

Antialiasing

Róbert Bohdal

`robert.bohdal@fmph.uniba.sk`

`https://flurry.dg.fmph.uniba.sk/webog/bohdal-vyucba`

Katedra algebry a geometrie
Fakulta matematiky fyziky a informatiky
Univerzita Komenského v Bratislave

Počítačová grafika (1)
prednáška č. 10



- 1 Vznik aliasu
- 2 Odstraňovanie aliasu – antialiasing
- 3 Odstraňovanie nepravého aliasu pre úsečky a krivky
- 4 Antialiasing (vyhladzovanie) v počítačových hrách

Základné pojmy

- Spojitý obraz v PG je definovaný spojitou funkciou $f(x, y)$ na definičnom obore $\langle x_{min}, y_{min} \rangle \times \langle x_{max}, y_{max} \rangle$ s oborom hodnôt v $\mathbb{R}^3$ (R,G,B)
- Pre zobrazenie, zaznamenanie či spracovanie obrazu je nutné spojitú funkciu diskretizovať v definičnom obore aj v obore hodnôt
- Diskretizácia v definičnom obore sa nazýva **vzorkovanie** (*sampling*)
- Diskretizácia v obore hodnôt sa nazýva **kvantovanie**
- Počet vzoriek udáva **rozlíšenie** a počet možných farieb udáva **farebnú hĺbku**
- Pixel s konkrétnou farbou je reprezentant jednej vybranej vzorky spojitého obrazu
- **Rekonštrukciou** obrazu/signálu rozumieme (znovu)vytvorenie spojitej funkcie, napr. pomocou interpolácie, zo vzoriek, ktoré vznikli diskretizáciou pôvodnej funkcie



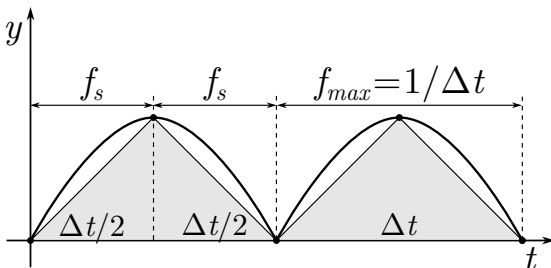
Vzorkovanie

- Vzorkovanie v 1D s rovnomerným krokom (intervalom) o dĺžke Δx so začiatkom v bode x_0 : $I_i = f(x_0 + \Delta x \cdot i)$, pre $i = 0, 1, \dots, n$
- Vzorkovanie v 2D: $I_{ij} = f(x_0 + \Delta x \cdot i, y_0 + \Delta y \cdot j)$, pre $i = 0, 1, \dots, n$ a $j = 0, 1, \dots, m$
- Vzorkovacia frekvencia v 1D je určená vzťahom $f_s = 1/\Delta x$
- V spojitom obraze je frekvencia (opakovanie určitého vzoru/signálu) neohraničená ($\Delta x \rightarrow 0$), avšak v diskretnom obraze nekonečná nemôže byť, preto $f_s \ll \infty$
- Pri vzorkovaní dochádza k strate určitej informácie a pri veľkom kroku aj k strate „detailov“
- Chyba pri vzorkovaní obrazu s nedostatočnou frekvenciou (veľkosťou kroku Δx a Δy) sa prejavuje tzv. **aliasom**, ktorý je sprevádzaný artefaktmi
- Chyba pri kvantovaní sa sa prejavuje tzv. Machovým efektom, pri ktorom človek vníma hrany na neprirodzenom (skokovom) prechode medzi jednotlivými farbami pixlov

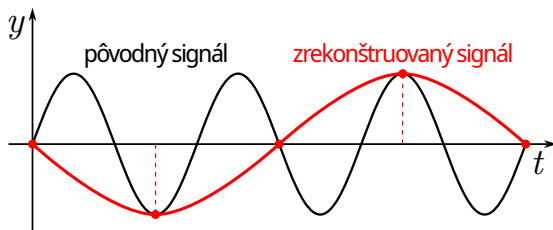


Vzorkovanie – Nyquistov limit, alias

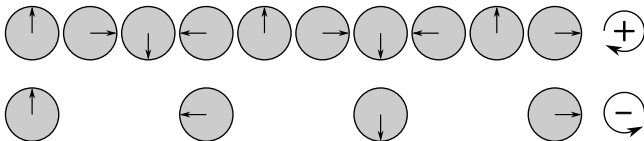
- Ak nechceme stratiť informáciu o „detailoch“ signálu, musíme dodržať **Nyquistov limit**, čo znamená, že pre vzorkovaciu frekvenciu musí platiť $f_s > 2f_{max}$, kde f_{max} je hodnota najvyššej frekvencie v signáli
- Alias vzniká v zrekonštruovanom signáli (obraz), ak bol vzorkovaný pod hodnotou Nyquistovho limitu, t.j. ak $f_s < 2f_{max}$
- Alias typicky vzniká pri zmenšovaní a opätovnom zväčšení obrazu bez použitia filtrovania. Napr. pri zmenšení textúry šachovnice sa stratia detaily (niektoré štvorčeky zmiznú), alebo sa posunú niektoré hrany



Vzorkovanie – alias

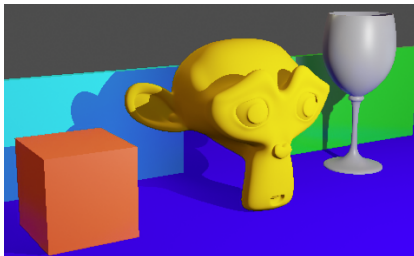


- Pri snímaní obrazu kamerou vzniká aj *temporálny* alias, ktorý sa prejavuje napr. zdanlivo opačným otáčaním kolies auta
- Aliasom (nepravým) nazývame aj zubatosť úsečiek, ktorá vzniká pri ich zobrazovaní v rastri. Nepravým ho nazývame preto, že nemá súvis so vzorkovacou frekvenciou a stratou detailov

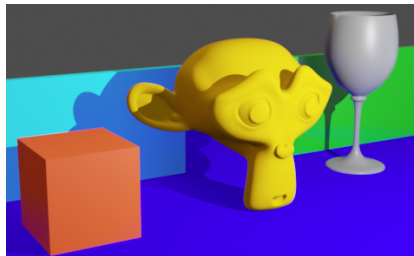


Odstraňovanie aliasu

- Pod antialiasingom rozumieme odstraňovanie aliasu
- Často sa alias neodstráni, len sa zmenší jeho viditeľnosť
- Metódy odstraňovania aliasu:
 - filtrovanie pred vzorkovaním (*prefiltering*),
 - vzorkovanie plochy (pixelu) (*area sampling*),
 - nadvzorkovanie (*supersampling*),
 - filtrovanie po vzorkovaní (*postfiltering*),



Obrázok s viditeľným aliasom



Odstránenie aliasu nadvzorkovaním



Filtrovanie pred vzorkovaním

- Filtrovanie pred vzorkovaním najčastejšie používa filter rozostrenia (*blur filter*), čím sa z obrazu odstráni vysoké frekvencie (details)
- Nevýhodou tejto metódy je redukcia detailov, ktorá spôsobuje vizuálne zníženie rozlíšenia obrazu



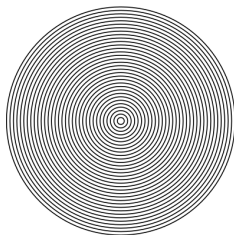
Originálny obrázok



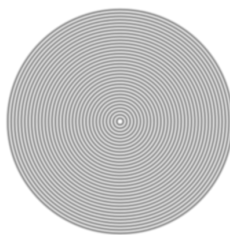
Preškáľovaný obrázok bez filtru



Preškáľovaný obrázok s filtrom



Originálny obrázok

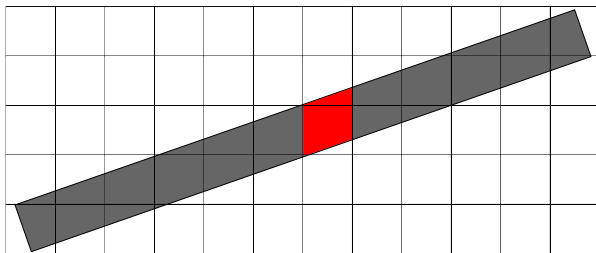


Filtrovaný obrázok



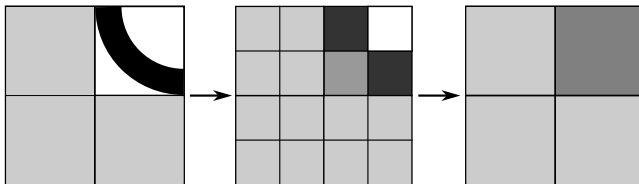
Vzorkovanie plochy

- Vzorkovanie plochy (pixlu) využíva to, že pixel má štvorcový tvar, pričom niektorá jeho časť alebo aj celý pixel je pokrytý časťou zobrazovaného objektu
- Intenzita pixlu sa vypočíta podľa plochy, ktorú zobrazovaný objekt v pixeli zaberá, t.j. ak objekt zaberá $1/4$ plochy, tak pixel bude mať $1/4$ intenzity
- Výsledná farba pixlu je súčtom príspevkov všetkých tých častí objektov, ktoré majú s daným pixlom prienik a sú v pixeli viditeľné
- V praktickej realizácii sa využíva akumulčný a hĺbkový (*depth*) buffer
- Metóda je výpočtovo náročná, je nutné počítat preniky objektov s pixelami
- Pri využití aproximácie prienikov sa použijú preddefinované útvary v podobe trojuholníkov a lichobežníkov s konkrétnymi veľkosťami strán



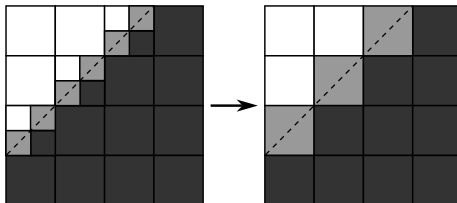
Nadvzorkovanie

- Pri mriežkovom uniformovanom nadvzorkovaní sa každý pixel rozdelí na n^2 ($n = 2, 3, \dots$) subpixelov, čím sa značne zvýši pamäťová aj výpočtová náročnosť, keďže sa rasterizácia vykonáva vo väčšom (jemnejšom) rozlíšení
- Okrem štandardného mriežkového usporiadania subpixelov sa využíva aj usporiadanie vedľa seba pre 2x (vodorovne, zvislo, diagonálne), do šachovnice či n -veží pre 4x, 8x a 16x
- Pri neuniformovanom nadvzorkovaní sa pixle rozdeľujú iba tam, kde susedné pixle majú odlišnú intenzitu (farbu), napr. na hranách objektov
- Keďže sa v oboch typoch nadvzorkovania vytvára obraz vo vyššom rozlíšení, treba ho prepočítať späť na pôvodné rozlíšenie



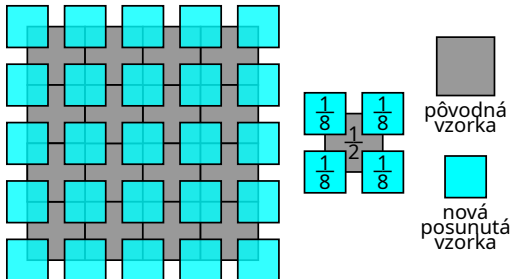
Nadvzorkovanie

- Pri prepočítaní výslednej farby pixla v nižšom rozlíšení sa používa aritmetický alebo vážený priemer zodpovedajúcich subpixlov
- Pri počítaní váženého priemeru sa používa štandardný rozostrojúci filter (napr. Gaussov), keďže však malé objekty môžu zaniknúť, používajú sa aj iné filtre
- Nadvzorkovanie obvyčajne alias neodstráni, ale ho iba posúva k vyšším frekvenciám, ktoré si ľudský mozog už nevšimne



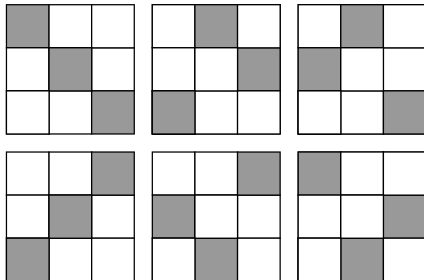
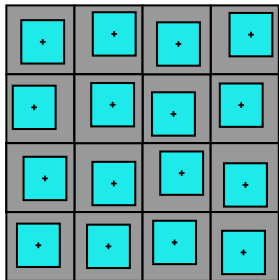
Nadvzorkovanie metódou quincunx

- Používa 5 „subpixlov“ na 1 pixel, pričom stredový subpixel má váhu $1/2$ a rohové subpixle majú váhu $1/8$
- Rohové subpixle zdieľa so susednými pixlami, preto sa počítajú iba dva obrazy
- Vizuálne sa tento typ AA podobá náročnejšiemu 4x nadvzorkovaniu
- Scéna sa vykreslí (renderuje) do dvoch rastrov v štandardnom rozlíšení, pričom pri vykresľovaní do druhého rastra sa posunie kamera v priestore obrazovky o vektor $(1/2, 1/2)$. Použitím váženého priemeru sa potom z dvoch rastrov (bitmáp) vypočíta výsledný obraz



Stochastické nadvzorkovanie

- Náhodným posunutím sa vysoké frekvencie zmenia na šum, ktorý ak je dostatočne malý, tak ho ľudský mozog nevníma, keďže svetlocitlivé bunky sú usporiadané ako vzorkovanie Poissonovho disku
- Metóda roztrásením (*jittering*) využíva uniformované rozdelenie pixlov na subpixle avšak stredy subpixlov sú posunuté o náhodný vektor s dĺžkou maximálne $1/2$ veľkosti subpixla
- Metóda n -veží (n -rooks) vyberá náhodné rozloženie subpixlov tak, aby sa subpixle v pixeli „nehrozovali“, podobne ako veže na šachovnici



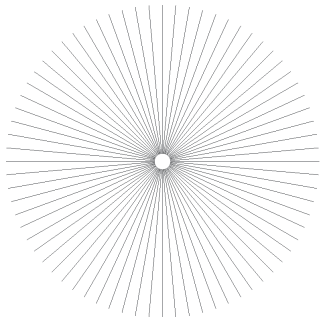
Filtrovanie po vzorkovaní

- Využívajú rôzne, najmä rozostrojúce filtre, ktoré sa použijú, až keď je obraz vyrenderovaný
- Cieľom filtrovania je „vyhladiť zubaté okraje“ hrán zobrazovaných objektov
- Ako filter môže byť použitý filter s aritmetickým priemerom (box) alebo s váženým priemerom (kubický, kvadratický, Gaussov), či iné druhy filtrov (Catmull-Rom, Mitchell-Netravali, atď.), ktoré obsahujú aj záporné hodnoty váh, spôsobujúce istú formu zaostrenia niektorých pixlov obrazu, aby sa predišlo prílišnému rozostrovaniu hrán, či malý objektov

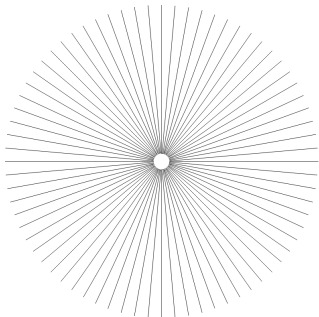


Odstraňovanie nepravého aliasu pre úsečky a krivky

- Pri klasickom vykresľovaní kriviek do rastra vzniká známa zubatosť a moiré
- Môže sa použiť nadvzorkovanie, pričom sa do jemnejšieho rastra vykresľuje n násobne hrubšia čiara. Výsledná intenzita sa potom vypočíta podľa počtu vykreslených subpixlov
- Pri jednofarebnom pozadí je rýchlejšie a pamäťovo výhodnejšie použiť napr. modifikáciu Bresenhamovho algoritmu, pri ktorej sa počítajú aj relevantné susedné pixle okrem štandardne vykresľovaných pixlov



Úsečky s viditeľným aliasom



Úsečky vykreslené s antialiasingom



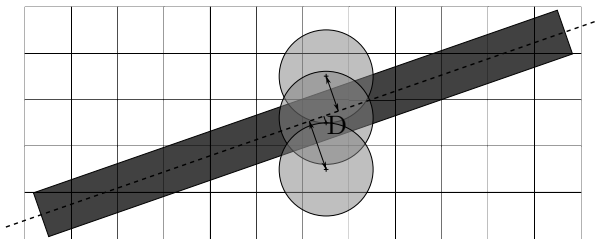
Metódy vzorkovania plochy

- Nevážené vzorkovanie plochy využíva namiesto úsečky obdĺžnik s hrúbkou 1 pixlu a výsledná intenzita farby vykresľovaného pixlu je určená plochou časti obdĺžnika, ktorá pokrýva daný pixel
- Vážené vzorkovanie plochy dávajú väčšiu váhu oblastiam, ktoré sú bližšie k úsečke. Využíva nejakú váhovú funkciu (napr. vzdialenosť k úsečke) alebo nejaký filter (napr. kužeľový)



Gupta-Sproull metóda pre kreslenie úsečky

- Gupta a Sproull použili vyhladzovací filter s kužeľovou funkciou rozloženia bodov o polomere jedného pixlu, pričom jeho konvolúcia s priamkou závisí iba od kolmej vzdialenosti kužeľa k priamke
- Intenzita zobrazovaného pixlu je úmerná objemu časti kužeľa, ktorý prekrýva rastrovanú úsečku
- Tento objem je vypočítaný pomocou vzdialenosti D stredu pixlu ku priamke reprezentujúcej danú úsečku
- Vzdialenosť D sa kvôli efektívnosti počíta inkrementálne
- Pre vykreslenie bodu úsečky používa 3 pixle, pre hrubšie úsečky viac pixlov



Chenova varianta metódy Gupta-Sproull pre 1. oktant

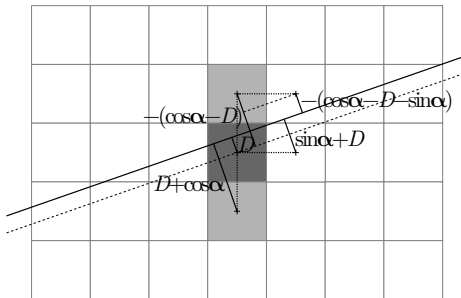
$$\cos \alpha = \frac{\Delta x}{\sqrt{\Delta x^2 + \Delta y^2}}, \sin \alpha = \frac{\Delta y}{\sqrt{\Delta x^2 + \Delta y^2}}$$

$$D_{curr}(y-1) = D + \cos \alpha$$

$$D_{curr}(y+1) = -(\cos \alpha - D) = D - \cos \alpha$$

$$D_{next}(y) = D + \sin \alpha$$

$$D_{next}(y+1) = -(\cos \alpha - D - \sin \alpha) = D + \sin \alpha - \cos \alpha$$



Chenova varianta algoritmu Gupta-Sproull pre 1. oktant

```

procedure draw_line(ax, ay, bx, by: integer);
x, y, deltax, deltay, d: integer;
D, length, sin, cos: real;
begin
  x := xa; y := ya;
  deltax := xb - xa;
  deltay := yb - ya;
  d := 2 * deltay - deltax;
  D := 0;
  length := sqrt(deltax * deltax + deltay * deltay);
  sin := deltay / length;
  cos := deltax / length;
  while x <= xb do begin
    drawpixeli(x, y - 1, D + cos);
    drawpixeli(x, y, D);
    drawpixeli(x, y + 1, D - cos);
    x := x + 1;
    if d <= 0 then begin
      D := D + sin;
      d := d + 2 * deltay;
    end
    else begin
      D := D + sin - cos;
      d := d + 2 * (deltay - deltax);
      y := y + 1;
    end
  end
end
end

```



Wuova metóda pre kreslenie úsečky

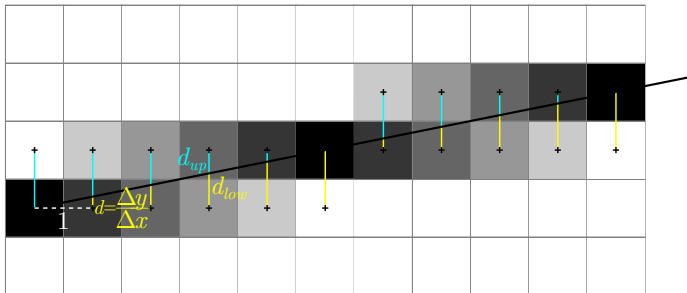
- Jednoduchšia a efektívnejšia metóda ako Gupta-Sproull
- Založená na úprave Bresenhamovho alebo DDA algoritmu kreslenia úsečky
- Vykresľuje dva pixle, jeden pod a druhý nad „geometrickým“ obrazom úsečky
- Intenzita pixlu je daná jeho y-ovou vzdialenosťou od úsečky

$$y = mx + b = \frac{\Delta y}{\Delta x}x + b$$

$$d_{next}(y) = d + \frac{\Delta y}{\Delta x}$$

$$d_{next}(y + 1) = d - 1$$

$$d_{low} = d, d_{up} = 1 - d_{low}$$



Varianta Wuovho algoritmu pre úsečku v 1. oktante

```

procedure draw_line(ax, ay, bx, by: integer);
x, y, deltax, deltay, dy: integer;
low: real;
begin
  x := xa; y := ya;
  deltax := xb - xa;
  deltay := yb - ya;
  dy := 0;
  while x <= xb do begin
    low := dy / deltax;
    drawpixeli(x, y, low);
    drawpixeli(x, y + 1, 1 - low);
    dy := dy + deltay;
    if dy > deltax then begin
      dy := dy - deltax;
      y := y + 1;
    end
    x := x + 1;
  end
end
end

```



Antialiasing (vyhladzovanie) v počítačových hrách

- Existuje viacero rôznych techník vyhladzovania, avšak využívajú sa najmä dva prístupy, včítane ich kombinovania a rôznych modifikácií:
 - nadvzorkovanie – počítajú obraz vo vyššom rozlíšení a potom ho prepočítajú na pôvodné rozlíšenie s využitím priemeru
 - filtrovanie – na vypočítaný obraz sa aplikuje vhodný filter, ktorý mierne rozostruje okolie hrán
- Vyhladzovanie sa vykonáva na celom zobrazovanom obraze, t.j. antialiasing je celoobrazovkový (FSAA)



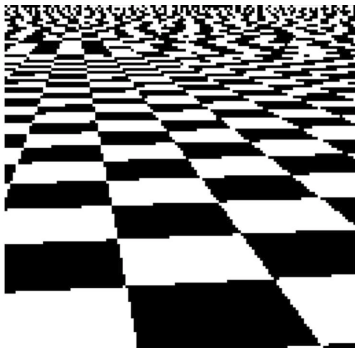
Typy antialiasingu v počítačových hrách

- Supersample Anti-aliasing (SSAA)
- Multisample Anti-aliasing (MSAA)
- Fast Approximate Anti-aliasing (FXAA)
- Temporal Anti-aliasing (TAA)
- Enhanced Subpixel Morphological Anti-aliasing (SMAA)
- Deep Learning Super-Sampling (DLSS)

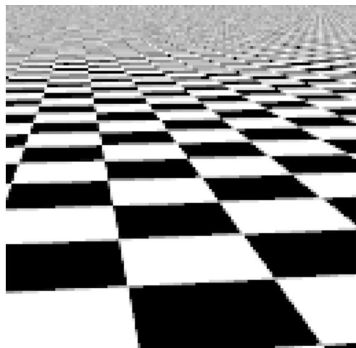


Supersample (nadvzorkovací) antialiasing

- Základna forma antialiasingu, pri ktorej sa renderuje obraz s vyšším rozlíšením
- Napr. pri SSAA 4x (2×2) sa na jeden pixel použijú 4 subpixle, vypočíta sa obraz pre jemnejšie rozlíšenie a potom sa váženým priemerom vypočíta zo 4 pixlov hodnota výsledného zobrazovaného pixlu
- SSAA dáva najlepšie výsledky vyhladzovania, ale je to výpočtovo najnáročnejšia metóda, najmä pri veľkom počte subpixlov



single sampling



32x32 supersampling



Multisample (viacvzorkový) antialiasing

- Nadvzorkujú sa iba tie časti obrazu, kde je alias viditeľný, t.j. na hranách (hraniciach objektov)
- Využíva sa informácia (v shaderi), či susedné pixle zobrazujú ten istý objekt (grafický primitív)
- Rasterizácia objektov sa robí v štandardnom rozlíšení, avšak pre z-buffer sa používa vyššie rozlíšenie
- Môže sa využiť aj *stencil* buffer, v ktorom sa uchováva informácia o okraji (hranách) zobrazovaného objektu
- Prehľadávaním pixlov v z-buffri a *stencil* buffri sa získa informácia, či výsledný zobrazovaný pixel je na hranici objektu a treba prepočítať hranu vo vyššom rozlíšení
- Táto metóda je výpočtovo menej náročná ako SSAA



Fast approximate (aproximačný) antialiasing

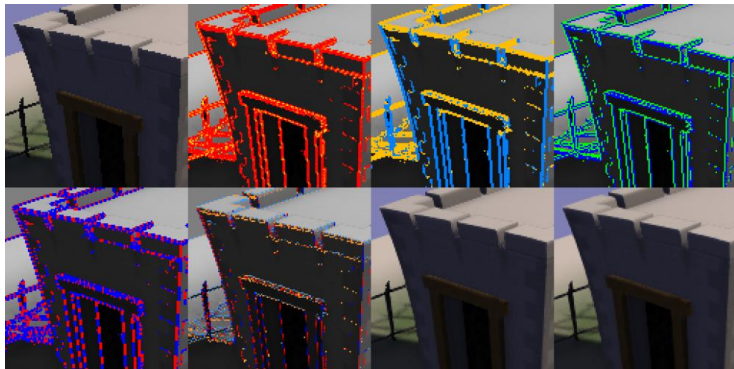
- Bol vyvinutý firmou Nvidia a predstavuje typické filtrovanie obrazu, ktoré je vykonávané ako postproces, až po výpočte pixlov obrazu
- Pri detekcii hrán porovnáva intenzitu susedných pixlov
- Tento druh AA je oveľa menej náročný ako SSAA a MSAA, avšak na úkor kvality obrazu

Algoritmus pozostáva z nasledujúcich krokov:

1. RGB hodnoty pixlov obrazu sa skonvertujú na hodnoty intenzít
2. Lokálnym kontrastným testom sa klasifikujú hrany ako horizontálne či vertikálne
3. Nájdené hrany sa rozdelia na vodorovné resp. zvislé kúsky (schodíky v dominantnom smere)
4. Tieto kúsky sa potom posunú v kolmom smere o subpixelovú vzdialenosť z oboch strán hrany
5. Na základe vytvorenej subpixelovej mriežky sa prevzorkuje vstupný obrázok iba tam kde je to treba, t.j. na nájdených hranách
6. Nakoniec sa na miesta, kde sa použilo prevzorkovanie použije dolnopriepustný filter s hodnotami určenými prevzorkovanými subpixelmi



Fast approximate (aproximačný) antialiasing



Temporal (časový) antialiasing

- Ide tiež o vyhladzovanie v rámci postprocesu
- Využíva fakt, že sa veľmi často farby objektov medzi susednými snímkami (frames) veľmi nemenia
- Ak scéna nie je statická, a kamera alebo objekty sa v scéne hýbu, tak susedné snímky predstavujú ten istý obraz/objekt, ktorý je mierne posunutý
- Používa predchádzajúce, nasledujúce a aktuálnu snímku na spriemerovanie hodnoty každého pixlu, pričom sa využíva vhodný filter s váženým priemerom
- Môže vytvárať tzv. duchov na hranách objektov

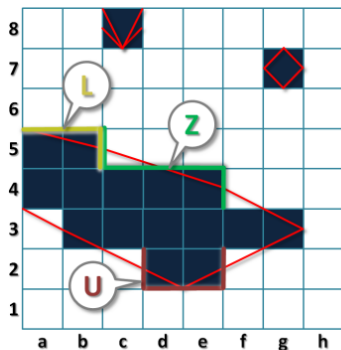


Enhanced subpixel morphological antialiasing

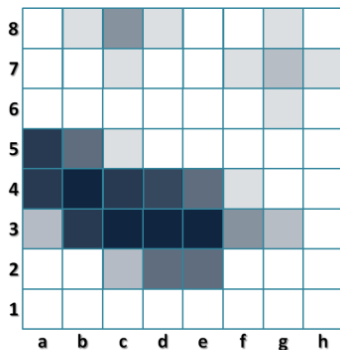
- Predstavuje vhodnú kombináciu viacerých metód
- V niečom je podobný FXAA ale využíva robustnejšiu klasifikáciu zvislých, vodorovných a navyše aj šikmých hrán
- Rozširuje štandardný morfológický antialiasing (MLAA), ktorý vo výslednom obraze hľadá hranice objektov v ich pôvodnej (nerasterizovanej) forme. MLAA v nich potom nachádza špecifické vzory (tvar), ktoré využije pri vytvorení vhodného váženého priemeru okolitých pixlov
- Oproti FXAA, nadvzorkovanie metódou MSAA kombinuje s TSAA, aby sa obmedzilo „blikanie“ pixlov na hranách zobrazovaných objektov počas animácie



Morphological antialiasing



Z-shapes: 
U-shapes: 
L-shapes: 



Z and U shape decomposition into L-shapes:

$$\begin{array}{c} \text{Z-shape} \end{array} = \begin{array}{c} \text{L-shape} \end{array} + \begin{array}{c} \text{L-shape} \end{array} \quad \begin{array}{c} \text{U-shape} \end{array} = \begin{array}{c} \text{L-shape} \end{array} + \begin{array}{c} \text{L-shape} \end{array}$$

Deep Learning supersampling

- Metóda bola vyvinutá spoločnosťou Nvidia pre jej grafické karty radu RTX
- Využíva neurónovú sieť pre zobrazenie vylepšeného obrazu alebo obrazu vo vyššom rozlíšení prostredníctvom rekonštrukcie obrazu
- Funguje na podobnom princípe ako TAA, avšak netestuje každý pixel v každej snímke. Namiesto toho vzorkuje rôzne pixle v rôznych snímkach a používa pixle navzorkované v minulých snímkach na vyplnenie nevzorkovaných pixlov v aktuálnej snímke
- V novšej verzii DLSS 3.0 sa pridalo vkladanie novej snímky medzi dve nasledujúce snímky, aby sa medzi nimi vytvoril plynulejší prechod. Využíva sa pri tom tzv. „Optical Flow Accelerator“
- Vo verzii DLSS 3.5 bola pridaná metóda rekonštrukcie lúčov založená na AI, ktorá sa používa na vylepšenie „zašumených“ renderovaných obrázkov, ktoré často vznikajú po použití ray tracingu s nedostatočným počtom lúčov na pixel

