PRACE NAUKOWE Akademii im. Jana Długosza w Częstochowie
Edukacja Plastyczna. Fotografia 2015, z. X

http://dx.doi.org/10.16926/ep.2015.10.08

Dariusz Pleśniak
Akademia im. Jana Długosza
Częstochowa

Grafika 3D – porównanie trzech silników renderujących z programu 3dsMax

3D graphics – a comparison of the three rendering engines 3dsMax program

Streszczenie

Ważnym elementem każdego programu do grafiki 3D jest silnik renderujący (ang. *rendering engine*), od którego zależy jakość renderingu (obrazowania) i czas, jaki trzeba będzie na niego poświęcić. W artykule przedstawione zostały testy, jakim poddane zostały silniki V-ray, Scanline i QuickSilver. Przeprowadzone próby pozwalają poznać ich zróżnicowany zakres możliwości, a w konsekwencji dokonać wyboru adekwatnego do projektu, jaki ma być zrealizowany.

Słowa kluczowe: silniki renderujący, rendering, grafika 3D, 3ds Max, obrazowanie komputerowe

Ważnym elementem każdego programu do grafiki 3D jest silnik renderujący (ang. *rendering engine*), od którego w dużym stopniu będzie zależała jakość renderingu (obrazowania) i czas, jaki trzeba będzie na niego

poświęcić. Ponieważ poszczególne silniki mają różny zakres możliwości, dlatego większość programów do grafiki 3D uwzględnia opcjonalną możliwość zainstalowania dodatkowego silnika innej firmy.

Dla osób nie zajmujących się na co dzień grafiką 3D, może wydawać się nieco dziwne, iż silnik renderujący ma tak duże znaczenie. Szukając analogii możemy odnieść się do rysownika, który zależnie od swojej wizji artystycznej sięgnie po ołówek lub węgiel. Dwa odmienne narzędzia dadzą zupełnie odmienne rezultaty, nawet jeśli artysta zdecyduje się narysować ten sam temat. Problem, czy rysunek ma być wykonany węglem, czy ołówkiem jest tak ważny dla rysownika, jak wybór silnika renderującego dla grafika 3D. Analogia jest jeszcze pełniejsza, ponieważ decydując się na pracę ołówkiem, nie jesteśmy ograniczeni do jednego typu ołówka, ale mamy tu całkiem sporą gamę narzędzi. Nie inaczej jest z silnikami renderującymi. Mają one wiele odmian i specyficzne możliwości, charakterystyczne dla konkretnego mechanizmu renderującego.

Zadaniem silnika renderującego jest zamiana określonej grupy danych matematycznych opisujących wirtualny świat na wyjściową formę dwuwymiarowego obrazu. Tworząc owe obrazowanie wirtualnego świata *engine* oblicza między innymi, w jaki sposób poszczególne obiekty są oświetlone i w jaki sposób będą rzucane przez nie cienie, oblicza załamania światła i odblaski dla poszczególnych obiektów zgodnych z symulowanym materiałem i jego właściwościami. Symuluje gęstość atmosfery czy też głębie ostrości. Zależnie od sposobu realizacji poszczególnych działań poszczególne silniki robią to lepiej lub gorzej. Na niektórych można uzyskać obraz fotorealistyczny, na innych tylko dosyć przybliżone odwzorowanie. Niektóre wyspecjalizowane są np.: do tworzenia obrazu charakterystycznego dla klasycznej animacji Disneya lub uproszczonych wizualizacji dla celów technicznych lub gier. „Najczęściej wykorzystywaną metodą renderowania w programach do grafiki trójwymiarowych jest śledzenie promieni, pozwalająca na bardzo wierne symulowanie obrazu z uwzględnieniem wielu rzeczywistych zjawisk fizycznych. (...) Inne analogiczne metody to raycasting oraz dwie metody oświetlenia globalnego (global ilumination): energetyczna (radiosity) i mapowanie fotono-

we (photon mapping), ponadto wykorzystuje się metody do obrazowania kaustyki (caustic) i cienie powierzchniowe (area shadows), które umożliwiają uzyskanie cieni uwzględniających wielkość emitera światła."[1]

Wybór silnika renderującego uzależniony jest od potrzeb jego zastosowania. Dla twórców gier będzie to wymóg pracy w czasie rzeczywistym. W takim wypadku najczęściej wiele elementów składowych, które się oblicza w przypadku fotorealistycznych renderingów jest albo w jakiś sposób uproszczona, albo w ogóle nieobsługiwana i trzeba stosować sztuczki, które mają ów efekt udawać. Np.: uproszczeniu może zostać poddane obliczenie wyglądu cienia obiektu. Generowanie cienia dla obiektów w pełni nieprzeźroczystych jest stosunkowo łatwe i w miarę szybkie. Sprawa znacznie się jednak komplikuje, jeśli obiekt jest przeźroczysty. W takim wypadku, aby przyspieszyć proces generowania cienia, traktuje się obiekt przeźroczysty dokładnie tak samo jak nieprzeźroczysty. Jednym słowem w scenie, gdzie mamy dwie kule: stalową i szklaną, można zastosować sztuczkę polegającą na pominięciu przeźroczystości obiektu i w efekcie zarówno stalowa kula jak i szklana będą rzucały taki sam eliptyczny jednolity cień. W niektórych przypadkach dla lepszego efektu wizualnego cień szklanej kuli jest w dalszym ciągu jednolitą elipsą, ale jest jaśniejszy, co ma odwzorować jej przeźroczystość. Taki uproszczony efekt w grach sprawdza się bardzo dobrze, ale daleko mu do odwzorowania rzeczywistych efektów, jakie daje światło po napotkaniu szklanej kuli. Brak charakterystycznej cechy tzw. kaustyki, czyli skupiania promieni słonecznych w określonych punktach, podobnie jak to ma miejsce w soczewce. Niemniej brak kaustyki nie jest na tyle mocno zauważalny, aby gry koniecznie musiały go odwzorować. Czasami gdy obiektów, które rzucają cienie, jest bardzo dużo, stosuje się jeszcze mocniejsze uproszczenia. Generowanie cienia dla wielu sylwetek ludzi może na tyle utrudnić generownie obrazu w czasie rzeczywistym, że zastępuje się ów cień sylwetki kształtem elipsy z mocno rozmytymi brzegami. W trakcie ruchu postaci elipsa tylko nieznacznie zwiększa i zmniejsza swoją średnice, co również daje imitację animacji cienia. Wbrew pozo-

[1] Źródło: http://pl.wikipedia.org/wiki/Renderowanie, [stan z 19.5.2015 r., kopia w zbiorach autora].

rom, mimo iż stosowane zabiegi wydają się dosyć prymitywne, to jednak spełniają swoje zadanie. Gra, w której postacie i przedmioty pozbawione są cieni, wygląda znacznie gorzej w stosunku do tych, które stosują najbardziej nawet prymitywne ich zamienniki. Sylwetki postaci z rozmytą elipsą udającą cień są przez nasz mózg lepiej odbierane i pozwalają lepiej odtworzyć wrażenie przestrzeni od tych, które w ogóle ich nie posiadają. Cienie to tylko jeden z elementów, na którym się oszczędza w silnikach renderujących w czasie rzeczywistym. Kolejnym elementem, który jest zmorą dla silników pracujących w czasie rzeczywistym, będzie efekt załamania się światła w obiektach przeźroczystych. Efekt, który możemy zaobserwować na co dzień, wkładając łyżeczkę do szklanki z herbatą i obserwując charakterystyczne złamanie łyżeczki na styku cieczy i powietrza wynikające z efektu załamania się światła, w przypadku symulacji komputerowych wymaga sporych obliczeń, które również mogą zaburzyć renderowanie w czasie rzeczywistym. W takim wypadku ten efekt całkowicie się pomija i obiekt nakładany jest na tło z jednakową przeźroczystością, albo przygotowuje się dla obiektu stałą specyficzną maskę, która określa uproszczone zniekształcenie tła za obiektem, które ma symulować załamywanie się światła. Nieco inny zabieg stosuje się dla powierzchni lustrzanych, choć podobnie jak w przypadku uproszczonego symulowania załamania się światła nie odzwierciedla odbić w sposób pełni realistyczny, to dla celów prostych symulacji sprawdza się dosyć dobrze. Jednym ze sposobów zaoszczędzenia na obliczeniach odbić w symulacjach czasu rzeczywistego jest wyznaczanie stref, w których ma być generowanie odbicie, np.: jeśli będzie symulowane mieszkanie składające się z trzech pomieszczeń: kuchni, łazienki i pokoju, to dla każdego z tych pomieszczeń będzie generowana odrębna tzw. mapa odbić, która jest obliczana i nakładana na sferę. Mapa ta najczęściej generowana jest z punktu widzenia usytuowanego w centrum danego pomieszczenia. Wszystkie drobne obiekty, które mają imitować odbicie światła, pobierają odpowiedni fragment obrazu z owej sfery. Aby łatwiej wyobrazić sobie ten proces, wystarczy, że staniemy w środku jakiegoś pokoju trzymając w ręce chromowaną kulkę. Obraz, który zaobserwujemy na takiej kulce,

to właśnie mapa odbić. Następnie wyobraźmy sobie, że kulka zwiększa swoją średnicę tak, iż całe pomieszczenie znajduje się w jej wnętrzu. Jeśli np.: w symulowanym pomieszczeniu będziemy chcieli umieścić metalowy czajnik, to aby zasymulować odbicia na jego powierzchni, będzie on pobierał obraz odbicia z naszej sfery. Powierzchnia czajnika zostanie podzielona na płaszczyzny, a każdy kawałek takiej płaszczyzny pobierze obraz z mapy odbić sfery z tego miejsca, które jest prostopadłe do danego fragmentu powierzchni sfery. W ten sposób uzyskuje się bardzo szybkie symulowanie odbić. Oczywiście to tylko jeden ze sposobów imitacji. Czasami dla lepszego efektu stosuje się równolegle kilka tego typu zabiegów i choć mogą się one wydać mocno skomplikowane i wymagające sporej liczby obliczeń, to trzeba pamiętać, iż mimo to są i tak znacznie szybsze niż obliczanie rzeczywistych odbić dla każdego obiektu. Przedstawione przed chwilą przykłady stosowania pewnych uproszczeń przy wirtualnym obrazowaniu nie dotyczą tylko silników renderujących w czasie rzeczywistym. Często można je stosować opcjonalnie w wielu silnikach renderujących przeznaczonych do fotorealistycznego kreowania obrazu.

Na potrzeby niniejszego artykułu dokonane zostanie porównanie trzech silników renderujących dostępnych dla programu 3D Studio Max firmy Autodesk. Jest to jeden z popularniejszych programów do grafiki 3D, znajdujący zastosowanie w wizualizacjach architektonicznych, produkcji gier, realizacji 3D na potrzeby filmu i jako narzędzie dla artystów zajmujących się grafiką 3D. Oprócz kilku własnych, można do niego zaimplementować większość silników renderujących dostępnych na rynku. Ideą tego testu było przygotowanie martwej natury, składającej się z kilkunastu obiektów i porównanie jakości obrazów uzyskanych przez różne silniki renderujące oraz czasu potrzebnego do ich obliczenia. Do testów zostały wytypowane trzy silniki o odrębnym sposobie renderownia:

1. Scanline
2. V-ray
3. QuickSilver

Komputer na którym dokonywano testów posiadał następującą konfigurację:

Procesor: AMD FX-8320 Eight-core procesor 3.50 GHz

Ram: 14 GB

Karta graficzna (wykorzystywana przez QuickSilver): NVIDIA GeForce GTX 770

Silniki renderujące:

Scanline - to podstawowy silnik renderujący dla programu 3ds Max. Jest uruchamiany jako domyślny renderer i stworzony przez twórców 3ds Max. W swojej podstawowej formie działa on na bazie obliczania sceny z uwzględnieniem oświetlenia lokalnego. Oznacza to, że program w pierwszej kolejności dokonuje analizy, które obiekty lub ich części będą widoczne, a następnie sprawdza, czy poszczególne płaszczyzny danego obiektu są oświetlone przez jakieś źródło światła, a jeśli tak, to oblicza jego natężenie zgodnie z odległością od źródła światła i kątem pomiędzy powierzchnią obiektu i światłem. Uwzględnia przy tym oczywiście właściwości fizyczne materiału jaki został przyporządkowany dla danego obiektu.

Specyfika takiego obliczania sceny powoduje, że jest to stosunkowo szybki silnik renderujący, dający zadowalające efekty wizualne. Jego dodatkowym atutem jest możliwość selektywnego renderowania obiektów w zaawansowanym trybie *raytrace*, poprzez nakładanie materiału o tej właśnie nazwie na wybrane obiekty. Dzięki takiemu wybiórczemu traktowaniu pewnych obiektów otrzymujemy możliwość renderownaia szkła i powierzchni lustrzanych z uwzględnieniem realnych załamań światła (*Index of refraction*) i odbić. Wadą tego systemu liczenia jest uproszczony system obliczania oświetlenia, przez co silnik nie uwzględnia w ogóle światła odbitego, np.: jeśli na białej płaszczyźnie stworzymy zielony sześcian i oświetlimy go jednym silnym źródłem światła, to płaszczyzny nie oświetlone i obszar cienia będą absolutnie czarne. Dodatkowo na białej powierzchni nie dojdzie do interakcji barwnej i nie pojawi się delikat-

na zielona poświata światła odbitego od zielonego sześcianu, bo ten silnik jej nie oblicza. Do uzyskania w miarę realistycznych odwzorowań światła stosuje się sztuczkę polegającą na dodawaniu tzw. świateł dopełniających o mniejszej intensywności od głównego źródła światła, które nie rzucają cienia i oświetlają zacienione obszary obiektu oświetlonego światłem głównym. Poszczególne obiekty oświetlane takim światłem ignorują ewentualny cień, który byłby na nie rzucany. Zawsze obliczane jest oświetlenie obiektu w taki sposób, jakby pomiędzy źródłem światła a obiektem nie było żadnych przeszkód, czyli uwzględniana jest jedynie intensywność oświetlenia zależna od odległości od źródła światła. Dobre i przemyślane rozmieszczenie świateł dopełniających dobrze imituje światło rozproszone i odbite.

Scanline posiada także bardziej zaawansowany tryb pracy oparty na radiosity.

„Metoda energetyczna (ang. „*Radiosity*") – metoda wykorzystywana w grafice komputerowej do wyznaczenia globalnego rozkładu oświetlenia scen trójwymiarowych. Algorytm wywodzi się z efektów badań nad promieniowaniem cieplnym, w dziedzinie grafiki komputerowej po raz pierwszy pojawił się w 1984, w pracy naukowców z amerykańskiego Cornell University. Radiosity wyznacza globalny rozkład natężenia światła, uwzględniając pochłonięcia i odbicia światła, jakie mają miejsce na wszystkich powierzchniach znajdujących się na scenie. Czyli modeluje prawie dokładnie to samo, co obserwujemy w rzeczywistym świecie, gdzie każda powierzchnia pochłania światło, ale także część odbija. Po lewej – oświetlenie bezpośrednie, po prawej – po użyciu metody energetycznej: światło odbite od czerwonej podłogi nadaje czerwone zabarwienie całemu pomieszczeniu, w narożnikach można dostrzec subtelne cienie. Radiosity uwzględnia wyłącznie odbicia rozproszone, tj. intensywność światła odbitego jest niezależna od kierunku – dzięki temu uzyskane wyniki są niezależne od położenia obserwatora, co pozwala na wielokrotną, dowolną wizualizację sceny bez ponawiania obliczeń. Metoda nie uwzględnia jednak efektów świetlnych zależnych od położenia obserwatora takich jak rozbłyski na powierzchniach metalicznych, odbicia zwierciadlane, załamanie

światła itp. Dobre efekty finalne uzyskuje się po połączeniu tej metody ze śledzeniem promieni, modelującej te zjawiska, które metoda energetyczna pomija."[2] W tym trybie program oblicza światło odbite i daje bardziej realistyczne efekty oświetlenia.

V-ray – to jeden z najpopularniejszych silników renderujących do bardzo precyzyjnych i doskonałych jakościowo renderingów fotorealistycznych. Swoją popularność zawdzięcza wysokiej jakości i względnie niedługim czasie renderingu. Przy obliczaniu sceny korzysta zazwyczaj równolegle z dwóch różnych (z czterech dostępnych) typów technik obliczania wyglądu sceny na bazie światła odbitego (*irradiance map, light cache, brute force i photon map*).

V-ray wykorzystuje metodę energetyczna (*radiosity*), Path tracing „komputerowa metoda Monte Carlo tworzenia fotorealistycznych obrazów scen trójwymiarowych, w której analizowane są losowo wybrane ścieżki promieni światła. Analizując dostatecznie dużo różnych promieni możliwe jest wyznaczenie bardzo dobrej aproksymacji globalnego rozkładu światła. Metoda została zaproponowana przez Jamesa Kajiya w 1986 roku, jako rozwiązanie tzw. równania oświetlenia. Algorytm path tracingu jest dość podobny do raytracingu rekursywnego. Jedną z przewag tej metody nad raytracingiem jest łatwe modelowanie promieniujących powierzchni – w raytracingu rozważane są jedynie światła punktowe. Wada: większa liczba obliczeń, zwykle należy dla jednego piksela obrazu przeanalizować kilkadziesiąt do kilkuset ścieżek."[3] oraz mapowanie fotonowe, polegające na wystrzeliwaniu wirtualnych fotonów ze źródła światła, a nastepnie interpolowaniu wartości energetycznych pomiędzy poszczególnymi próbkami. Im więcej fotonów tym lepsze odwzorowanie oświetlenia.

V-ray jest silnikiem stworzonym przez odrębną firmę i jeśli korzysta się z dedykowanych dla V-ray narzędzi i materiałów, posługuje się wyłącznie swoimi algorytmami. Może jednak wykorzystywać standardowe materiały i wtedy częściowo korzysta z algorytmów 3ds Max.

[2] Źródło: http://pl.wikipedia.org/wiki/Metoda_energetyczna [stan z 19.5.2015 r., kopia w zbiorach autora].
[3] Źródło: http://pl.wikipedia.org/wiki/V-ray [stan z 19.5.2015 r., kopia w zbiorach autora].

QuickSilver – to silnik renderujący wykorzystujący sprzętowe wspomaganie renderowania i bazujący na wbudowanej karcie graficznej. Oblicza on scenę z uwzględnieniem światła lokalnego, a także z wykorzystaniem światła rozproszonego. Nie potrafi uwzględniać *Index of refraction* obiektów przeźroczystych, więc wszystkie szklane obiekty nie załamują obrazu znajdującego się za nimi i wyglądają płasko. Kiepsko radzi sobie także z odbiciami lustrzanymi. O ile na powierzchniach kulistych wygląda to względnie poprawnie, to przy płaskich powierzchniach lub o niewielkim zakrzywieniu często pojawiają się błędy.

Wybór tych konkretnych silników renderujących został podyktowany ich odmiennością w sposobie obliczania sceny, co daje w konsekwencji duże różnice w końcowej wizualizacji. Przy czym silnik Scanline wykorzystywany był bez trybu radiosity, tylko w swojej podstawowej wersji. W trybie radiosity stosowałby podobne algorytmy jak V-ray, a chodziło o porównanie zupełnie odrębnych koncepcji wizualizacji obrazu. Natomiast silnik V-ray testowany był w dwóch trybach jakościowych. W trybie niskiej jakości oraz wysokiej z wykorzystaniem dedykowanych dla silnika V-ray materiałów i źródeł światła oraz z wykorzystaniem standardowych materiałów i świateł.

Do testów porównawczych wykorzystana została prosta scena składająca się z szachownicy oraz figur i pionków szachowych. Przy tworzeniu sceny posługiwano się takimi samymi materiałami o jednakowo zdefiniowanych właściwościach fizycznych. Wykorzystane zostały cztery materiały. Pierwszy materiał dla podłoża, na którym leży szachownica. Drugi materiał dotyczył szachownicy, a trzeci i czwarty – odpowiednio pionków czarnych i białych.

Ilustracja nr 1 przedstawia testową scenę oraz wygląd siatek poszczególnych obiektów. Czarne linie pokazują, z jakich wielokątów zbudowane są poszczególne elementy. Powierzchnie zakrzywione potrzebują zazwyczaj sporej ilości wielokątów, aby wyglądały gładko i nie było widać, że składają się z wielokątów. Dlatego teoretycznie szachownica nie musi posiadać siatki dokładnie odwzorowującej poszczególne pola szachowe.

A na ilustracji czarne linie pokazują, że szachownica składa się ze sporej liczby wielokątów. W tym wypadku podyktowane zostało to sposobem nakładania materiału na obiekt. Każdy wielokąt może otrzymać specyficzny dla siebie materiał. Przy takiej budowie siatki, tworzy się tzw. multi-materiał zawierający w sobie kilka materiałów. W tym wypadku dla tej szachownicy potrzeba trzech: dla białych pól i zewnętrznego obramowania szachownicy, dla czarnych pól i dla cyfr na bokach szachownicy.

Pierwszy test (neutralna scena) – na wszystkie obiekty w scenie nakładany jest dokładnie ten sam materiał o standardowych właściwościach fizycznych. Dodane zostaje także jedno źródło światła oświetlające testową scenę z prawej strony. Przy czym silnik V-ray podmienił źródło światła ze standardowego na jego własny.

Ilustracja 2 przedstawia testową scenę wyrenderowaną silnikiem Scanline. Od razu rzuca się w oczy nienaturalna czerń w nieoświetlonych partiach obrazu. Cień pionka i jego nieoświetlona część jest całkowicie i nienaturalnie czarna. Jest to wynik braku obliczania oświetlenia odbitego od innych obiektów. Krawędzie cienia są jednakowo zmiękczone niezależnie czy są blisko obiektu, czy daleko. Dodatkowo widać jeszcze jeden defekt. Jeśli spojrzy się dokładnie na cień poszczególnych obiektów, to można zauważyć, że nie do końca jest poprawnie odwzorowany. Odnosi sie wrażenie, że cień jest przesunięty za mocno w lewo w stosunku do obiektu, który go rzuca. Tak jest w istocie i wynika ze sposobu obliczania cieni przez Scanline. Oczywiście efekt ten można odpowiednio zmodyfikować i poprawić, ale w założeniach scena nie miała być przygotowywana pod konkretny silnik.

Ilustracja 3 przedstawia tę samą scenę wygenerowaną przez QuickSilver. Pomimo iż silnik ten ma uwzględniać światło odbite, to cienie są stosunkowo ciemne. Jednak poszczególne pionki już nie są jednolicie czarne w strefie zacienionej. Nie ma także wrażenia, iż cień poszczególnych obiektów jest nienaturalnie przesunięty w lewo. Jednak minusem jest krawędź cienia, która podobnie jak w Scanline jest jednakowo rozmyta na całym obwodzie.

Ilustracja 4 przedstawia tę samą scenę wykonaną przez V-ray, która

zdecydowanie prezentuje się najlepiej z tej trójki. Cienie rzucane przez obiekty nie są czarne, a krawędź cienia jest ostra blisko podstawy pionków, by ulegać stopniowemu rozproszeniu wzrastającym z odległością od obiektu. Odbite od podłoża i innych pionków światło rozświetla zacienione miejsca pionków. Dodatkowo V-ray dokonał automatycznej korekty ekspozycji przez co obraz jest jaśniejszy.

W tym teście pod względem jakościowym zdecydowanie wygrywa V-ray, drugi jest QuickSilver a trzeci Scanline. Porównanie czasów renderingów dla obrazka o rozdzielczości 1920 x1 080 pikseli wygląda następująco:
Scanline – 00:15,
QuckSilver – 00:05,
V-ray – 02:28.

Czasy potrzebne do wyrenderowania końcowego obrazu są mocno zróżnicowane. Scanline i V-ray to silniki programowe opierające swoją szybkość na procesorze. QuckSilver wspomaga się sprzętowo i wykorzystuje moc karty graficznej. Widać, że wykorzystanie sprzętowego wspomagania dało mu bezsprzecznie pierwsze miejsce, jeśli chodzi o szybkość renderingu. Najgorzej wypada tu V-ray, który potrzebował dwóch minut i dwudziestu dwóch sekund do wyrenderowania obrazu. Jeśli jesteśmy zainteresowani stworzeniem pojedynczego obrazka, to nie stanowi to problemu. Jeśli jednak wynikiem ma być animacja, to każda sekunda na wygenerowanie jednej klatki obrazu ma tu znaczenie. Dwuminutowy film odtwarzany z prędkością 25 klatek na sekundę potrzebuje ich aż 3000.

Przy utrzymaniu takiego czasu renderingu dla wszystkich klatek dwuminutowy film będzie generowany przez QuickSilver przez 4 godziny i 10 minut. Scanline będzie potrzebował 12 godzin i 30 minut, a V-ray 118 godzin i 20 minut.

Drugi test – scena wykorzystana w pierwszym teście zostaje zmodyfikowana przez dodanie materiałów dla poszczególnych obiektów. Odrębne materiały otrzymuje szachownica oraz podłoże, na którym spoczywa. Pionki otrzymują materiał o tych samych cechach, ale odpowiednio o bia-

łej i czarnej kolorystyce. Poszczególne pionki nie są gładkie, każdy z nich posiada drobne poziome wgłębienia, które mogą przypominać ślady noża pozostałe po niezbyt precyzyjnym toczeniu na tokarce, które następnie zostały wypolerowane. Drobne wgłębienia posiada także sama szachownica.

Ilustracja 5 (Scanline) – Obraz wygląda zdecydowanie lepiej niż. w przypadku poprzedniego testu. Pojawiło się sporo szczegółów w zacienionym obszarze pionków. Co prawda w dalszym ciągu światło odbite nie jest uwzględniane, ale liczone są refleksy światła wynikające z faktury pionków, które wydobywają sporo szczegółów. Cienie wyglądają dokładnie tak jak w teście nr 1.

Ilustracja 6 (QuckSilver) – Pomimo iż materiały w Scanline są dokładnie takie same jak w QuickSilver, końcowy obraz dla tych dwóch silników wyraźnie się różni. Mimo iż rozproszone światło nie jest szczególnie dobrze obliczane przez QuickSilver, to w połączeniu z refleksami na pionkach dobrze buduje ich powierzchnię w zacienionej strefie. Wyraźnie widać, że silnik nie poradził sobie z odbiciami pionków w szachownicy.

W przypadku silnika V-ray zaprezentowane są aż trzy ilustracje. Ilustracja 7 i 8 zostały wykonane w różnych trybach jakości. Silnik V-ray w odróżnieniu od Scanline posiada kilka trybów jakości: bardzo szybki (*Fast Draw*), podgląd (*preview*), niskiej jakości (*low quality*), średniej jakości (*medium quality*), dobrej jakości (*good quality*), produkt (*production*) i ultra (*ultra*). Ilustracja 7 jest wyrenderowana w tzw. niskiej jakości (*low quality*), a ilustracja 8 w tzw. dobrej jakości. Trudno jest przewidzieć, czy te ilustracje po wydruku w stosunkowo małym formacie pozwolą zaobserwować różnicę pomiędzy niską a dobrą jakością. Na ekranie monitora jest to wyraźnie widoczne, szczególnie patrząc na cienie, które są mniej ziarniste. Jakość sceny na ilustracji nr 8 jest zdecydowanie lepsza. Jednakże w obu wypadkach scena jest dobrze oświetlona. Wyraźnie widać fakturę pionków i szachownicy. Cienie są zauważalne, ale zdecydowanie doświetlone przez światło odbite od innych obiektów. Dzięki czemu całość wygląda naturalnie.

Główna różnica pomiędzy tym, co widzimy na ilustracjach nr 7 i 8

względem ilustracji nr 9, wynika z faktu, iż w pierwszych dwóch wypadkach V-ray dokonał konwersji materiałów na swój dedykowany format. Podobnie stało się ze źródłem światła. Natomiast w przypadku sceny z ilustracji nr 9 zostało wymuszone utrzymanie standardowych materiałów i oświetlenia. Mimo iż V-ray generował tą scenę opierając się na standardowych światłach, wygląda ona zdecydowanie lepiej niż ta wygenerowana przez Scanline. To efekt uwzględnienia GI, czyli doświetlenia sceny przez światło odbite od innych obiektów. W ilustracji nr 9 możemy jednak dostrzec, iż wykorzystując standardowe oświetlenie V-ray popełnia te same błędy, co Scanline: cienie wydają się przesunięte lekko w lewo, a krawędzie cienia są na całym jego obwodzie jednolicie rozmyte. Wyraźnie widać, że przy oświetleniu standardowym V-ray korzysta z algorytmów 3ds Max. Scena ze standardowymi materiałami została wygenerowana także w dwóch trybach jakości. Ale chociaż czas renderingu zdecydowanie się różnił, to różnice w obrazie pomiędzy obrazami wygenerowanymi w standardzie *low quality* a *good quality* nie są zauważalne nawet na dużym 27" monitorze. Dlatego została zamieszczona tylko jedna ilustracja.

Czas renderingu dla obrazu 1920 x 1080 pixeli:
Scanlie 02:22,
QuickSilver 00:08,
V-ray (własne materiały i światło) 08:15 (*low quality*) i 21:18 (*good quality*),
V-ray (standardowe materiały i światło) 04:26 (*low quality*) i 10:04 (*good quality*).

W odróżnieniu od czasów poprzedniego testu tylko QuickSilver zdołał utrzymać względnie krótki czas renderingu. Tak duża zmiana wynika ze specyfiki materiałów użytych dla poszczególnych obiektów. Obliczanie faktury błyszczących powierzchni wpłynęło znacząco na czas renderingu. Ciekawe są wyniki dla silnika V-ray w przypadku renderowania sceny ze standardowymi materiałami i oświetleniem w trybach *low quality* i *good quality*, pomimo ponad dwukrotnie większego czasu renderingu. Na 27" ekranie nie można było zauważyć różnicy pomiędzy tymi obraza-

mi. Obraz w *low quality* był delikatnie tylko jaśniejszy w lewym górnym rogu.

Ponownie jako dodatkowa informacja, czas potrzebny na wyrenderowanie 3000 klatek (2–minutowy film):
Scanline = 118 godzin i 20 minut (prawie 5 dni),
QuickSilver = 6 godzin i 40 minut,
V-ray (własne materiały i światło, *low quality*) = 412 godzin i 30 minut (około 17 dni),
V-ray (własne materiały i światło, *good quality*) = 1065 godzin (około 44 dni),
V-ray (standardowe materiały i światło, *low quality*) = 221 godzin i 40 minut (około 9 dni),
V-ray (standardowe materiały i światło, *good quality*) = 503 godziny i 20 minut (około 21 dni).

Trzeci test – zmiany w stosunku do sceny z testu nr 2 dotyczą wszystkich pionków szachowych. Materiał dla pionków jest teraz gładki i częściowo przeźroczysty (50%) z IOR równym 1.6 (jest to standardowe ustawienie) IOR =1.6 jest charakterystyczne dla niektórych typów szkła. Takie zakrzywienie światła uzyskuje także tremolit z IOR =1.600 i zbliżone jest parametrami do szmaragdu IOR = 1.608 oraz kilku innych substancji.

Ilustracja 10 dla silnika Scanline pozwala bez problemu dostrzec w przewróconej figurze konia zakrzywienie obrazu znajdującego się pod nim. Czuje się, że figura jest z przeźroczystego materiału. Jednak widać zafałszowanie obrazu przez nienaturalne jednolite czarne cienie. Przeźroczystość na poziomie 50% powoduje, że białe figury są zmatowione, ale na 27” monitorze dostrzega się bez problemu zakrzywiony obraz tła znajdujący się za nimi. Nieco lepiej widać przejrzystość w czarnych pionkach. W czarnym pionku i w wieży w prawym dolnym rogu obrazu widać błąd przy generowaniu bryły. Wyraźnie rysują się trójkąty zbiegające się w centrum, z których zbudowana jest podstawa pionków. Nie zadziałał tu prawidłowo algorytm wygładzania krawędzi. Scanline opiera się na liczeniu światła lokalnego, ale przeźroczysty materiał wymusza na silniku

przejście w tych miejscach obrazu na typ liczenia Raytrace, dzięki czemu widzimy zakrzywioną przestrzeń za poszczególnymi figurami.

Ilustracja 11 – przedstawia wynik działania silnika QuickSilver. Wspomniany silnik nie potrafi obliczać IOR, dlatego czarne pionki wyglądają jak duchy, a nie jak obiekty przeźroczyste. Nieco lepiej prezentują się białe pionki ze względu na mlecznobiałe zmętnienie. Jednak ogólnie wygląda to bardzo nienaturalnie i nieprzekonująco.

Ilustracja 12 – V-ray w *low-quality* z dedykowanymi materiałami i światłem. Obraz jest bardzo ładny. Zdecydowanie czuje się, że pionki są obiektami przeźroczystymi. obraz wygląda naturalnie.

Ilustracja 13 – V-ray w *good quality* z dedykowanymi materiałami i światłem. Wynik renderingu jest lepszy niż w *low quality*. W tym trybie częstotliwość próbkowania dla kaustyki jest na tyle wysoka, iż dostrzegamy wyraźnie, gdzie w obszarze cienia następuje skupienie światła. Wystarczy porównać cień rzucany przez pionek stojący obok przewróconej figury konia z poprzednią ilustracją. Wyraźnie widać mocne rozświetlenie tuż przy krawędzi pionka. Efekt obliczeń kaustyki dobrze widać w cieniu obu przewróconych figur.

V-ray w *low* i *good quality* ze standardowymi materiałami i światłem niczym sie nie różnią, poza czasem renderingu. Nie widać żadnej poprawy jakości (ilustracja 14). W tym przypadku ponownie V-ray korzysta częściowo z algorytmów 3ds Max, które nie wykorzystują dokładniejszego próbkowania zbieranego przez silnik V-ray. Stad brak różnicy w jakości. Mimo to obraz jest zdecydowanie lepszy jakościowo od tego, co oferuje Scanline. Podobnie jak w obrazku z silnika Scanline wyraźnie rysują się trójkąty zbiegające się w centrum, z których zbudowana jest podstawa pionków. To zrozumiałe, bo wykorzystywane są te same algorytmy. Brak kaustyki i jednolite cienie to kolejny mankament. Jednak nawet takie cienie wyglądają znacznie lepiej od tych z Scanline, dzięki obliczaniu światła rozproszonego (GI).

Czasy:
Scanline = 07:15,
QuickSilver = 00:10,

V-ray (własne materiały i światło) 09:03 (*low quality*) i 22:28 (*good quality*),
V-ray (standardowe materiały i światło) 05:44 (*low quality*) i 11:02 (*good quality*).

Co prawda QuickSilver ponownie ma bardzo dobry wynik jeśli chodzi o czas renderingu, to jednak jakość tego obrazu w zasadzie dyskwalifikuje go w tym teście.

Ciekawy jest wynik silnika V-ray ze standardowymi materiałami i światłem. Daje nam znacznie ładniejszy i naturalny obraz końcowy niż Scanline i robi to szybciej. V-ray z dedykowanymi dla siebie materiałami jest zdecydowanym liderem, jeśli chodzi o odwzorowanie obrazu ale, jeśli chcemy, żeby efekty kaustyki były dobrze widoczne, to musimy renderować w trybie *good quality,* inaczej częstotliwość próbkowania jest zbyt rzadka i kaustyka jest ledwo widoczna.

Czwarty test – ponownie zmiany dotyczą materiału nakładanego na pionki szachowe. Tym razem jest to gładki materiał o silnie odbijającej światło powierzchni, co w przypadku czarnych pionków daje efekt prawie lustrzanej powierzchni. Do sceny zostały dołączone także dwa nowe obiekty.

Ilustracja 15 przedstawia wynik działania silnika Scanline. Obserwując nowe obiekty, widzimy, że zarówno na kuli jak i na sześcianie wyraźnie odbija się otoczenie. Jest to mniej dostrzegalne w przypadku figur i pionków szachowych, ze względu na ich specyficzny kształt oraz budowę sceny. Składa się ona z podłoża, „sufitu" i pustej (czarnej) przestrzeni dookoła. Brak obliczeń światła rozproszonego powoduje, że sufit jest tylko lekko rozświetlany przez stożek światła. Podobnie jak w poprzednich testach cienie są jednolitą ciemną plamą.

Ilustracja 16 – Jak widać silnik QuickSilver nie radzi sobie zbyt dobrze z odbiciami na płaskich powierzchniach, co widać na przykładzie sześcianu jak i samej szachownicy. Niemniej jednak w przypadku pozostałych obiektów wygląda to nieco lepiej. Pomimo to obraz nie jest zbyt naturalny ze względu na niezbyt dobre jakościowo obliczanie świateł odbitych i nienaturalne cienie, podobnie jak ta ma miejsce w silniku Scanline.

Ilustracja 17 i 18 to wynik pracy silnika V-ray z dedykowanym materiałami i odpowiedni w niskiej i dobrej jakości. W tym wypadku scena wygląda dobrze, a różnice pomiędzy niską a dobrą jakością widać w cieniach oraz w większej ilości wzajemnych odbić przedmiotów. Wystarczy porównać kulę na obu ilustracjach.

Ilustracja 19 to V-ray ze standardowymi materiałami i światłem. Podobnie jak w poprzednich testach nie ma różnicy jakościowej pomiędzy obrazami w *low quality* i *good quality*. Pomimo że sposób obliczenia cieni jest taki sam jak w Scanline, to ponownie dzięki obliczaniu GI obrazek końcowy z V-ray prezentuje się lepiej niż ten bezpośrednio ze Scanline.

Czasy:
Scanline = 05:16,
QuickSilver = 00:07,
V-ray (własne materiały i światło) 09:22 (*low quality*) i 28:48 (*good quality*),
V-ray (standardowe materiały i światło) 05:41 (*low quality*) i 14:01 (*good quality*).

Co prawda QuickSilver nie do końca poradził sobie z tym zadaniem, to jednak pozostaje niekwestionowanym liderem szybkości, pod względem jakości obrazu nawet przy wykorzystaniu standardowych materiałów.

Piąty test – scena została rozbudowana o dodatkowe źródła światła. W poprzednich testach było tylko jedno źródło, a w tym teście jest ich sześć. Dodatkowo na pionki szachowe zostały nałożone materiały, które były wykorzystane w teście nr 2.

Ilustracja 20 – Dzięki dodaniu dodatkowych świateł scena rozświetliła się i generowane przez Scanline jednolite cienie nie są już tak zauważalne, przez co scena zdecydowanie zyskała na jakości.

Ilustracja 21 – W przypadku silnika QuickSilver dodanie dodatkowych świateł także poprawiło wcześniejsze niekorzystne oddziaływanie jednolitych ciemnych cieni. Jednak obraz z QuickSilver jest uboższy w stosunku do Scanline.

Ilustracja 22 to wynik pracy V-ray z dedykowanymi dla tego silnika materiałami i światłami. Prezentowana jest tylko jedna ilustracja, ponieważ

w tym teście różnice pomiędzy wersją *low quality* i *good quality* są ledwo dostrzegalne nawet na 27" monitorze. Obraz wygląda w miarę naturalnie i zdecydowanie lepiej niż w przypadku pozostałych dwóch silników.

Ilustracja 23 to ponownie V-ray ze standardowymi materiałami i światłem. W poprzednich testach na standardowych materiałach i jednolitych cieniach V-ray zyskiwał ze względu na obliczanie GI. Dodatkowe światła sprawiły, że problem ten nie jest już tak widoczny w obrazie z Scanlline, dlatego w tym wypadku trudno jednoznacznie się wypowiedzieć.

Czasy:
Scanline = 03:54,
QuickSilver = 00:19,
V-ray (własne materiały i światło) 18:13 (*low quality*) i 28:51 (*good quality*),
V-ray (standardowe materiały i światło) 05:32 (*low quality*) i 15:15 (*good quality*).

W ostatnim teście wyraźnie zwolnił silnik QuickSilver, który potrzebował prawie 19 sekund na wygenerowanie obrazka. To oczywiście efekt dodania dodatkowego oświetlenia. Jednak mimo to pozostaje liderem w szybkości generowania obrazu. Dla silnika Scanline dodatkowe oświetlenie było znacznie mniej obciążające (w porównaniu do testu nr 2), niż w przypadku gdy musiał obliczać dużo obiektów przeźroczystych lub mocno odbijających światło. W przypadku silnika V-ray scena z dedykowanymi materiałami i światłami była obliczana zdecydowanie wolniej, przy generowaniu w niskiej jakości obrazu. W poprzednich testach w przypadku *low quality* czas był zawsze poniżej 10 minut, a dodatkowe źródła światła zwiększyły go do ponad 18 minut.

Tab. 1. Zbiorcze zestawienie wykonanych testów

typ testu	nr testu	Scanline	Quick render	V-ray (low)	V-ray (good)	V-ray(low) Standard	V-ray(good) Standard
bez materiału	1	00:15	00:05	01:25	02:28		
Z materiałami	2	02:22	00:08	08:15	21:18	04:26	10:04
przeźroczysty materiał	3	07:15	00:10	09:03	22:28	05:44	11:02
„lustrzany" materiał	4	05:16	00:07	09:22	28:48	05:41	14:01
Dodatkowe światła	5	03:54	00:19	18:13	28:51	05:32	15:15

Podsumowanie

Oglądając wyniki poszczególnych testów trzeba pamiętać, że zazwyczaj tworząc projekt, konstruuje się scenę w taki sposób, aby jak najbardziej wykorzystać walory danego silnika. W tym wypadku testowa scena była przygotowana w sposób minimalistyczny, dlatego w żadnym teście nie został uzyskany efekt fotorealizmu. Dobór silnika renderującego zależy w dużej mierze od projektu, jaki ma być zrealizowany.

Pod względem jakości zdecydowanym faworytem jest silnik V-ray, ale aby w pełni wykorzystać jego potencjał należy dobrze zdefiniować materiały, które on udostępnia. Zwykła konwersja standardowego materiału na materiał V-ray nie wystarczy. Jakość uzyskiwaną z tego silnika trzeba często opłacić długim czasem oczekiwania. W przypadku pojedynczych obrazów nie stanowi to problemów, ale przy realizacji animacji warto mieć do dyspozycji tzw. render farmę.

Silnik Scanline potrafi dać bardzo zadowalające efekty. Zazwyczaj jednak trzeba dobrze popracować nad rozmieszczeniem świateł. Często wykorzystuje się w nim światła dopełniające, które nie rzucają cienia, a tylko doświetlają wszystkie obiekty. Co ciekawe, takie światła nie zwiększają znacząco czasu renderingu (w teście nr 5 wszystkie światła rzucały cień). Wybór Scanline to kompromis pomiędzy szybkością renderingu a jakością.

QuickSilver mimo swoich wad może być całkiem sensowną alternatywą w przypadku niektórych typów projektów. Jego duża prędkość jest sporym atutem. Poważną wadą jest nieradzenie sobie z obiektami przeźroczystymi i szklanymi płaskimi powierzchniami.

Bibliografia

Druki zwarte

Eliott Steven, Miller Phillip, *3D Studio Max. Doskonałość i precyzja*, t. 1, Helion, Gliwice 1997,

Eliott Steven, Miller Phillip, *3D Studio Max. Vademecum profesjonalisty*, t. 2, Helion, Gliwice 1997,

Kuklo Kamil, Kolmaga Jarosław, *Blender, Kompendium*, Helion, Gliwice 2011,
Murdock Kelly L., *3ds Max 2012. Biblia*, Helion, Gliwice 2012,

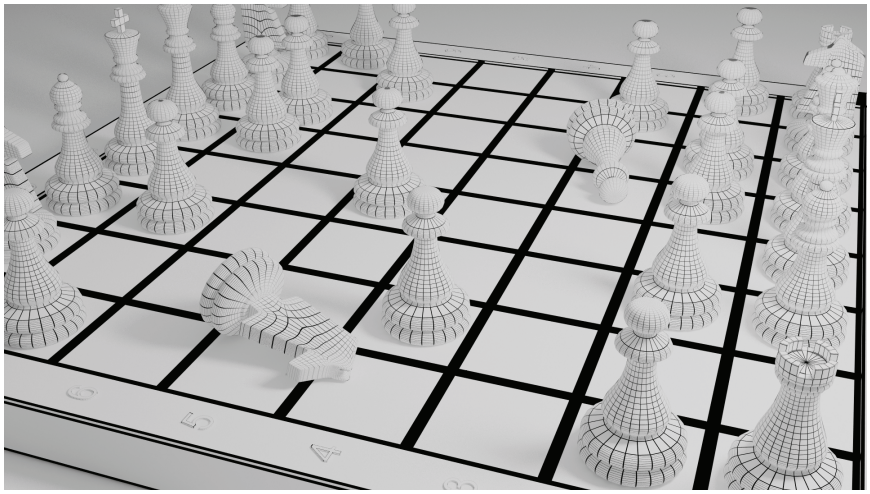
Strony www

http://docs.autodesk.com/3DSMAX/15/ENU/3ds-Max-Help/
http://pl.wikipedia.org/wiki/Metoda_energetyczna.
http://pl.wikipedia.org/wiki/Renderowanie,
http://pl.wikipedia.org/wiki/V-ray

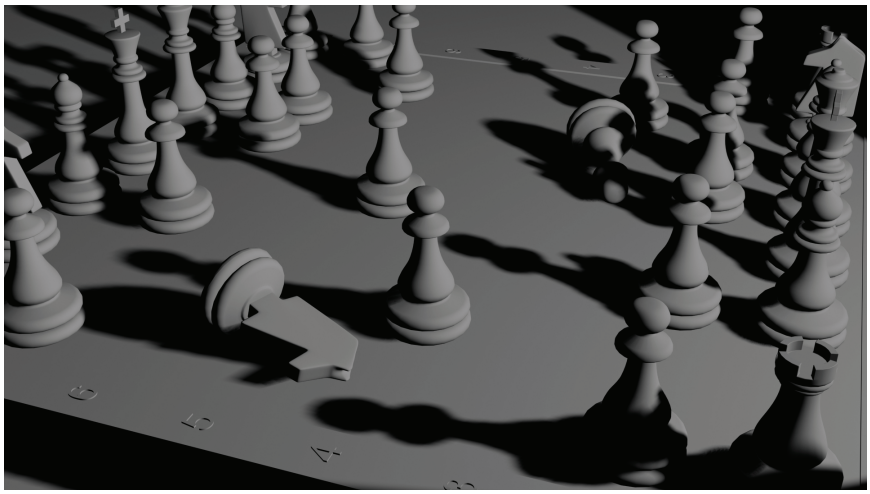
Summary

Rendering engine is an important element of every 3D programme. Quality of rendering (imaging) and the time it takes depends on the engine. The article presents tests of V-ray, Scanline and QuickSilver engines. The author received information on diverse range of their capabilities. In consequence it allows him to choose the right engine for each project.

Key words: rendering engine, rendering, 3D graphics, 3ds Max, computer imaging



1. Scena testowa – siatki obiektów



2. Test 01 – scena bez materiałów – silnik Scanline

3. Test 01 – scena bez materiałów – silnik QuickSilver

4. Test 01 – scena bez materiałów – silnik V-ray, *low quality*

5. Test 02 – silnik Scanline

6. Test 02 – silnik QuickSilver

7. Test 02 – silnik V-ray, *low quality*

8. Test 02 – silnik V-ray, *good quality*

9. Test 02 - standardowe materiały w silniku V-ray, *good quality*

10. Test 03 - przeźroczyste pionki - silnik Scanline

11. Test 03 – przeźroczyste pionki – silnik QuickSilver

12. Test 03 – przeźroczyste pionki – silnik V-ray, *low quality*

13. Test 03 – przeźroczyste pionki – silnik V-ray, *good quality*

14. Test 03 - przeźroczyste pionki – standardowe materiały w silniku V-ray, *good quality*

15. Test 04 – „lustrzane” pionki – silnik Scanline

16. Test 04 – „lustrzane” pionki – silnik QuickSilver

17. Test 04 – „lustrzane” pionki – silnik V-ray, *low quality*

18. Test 04 – „lustrzane” pionki – silnik V-ray, *good quality*

19. Test 04 – „lustrzane” pionki – standardowe materiały w silniku V-ray, *good quality*

20. Test 05 – sześć źródeł światła - silnik Scanline

21. Test 05 – sześć źródeł światła – silnik QuickSilver

22. Test 05 – sześć źródeł światła – silnik V-ray, *good quality*

23. Test 05 – sześć źródeł światła – standardowe materiały w silniku V-ray, *good quality*

24. Szklane pionki – silnik Vray, *low quality*