POČÍTAČOVÉ ZPRACOVÁNÍ STATICKÉHO OBRAZU

Pavel Pokorný

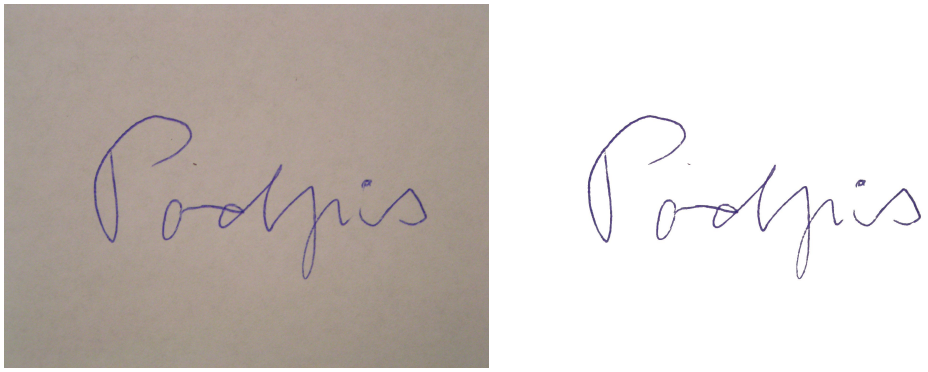
Počítačová grafika je matematická archeologie.

1. Úvod

Ukážeme si použití některých jednoduchých matematických postupů pro počítačové zpracování obrazu. Naším cílem bude vyčistit pozadí statického obrazu. Postup a výsledek ukážeme na dvou příkladech: rukou psaný podpis, kdy stačí použít pouze jednu barevnou složku obrazu, a složitější případ, kdy musíme vyhodnotit více barevných složek. K tomu s výhodou použijeme počítačový algebraický systém *Mathematica*, který umožňuje kombinovat matematické a grafické operace na uživatelsky přívětivé úrovni.

2. Příklad 1: Vyčištění pozadí podpisu

Uvažujme podpis na nečistém pozadí, viz obr. 1 vlevo.

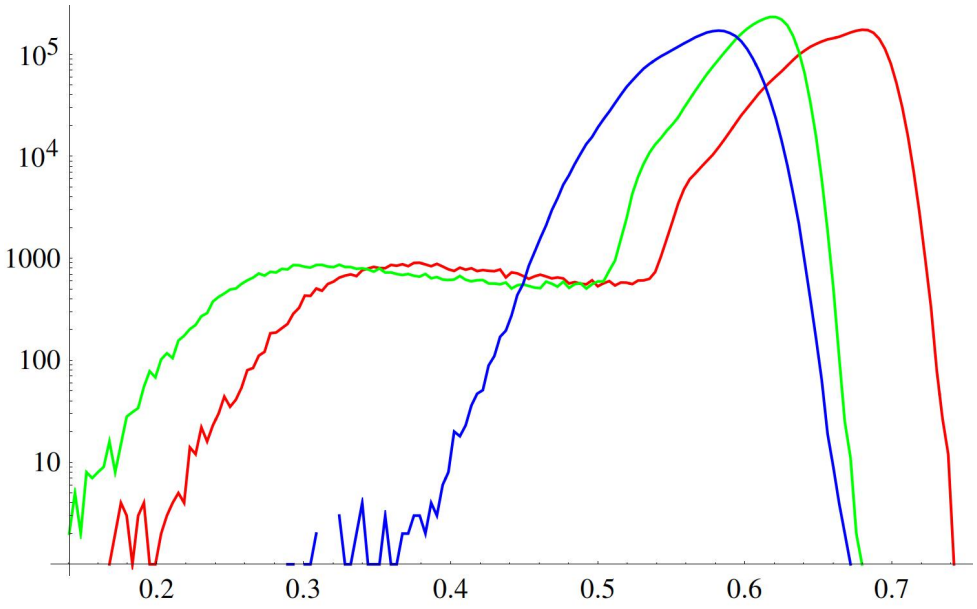


Obrázek 1: Vlevo: příklad podpisu na nečistém pozadí. Vpravo: tentýž obrázek s čistým bílým pozadím po použití transformace.

Takový obrázek je v počítači uchován jako matice (zde matice 2048 x 1536) pixelů, kde každý pixel je trojice čísel z intervalu $\langle 0, 1\rangle$ udávajících intenzitu červené (red), zelené (green) a modré (blue) barvy daného pixelu.

Nejdříve si příkazem `ImageLevels` připavíme histogram obrázku, tedy závislost četnosti na intenzitě pro všechny tři barevné složky, a vykreslíme jej příkazem `ListLogPlot` viz obr. 2.

Histogram na obr. 2 ukazuje, že modrá složka je obsažena s velkou intenzitou ve většině pixelů. Pixely s malou intenzitou modré se v obrázku téměř nevyskytují. Oproti tomu červená a zelená složka jsou obsaženy s velkou intenzitou ve velkém počtu pixelů (pozadí obrázku), ale také ve velkém (i když menším) počtu pixelů s malou intenzitou (modrá čára podpisu). Toho lze využít pro identifikaci pixelu pozadí. Nevýrazné minimum histogramu červené barvy nastává pro hodnotu intezity červené přibližně 0,5. Můžeme tedy navrhnout



Obrázek 2: Histogram nečistého podpisu na obr. 1.

transformaci v podobě funkce $f : R^3 \to R^3$, která pixely, jejichž červená složka je větší než 0,5 (pozadí), nahradí trojicí jedniček (čistá bílá) a ostatní pixely ponechá beze změny `f[{r_, g_, b_}] := If[r > 0.5, {1, 1, 1}, {r, g, b}]`. Tuto funkci použijeme na obrázek příkazem `ImageApply` a výsledek (obr. 1 vpravo) uložíme příkazem `Export`.

Máme-li výchozí obrázek v dostatečném rozlišení, můžeme použít funkci `Blur` pro částečné potlačení šumu. Celý program pak může vypadat např. takto

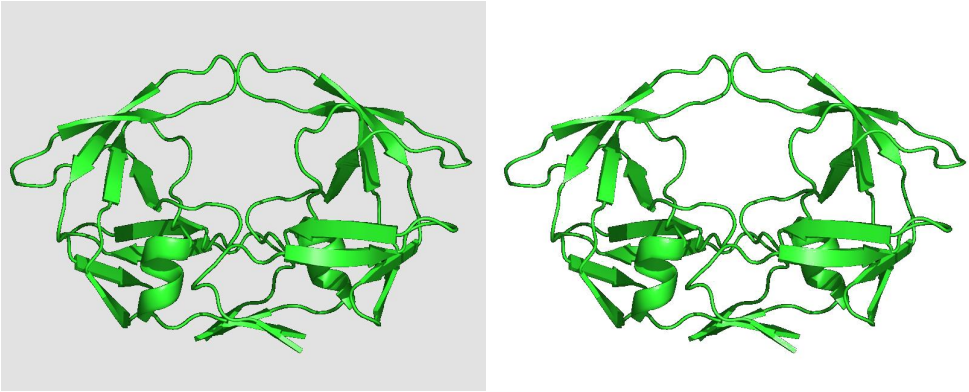
```
s = Import["spinavy1.jpg"];
ss= Blur[s];
ListLogPlot[ImageLevels[ss], Joined->True, PlotStyle->{Red,Green,Blue}]
f[{r_, g_, b_}] := If[r > 0.5, {1, 1, 1}, {r, g, b}];
c = ImageApply[f, ss]
Export["cisty1.jpg", c];
```

Nezanedbatelnou výhodou výše uvedeného postupu je výrazné zmenšení objemu dat. Původní podpis na nečistém pozadí byl uložen v souboru o velikosti přes 2 MB, vyčištěný podpis má objem 81 kB.

3. Příklad 2: Použití vícerozměrného histogramu

Jako příklad, kdy výše uvedený způsob nestačí, se pokusíme vyčistit pozadí na obrázku proteázy z viru HIV, viz obr. 3. Potíž je v tom, že zelené části obrazu se vyskytují s různými hodnotami jasu a sytosti až skoro do bílé a přesto je potřeba je odlišit od pozadí.

Pro pokročilejší zpracování obrazu je vhodné převést obraz na číselná data příkazem `ImageData`. Tím dostaneme matici pixelů. Příkazem `Flatten` ji převedeme na posloupnost



Obrázek 3: Stužkový model proteázy z viru HIV. Vlevo: s nečistým pozadím. Vpravo: tentýž obrázek s čistým bílým pozadím po použití transformace.

pixelů.

V předcházejícím příkladě jsem pracovali se třemi samostatnými 1-dim histogramy pro jednotlivé barevné složky. Každý takový histogram lze považovat za reálnou funkci jednoho reálného argumentu, kde argumentem je intenzita určité barvy a funkční hodnotou je hustota výskytu této intenzity v obraze.

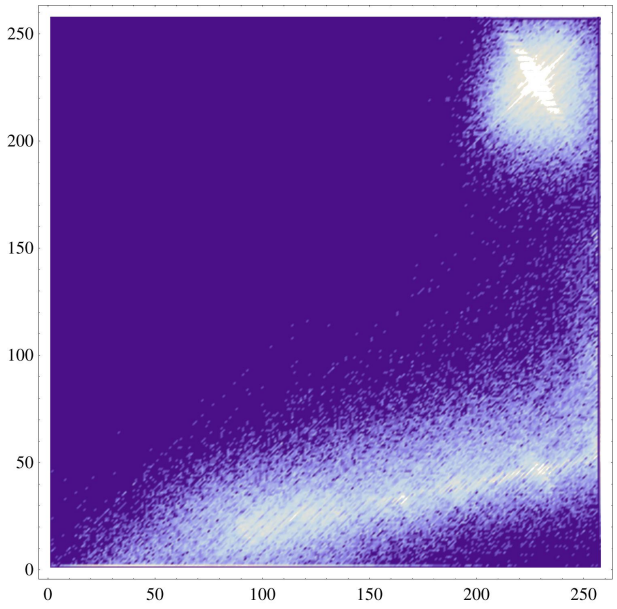
Data z barevného obrázku by bylo vhodnější popsat 3-dim histogramem, tedy reálnou funkcí třech argumentů (r, g, b). Argumenty jsou tři intenzity červené, zelené a modré barvy a funkční hodnotou (p) je opět hustota výskytu. Pro pohodlnější práci se omezíme na 2-dim histogram, který dostaneme průmětem z 4-dim prostoru (r, g, b, p) do 3-dim prostoru (r, g, p). Hodnoty p zobrazíme pomocí stupně jasu a vystačíme s 2-dim zobrazovacím zařízením (stránky tohoto sborníku). Tento průmět snadno provedeme tak, že vybereme první dvě složky (r, g) ze třech složek (r, g, b). Příkazem `BinCounts` pak vytvoříme 2-dim histogram, (omezíme se např. na 256 hodnot intenzit každé barvy). Příkazem `ListDensityPlot` jej zobrazíme, viz obr. 4.

Při velice pečlivém vyhodnocení histogramu zjistíme, že vpravo od velkého ostrovu, který odpovídá pozadí, existují malé ostrůvky, které odpovídají světlezeleným částem obrazu, které nechceme ztotožnit s pozadím. Pro jejich správné odlišení bychom nevystačili s testováním jediné barevné složky. Mnohem lepšího výsledku dosáhneme vyhodnocením jak červené (r), tak zelené (g) složky pomocí funkce

```
f[{r_, g_, b_}] := If[r > 0.5 && g < 0.95, {1, 1, 1}, {r, g, b}];
```

Tuto funkci opět použijeme na obraz příkazem `ImageApply` a dostaneme obrázek s krásným čistým pozadím, viz obr. 3. Celý úkol lze provést tímto programem:

```
s = Import["spinavy2.jpg"]
id = ImageData[s];
idf = Flatten[id, 1];
```



Obrázek 4: Průmět 3-dim histogramu z prostoru (r, g, b, p) do 3-dim prostoru (r, g, p). Hodnoty hustoty p jsou vyneseny pomocí stupně jasu. Ostrov vpravo nahoře odpovídá světlešedému pozadí, protáhlý pás odpovídá zeleným částem s různým jasem.

```
idf2 = idf[[All, {1, 2}]];
bc = BinCounts[idf2, 1/256, 1/256];
h2 = ListDensityPlot[ArcSinh[bc]]
f[{r_, g_, b_}] := If[r > 0.5 && g < 0.95, {1, 1, 1}, {r, g, b}];
c = ImageApply[f, s]
Export["cisty2.jpg", c];
```

Použití funkce `ArcSinh` umožní zobrazit současně malé i velké hodnoty v jediném obrázku.

4. Závěr

Ukázali jsme si některé jednoduché matematikcé operace pro zpracování statického obrazu s využitím počítačového algebraického systému *Mathematica*. Výsledkem je nejen vyšší kvalita obrazu, ale také menší objem dat.