

Spektralno upodabljanje

Ivo Pajer, 63180218

Abstract

V okviru seminarja za predmet Napredna Računalniška Grafika sem izbral naslov Spectral rendering (sl. Spektralno upodabljanje). Cilj seminarja je bil nadgraditi PathTracer okolje, ki smo ga uporabljali pri drugi domači nalogi tako, da bo podpiral spektralno upodabljanje. Implementacija je uspešna, a je približno 20-krat počasnejša od običajne implementacije z RGB barvno shemo.

1. Uvod

V okviru seminarske naloge pri predmetu Napredna Računalniška Grafika sem si izbral naslov Spectral Rendering. Za cilj seminarja sem si zadal nadgradnjo obstoječega okolje PathTracer, uporabljen pri 2. domači nalogi, ki bazira na prbt upodobljalniku opisanem v knjigi [PH10]. Odločil sem se, da bom poleg spektralnega upodabljanja implementiral tudi simulacijo stekla, da bomo lahko opažali spektralne efekte refrakcije.

2. Spektralno upodabljanje

Spektralno upodabljanje (angl. Spectral Rendering) je tehnika upodabljanja, kjer je prenos svetlobe modeliran s frekvencami svetlobe namesto samo barvami. Ta tehnika je počasnejša od običajne, ki upodobi sceno v rdeči, zeleni in modri komponenti, ki jo nato združi. Spektralno upodabljanje se pogosto uporablja v sodelovanju s sledenjem žarkov, saj tako lahko simuliramo fotone. Rezultat naj bi bila bolj realistična slika, saj simuliramo svetlobne pojave v realnem svetu.

2.1. Reflekcija in Refrakcija

V osnovnem načinu smo reflekcijo in refrakcijo modelirali z uporabo BSDF (angl. *bidirectional scattering distribution function*). Pri spektralnem upodabljanju, pa moramo poleg integriranja skozi vseh smeri žarkov, integrirati še po vseh frekvencah svetlobe. S tem se enačba za računanje BSDF spremeni na:

$$L_o(p, \omega_o) = \int \int f(p, \omega_o, \omega_i, \lambda) L_i(p, \omega_i, \lambda) \cos\Theta_i d\omega_i d\lambda$$

Ker pa se različne frekvence svetlobe lomijo pod drugačnimi koti, moramo tudi to vzeti v obzir pri implementaciji spektralnega upodabljanja.

2.2. Konverzija iz Frekvenčnega prostora v RGB

Ker se pri spektralnem upodabljanju ukvarjam z svetlobo v frekvencah, moramo znati pretvoriti frekvenco barve (ali nabora frekvenc) v njeno RGB predstavitev, da jo lahko prikažemo na ekranu. S tem razlogom bomo uporabili CIE 1931 color space, ki frekvence pretvori v aktivacijo posameznih senzorjev svetlobe v človeškem očesu (palice in stožci). Definiran je bil leta 1931 po naročilu Mednarodne Iluminacijske komisije [wik22]. Sam pa bom v nalogi uporabil približek CIE funkcije, ki so ga predlagali znanstveniki iz NVIDIA leta 2013 [WSS13]. Spodnja slika prikazuje razliko med približno funkcijo in CIE funkcijo.

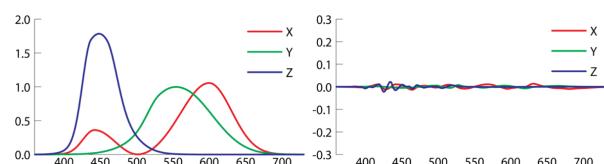


Figure 1: Približna funkcija (levo), primerjava s CIE funkcijo (desno)

Da lahko pridobimo RGB reprezentacijo frekvenc moramo najprej pretvoriti iz frekvenčnega prostora v CIE prostor ali XYZ reprezentacije, iz katere lahko z matričnim množenjem pretvorimo v RGB barvo.

3. Implementacija

Za implementacijo sem se veliko zgledoval po knjigi [PH10], ki zelo podrobno programsko opiše in matematično podpre vse implementacije.

3.1. Specular Transmission

Najprej sem se lotil implementacije refrakcije, ki je v knjigi opisana kot Specular Transmission ali prenos svetlobe. Za refrakcijo moramo vedeti še refrakcijski indeks, ki nam pove, koliko se svetlobi spremeni hitrost v nekem mediju, in posledično za koliko se bo spremenil kot svetlobe pri prehajanju med različnimi mediji po Snellovem zakonu [Dav]. Pri implementaciji refrakcije sem se odločil še implementirati integracijo po frekvenci. To sem naredil tako, da sem z metodo Monte Carlo, uteženi na moči frekvenc izbiral med frekvencami. Vsaka frekvenca ima tudi drugačen lomni koeficient. Do sedaj sem implementiral linearno vpadnost lomnega koeficiente, po višanju frekvence, kar ni popolnoma pravilno, a je dovoljen približek.

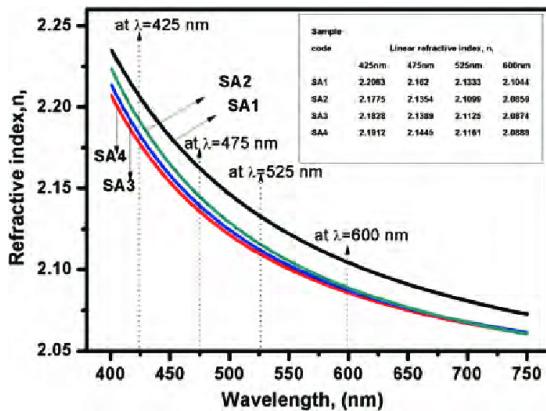


Figure 2: Refrakcijski indeks stekla v primerjavi s frekvenco svetlobe. vir: [ARY*19]

3.2. Glass

Implementacija stekla ni podborneje opisana v knjigi, zato sem v kombinaciji z izvorno kodo PBRT in projektom [Xu]. Postopek modeliranja stekla je zato sledeč: Najprej pogledamo ali žarek potuje skozi notranjost predmeta ali ne. Če potuje skozi notranjost vse ostane isto, le da se glede moči spektra upošteva Fresnel efekt. Nato pa naključno (uteženo na kot med predmetom in žarkom), izberemo ali bomo žarek odbili ali lomili. V primeru lomljenja moramo upoštevati še integriranje po frekvenci. S tem razlogom se stekleni objekti počasi osvetljijo, a je končen rezultat bolj natančen. Za lomljenje svetlobe uporabljamo vse pojave, ki smo jih omenili že zgoraj.

3.3. SampledSpectrum

Za implementacijo frekvenčnega spektra sem uporabil implementacijo iz knjige, ki temelji na naboru valovnih dolžin od 400nm do 700nm z 60 vzorci. Implementirali smo metode za pretvarjanje v XYZ reprezentacijo in nato iz XYZ

v RGB in obratno. Implementirali smo tudi, da lahko konvertiramo iz RGB predstavitev v frekvenčni spekter, kar pomeni, da ne potrebujemo imeti frekvenčne reprezentacije za vsako barvo. SampledSpectrum vsebuje vse metode, ki jih vsebuje že osnovni razred spectrum, zato pretvarjanje med RGB in frekvenčno reprezentacijo ni zahtevno. Za pretvarjanje iz frekvenčnega prostora v XYZ reprezentacijo smo kot omenjeno uporabili približek. Na našo srečo ima članek [WSS13] že napisano kodo za pretvorbo. Za pretvarjanje iz XYZ v RGB pa je uporabljen enačba CIE 1931.

V okviru SampledSpectrum smo lahko implementirali tudi različne iluminatorje, ki so standardni. V našem primeru smo implementirali še iluminator CIE standard illuminant D, ki simuliira sončno svetobo opoldne v evropi. Implementiramo lahko poljuben spekter barv, če vemo frekvenčni svetlobni odziv tega spektra barv.

4. Rezultati

Odločil sem se, da bom za prikaz spektralnega upodabljanja izbral sceno, ki vsebuje en rdeč, en zelen in 3 bele zidove z lučjo na vrhu. V sceni bosta tudi modra krogla, ki je mat materijala, ter steklena krogla, ki bo pred njo in rahlo višje. Takšna scena nam bo pokazala, kako deluje refrakcija in spektralno upodabljanje.

Program je občutno počasnejši od osnovne implementacije, saj je namesto vektorja s 3 elementi treba množiti vektorje s 60 elementi. To prinaša približno 20-kratno povečanje časovne zahtevnosti. En prehod čez sceno sedaj traja približno 30 sekund, v primerjavi z osnovno implementacijo, ki prehod naredi v rahlo več kot sekundi.

V spodnji sliki lahko vidimo problem, ki ga prinaša vzorčenje frekvenc pri refrakciji. Opazimo, da se steklena krogla barva počasi.

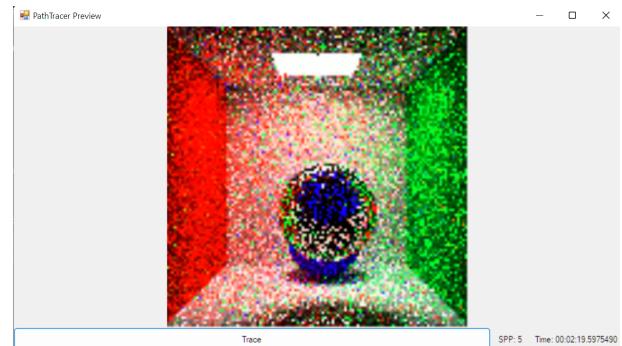


Figure 3: Scena po 5 ciklih

V naslednji sliki, pa lahko opazimo sceno po 124 ciklih, kjer lahko vidimo, da se krogla sčasomaobarva. Tu opazimo tudi lom svetlobe, ki deluje tako kot bi pričakovali, da sceno zrcali iz obeh strani. Opazimo tudi, da se pod kroglo

pojavi snop svetlobe, ki je fokusiran kot v leči, kar nam daje pozitivne povratne informacije.

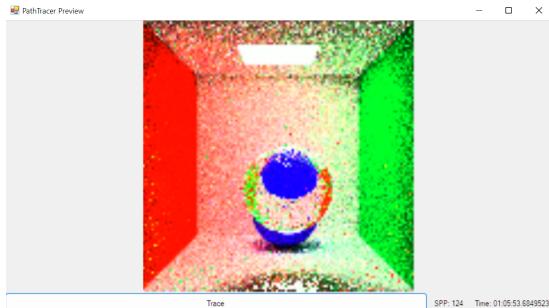


Figure 4: Scena po 124 ciklih

V naslednji sliki smo stekleno kroglo premaknili pod luč in ji dali večji refrakcijski indeks, pravtako pa smo povečali razlike refrakcijskega indeksa med frekvencami, da bi lažje opazili spektralne efekte, ki jih lahko tudi rahlo vidimo na spodnji strani krogle, kjer naj ne-bi bilo nič. Opazimo tudi, da je snop žarkov bolj izostren, kot je bil v prejšnjem primeru.

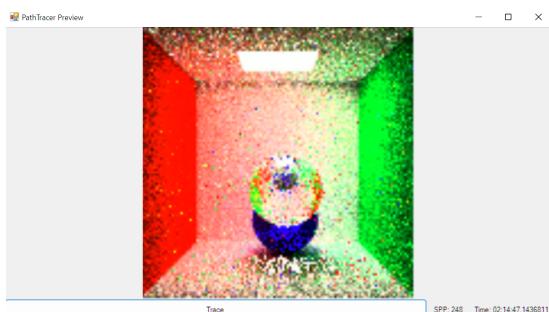


Figure 5: Scena po 248 ciklih

Na zadnje pa smo lučpomanjšali in kroglo premknili rahlo levo od luči, da bi opazili refrakcijsko mavrico; kar lahko opazimo, če podrobno pogledamo na levi strani krogle. Spektralni efekt ni tako močen, saj uporabljamo difuzno luč. Efekt bi bil še močnejši, če bi namesto krogle imeli prizmo. V tem primeru je program šel skozi 695 iteracij.

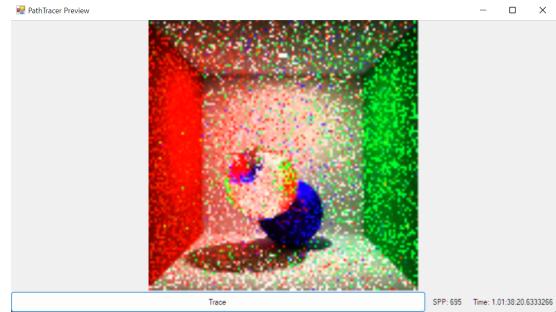


Figure 6: Scena po 695 ciklih

5. Zaključek

V okviru tega seminarja sem implementiral Spektralno upodabljanje v PathTracer okolju. Najbolj sem se zgledoval po knjigi Physically Based Rendering, ki že opisuje postopek implementacije spektralnega upodabljanja. Implementiral sem še refrakcijo glede na frekvence, ki zahteva več časa, da se obarva pravilno, kot navadna metoda, saj temelji na utezenem Monte Carlo pristopu. Implementiral sem tudi material steko, ki deluje kot pričakovano. Program, ki temelji na spektralnem upodabljanju je približno 20-krat počasnejši kot navadni program, saj so vektorji 20-krat daljši in posledično množenje njih teže. Rezultati so spodbudni. Vidimo lahko, da refrakcija deluje, kot bi pričakovali, saj se žarki posamezne frekvence lomijo pod drugačnim kotom. Končna slika je dober približek realnega življenja.

References

- [ARY*19] ABOUDEIF Y., REBEN M., YOUSEF E., AL-SALAMI A., SHEHRI A.: Thermal stability and optical properties of tellurite glasses doped with rare earth containing lithium chloride. *Journal of Nanoelectronics and Optoelectronics* 14 (04 2019), 448–455. doi:10.1166/jno.2019.2585.
- [Dav] DAVIDSON M. W.: Refractive index (index of refraction). URL: [https://www.microscopyu.com/microscopy-basics/refractive-index-index-of-refraction#:~:text=Refractive%20Index%20\(Index%20of%20Refraction\)%20is%20a%20value%20calculated%20from,descriptive%20text%20and%20mathematical%20equations.](https://www.microscopyu.com/microscopy-basics/refractive-index-index-of-refraction#:~:text=Refractive%20Index%20(Index%20of%20Refraction)%20is%20a%20value%20calculated%20from,descriptive%20text%20and%20mathematical%20equations.)
- [PH10] PHARR M., HUMPHREYS G.: *Physically Based Rendering, Second Edition: From Theory To Implementation*, 2nd ed. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2010.
- [wik22] Cie 1931 color space, Mar 2022. URL: https://en.wikipedia.org/wiki/CIE_1931_color_space.
- [WSS13] WYMAN C., SLOAN P.-P. J., SHIRLEY P.: Simple analytic approximations to the cie xyz color matching functions.
- [Xu] XU H.: Pathtracer part 2. URL: <https://henryzxu.github.io/pathtracing-p2/>.