieletu, zaczynając od dowolnej kości i przechodząc do jej rodzica oraz kolejnych przodków. Okazuje się jednak, że bardziej efektywne jest odwrotne podejście, w którym zaczynamy od kości głównej i poruszamy się w dół drzewa. Otrzymujemy następujący wzór dla i-tej kości:

$$koscSzkielet_i = dzieckoRodzic_i * koscSzkielet_r \tag{2}$$

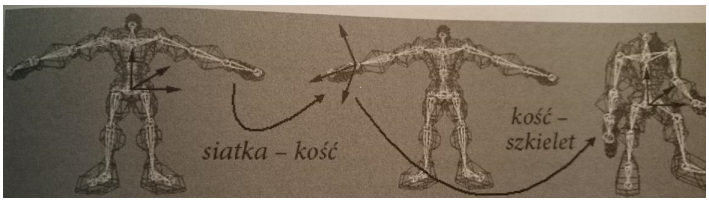
W powyższym wzorze r oznacza rodzica kości i. Przekształcenie $koszSzkielet_r$ przenosi geometrię bezpośrednio z układu współrzędnych kości r do układu współrzędnych kości głównej.Zatem aby znaleźć się w układzie współrzędnych kości głównej, wystarczy przekształcić geometrię z układu kości i do układu kości rodzica r, a resztę załatwi przekształcenie $dzieckoRodzic_i$.

Pojawia się jednak pewien problem. Podczas przetwarzania i-tej kości musimy znać przekształcenie kość - szkielet jej rodzica. Warto jednak zauważyć, że jeżeli poruszamy się w dół drzewa, przekstałcenie kość - szkielet rodzica będzie zawsze obliczane przed przekształcenie kość - szkielet dziecka. Technika zstępująca gwarantuje, że dla każdej kości i dysponujemy gotową macierzą przekształcenia kość - szkielet rodzica, a więc jesteśmy tylko o krok od obliczenia przekształcenia kość - szkielet dla kości i.

1.2.2 Przekształcenie siatka - kość

Wierzchołki siatki kontrolowane przez kość nie są określone w jej układzie współrzędnych (są opisane w przestrzeni łączenia - układzie współrzędnych, w którym była modelowana siatka). Przed zastosowaniem wzoru 2 musimy zatem

przekształcić najpierw wierzchołki z przestrzeni łączenia do przestrzeni kontrolującej je kości. Służy do tego tak zwane przekształcenie siatka - kość. Zasadę jego działania pokazano na rysunku 5.



Rysunek 5: Kolejne przekształcenia na siatce

A więc przekształcając wierzchołki za pomocą macierzy siatka - kości B, przenosimy je z przestrzeni łączenia do przestrzeni kości B. Kiedy wierzchołki są już w przestrzeni kości B, stosujemy przekształcenia kość - szkielet B, aby umieścić kość ponownie w przestrzeni postaci w aktualnej pozie. Wprowadzimy teraz nową transformację: przekształcenie finalne będące złożeniem przekształcenia siatki - kość oraz przekształcenia kość - szkielet. Macierz przekształcenia finalnego i-tej kości F_i obliczamy ze wzoru:

$$F_i = siatkaKosc_i * koscSzkielet_i \tag{3}$$

2 Skinning

Naszym pierwszym celem będzie stworzenie prostej sceny przedstawiającej model zawierający szkielet, który będziemy mogli modyfikować przy użyciu GUI. W tym celu będziemy potrzebować podstawowych struktur do reprezentacji szkieletu i danych w siatce.

W pliku vertexDef.h w projekcie *DirectX 11 Utils* została już stworzona struktura **VertexA**.

```
struct VertexA
{
    XMFLOAT3 position;
    XMFLOAT3 normal;
    XMFLOAT3 weights;
    BYTE indices[4];
};
```

Zwróćmy uwagę na dwa ostatnie pola (*weights i indices*). Tak jak było wcześniej mówione pozycja wierzchołka jest wyznaczana na podstawie średniej ważonej przekształceń finalnych, które oddziaływują na niego. Zatem musimy przetrzymywać w wierzchołku także informacje o wagach oraz indeksach kości, które mają na niego wpływ. W praktyce zauważono, że rzadko zdarza sie tak, że na

wierzchołek oddziałowują więcej niż cztery kości, dlatego w naszym rozwiązaniu przyjeliśmy taką też liczbę. Dodatkowo wagi powinny sumować się do 1, więc dla zmniejszenia przekazywanych danych przekazujemy tylko trzy wartości, a czwarta będzie wyliczana już w shaderze.

Teraz musimy zająć się reprezentacją szkieletu. W pliku *Skeleton.h* znajdują się struktury **Skeleton** i **Bone** reprezentujące kolejno szkielet i kość. Skupmy się na tej drugiej. Macierz *offset* jest przekształceniem siatka - kość, o którym była mowa w rozdziale 1.2.2. Natomiast *parent_index* odpowiada indeksowi rodzica, dzięki temu możemy korzystać z wcześniej obliczonych danych (rozdział 1.2.1); a składowa *trasform* pozwala na wyliczenie przekształcenia dziecko - rodzic.

Teraz już jak mamy omówione wszystkie potrzebne nam struktury możemy przejść do implementacji skinnigu. Pierwszym etapem będzie wyznaczenie macierzy przekształcenia dla każdej kości w następujący sposób (funkcja Update):

- Wyznaczyć przekształcenie *dzieckoRodzic* dla każdej kości. Korzystamy więc ze wspomnianej wcześniej składowej *transform*. W DirectX11 dostępna jest funkcja *XMMatrixAffineTransformation*, która na podstawie skali, punktu (wokół którego jest rotacja), rotacji i translacji wyznacza macierz przekształcenia. Rotacje należy reprezentować w postaci kwaterniona (należy przekonwertować z Eulera).

- Wyznaczyć przekształcenie *koscSzkielet* zgodnie ze wzorem 2. Dla pierwszej kości przekształceniem tym jest przekształcenie *dzieckoRodzic*.

- Wyznaczyć przekształcenie finalnego zgodnie ze wzorem 3.

Kolejnym krokiem jest napisanie shadera wierzchołków. Znajduje się on w pliku *SkinningEffect.fx*. Będziemy postępować następująco:

- Wyznaczamy wszystkie cztery wagi (powinny sumować się do 1) na podstawie danych z wierzchołka

- Inicjalizujemy wektor pozycji i normalny na wektory zerowe

- Do powyższych wektorów będziemy dodawać ich odpowiedniki wyliczone na postawie poniższego wzoru

$$pos+ = weight_i * BoneMatrix[vertex.indices_i] * vertex.pos \quad (4)$$

  Uwaga! W plikach efektów (.fx) wektor mnożymy z prawej strony przez macierz, a nie tak jak to jest w plikach shaderów (.hlsl) z lewej strony.

- Dalej postępujemy tak jak w przypadku zwykłego Vertex Shadera.

3 Animacje

Kolejnym krokiem w naszym kursie będzie zaimplementowanie animacji szkieletu. Każda animacja składa się z sekwencji klatek kluczowych, które wyznaczają

przybliżony przebieg animacji, a dokładniej mówiąc określają położenie, rotacje i skalę obiektu w danym momencie.

W pliku *AnimationClip.h* znajdują się potrzebne struktury do animacji. Struktura **Keyframe** odpowiada klatce kluczowej w animacji. Przechowywana jest w niej pozycja klatki w czasie, pozycja, rotacja (podana w postaci kwaternionu) i skala dla animowanego obiektu. Natomiast w strukturach **BoneAnimation** i **AnimationClip** przechowywane są odpowiednio: sekwencja klatek kluczowych dla pojedyńczej kości oraz dla wszystkich kości.

3.1 Interpolacja pomiędzy klatkami

W powyższym rozwiązaniu każda animacja składa się ze skończonej sekwencji klatek. Nie zawsze pozycja w czasie wskazuje dokładnie na konkretną klatkę, natomiast wyświetlanie tylko tego zbioru, sprawia, że przy małej gęstości animacji, może być widoczny efekt klatkowania. Z pomocą przychodzi interpolacja. Wybieramy dwie klatki pomiędzy którymi znajduje się aktualna pozycja i interpolujemy poszczególne elementy animacji (LERP dla pozycji i skali) oraz (SLERP dla rotacji)

3.2 Przechodzenie po animacji

Kolejnym krokiem przy animowaniu animacji jest przechodzenie pomiędzy nimi. Nie zawsze przełączanie z jednej animacji na drugą będzie wyglądało dobrze, m.in. gdy pozycja końcowa pierwszej animacji znacznie różni się od pozycji w pierwszej klatce drugiej animacji. W takim przypadku możemy postąpić analogicznie jak w rozdziale wyżej. Tworzymy nową animacje, która składa się z dwóch klatek: ostatniej klatki pierwszej animacji i pierwszej klatki drugiej animacji. Należy także zaktualizować pozycje klatek w animacji.