Programovacia úloha č. 2

(6b)

Téma: 1D Z-buffer (chromatická úsečka)

Termín: uvedený na stránke.

Cieľ: Cieľom druhej programovacej úlohy je:

1. naprogramovať samostatnú aplikáciu, ktorá umožní používateľovi pridávať rôznofarebné úsečky a zároveň pridať špeciálnu „chromatickú" úsečku v príslušnom režime.
2. implementovať 1D prípad algoritmu *Z-buffer* riešiaci viditeľnosť v scéne a ofarbujúci chromatickú úsečku.

Zadanie: Implementujte aplikáciu vytvárajúcu a vizualizujúcu algoritmus *Z-buffer*, ktorá ofarbuje chromatickú úsečku. Konkrétne:

- **1b** Program umožňuje užívateľovi zadať farebné úsečky (zadávaním ich koncových bodov, klikaním ľavého tlačidla myši) a jednu špeciálnu chromatickú úsečku (zadávaním jej koncových bodov, klikaním pravého tlačidla myši).
- **5b** Po stlačení tlačidla sa chromatická úsečka ofarbí na základe kolmého premietania ofarbenej úsečky a princípu *Z-buffer*.

  Pomôcka: Najjednoduchšie riešenie je využiť afinnú transformáciu, ktorá presunie chromatickú úsečku do jednej zo súradnicových osí.

Chromatický režim: Chromatickú úsečku chápte ako 1 rozmernú obrazovku s určitým rozlíšením, ktoré je možné nastaviť v textovom okne zadaním počtu jej pixlov. Skutočná veľkosť chromatickej úsečky nie je dôležitá – ak má úsečka v textovom okne nastavených napr. 10 pixlov a úsečku natiahnete cez polovicu obrazovky, úsečka sa skladá z 10-tich segmentov, ktoré predstavujú 10 jej pixlov.

Po zadaní chromatickej úsečky sa do tejto vyrenderuje časť scény, ktorá sa nachádza na jej pravej strane, napravo od smeru definovaného spojnicou jej začiatočného a koncového bodu, čiže v kladnej polrovine definovanej danou chromatickou úsečkou.

Pomôcka: Uvažujme, že na monitore je bod $(0,0)^\top$ v ľavom dolnom rohu, x-ová os smeruje doprava, a y-ová nahor. Potom, keď A je začiatočný bod a B koncový bod chromatickej úsečky, tak $\mathbf{s}$ chápeme ako vektor $B - A$. Následne kolmý vektor na chromatickú úsečku je vektor $(\mathbf{s}.y, -\mathbf{s}.x)^\top$. Normalizáciou tohto vektora získame normálu polroviny, ktorá udáva kladný smer.

Renderovanie scény do chromatickej úsečky naprogramujte pomocou metódy *Ray-casting*, pričom pole jej virtuálnych pixlov (segmentov) chápte ako *frame-buffer*.

Pomôcka: Na vyriešenie problému viditeľnosti v scéne použite algoritmus *Z-buffer*. Zo stredu každého virtuálneho pixla (segmentu) zadanej chromatickej úsečky, vystreľte polpriamku, kolmú na chromatickú úsečku, smerujúcu do scény. Každý prienik vystrelenej polpriamky s ľubovoľnou úsečkou v scéne je potenciálne viditeľný, preto použite *Z-buffer*. Vo výsledku si vo *frame-bufferi* pamätajte farbu úsečky zo scény, s ktorou mala vystrelená polpriamka prvý prienik. Keď budete mať takto vypočítané farby pre celý *frame-buffer*, vykreslite chromatickú úsečku, a to tak, že jednotlivé jej segmenty vykreslite príslušnými farbami z *frame-buffera*.

Po vykreslení chromatickej úsečky môže užívateľ zadať novú.

Výstup: Okrem používateľského prostredia, pripojte aj kód, ktorý implementuje vaše riešenie. Táto časť musí byť ľahko identifikovateľná a vytvorená autorom, t.j. vytvorená výhradne pre účely tejto úlohy. Použitie výlučne externých knižníc je prísne zakázané!

Požiadavky: Aplikácia musí spĺňať požiadavky uvedené na adrese: `https://mkvnk.sk/PG1/#submit`.

Okrem toho sa vyžaduje:

- kód musí byť **dostatočne** komentovaný a **prehľadne** formátovaný. Nedostatočné komentáre a neprehľadné formátovanie môže byť penalizované stratou až *3 bodov*.
- neintuitívne používateľské prvky či časti prostredia (ak nejaké sú) musia byť opísané v osobitnom `readme.txt` súbore.

Vzorová aplikácia je dostupná na webstránke, spolu s týmto .pdf súborom.