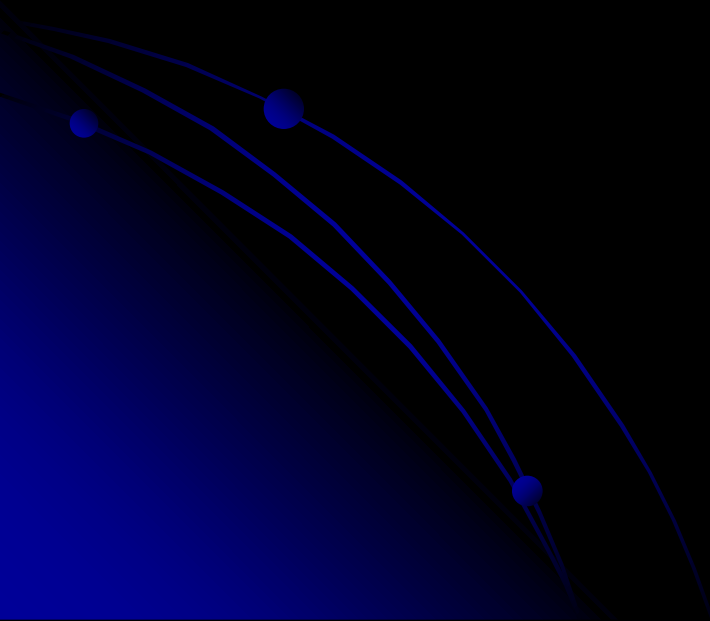


Kvantovanie - metódy



Originál

500x362 = 181 000 pixlov

ak máme pre každý pixel 3x8 bitov (8 bitov pre R,G,B)

= 4 344 000 bitov



Čo chceme?

zmenšiť toto číslo

- Čo najviac (2 farby)
- Aby sa výsledný obrázok podobal originálu



30/36/48-bitová reprezentácia

Extrémne vysoký počet odtieňov

30 bits (1.073 billion colors),

36 bits (68.71 billion colors) a

48 bits (281.5 trillion colors).

Deep Color

24-bitová reprezentácia

každá farba je reprezentovaná 8 bitmi

máme ~16 miliónov možných farieb

Truecolor



16-bitová reprezentácia

16 bitov môžeme rozdeliť

5-5-5-1 (R-G-B-transparentnosť)

5-6-5 (R-G-B)

~65 tisíc možných farieb

Hicolor

8-bitová reprezentácia

1 byte na pixel

256 farieb rozdelíme na 3-3-2 (R-G-B)

Alebo použijeme Look-Up Table (LUT) – index do palety farieb

Indexed color

1-bitová reprezentácia

1 bit na pixel

Binárny obraz

Kvantovanie

- Redukcia počtu farieb s minimálnou vizuálnou distorziou (deformáciou)
- Stratová obrazová kompresia
- Znižuje nároky
 - na úložný priestor
 - na šírku prenosového pásma
- Dôležité
 - Výpočtová efektívnosť
 - Distorzia obrazu čo najmenšia

Kvantovanie matematicky

C – priestor farieb

P – kvantovaný priestor ($P \subset C$), paleta, color map

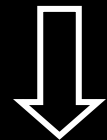
$$P = \{c_1, c_2, \dots, c_n \mid c_i \in C, n \ll \|C\|\}$$

Kvantizátor Q :

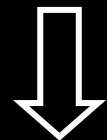
$$Q: C \rightarrow P$$

Fázy kvantovania

- Preskúmanie originálneho obrazu, zistenie informácií o použitých farbách
- Určenie palety na základe týchto informácií
- Namapovanie farieb na vybrané reprezentatívne farby
- Vykreslenie nového (kvantovaného) obrazu



Algoritmus



Mapovanie...



Chyba kvantovania

spôsob merania kvality aproximácie

pre každú farbu x v origináli zadefinuje $d(x,c)$
vzdialenosť od novej farby c

zvyčajne sa používa euklidovská metrika v RGB
mala by byť v CIE Lab

priemerná kvadratická chyba všetkých bodov

Chyba kvantovania

Mean square error (MSE) pre daný obraz

$$E = \frac{1}{n} \sum_i d^2(x_i, Q(x_i))$$



Metóda 1



Metóda 2



Error map

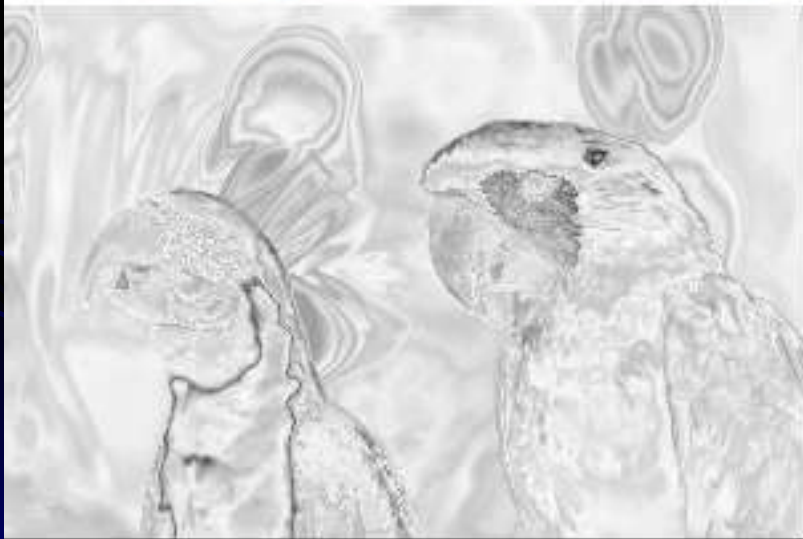
256 farieb



1. metóda



2. metóda



Error map

16 farieb



Chyba kvantovania

Používa sa ako podmienka pri iteratívnych metódach kvantovania

Zlepšujeme, kým je rozdiel chýb 2 iterácií väčší ako daný prah



Podmienky optimality

Vyplývajú z minimalizácie MSE

Podmienka najbližšieho suseda

Pre danú paletu P optimálne regióny $\{R_i: i=1, \dots, N\}$ spĺňajú podmienku:

$$R_i \subset \{x: d(x, c_i) \leq d(x, c_j); \forall j\}$$

a teda

$$Q(x) = c_i \Leftrightarrow d(x, c_i) \leq d(x, c_j) \forall j$$

Podmienky optimality

Podmienka centroidu

Pre dané regióny $\{R_i: i=1, \dots, N\}$, prvky palety spĺňajú podmienku:

$$c_i = \text{cent}(R_i),$$

kde centroid množiny R je aritmetický priemer:

$$\text{cent}(R) = \frac{1}{|R|} \sum_{i=1}^{|R|} x_i$$

pre $R = \{x_i : i = 1, \dots, |R|\}$

Kvantovanie

podľa prístupu k informáciám:

Obrazovo nezávislé: Priestor farieb je rozdelený na pravidelné regióny, bez ohľadu na farebné vlastnosti obrazu.

• **Obrazovo závislé:** Rozdelenie priestoru farieb závisí od skutočného rozloženia farieb v obraze.



Kvantovanie – zhlukovacie metódy

Clustering methods

3 kanály – RGB, HSV, Lab, ...

podľa prístupu k zložkám:

- skalárne – po zložkách (SQ)
 - 3 x 1D splitting methods
- vektorové – celá trojicu naraz (VQ)
 - Rozdeľovanie - 3D splitting methods
 - Zlučovanie - Grouping methods
 - Kombinované metódy - Split and merge methods

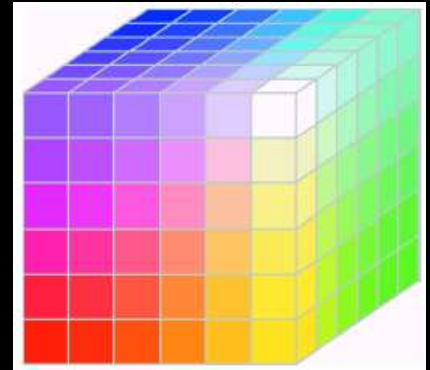
Uniformné kvantovanie

RGB kocka na

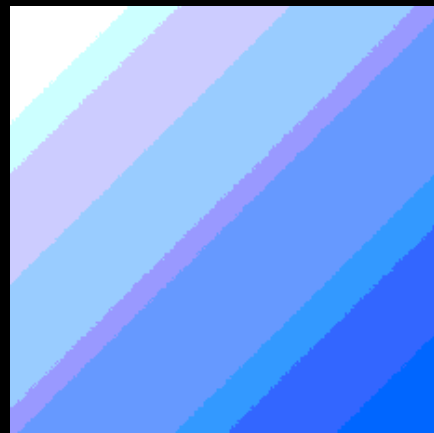
8x8x4 (rozdelenie 3-3-2)

6x6x6

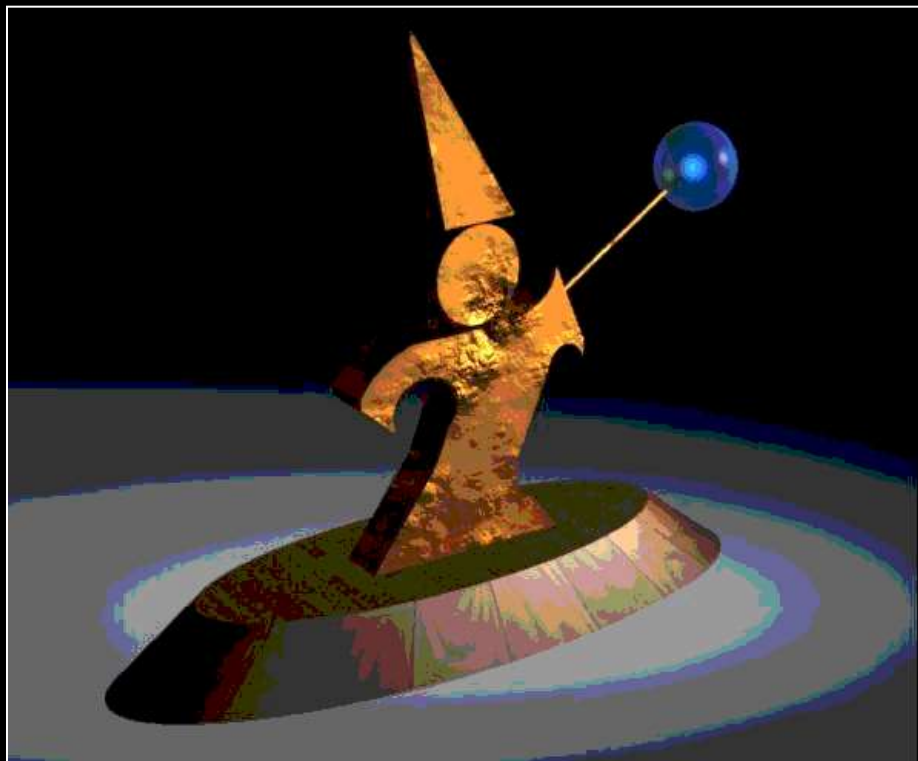
6x7x6



- Pre každú farbu určíme do ktorého chlievika patrí.
- Reprezentatívna farba sa určí ako priemer všetkých farieb v danom regióne.
- Rýchle, jednoduché, ale výsledky nie sú dobré.

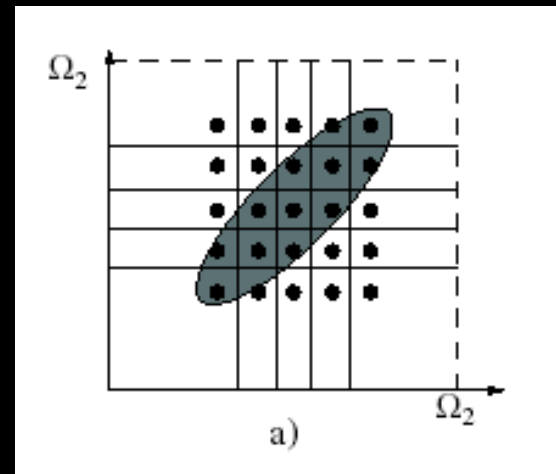


Paleta 6x6x6



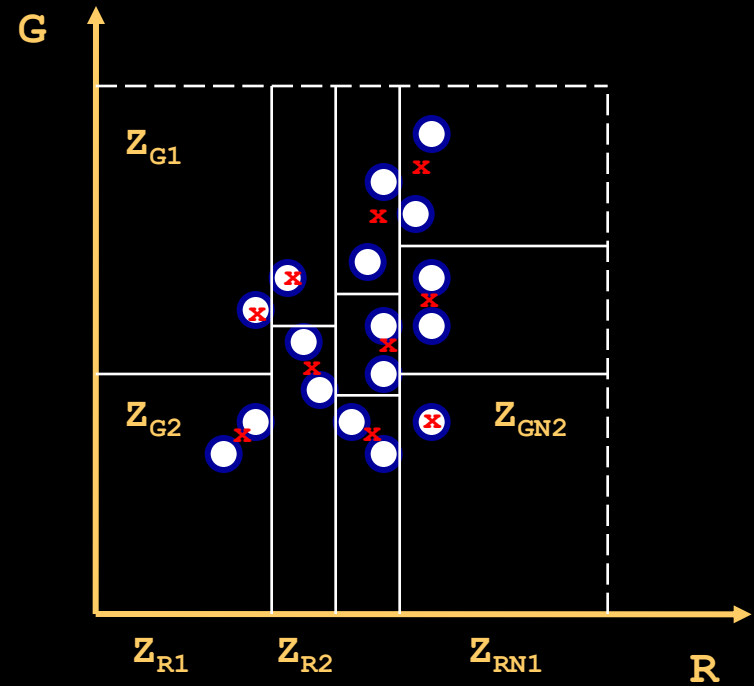
Nezávislé skalárne kvantovanie

- 3 marginálne histogramy zložiek R, G a B
- Kvantizačný algoritmus aplikovaný nezávisle na každú zložku
- Nemôže zohľadniť korelácie medzi zložkami
- Ak sú farby obrazu len určitej časti RGB kocky – zvyšok je prázdny a zbytočne zaberá priestor v palette.



Sekvenčné skalárne kvantovanie

- kvantuj R zložku na N_1 úrovni vzhľadom na rozloženie farieb (Z_{R1} až Z_{RN1})
- kvantuj G zložku v každom novom zhluku podľa rozloženia farieb (Z_{G1} až Z_{GN2})
- kvantuj B zložku v každom novom zhluku podľa rozloženia farieb (Z_{B1} až Z_{BN3})
- vyber centroid každého zhluku
- $N_1 + N_2 + N_3 = N$ zhlukov - **problém** ako vybrať N_1 , N_2 a N_3
- Výsledok závisí od výberu poradia zložiek R, G, B



Sekvenčné skalárne kvantovanie



2



16



4

256



Rozdeľovací prístup

Daj všetky vektory do jedného zhluku

REPEAT

Vyber zhluk, ktorý treba rozdeliť

Rozdeľ zhluk

UNTIL daný počet zhlukov

Table 9.1 Different Strategies Used by Five Tree-Structured Vector Quantizers

		1	2	3	4	5
Cluster selection	Greatest squared error		*		*	*
	Greatest eigenvalue			*		
	Greatest cardinality	*				
Cutting axis	Longest coordinate axis	*				
	Coordinate axis with greatest variance					*
	Major axis		*	*	*	
Cutting position	Median cut	*				
	Marginal squared error minimization		*			
	TSE minimization				*	*
	Pass through the mean			*		

Note: 1 = Heckbert,³⁵ 2 = Wan and Wong,⁷¹ 3 = Bouman and Orchard,^{12,13} 4 = Wu,⁷⁵ and 5 = Braquelaire and Brun.¹⁴

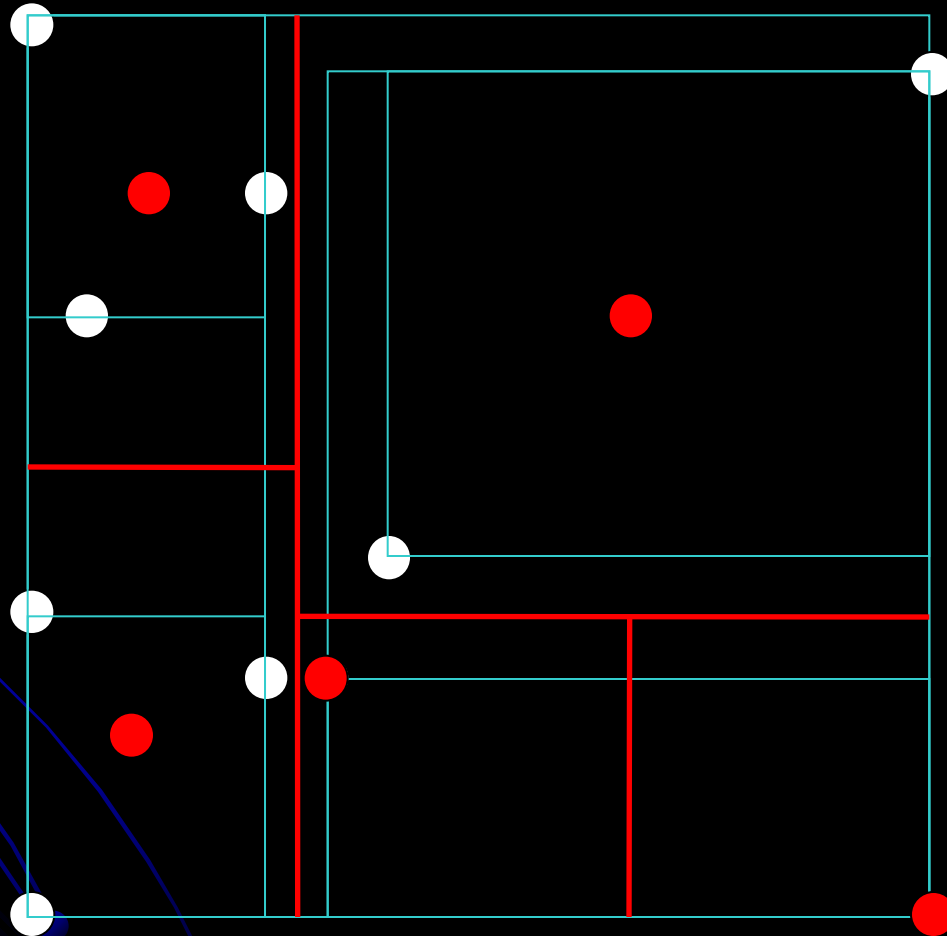
Algoritmus rozdelenia podľa mediánu (median cut)

Paul Heckbert in 1980

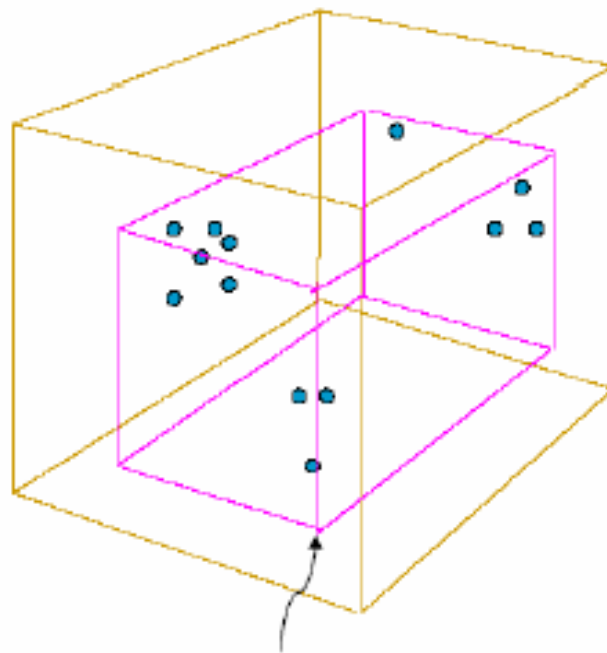
Koncept – reprezentatívne farby zastupujú približne rovnaký počet pôvodných farieb

- nájsť najmenší obal obsahujúci všetky farby
- zoradiť farby podľa najdlhšej osi
- rozdeliť obal v bode mediánu
- opakuj, kým nemáme K farieb

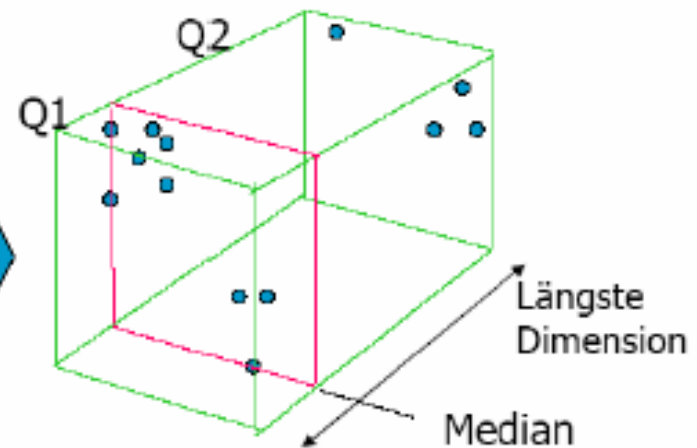
Median Cut



Median Cut



Urspr. Bounding Box Q



Median Cut

Reprezentant = priemer

časová a priestorová náročnosť

Vylepšenie – prestať s delením, keď už oko nerozlišuje susedné farby

V modrých prestaneme deliť skôr, než v červených a potom v zelených farbách



4

8




32

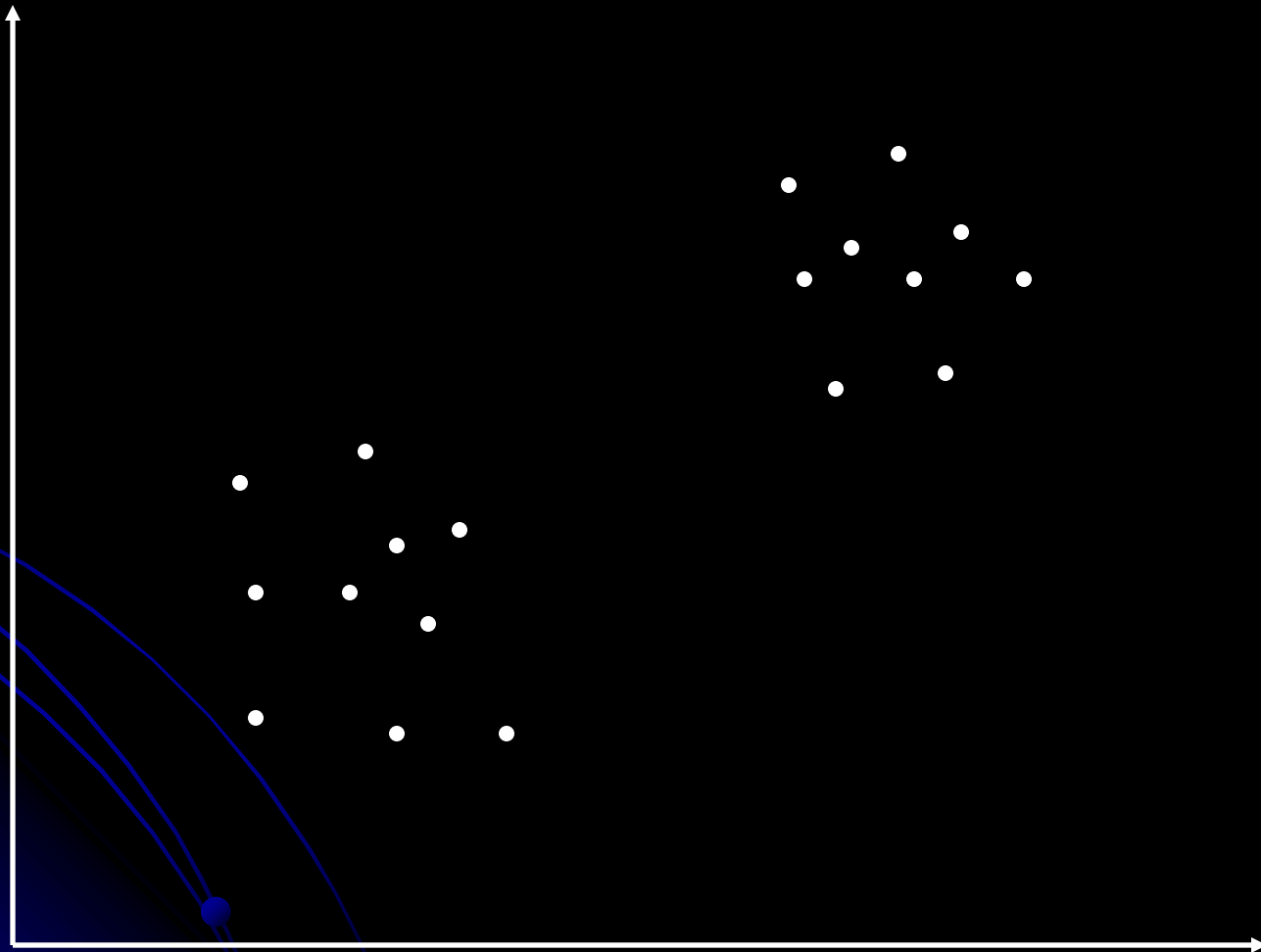
256



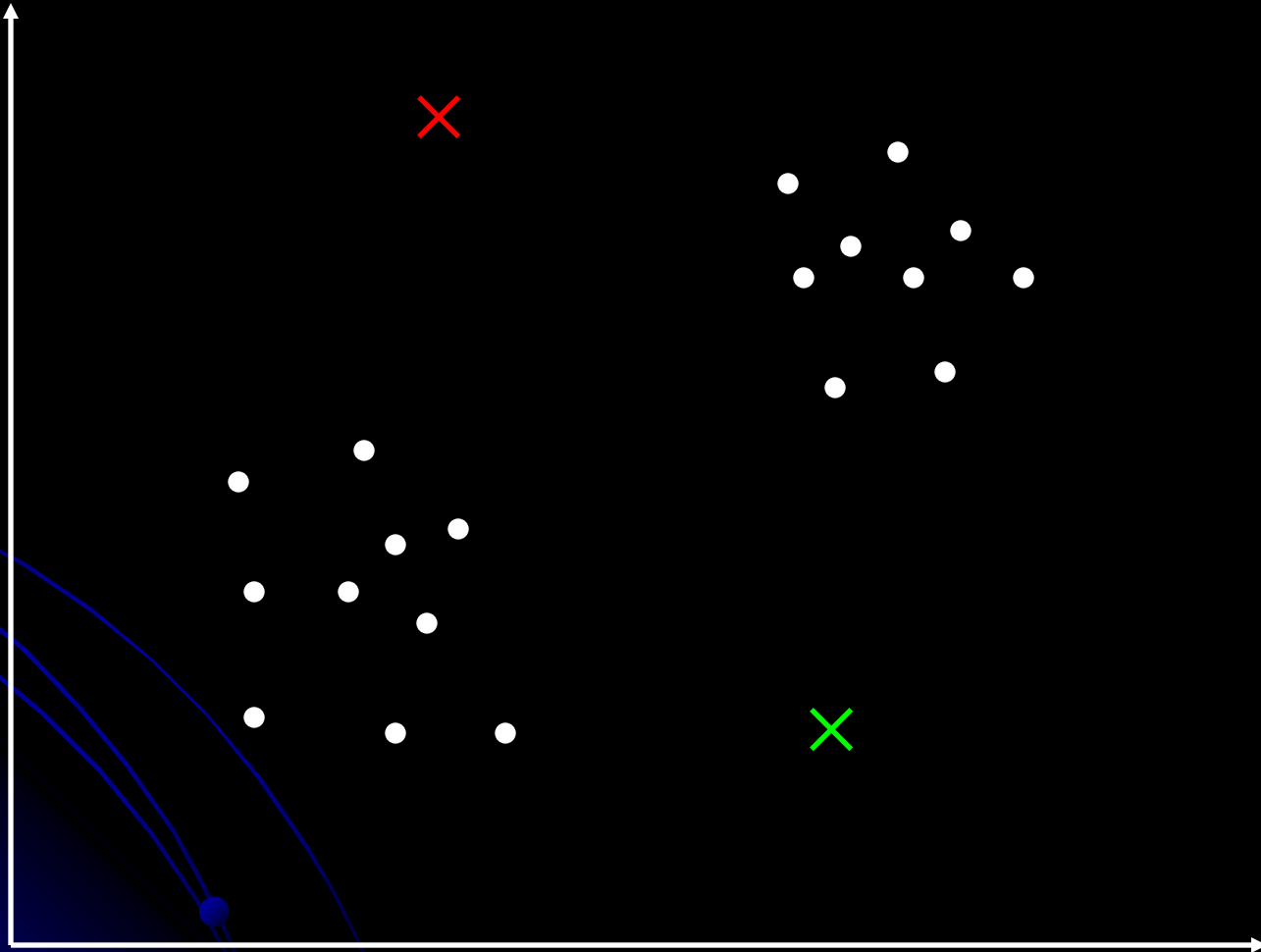
K-Means Clustering

- Vygeneruj začiatočné rozdelenie
 - Nájdí centroid každého zhluku
 - Pre každú farbu:
 - Vyrátaj vzdialenosť od každého centroidu
 - Prirad' k najbližšiemu zhluku
 - Vyrátaj nové centroidy
 - Opakuj, kým nie sú zhluky stabilné ($MSE < \text{prah}$)
- 

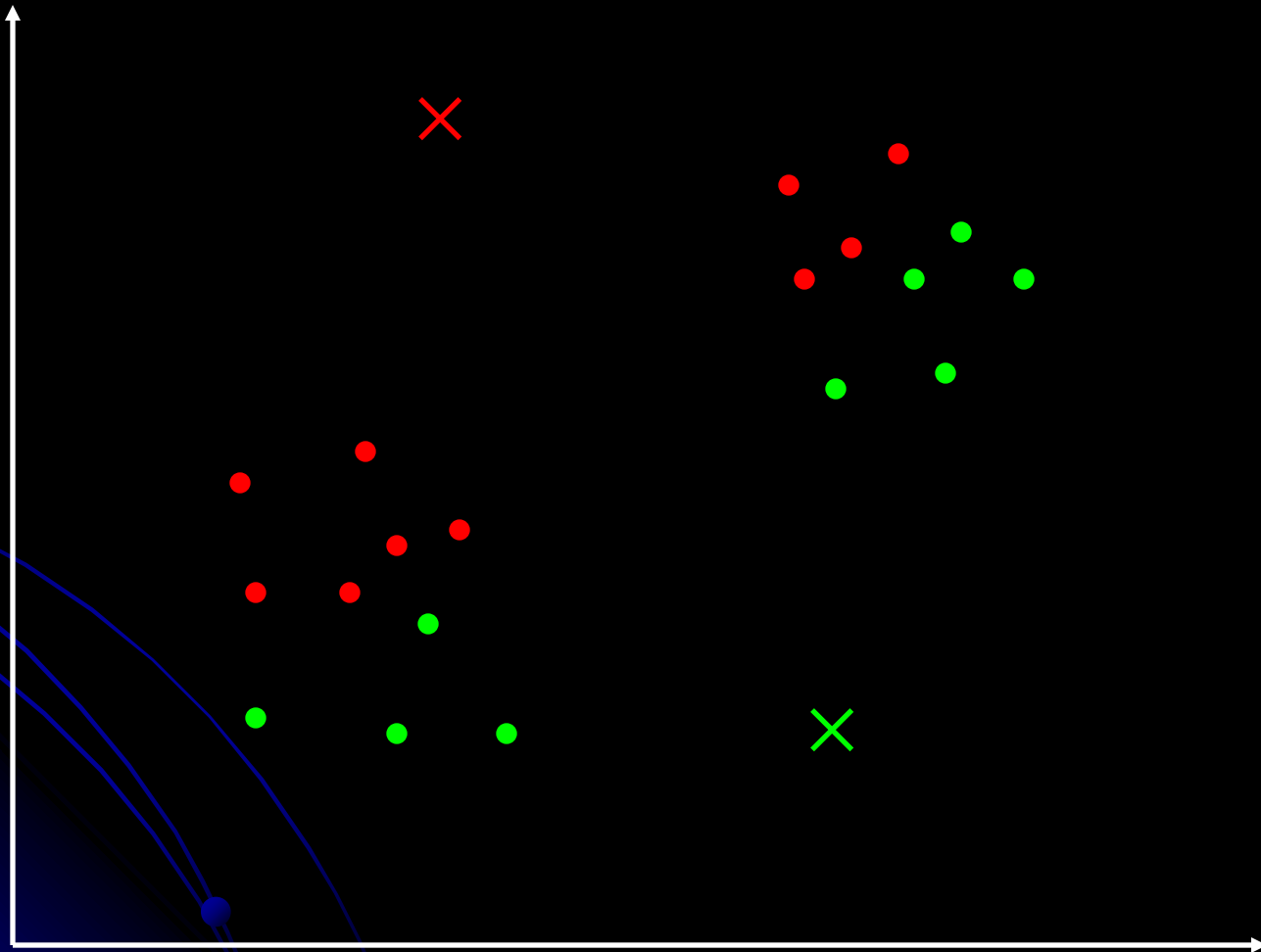
K-Means Clustering



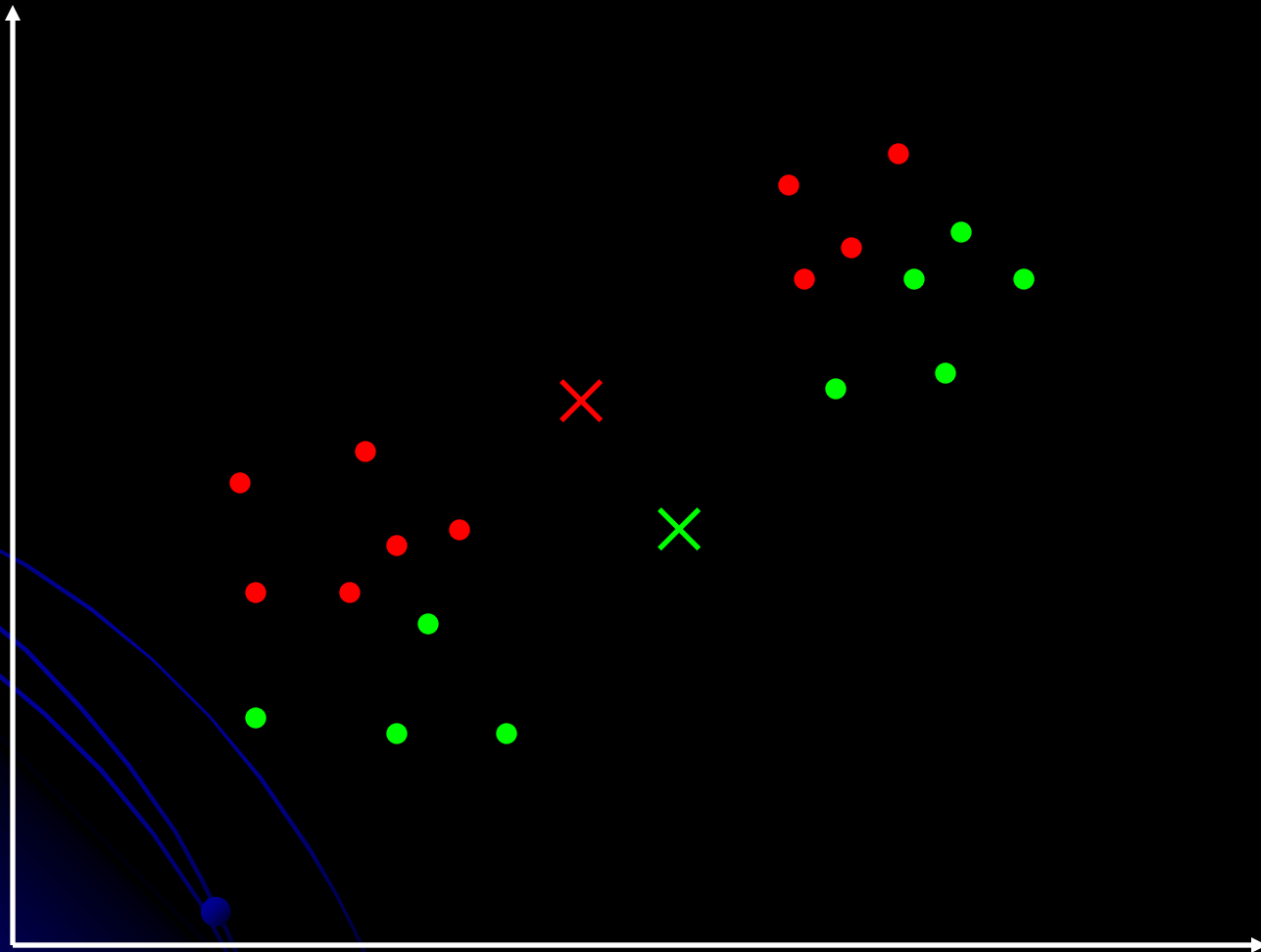
K-Means Clustering



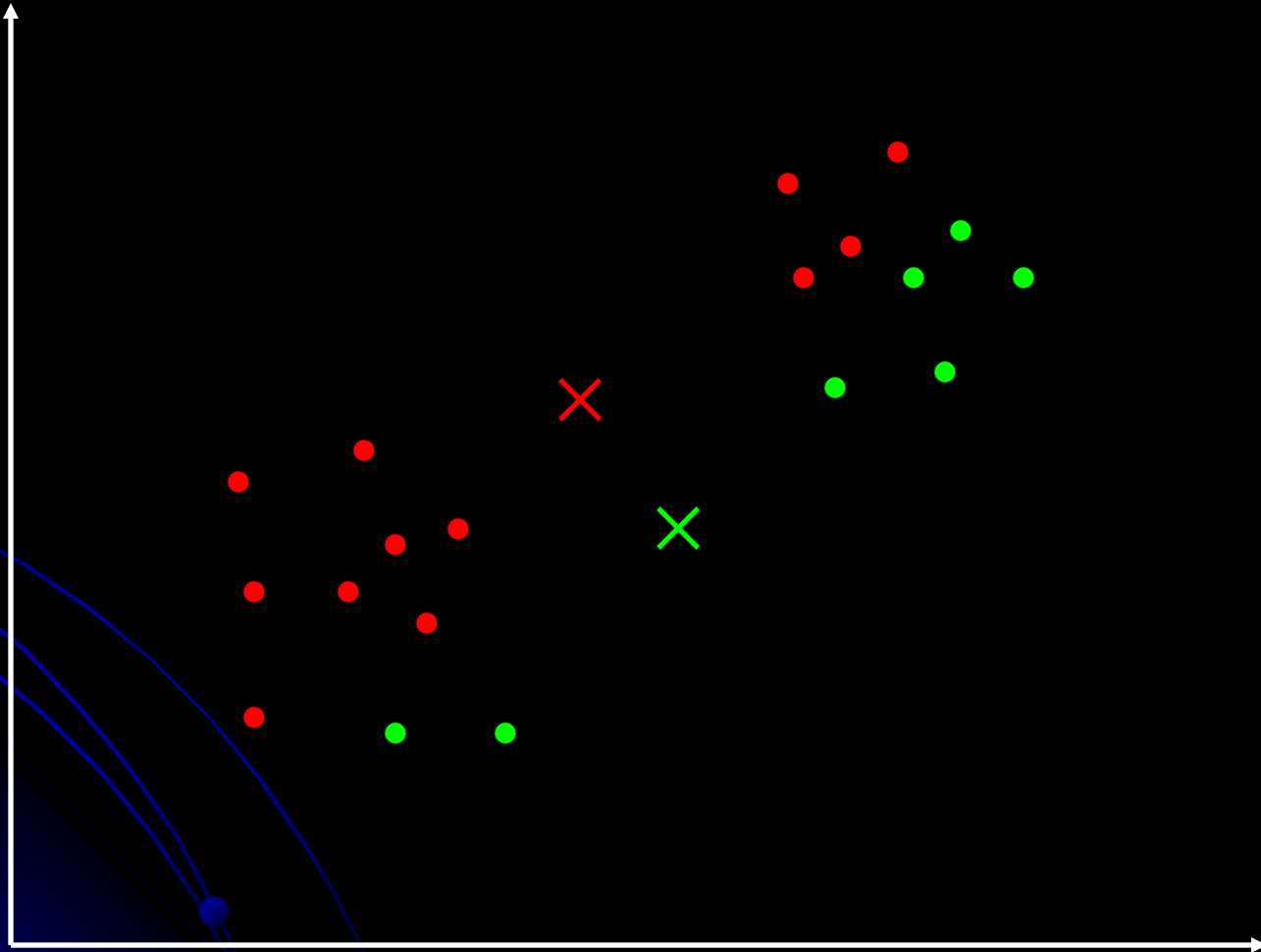
K-Means Clustering



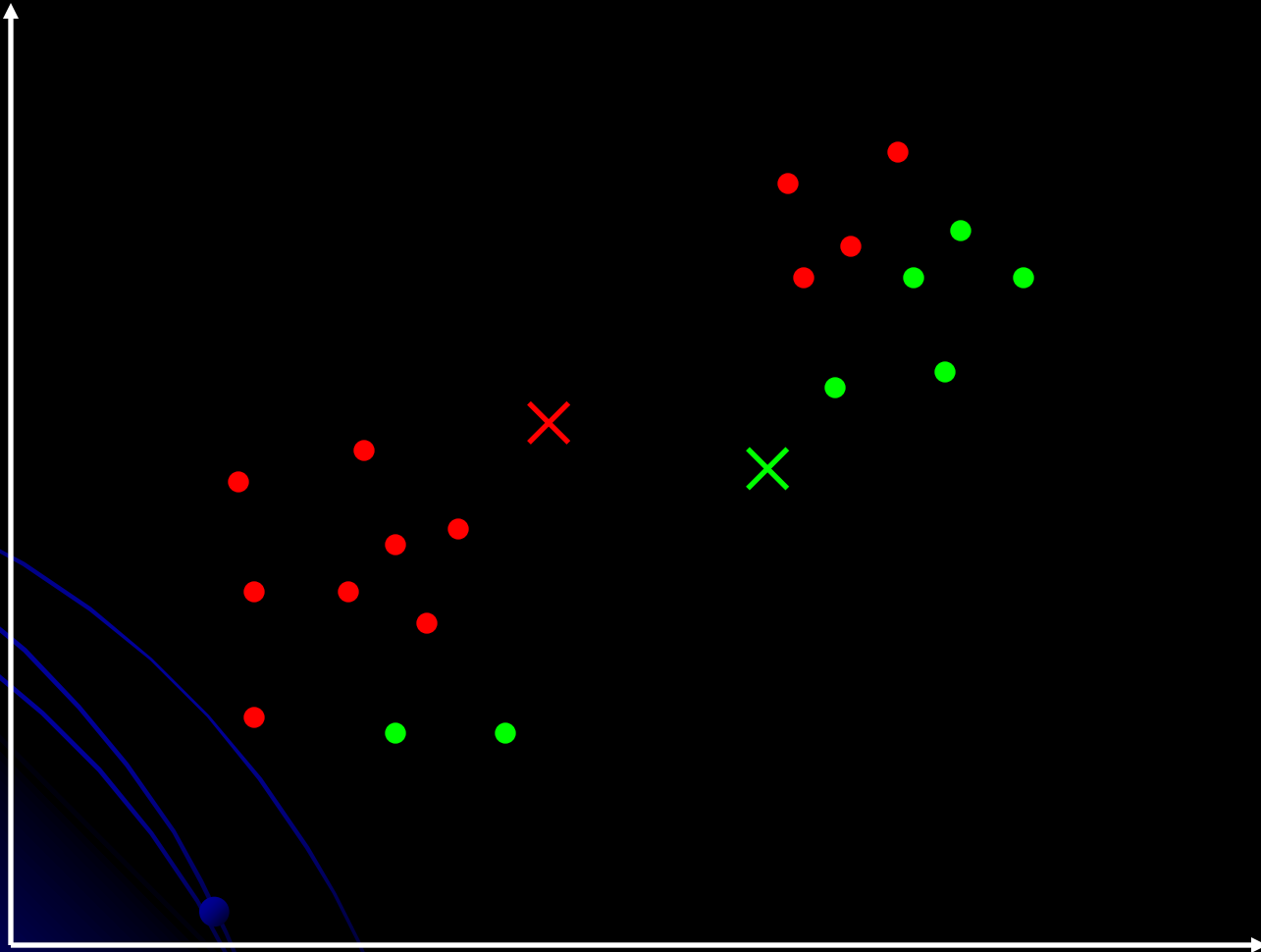
K-Means Clustering



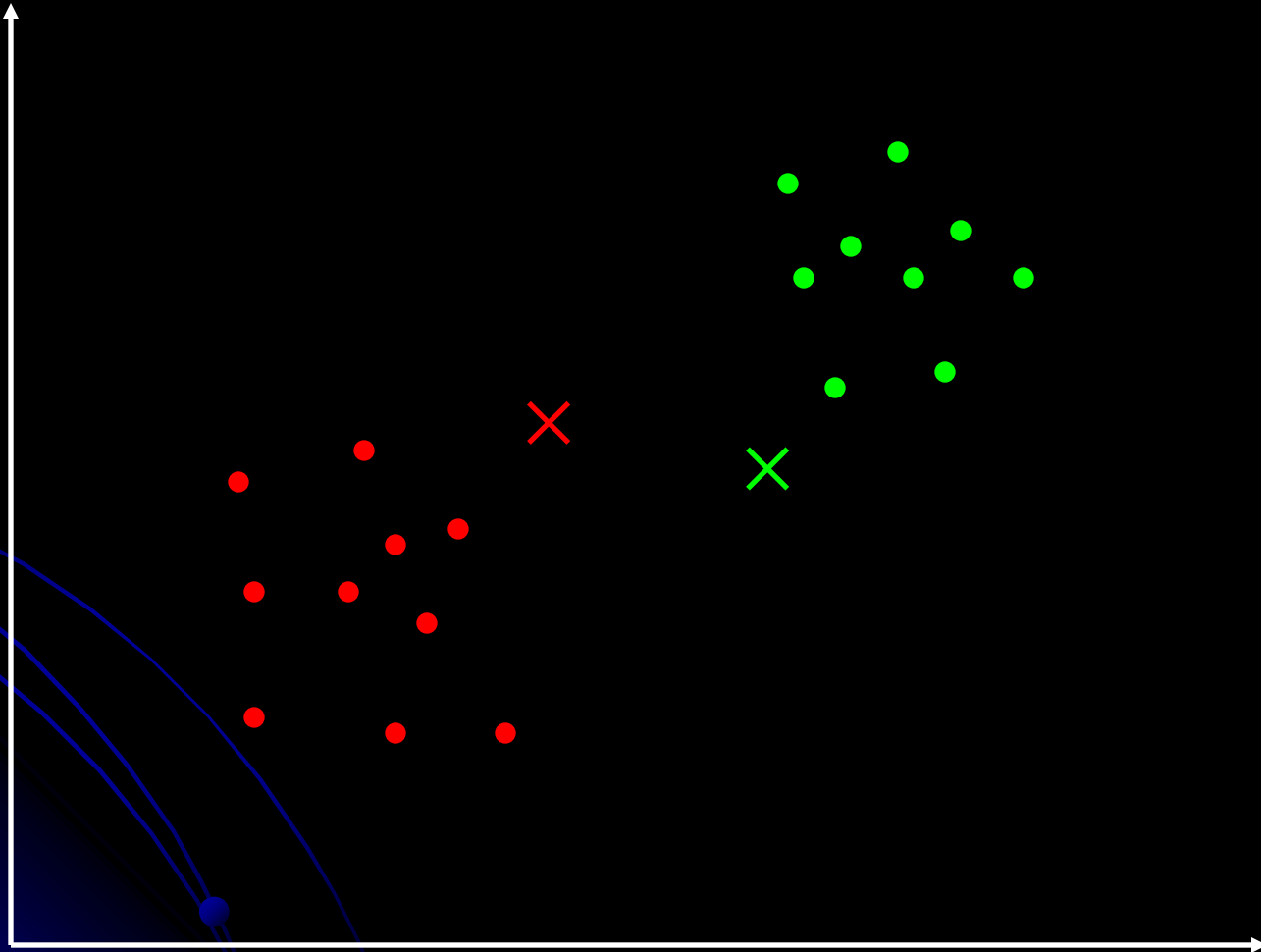
K-Means Clustering



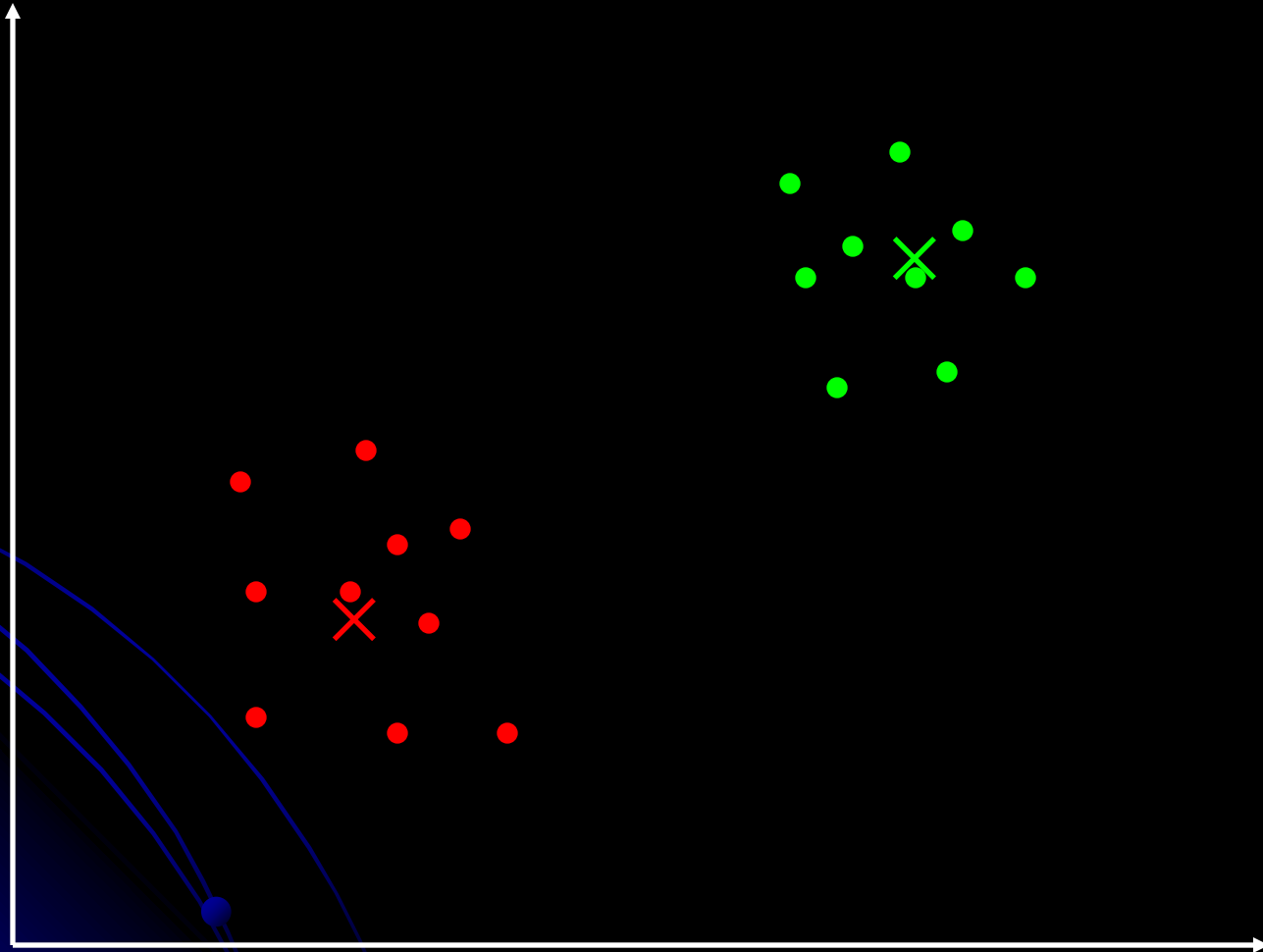
K-Means Clustering



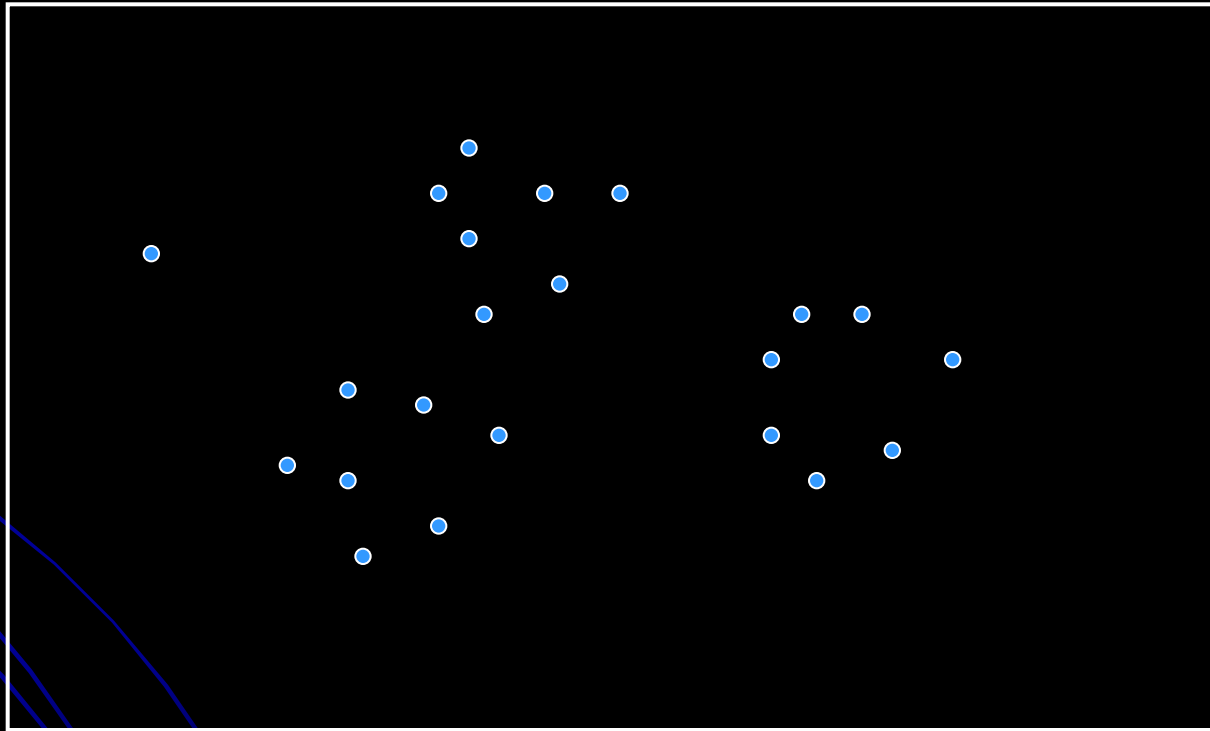
K-Means Clustering



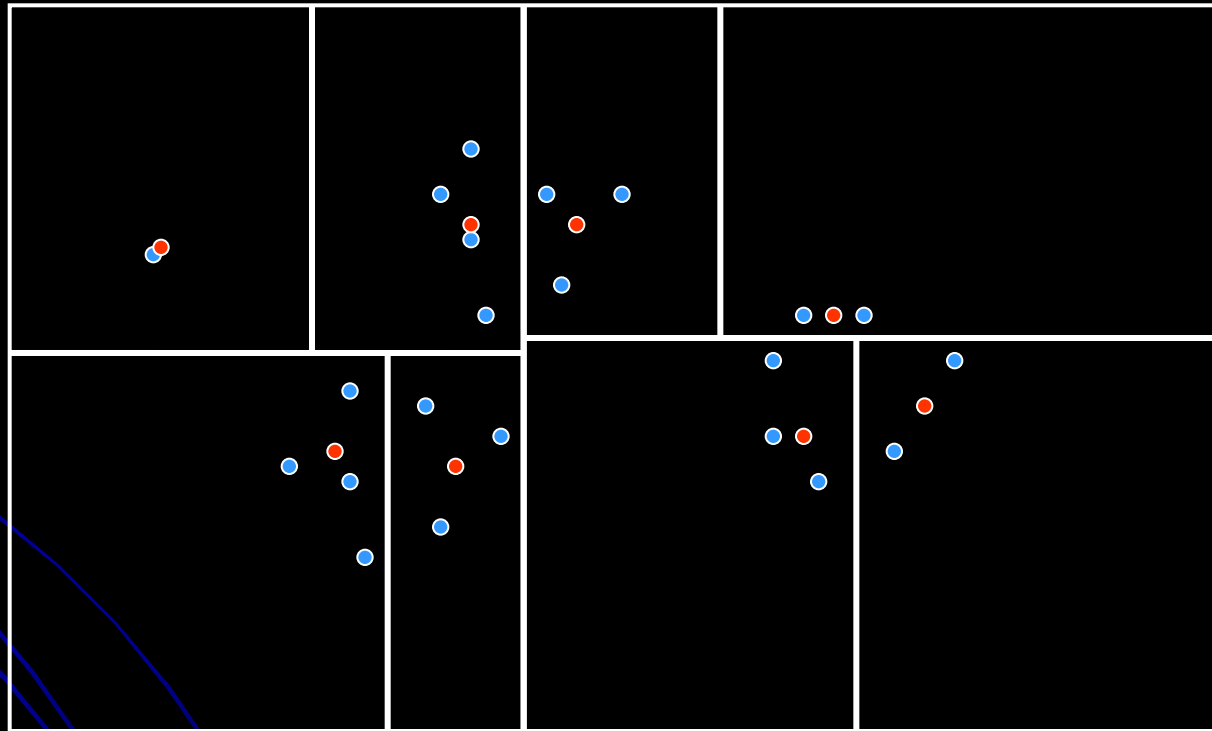
K-Means Clustering



Median Cut

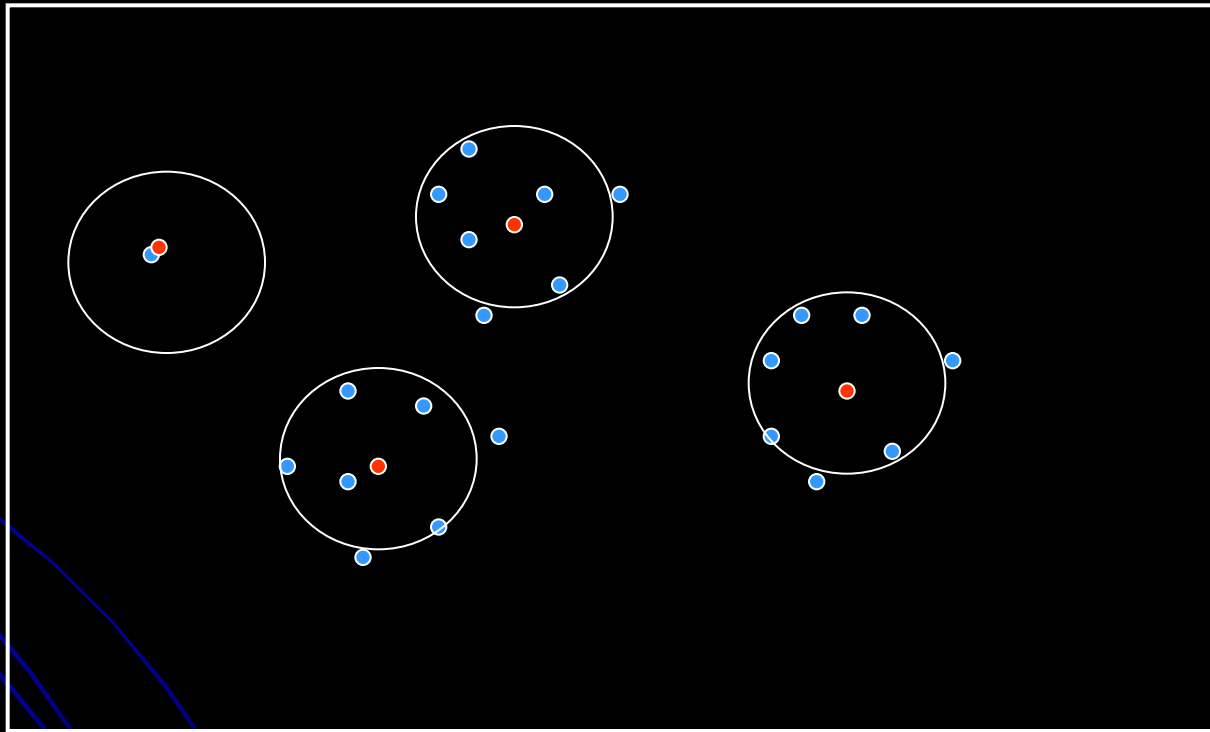


Median Cut



Lepšie riešenie

median-cut, potom k-means, potom zlúčiť blízke
zhluky ($d(c_i, c_j) < t$)

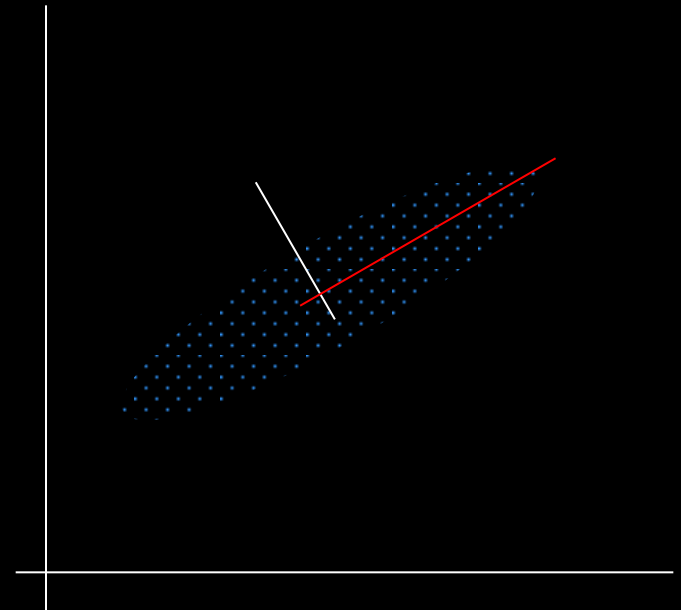


Delenie na báze PCA


PCA – principal component analysis

Karhunen-Loeve transformácia (výpočtovo náročná, kovariančná matica, vlastné vektory a čísla ...)

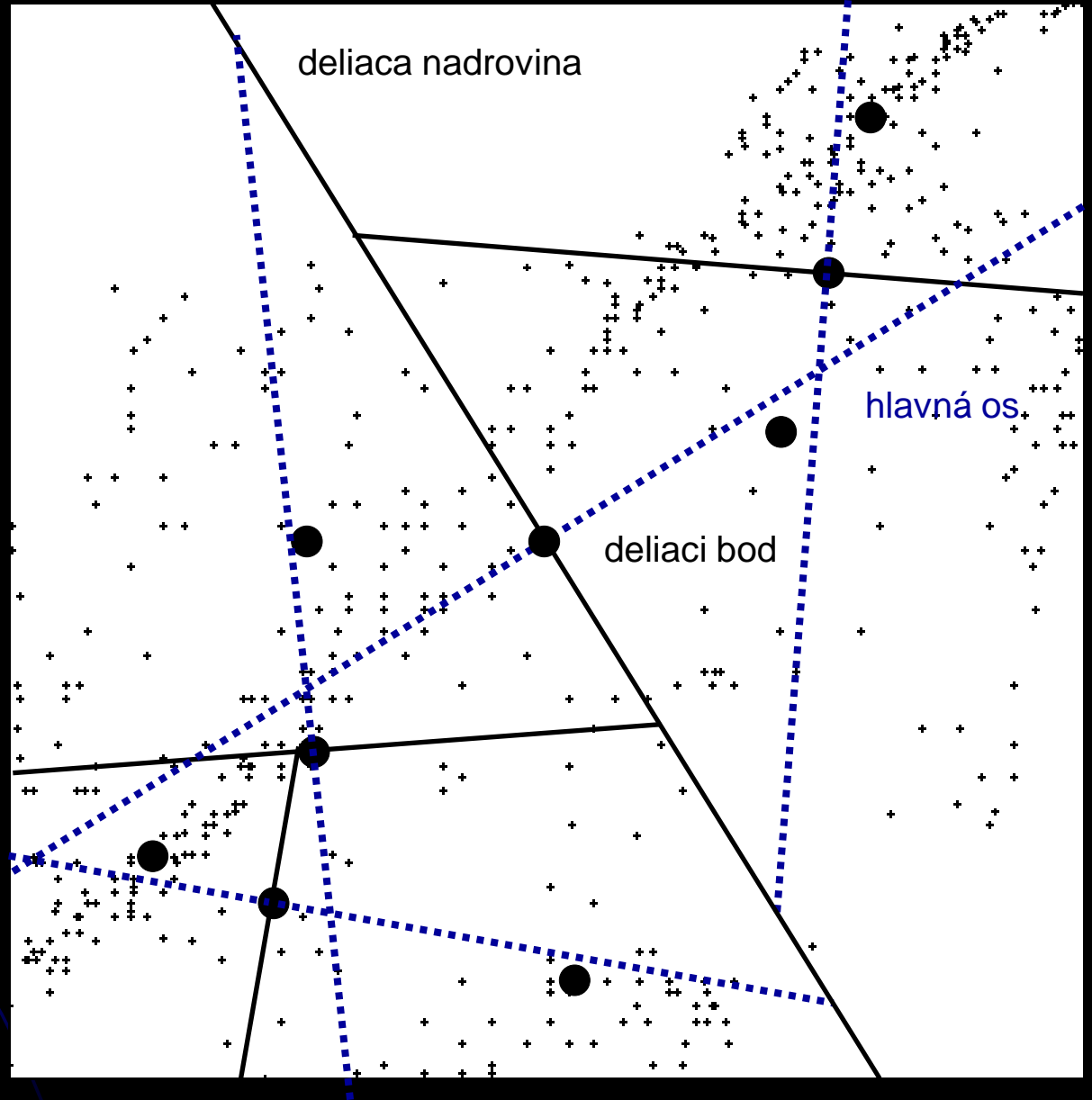
Vlastný vektor zodpovedajúci najväčšiemu vlastnému číslu = hlavná os




Delenie na báze PCA

1. vyrátaj hlavnú os (*principal axis*).
 2. vyber deliaci bod P na hlavnej osi.
 3. rozdeľ vzhľadom na nadrovinu (*hyperplane*).
 4. vyrátaj centroidy 2 nových zhlukov.
- 

1. vyrátaj hlavnú os (*principal axis*).
2. vyber deliaci bod P na hlavnej osi.
3. rozdeľ vzhľadom na nadrovinu (*hyperplane*).
4. vyrátaj centroidy 2 nových zhlukov.



Zlučovací prístup

- každý bod tvorí samostatný zhluk
 - zlúč 2 zhluky, ktoré sú si najbližšie vzhľadom na danú metriku
 - opakuj, kým sa dajú zhluky zlučovať
- 

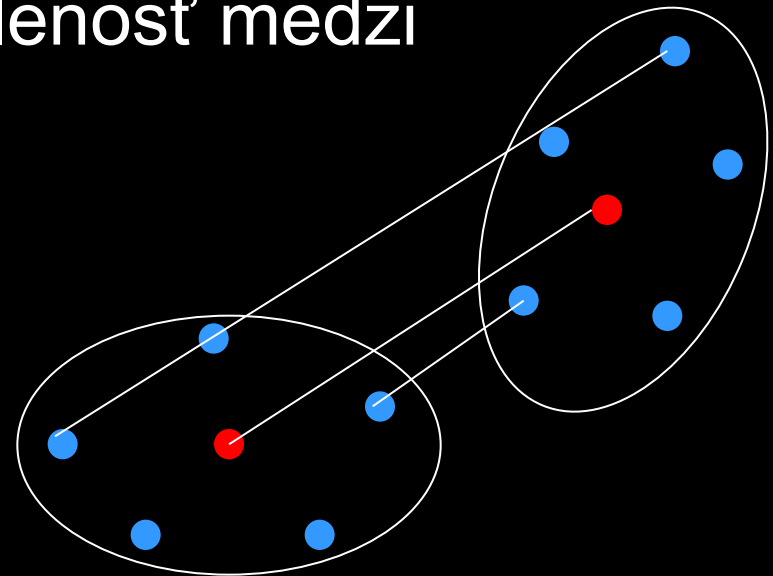
Vzdialenosť zhlukov

single-link: Minimálna vzdialenosť medzi prvkami A a B

complete-link: Maximálna vzdialenosť medzi prvkami A a B

centroid-link: Vzdialenosť medzi cetroidmi

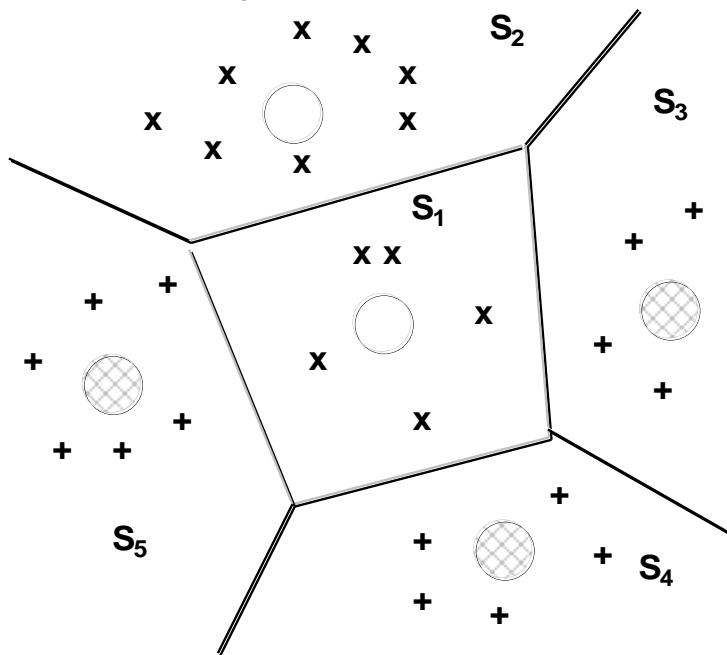
average-link: Priemerná vzdialenosť medzi prvkami A a B



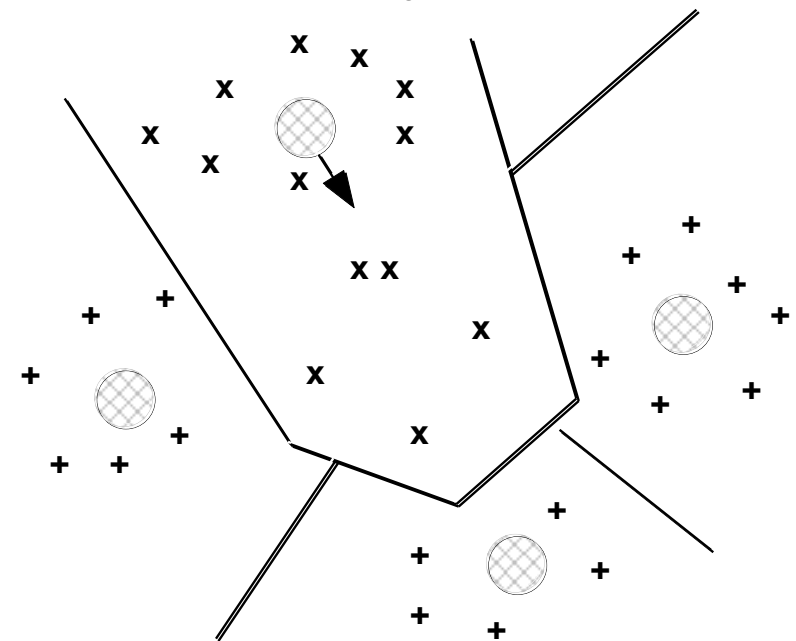
Zlučovanie

Pairwise Nearest Neighbor (PNN alg.)

Before cluster merge



After cluster merge



Code vectors:



Vectors to be merged



Remaining vectors

Training vectors:

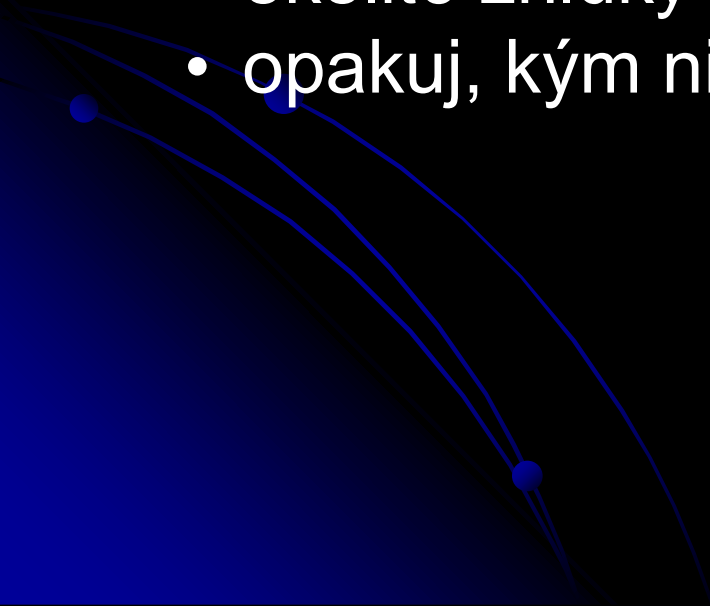
x

Training vectors of the clusters to be merged

+

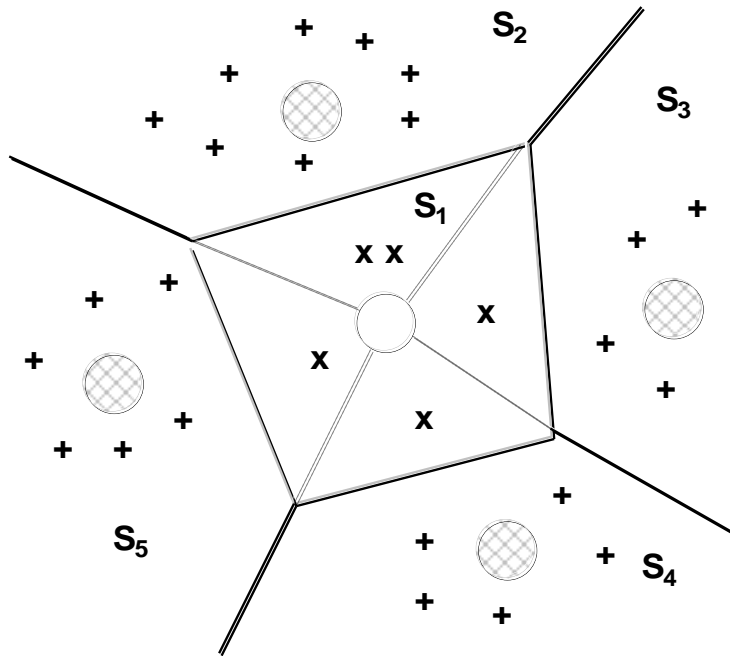
Other training vectors

Iteratívne zmršťovanie (shrinking)

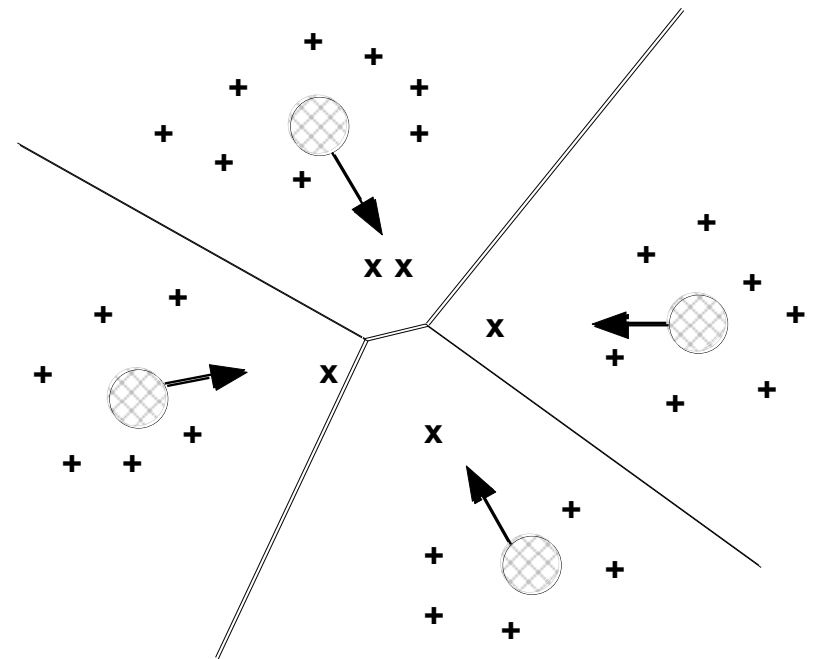
- každý bod tvorí samostatný zhluk
 - vyber zhluk na odstránenie
 - rozdeľ body z vybraného zhluku a medzi okolité zhluky
 - opakuj, kým nie je daný počet zhlukov
- 

Shrinking

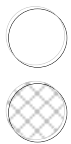
Before cluster removal



After cluster removal



Code vectors:



Vector to be removed

Remaining vectors

Training vectors:

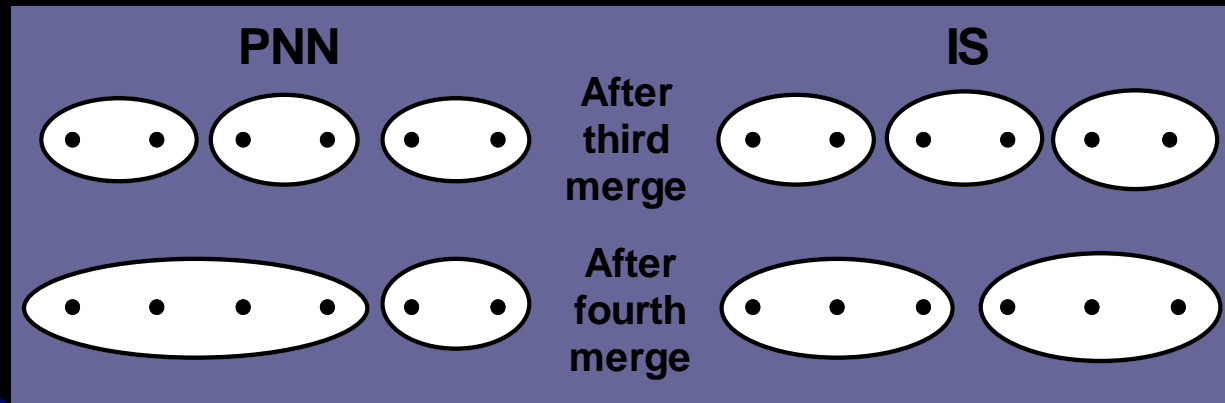
x

+

Training vectors of the cluster to be removed

Other training vectors

Rozdziel



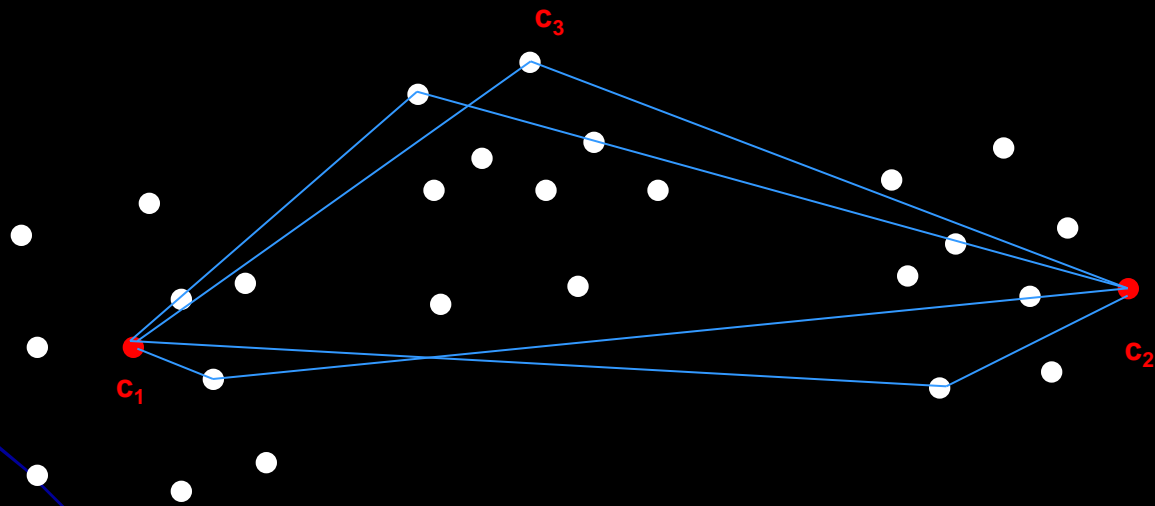
Max – min algoritmus

- Vyber farbu c_1
- Ďalší reprezentant farba c_k je ešte nevybratá farba ktorej minimálna vzdialenosť od akejkolvek reprezentanta je maximálna

$$\min_{k' = 1, \dots, k-1} \|c_k - c_{k'}\|^2 \geq \min_{k' = 1, \dots, k-1} \|c - c_{k'}\|^2 \quad \forall c \in \{\Omega^I - \Omega^Q\}$$

- Opakuj až vyberiem K reprezentatívnych farieb

Max – min algoritmus



Kombinovaný prístup

Rozdeľ a zlúč (Split and merge)

Vygeneruj začiatkové rozdelenie hocíjakým algoritmom

Repeat

 Vyber zhluk na rozdelenie

 Rozdeľ

 Vyber 2 zhluky na zlúčenie

 Zlúč

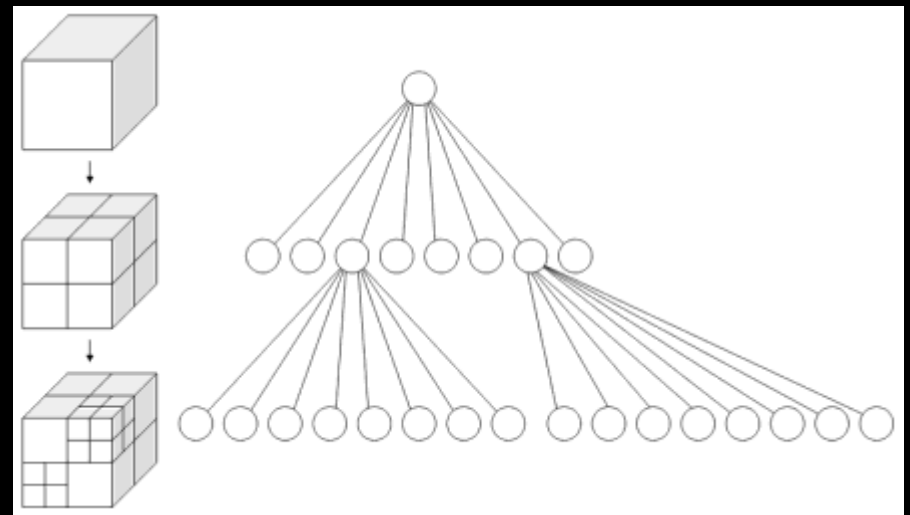
Until žiadne možné zlepšenie

Algoritmus Octree

Idea – vytvoriť stromovú štruktúru obsahujúcu maximálne K farieb

Vnútorne uzly stromu obsahujú maximálne 8 nasledovníkov

Listy obsahujú informácie o farbách, index farby a doterajšiu početnosť danej farby



Algoritmus Octree

Binárne vyjadrenie farieb v RGB – pre každú zložku 8 bitov

```
10011101  
01101100  
11010100
```

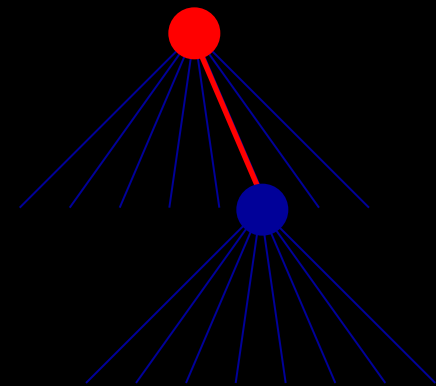
Sme v koreni stromu

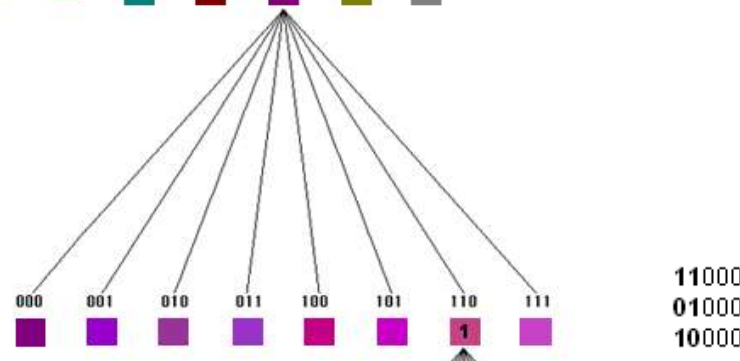
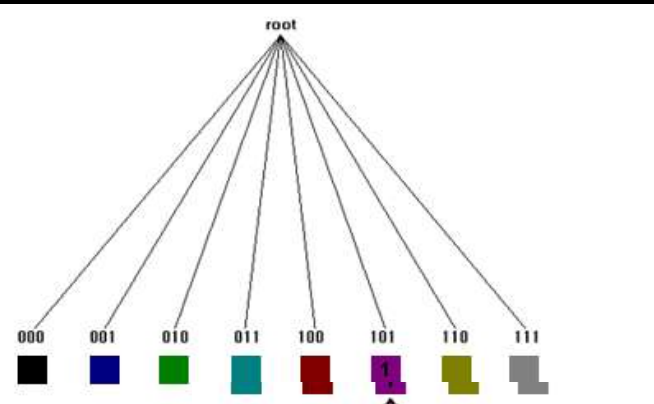
Vezmeme najvýznamnejší bit z každej zložky

- Zložíme ich – dostaneme číslo 0-7, čiže index nasledovníka (ak neexistuje, vytvoríme ho)

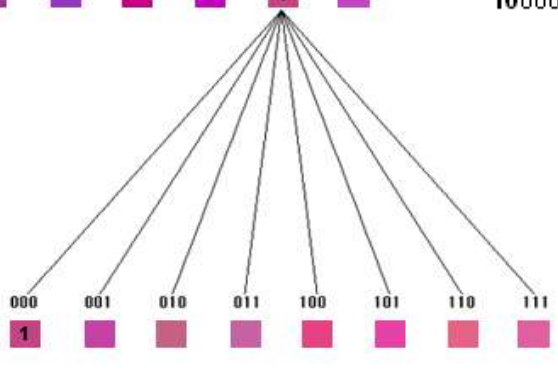
Pokračujeme na nasledovnej úrovni s ďalšími bitmi z RGB

$$101_2 = 5$$

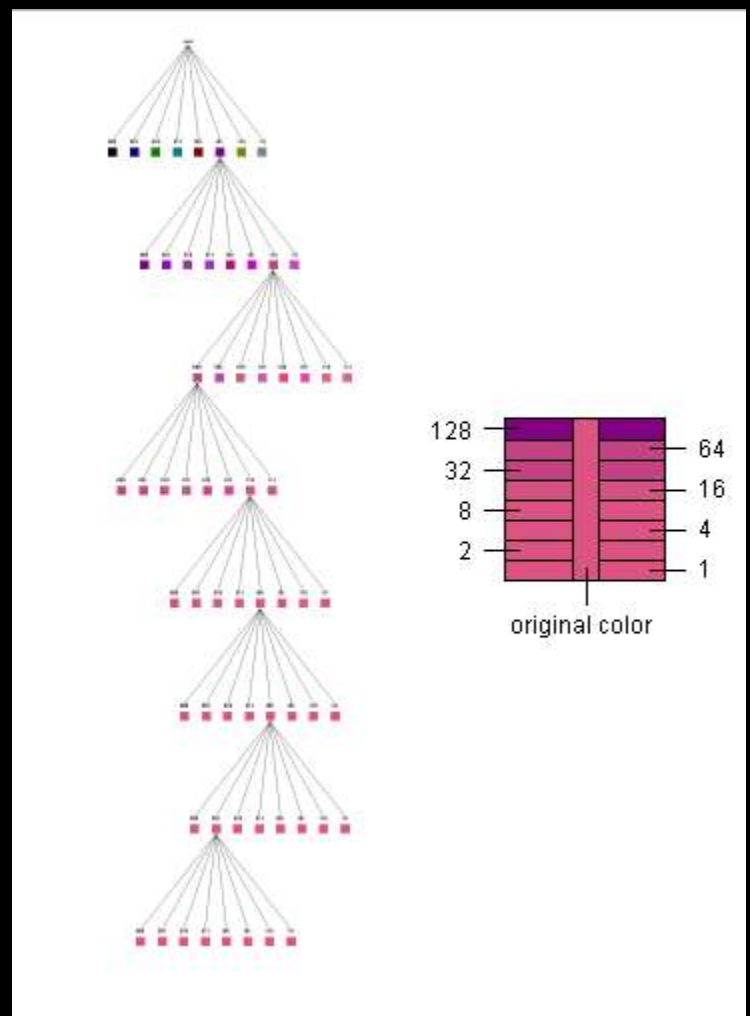




11000000
01000000
10000000



11011100
01010001
10000011



Algoritmus Octree

Strom môže mať maximálne 8 úrovní

Keď prídeme na koniec, prirátame hodnoty R,G,B zložiek a zvýšime početnosť

Prvých K rôznych farieb je reprezentovaných presne, potom sa začínajú farby zlučovať

Ak by mala byť vložená farba $K+1$, nájdeme uzol, ktorý má sumu početností u detí najmenšiu a zredukujeme ho

Algoritmus Octree

Redukcia:

Zrátame R, G, B zložky detí, početnosti detí a nastavíme uzol na 0 detí.

Ďalej pridávame farby.

- Na konci máme maximálne K listov.

Reprezentatívne farby vyrátame ako priemer farieb v listoch. R, G, B zložka / početnosť.

Algoritmus Octree

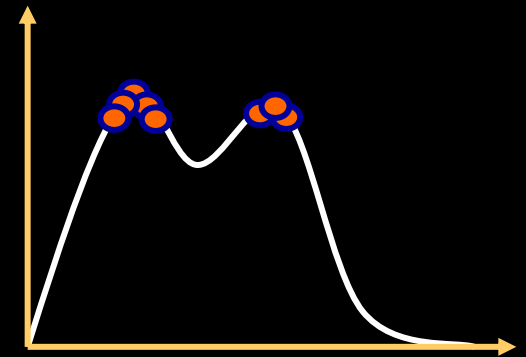
Kvantizácia – opätovné prejsenie stromu a priradenie reprezentatívnej farby

Veľmi pekné interaktívne vysvetlenie

<http://www.cs.wfu.edu/%7Eburg/nsf-due-0340969/interactive/Octree.htm>

Algoritmus popularity

- 3D histogram
- Rovnomerné rozdelenie RGB na kocôčky 4x4x4
Určíme ktorá farba obrazu kam patrí
- Vyrobitme priemerných reprezentantov
- Vyberieme **K** „najokývanejších“ regiónov
- Ostatné regióny namapujeme na najbližší región v palete



Algoritmus popularity

Môže sa stať, že dôležitú, ale málo zastúpenú farbu vynechá



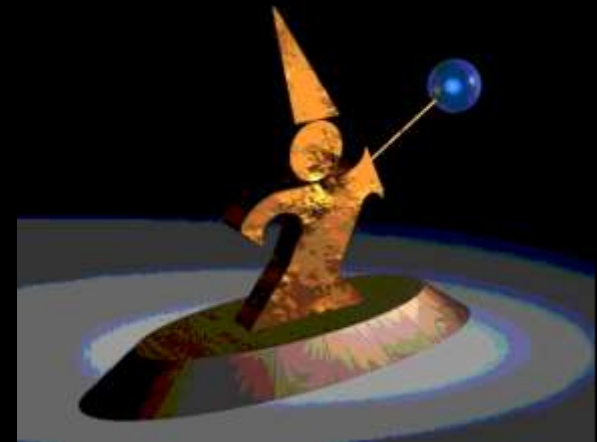
Algoritmus popularity vs. uniformný

Lepšie výsledky

Väčšie nároky na priestor

Časovo náročnejšie

Závisí od obrazu, či výsledky budú dobré



Algoritmus popularity - príklady



2



16



4



256

Originálny algoritmus diverzity

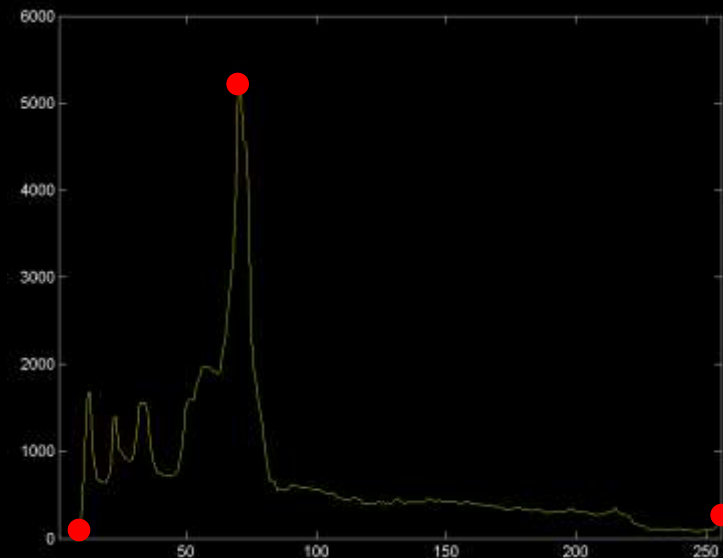
Vyrob histogram obrazu

Vyber farbu najväčším výskytom

Repeat

vyber nevybratú farbu, ktorá je najďalej od
všetkých vybratých farieb

Until nie sú vybraté všetky farby



Originálny algoritmus diverzity



2



4



16



256

Modifikovaný algoritmus diverzity

Prihliada aj na popularitu farieb

Vyrob histogram obrazu

Vyber farbu najväčším výskytom

2. – 10. farba: použi normálny algoritmus diverzity

Repeat

- (a) vyber farbu podľa popularity

- (b) vyber farbu podľa diverzity

Until nie sú vybraté všetky farby

Pri veľkom počte zhlukov môžeme hneď od začiatku striedať (a) a (b)

Modifikovaný algoritmus diverzity



2



16



4



256



original



Diverzita
4 farby



Diverzita
+ k-means



original



Popularita
16 farieb



Popularita
+ k-means



4

8



32



256



Inverse colormaping

- Proces ktorý mapuje farby obrazu do limitovanej množiny reprezentatívnych farieb
- Reprezentatívne farby definované
 - kvantovacími algoritmami
 - Farebnou paletou
- Pre každú farbu c nájdeme najbližšieho reprezentanta $Q(c)$

$$C \rightarrow \{c_1, \dots, c_K\}$$

$$c \rightarrow Q(c) = \arg \min_{z \in \{c_1, \dots, c_K\}} \|z - c\|$$

Inverse colormaping

Primitívny algoritmus

- Pre každú farbu c obrazu prehl'adáme všetkých reprezentantov
- Obraz 256×256 , paleta farieb 256 -> **256^3 porovnaní**

Inverzné mapovanie pomocou 3D Voronoiiovho diagramu

- Thomas
- Diskrétny Voronoiiov diagram definovaný reprezentatívnymi farbami palety.
 - Definovaný p -bodmi = rozdelenie obrazu na p buniek
- Kódované 3D pole integerov
 - Farba a index na najbližšieho reprezentanta
- Raz vypočítame diagram potom mapovanie bez výpočtov

Inverzné mapovanie pomocou 3D Voronoiiovho diagramu

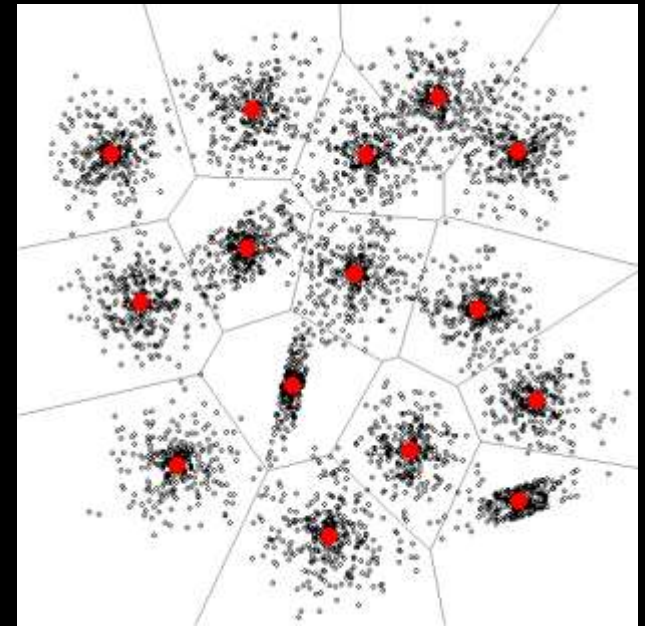
Nevýhody

Konštrukcia 3D Voronoiovho diagramu - výpočtovo náročné
3D diagram v RGB = 256^3 indexov

Riešenie

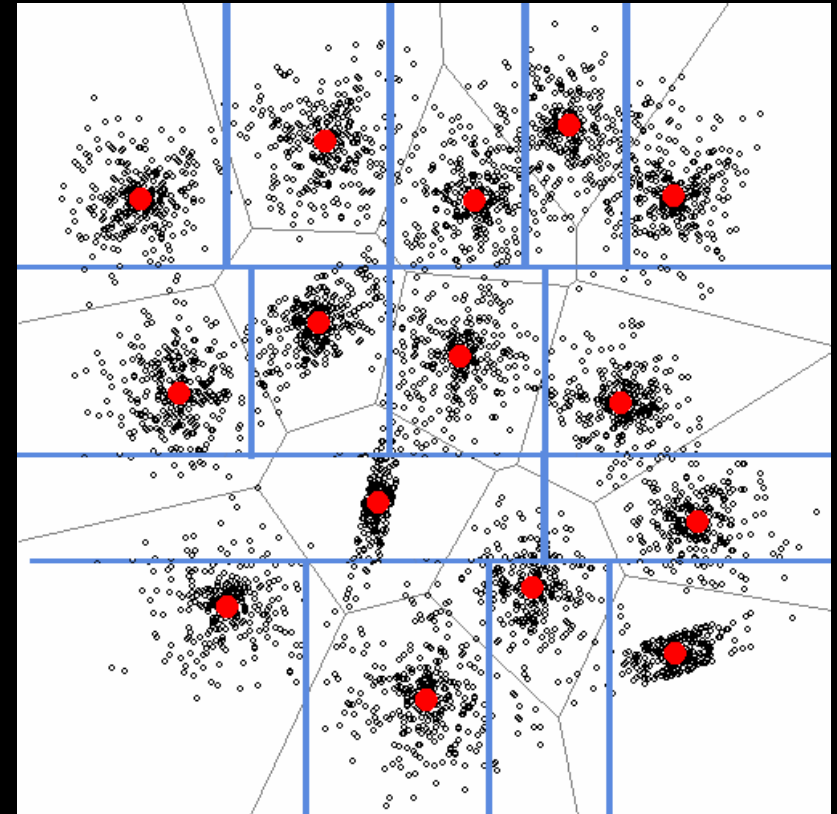
Thomas odstráni 3 najmenej dôležité bity z každého R,G,B
komponentu – 32^3 indexov

Euklidovská metrika?
RGB vs. **Lab** vs. HSV



Rozdelenie priestoru

Pevné prahy, nie nutne pravidelné
Jednoduché, ale nie presné

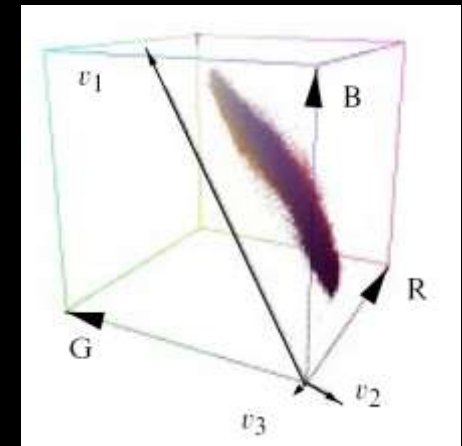


Inverzné mapovanie pomocou 2D Voronoiiovho diagramu

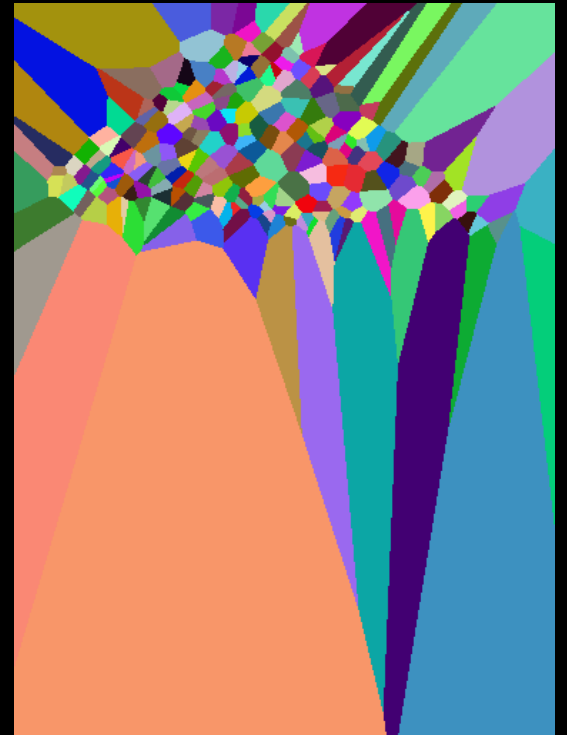
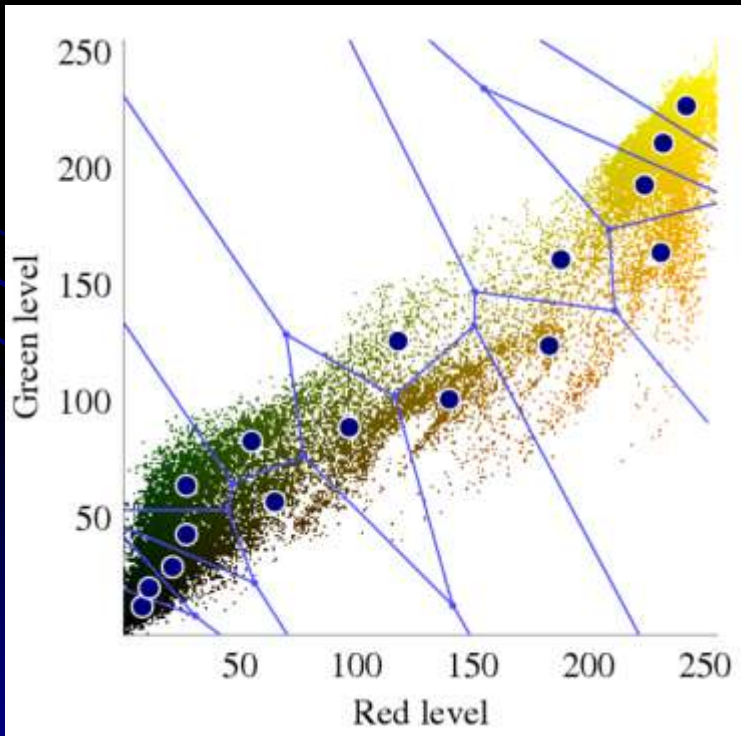
- Brun
- projekcia 3D Voronoiovho diagramu do 2D
- Prvé 2 vlastne vektory

$$Q(c) = \arg \min_{i \in D_I[V_I(p(c))]} \|c - c_i\|^2$$

- V_I – 2D diagram
- D_I – Delaunayov graf



The set of colors of the Lenna test image and the 3 eigenvectors (v_1 ; v_2 ; v_3) of its covariance matrix. The length of each vector is proportional to its eigenvalue



Ďalšie metódy

Simulované žíhanie
Genetické algoritmy

Hexagonálna mriežka
Polárna kvantizácia

