Projekt: rendering obrazów

1 Ogólna charakterystyka projektu

Projekt polega na zaprojektowaniu i implementacji prostego systemu renderującego widok trójwymiarowej sceny, składającej się z obiektów i źródeł światła. Poniżej przedstawiona jest podstawowa specyfikacja techniczna, jak również kilka propozycji rozszerzeń, wpływających na efektywność i funkcjonalność systemu. Istotną częścią realizacji projektu jest przygotowanie krótkiego raportu, opisującego rozpoznane problemy projektowe i decyzje odnośnie projektu systemu podjęte w celu ich rozwiązania. Bardzo istotnym aspektem jest adekwatne i przemyślane użycie wybranego języka programowania, w szczególności systemu modułów (w OCamlu/SMLu) lub klas typów (w Haskellu) w celu zapewnienia rozszerzalności projektu; ponadto, każde zastosowanie modyfikowalnego stanu czy leniwości powinno być bardzo przekonująco umotywowane.

1.1 Punktacja

Aby projekt został uznany za wykonany (tj. liczba otrzymanych punktów przekroczyła 50), należy

- zaimplementować wszystkie funkcje z sekcji wymaganych
- nadać projektowi odpowiednią, utrzymywalną strukturę z użyciem modułów, klas typów, itp., jak również udokumentować kod
- sporządzić krótki raport opisujący napotkane problemy i wybrane rozwiązania (wraz z uzasadnieniem wyboru).

Aby uzyskać maksymalną liczbę punktów, należy dodatkowo zaimplementować (wraz z dokumentacją i raportem) co najmniej dwa z rozszerzeń "łatwych", lub co najmniej jedno rozszerzenie "trudne". Lista rozszerzeń podana w specyfikacji jest pewną sugestią — można proponować własne, przy czym warto je wtedy skonsultować z prowadzącym. Stopień implementacji rozszerzeń i jakość kodu w oczywisty sposób może wpływać na ocenę projektu.

2 Specyfikacja techniczna

Podstawowym zadaniem programu jest wygenerować widok trójwymiarowej sceny (z pewnego punktu) przy pomocy metody śledzenia promieni (ray-tracing). Metoda ta polega, w skrócie, na prześledzeniu biegu promienia od każdego z punktów obrazu (przez punktowe ognisko lub w konkretnym, danym kierunku) do obiektu, który jest z tego punktu widoczny, a następnie ustalenie widocznego punktu widzianego obiektu. W zależności od właściwości tego obiektu, może to wymagać wykonania dodatkowych obliczeń (w zależności od tego czy i w jakim stopniu obiekt światło rozprasza, odbija, załamuje lub generuje), w tym wypuszczenia kolejnych "promieni" w odpowiednich kierunkach.

Problem wymaga odpowiedniego zareprezentowania sceny, w tym obiektów i źródeł światła, a także samego widoku, w postaci odpowiednich typów danych (w tym typów abstrakcyjnych), a także implementacji odpowiednich funkcji, w tym dotyczących geometrii w przestrzeni trójwymiarowej. Specyfikacja projektu jest w naturalny sposób funkcyjna, z potencjalnie rekurencyjnymi promieniami rzucanymi w celu wyliczenia koloru widzianego punktu. Jednocześnie, chcemy wygenerować pojedynczy widok sceny, więc wysoka efektywność nie jest konieczna (co nie oznacza że należy ją trwonić bez sensu).

2.1 Funkcje wymagane

- Wczytanie opisu sceny i widoku do wyrenderowania z pliku. Format opisu sceny należy zaproponować samodzielnie, dbając przy tym o możliwość łatwego rozszerzenia o nowe rodzaje obiektów, źródeł światła, widoków, itd.
- Wyświetlenie wyrenderowanego widoku.
- Zapisanie wyrenderowanego widoku na dysku (w wybranym formacie pliku).
- Wyrenderowanie sceny zawierającej:
  - obiekty dwóch rodzajów. Silnie sugerowane są: kula i płaszczyzna (ew. fragment płaszczyzny), dla których implementacja przecięcia z promieniem jest łatwa
  - powierzchnia obiektów musi umożliwiać co najmniej całkowite rozpraszanie światła
  - co najmniej jeden rodzaj źródła światła (punktowe o intensywności słabnącej z kwadratem odległości lub "słoneczne" o jednostajnej intensywności zlokalizowane w określonym kierunku) pozwalające na wyliczenie intensywności koloru powierzchni rozpraszającej

– widok dany przez prostokąt o określonym położeniu w przestrzeni i rozdzielczości wraz z ogniskiem przez które powinny przechodzić promienie wychodzące z punktów obrazu (lub, alternatywnie, z kierunkiem w którym zwrócone są wszystkie promienie).

2.2 Funkcje rozszerzone

- Dodanie jako rodzaju powierzchni obiektu powierzchni świecącej własnym światłem (o określonej barwie), powierzchni odbijającej światło (lustro), **oraz** umożliwienie mieszania wszystkich trzech typów powierzchni w wyspecyfikowanym stosunku (to ostatnie powinno umożliwiać również pochłanianie części światła przez obiekt). (**łatwe**)

- Dodanie czwartego (i ostatniego) rodzaju powierzchni: załamania (z określonym współczynnikiem załamania). Należy wziąć pod uwagę możliwość całkowitego odbicia wewnętrznego **i** zewnętrznego (tj. niemożność załamania światła ze względu na kąt padania). (**trudne**, przy tym wypada zrobić również poprzednie)

- Dodanie drugiego z wymienionych wyżej rodzajów źródeł światła. (**łatwe**)

- Użycie struktury danych która umożliwi efektywne znajdowanie obiektów w które może uderzyć promień. (**łatwe**, choć trzeba doczytać)

- Dodanie kolejnego rodzaju obiektów (innego niż kula i fragment płaszczyzny), np. którejś z powierzchni powstałych przez obrót krzywych stożkowych (paraboloida, hiperboloida, etc.). (**łatwe**)

- Dodanie trybu widoku z przesłoną (zamiast punktowego ogniska). Aby zaimplementować ten tryb, trzeba zmienić proces wyznaczania koloru piksela stochastycznie, zarówno w momencie wyznaczania pierwotnego kierunku (promień może przyjść z dowolnego punktu przesłony), jak i przy odbiciu (promień rozproszony może przyjść z dowolnego współpłaszczyznowego kierunku — choć z nierównomierną gęstością prawdopodobieństwa), a także nadać źródłom światła fizyczny rozmiar (kątowy w przypadku światła jednostajnego). (**trudne**, ale też pozwalające na uzyskanie bardzo ciekawych efektów)

- Dodanie możliwości korekcji gamma: w pewnych sytuacjach scena może być bardzo ciemna lub bardzo jasna. Prosta korekcja wykładnicza pozwala wydobyć nadmiernie zaciemnione lub rozjaśnione szczegóły. (**łatwe**)