Bevezetés a számítógépi grafikába

Élsimító algoritmusok

Troll Ede Mátyás

Matematikai és Informatikai Intézet
Eszterházy Károly Katolikus Egyetem

Eger, 2024

Áttekintés

Élsimítás

A korábban tanult raszteres algoritmusok még nagyítás nélkül is rendelkeznek bizonyos lépcsőhatással. Az élsimító algoritmusok feladata a megjelenített képen a lépcsőhatás csökkentése.

Élsimítás

A korábban tanult raszteres algoritmusok még nagyítás nélkül is rendelkeznek bizonyos lépcsőhatással. Az élsimító algoritmusok feladata a megjelenített képen a lépcsőhatás csökkentése. Ennek kezelésére alapvetően két irányról beszélhetünk

- A raszteres algoritmus módosítása úgy, hogy az eredmény simábbnak tűnjön

Élsimítás

A korábban tanult raszteres algoritmusok még nagyítás nélkül is rendelkeznek bizonyos lépcsőhatással. Az élsimító algoritmusok feladata a megjelenített képen a lépcsőhatás csökkentése. Ennek kezelésére alapvetően két irányról beszélhetünk

- A raszteres algoritmus módosítása úgy, hogy az eredmény simábbnak tűnjön
- Az elkészült kép (bit térkép) rasztereinek színinformációinak felhasználásával a teljes kép módosítása

Áttekintés

Szakasz élsimítása

Az eljárás lényege, hogy a megtalált pontokon túl (korábban megismert vonalrajzoló algoritmus) további pixeleket is megvilágítunk meghatározott intenzitással

Szakasz élsimítása

Az eljárás lényege, hogy a megtalált pontokon túl (korábban megismert vonalrajzoló algoritmus) további pixeleket is megvilágítunk meghatározott intenzitással

- Az elméleti szakaszra egy 1 pixel széles téglalapot helyezünk

Szakasz élsimítása

Az eljárás lényege, hogy a megtalált pontokon túl (korábban megismert vonalrajzoló algoritmus) további pixeleket is megvilágítunk meghatározott intenzitással

- Az elméleti szakaszra egy 1 pixel széles téglalapot helyezünk
- Ha egy pixelnek van a téglalappal közös része, akkor azt a megfelelő színintenzitással megfestjük

Szakasz élsimítása (súlyozatlan antialiasing)

Az eljárás lényege, hogy a megtalált pontokon túl (korábban megismert vonalrajzoló algoritmus) további pixeleket is megvilágítunk meghatározott intenzitással

- Az elméleti szakaszra egy 1 pixel széles téglalapot helyezünk
- Ha egy pixelnek van a téglalappal közös része, akkor azt a megfelelő színintenzitással megfestjük (A pixel intenzitását az határozza meg, hogy annak hány százalékát fedi a téglalap)

Szakasz élsimítása (súlyozott antialiasing)

Az eljárás lényege, hogy a megtalált pontokon túl (korábban megismert vonalrajzoló algoritmus) további pixeleket is megvilágítunk meghatározott intenzitással

- Az elméleti szakaszra egy 1 pixel széles téglalapot helyezünk
- Ha egy pixelnek van a téglalappal közös része, akkor azt a megfelelő színintenzitással megfestjük

Szakasz élsimítása (súlyozott antialiasing)

Az eljárás lényege, hogy a megtalált pontokon túl (korábban megismert vonalrajzoló algoritmus) további pixeleket is megvilágítunk meghatározott intenzitással

- Az elméleti szakaszra egy 1 pixel széles téglalapot helyezünk
- Ha egy pixelnek van a téglalappal közös része, akkor azt a megfelelő színintenzitással megfestjük
  - A pixeleket kör alakúnak feltételezzük
  - Egységnyi kúpot helyezünk mindegyikre
  - Az elméleti szakaszra pedig egységnyi magas hasábot
  - A színintenzitást az határozza meg, hogy a hasáb mekkora részben metszi az egyes kúpokat
  - Ebben az esetben nem csak a lefedettség mértéke, hanem az elméleti egyenestől való távolság is bele van kalkulálva a színintenzitásba

Xiaolin Wu szakaszrajzoló algoritmus

Xiaolin Wu 1991-ben publikálta algoritmusát, mely a Bresenham által megalkotott MidPoint algoritmus lépcsőhatását hivatott javítani.

Az algoritmus ugyan lassabb, mint elődje, de az eredmény lényegesen simább érzetet kelt.

Xiaolin Wu szakaszrajzoló algoritmus

Tegyük fel, hogy meg szeretnénk rajzolni egy szakaszt az $(1, 1)$ és $(8, 4)$ pontok között.

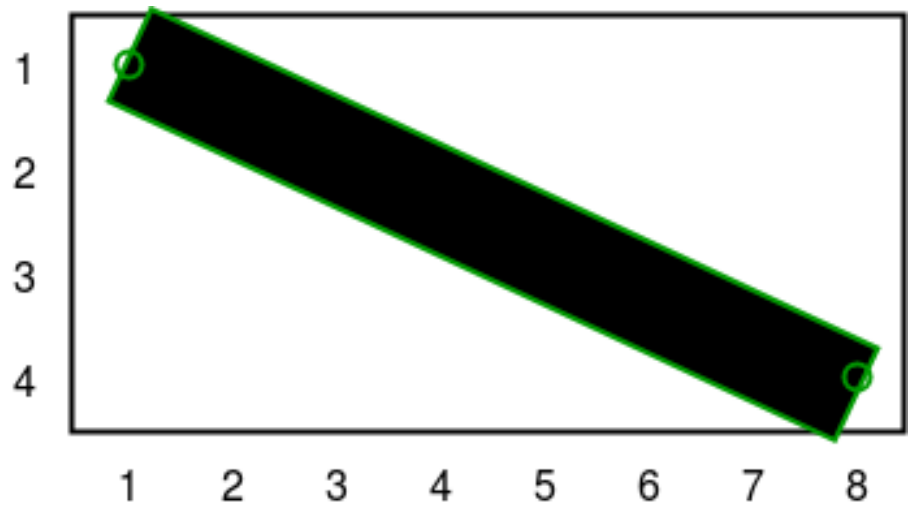


Fig. A: unit width line

Xiaolin Wu szakaszrajzoló algoritmus

Tegyük fel, hogy meg szeretnénk rajzolni egy szakaszt az $(1, 1)$ és $(8, 4)$ pontok között.

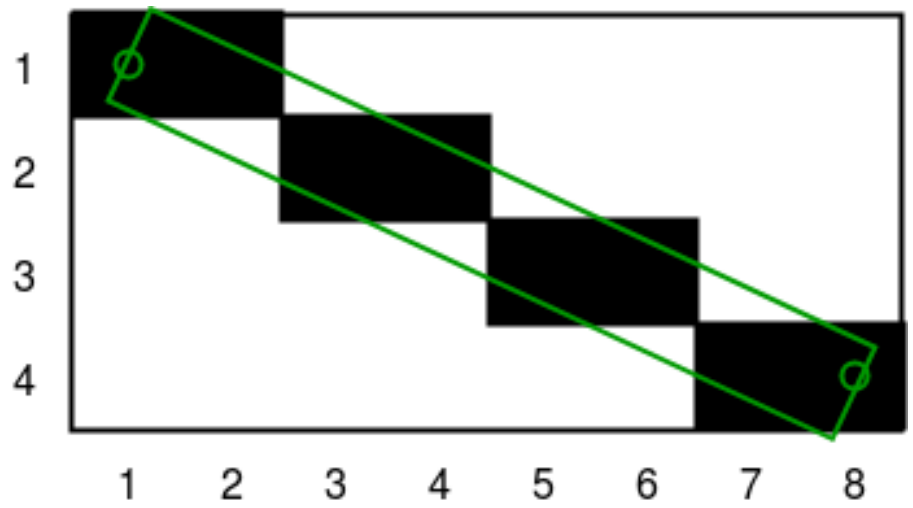


Fig. B: y=f(x) approximation

Mivel képernyőn szeretném megjeleníteni, ezért az elméleti egyenes helyett bizonyos rasztereket kell kiválasztanom, melyeket megvilágítok.

Xiaolin Wu szakaszrajzoló algoritmus

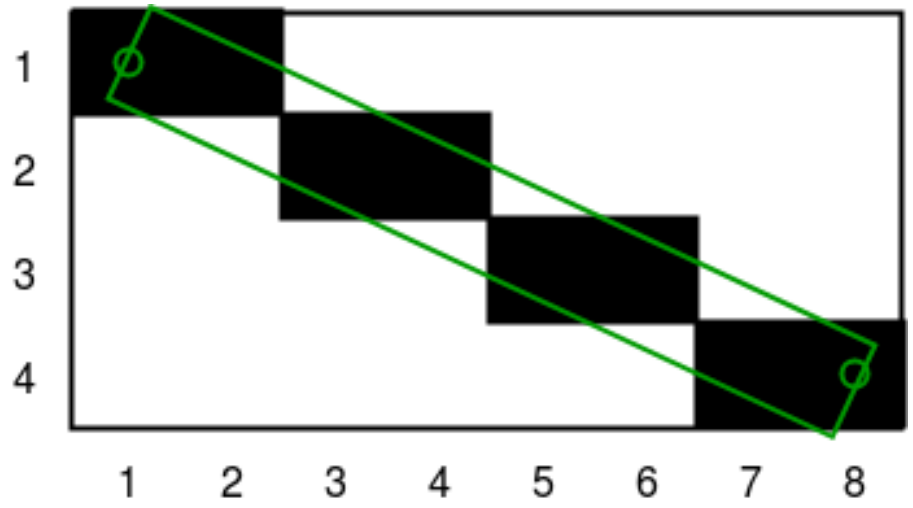


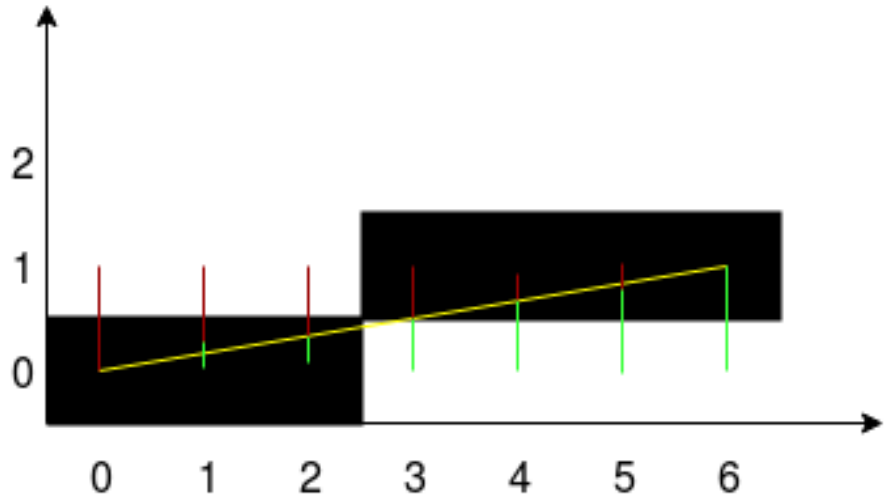
Fig. B: y=f(x) approximation

Felmerül néhány probléma

- A $(3, 2)$ nagyobb lefedettséggel rendelkezik, mint a $(4, 2)$, mégis mind a kettő fekete színnel jelenik meg
- A $(2, 2)$ lefedettsége nem sokban tér el a $(4, 2)$-től, mégis fehérrel jelenik meg

Xiaolin Wu szakaszrajzoló algoritmus

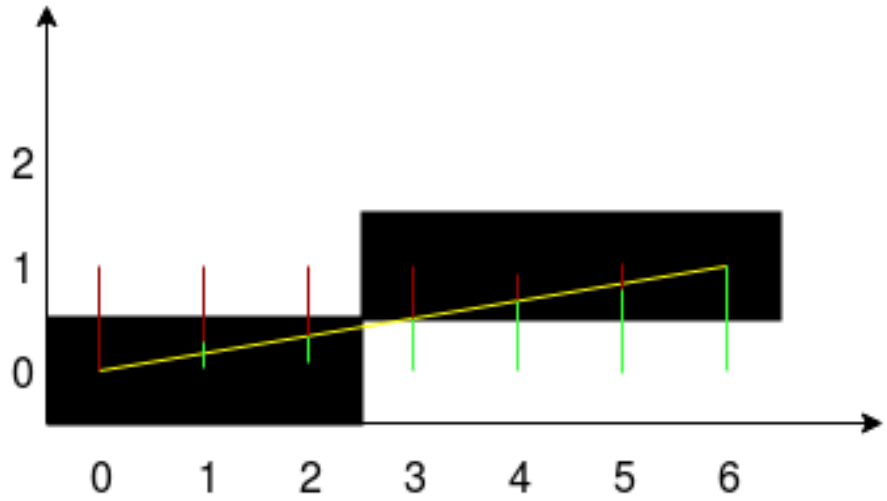
Az alábbi kép a Bresenham algoritmus alapján készült



Vízszintesen lépünk az x változóval.

Xiaolin Wu szakaszrajzoló algoritmus

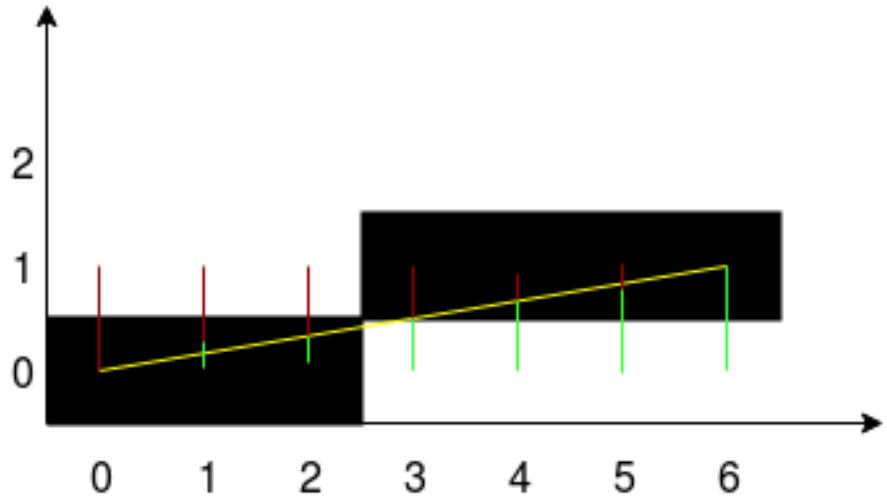
Az alábbi kép a Bresenham algoritmus alapján készült



Vízszintesen lépünk az x változóval.Minden lépésben kiszámoljuk a tényleges y koordináta és a legközelebbi rácspont közötti távolságot (hiba).

Xiaolin Wu szakaszrajzoló algoritmus

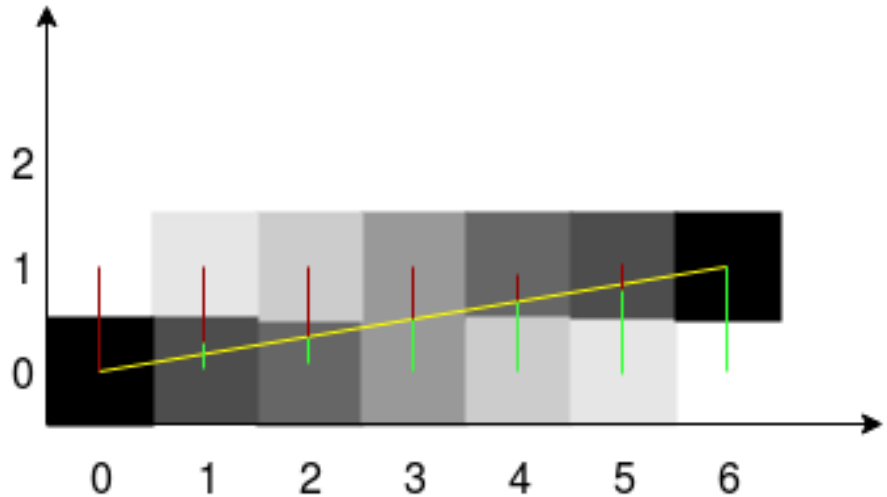
Az alábbi kép a Bresenham algoritmus alapján készült



Vízszintesen lépünk az x változóval.Minden lépésben kiszámoljuk a tényleges y koordináta és a legközelebbi rácspont közötti távolságot (hiba).Ha a fenti hiba nem haladja meg a rács magasságának felét, akkor kigyújtjuk.

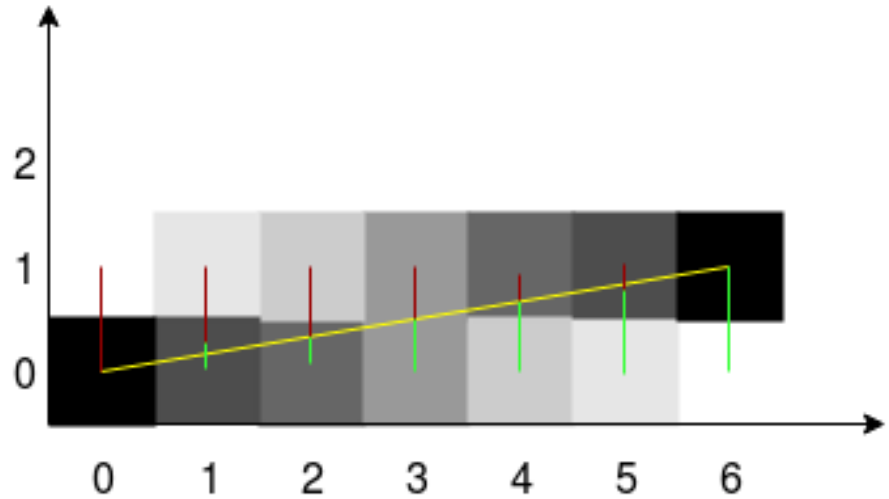
Xiaolin Wu szakaszrajzoló algoritmus

Az algoritmust úgy módosítjuk, hogy a szakasz éle el legyen simítva



Xiaolin Wu szakaszrajzoló algoritmus

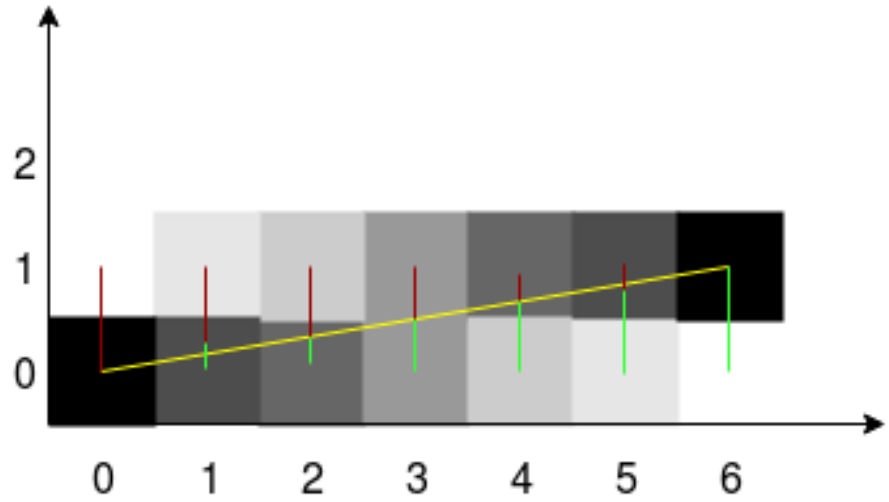
Az algoritmust úgy módosítjuk, hogy a szakasz éle el legyen simítva



- Az x irányú lépések esetén minden a két legközelebb eső pixelt kiválasztjuk
- Ezeket a távolságtól függő színintenzitással gyújtjuk ki arányosan
  - Tehát a pixel teljes színezettségű, ha a távolság 0

Xiaolin Wu szakaszrajzoló algoritmus

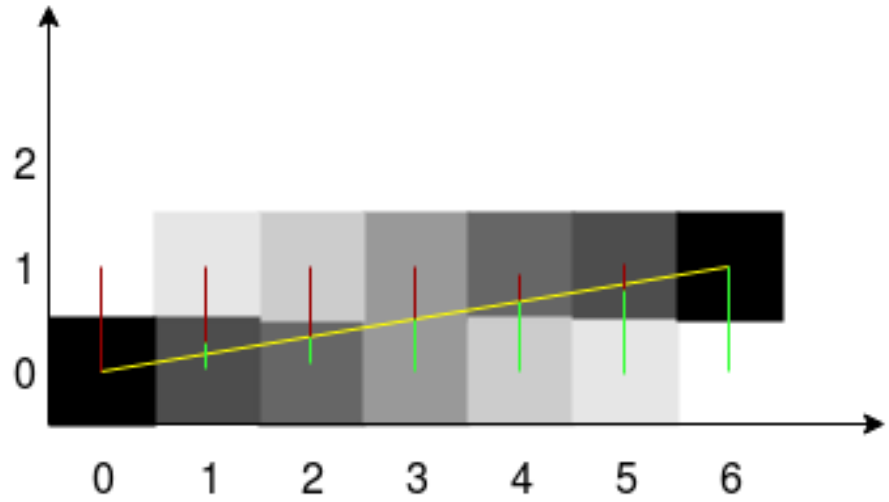
Az algoritmust úgy módosítjuk, hogy a szakasz éle el legyen simítva



- Az x irányú lépések esetén minden a két legközelebb eső pixelt kiválasztjuk
- Ezeket a távolságtól függő színintenzitással gyújtjuk ki arányosan
  - Tehát a pixel teljes színezettségű, ha a távolság 0
  - A pixel teljesen átlátszó, ha a távolság 1

Xiaolin Wu szakaszrajzoló algoritmus

Az algoritmust úgy módosítjuk, hogy a szakasz éle el legyen simítva



- Az x irányú lépések esetén minden a két legközelebb eső pixelt kiválasztjuk
- Ezeket a távolságtól függő színintenzitással gyújtjuk ki arányosan
  - Tehát a pixel teljes színezettségű, ha a távolság 0
  - A pixel teljesen átlátszó, ha a távolság 1
  - Az intenzitás megoszlik a két pixel között

Segédmetódusok

```
FÜGGVÉNY EGÉSZRÉSZ(VALÓS: X): EGÉSZ;
    ALGORITMUS
        EGÉSZRÉSZ <- (EGÉSZ)A;
ELJÁRÁS_VÉGE;

FÜGGVÉNY VALÓSRÉSZ(VALÓS: X): VALÓS;
    ALGORITMUS
        HA (X > 0) AKKOR
            VALÓSRÉSZ <- X { EGÉSZRÉSZ(X);
        KÜLÖNBEN
            VALÓSRÉSZ <- X { EGÉSZRÉSZ(X)
ELJÁRÁS_VÉGE;

FÜGGVÉNY VALÓSMARADÉKRÉSZ(VALÓS: X): VALÓS;
    ALGORITMUS
        VALÓSMARADÉKRÉSZ <- 1 { VALÓSRÉSZ(X);
FÜGGVÉNY_VÉGE;
```

Algoritmus

```
ELJÁRÁS WU_SZAKASZ_VÁZLAT(EGÉSZ: X0, EGÉSZ: Y0,
                          EGÉSZ: X1, EGÉSZ: Y1, SZÍN: SZ)
    ALGORIMTUS
        HA (MEREDEKSÉG > 45°) AKKOR
            X ÉS Y KOORDINÁTÁK CSERÉJE;
        HA_VÉGE;
        HA (X0 > X1) AKKOR
            A PONTOK CSERÉJE;
        HA_VÉGE;

        ...
ELJÁRÁS_VÉGE;
```

Algoritmus

```
ELJÁRÁS WU_SZAKASZ_VÁZLAT(EGÉSZ: X0, EGÉSZ: Y0,
                          EGÉSZ: X1, EGÉSZ: Y1, SZÍN: SZ)
    ALGORIMTUS
        ...

        HA (MEREDEKSÉG > 45°) AKKOR
            CIKLUS X <- X0..X1
                PIXEL(EGÉSZRÉSZ(Y0), X, VALÓSMARADÉKRÉSZ(Y0));
                PIXEL(EGÉSZRÉSZ(Y0) - 1, X, VALÓSRÉSZ(Y0));
                Y <- Y + MEREDEKSÉG;
            CIKLUS_VÉGE;
        KÜLÖNBEN
            CIKLUS X <- X0..X1
                PIXEL(X, EGÉSZRÉSZ(Y0), VALÓSMARADÉKRÉSZ(Y0));
                PIXEL(X, EGÉSZRÉSZ(Y0) - 1, VALÓSRÉSZ(Y0));
                Y <- Y + MEREDEKSÉG;
            CIKLUS_VÉGE;
        HA_VÉGE;
ELJÁRÁS_VÉGE;
```

Áttekintés

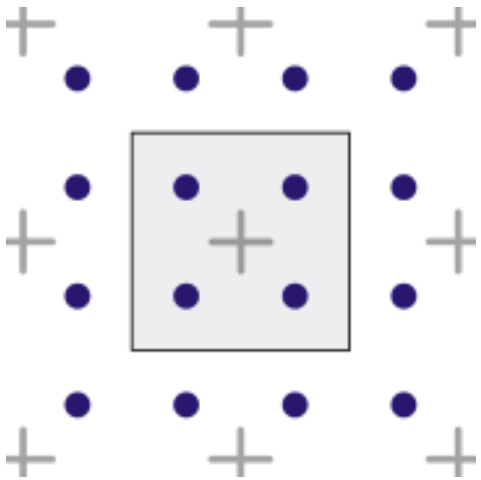
Supersampling

Az eljárás a már elkészült képet simítja, de könnyen átírható úgy, hogy például egy szakasz mentén végezze el az eljárást

Supersampling

Az eljárás a már elkészült képet simítja, de könnyen átírható úgy, hogy például egy szakasz mentén végezze el az eljárást

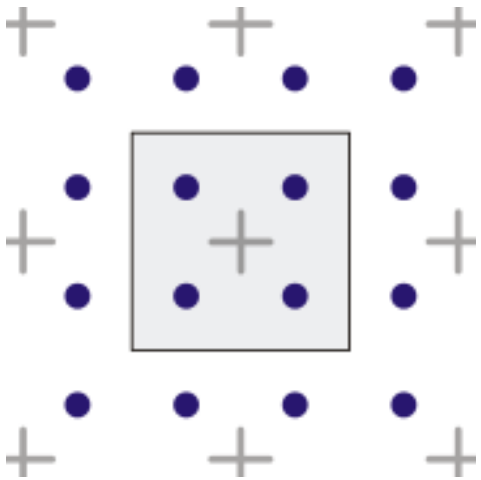
- Minden pixelt felbontunk további úgynevezett subpixelekre



Supersampling

Az eljárás a már elkészült képet simítja, de könnyen átírható úgy, hogy például egy szakasz mentén végezze el az eljárást

- Minden pixelt felbontunk további úgynevezett subpixelekre

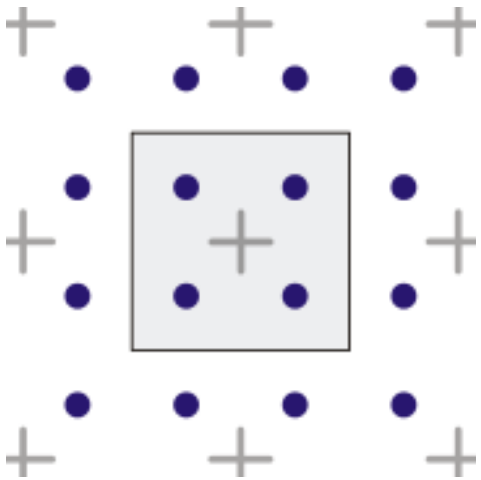


- Ez azt jelenti, hogy elméletben egy nagyobb felbontású képpel dolgozunk

Supersampling

Az eljárás a már elkészült képet simítja, de könnyen átírható úgy, hogy például egy szakasz mentén végezze el az eljárást

- Minden pixelt felbontunk további úgynevezett subpixelekre



- Ez azt jelenti, hogy elméletben egy nagyobb felbontású képpel dolgozunk
- A megjelenített pixel színe a részpixelek színértékeinek átlagából származtatható

Köszönöm a figyelmet!