M Ű E G Y E T E M 1 7 8 2

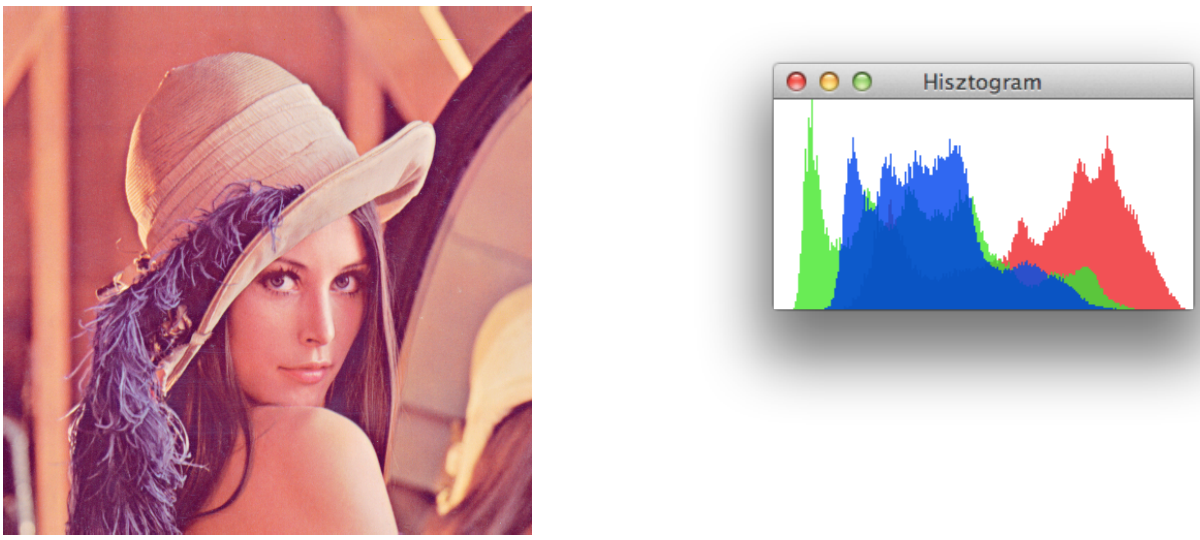
KÉPFELDOLGOZÁS
(BMEGEFOAMK1)

Színstatisztika készítés

Budapest, 2011. december 8. Ébert Dávid (W81GPX)

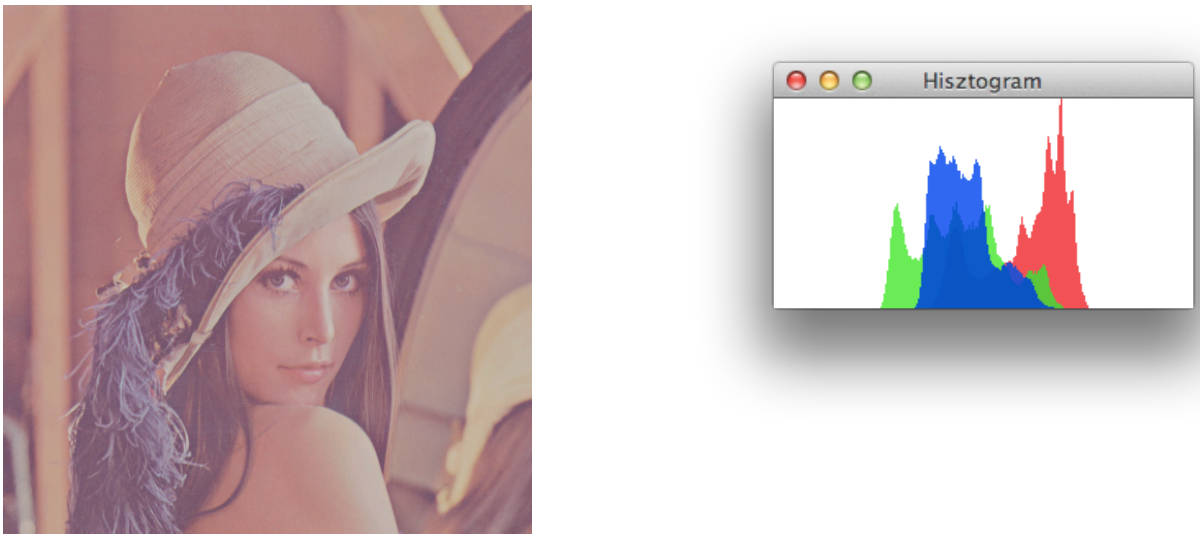
A hisztogram egy digitális kép tónusainak eloszlását ábrázolja, segítségével gyorsan megállapítható, hogy a kép helyesen lett-e kiexponálva, illetve a globális kontrasztra is tudunk következtetéseket tenni. Megszerkesztése egyszerű: a lehetséges világosságkódok függvényében – ami a legtöbb ma használt rendszer esetén színcsatornánként 8 biten kerül tárolásra, azaz értéke 0 és 255 közé esik – ábrázoljuk a képpontok relatív gyakoriságát.

Néhány jellegzetességet Lena Söderberg svéd Playmate fotóját használva mutatok be, aki 1972-ben Miss November volt.



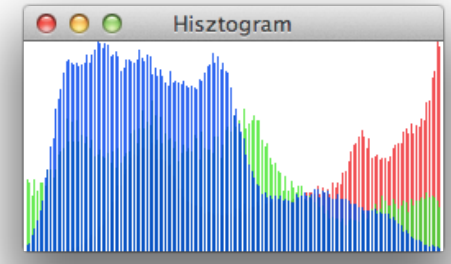
1. ábra

Az eredeti kép hisztogramja (1. ábra) kiegyensúlyozott: nincsenek se bebukott (a diagram bal széle), sem beégett részek (a diagram jobb széle), továbbá a kép kontrasztos, a teljes rendelkezésre álló dinamikai tartományt kihasználja.



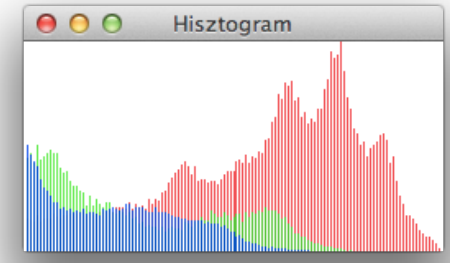
2. ábra

A kontraszt globális csökkentése esetén az egész diagram „összenyomódik” (2. ábra), így a finom részletek, különbségek elvesznek – csak közepes tónusok vannak a képen, nincs kihasználva a csatornánkénti 8 bit. Az ilyen képeket feldolgozáshoz kontrasztosíthatjuk a küszöbértékek megkeresésével, majd az értékek arányos elhúzásával. Az elveszett részletek értelemszerűen nem nyerhetőek vissza, így a kontrasztosított kép hisztogramjában lyukak lesznek.

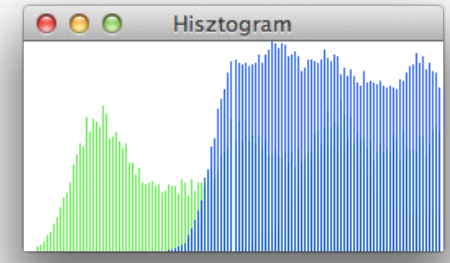


3. ábra

Ez a jelenség figyelhető meg a 3. ábrán is, ahol az eredeti kép kontrasztját növeltük. Ha a forrás képünk rendesen kihasználja a rendelkezésre álló biteket (nem úgy, mint a 2. ábrán), akkor ilyen esetben a sötét és a világos részletek elvesznek: vagy teljesen fekete vagy teljesen fehér képpontok kerülnek a helyükre.



4. ábra



5. ábra

Az alul- (4. ábra) és túlexponált (5. ábra) képek szintén könnyen felismerhetőek a hisztogramjuk alapján. Az előbbi esetben a pixelek túlnyomó része a grafikon bal, míg utóbbiban a jobb oldalára tömörülnek.

A sok bebukott és beégett pixel miatt ezen hisztogramokban a 0 és 255 értéket felvevő pixelek számát nem tüntettem fel.

Egy hisztogram megrajzolásának legegyszerűbb módja, ha egy kép minden egyes pixelének lekérdezzük mindhárom (vörös, zöld, kék) világosságkódját, összeszámoljuk őket, majd ezen adatok alapján létrehozzuk a diagramot. Az értékek lekérdezése előtt beolvassuk a fájlt, s létrehozzuk a kiértékeléshez szükséges változókat:

```
/** beolvassuk a kivalasztott fajlt */
kep = ImageIO.read(file);
/** mereteket kiolvassuk */
int width = kep.getWidth();
int height = kep.getHeight();
/** relativ gyakorisag szamitasahoz az osszes keppont szama */
Rajz.osszes = width * height;
/** 3 tomb a 3 csatornanak */
int[] piros;
piros = new int[256];
int[] zold;
zold = new int[256];
int[] kek;
kek = new int[256];
```

Majd két for ciklussal végigmegyünk az összes képponton, lekérdezzük a világosságkódokat, s mindegyiket külön-külön számláljuk:

```
for (int i=0; i < width; i++) {
      for (int j=0; j < height; j++) {
            /** sorban vegigmegy az oszlopokon, s lekeri
             *  mindegyik pixel szinet
             */
            int RGB = kep.getRGB(i, j);
            /** igy adja vissza, biteket kell shiftelni */
            int R = (RGB >> 16) & 0xff;
            int G = (RGB >> 8) & 0xff;
            int B = (RGB) & 0xff;

            /** szamoljuk, hogy adott szin hanyszor szerepel a kepen */
            piros[R]++;
            zold[G]++;
            kek[B]++;
      }
}
```

A hisztogram függőleges tengelyén az adott értékű képpontok relatív gyakoriságával arányos értékeknek kell lennie, tehát az eddig kapott értékeket el kell osztani az összes képponttal – mivel ez konstans minden értékre nézve, az ábrázolás megkönnyítése érdekében más

konstanssal is leoszthatjuk a képpontok számát. Ha szeretnénk, hogy minden hisztogram meghatározott méretű legyen, az a legegyszerűbb, ha a legnagyobb számban előforduló értéket válasszuk a grafikon maximumának.

```
/** Ez a ket ertek atirhato! :)
 *  a magassag a grafikon max. magassaga
 *  a szelesseg azt adja meg, hogy egy "vonal" (=szinertek)
 *  milyen szeles oszlopot jelentsen
 */
public static final int magassag = 128;
public static final int szelesseg = 1;

/** megkeressuk a tomb maximumat - ez pont toltse ki a kapott helyet */
int maxertek = 0;
for (int i=0; i<768; i++) {
     if (rszinek_l[i]>maxertek) {
          maxertek = rszinek_l[i];
     }
}
float oszto = (float)maxertek/magassag;

/** atkergetjuk inkabb egy uj tombbe az egeszet, ugy lesz a jo :)
 *  igy meretezzuk at a kapott magassagnak megfelelove az egeszet
 */
int[] rszinek_korr;
rszinek_korr = new int[768];
for (int i=0; i<768; i++) {
     rszinek_korr[i] = Math.round(rszinek_l[i]/oszto);
}
```

Ha ezzel is kész vagyunk, akkor pedig már csak ki kell rajzolnunk a grafikont.

A mintaprogram Javában készült, így platformfüggetlen, s az Image I/O-nak köszönhetően BMP, GIF, JPEG, PNG és TIFF formátumú képeket is képes beolvasni.

Indítás után a „Kép betöltése…” gombra kell kattintanunk. Ekkor megjelenik egy fájlválasztó ablak, ahol ki kell választanunk egy képet. A kép betöltése után a program kielemzi a képpontokat, s feltölti a kirajzoláshoz szükséges tömböket – ezt a program alján lévő szövegdobozban tudjuk figyelemmel követni. A „Tömbök feltöltve.” sor megjelenése után a „Rajzolás…” gombra kattintva megjelenik egy új ablak, rajta a hisztogrammal.

Újabb kép betöltéséhez (és a hisztogram kirajzolásához) nem kell kilépnünk a programból, sőt a már megjelenített hisztogramok megmaradnak, így egymás mellé tehetjük őket összehasonlítás céljából.

Felhasznált források és segédletek

A program Eclipse-ben készült: http://www.eclipse.org

Java SE dokumentáció, Image I/O: http://docs.oracle.com/javase/6/docs/technotes/guides/imageio/spec/apps.fm1.html

Tanszéki diasorok: http://canopus.mogi.bme.hu/letoltes/KEPFELDOLGOZAS%20(BSC)/

Lenna sztenderd teszt fotó: http://www.lenna.org

Saját órai jegyzet

A program forráskódja: http://dl.dropbox.com/u/39760/kepfel/forras.zip

A program: http://dl.dropbox.com/u/39760/kepfel/histogram.jar

Ez a dokumentum: http://dl.dropbox.com/u/39760/kepfel/histogram.pdf