FalseVisir

Obsah

Úvod

Program pro automatické vytvoření obrazu ve falešných barvách spojením snímků ve viditelném a infračerveném světle.

https://github.com/almaavu/falsevisir

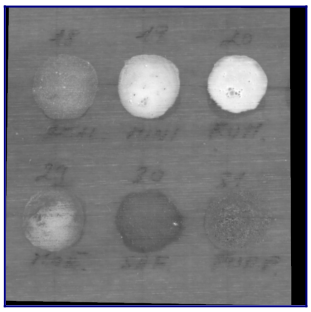
Zobrazení ve falešných barvách je technika zpracování obrazu používaná při průzkumu uměleckých děl (např. závěsných obrazů, nástěnných maleb, polychromovaných plastik). Pro vyhodnocení je vhodné porovnat snímky získané infračervenou reflektografií (IRR) se snímky ve viditelném světle (VIS). Spojení obou obrazů do snímku ve falešných barvách může pomoci při studiu podmalby nebo pro identifikaci některých pigmentů. [1], [2].

Ve výsledném obrazu jsou RGB kanály využity takto:

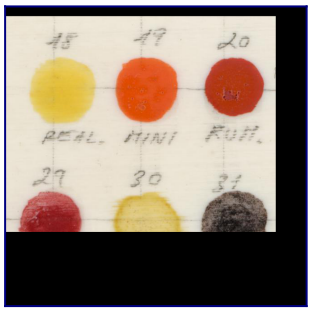
```
IRR   -> R
Vis R -> G
Vis G -> B
Vis B ->
```

Příklad složení snímku v infračerveném světle s červeným a zeleným kanálem snímku ve viditelném světle:

IRR

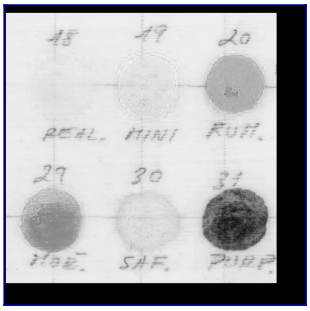


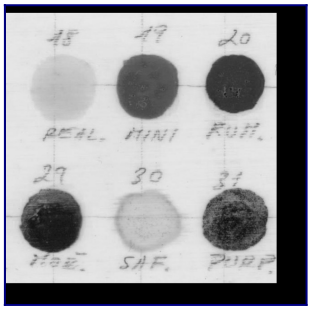
VIS

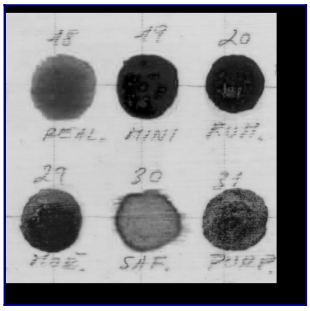


VIS R

G B



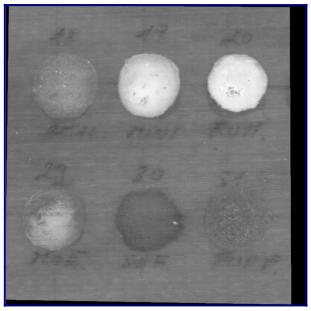


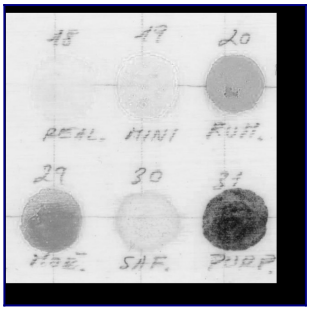


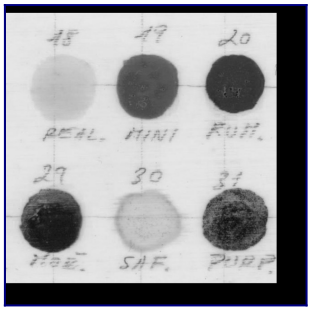
False

color

R G B

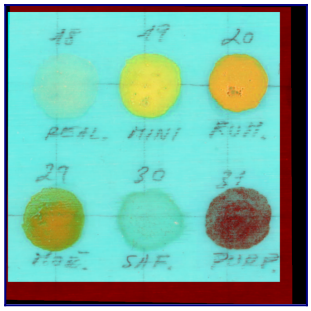






False

color



Pro složení snímků jsou obvykle využívány grafické editory (Adobe Photoshop, GIMP, ...). Snímky jsou zobrazeny přes sebe, pro přesný překryv je obvykle je nutné je transformovat a napravit tak zkreslení způsobené rozdíly v geometrii zobrazení (natočení, perspektivní zkreslení) a použitých objektivech (soudkovité zkreslení).

Pokud mají oba snímky podobné rysy, je možné transformaci a následné složení do falešných barev provést automaticky programem FalseVisir. To je výhodné zejména při zpracování většího počtu snímků.

Instalace:

Instalace programovacího jazyka **Python3**

```
https://www.python.org/downloads/
```

Instalace programu **FalseVisir**

```
python -m pip install --upgrade
git+https://github.com/almaavu/falsevisir.git#egg=falsevisir
```

Instalace knihoven:

```
python -m pip install --upgrade requirements.txt
```

- numpy
- matplotlib
- scikit-image
- scipy
- imageio

Použití:

Příkazový řádek

Program lze spustit z příkazového řádku se zadáním cesty ke vstupním souborům - obrázku ve viditelném a infračerveném světle.

```
python -m falsevisir "vis_soubor.jpg" "ir_soubor.jpg"
```

Skript je možné spustit i bez instalace:

```
python falsevisir.py "vis_soubor.jpg" "ir_soubor.jpg"
```

Vstupní data:

Cesta a název souboru snímku ve viditelném světla (formát RGB) a infračerveného snímku (formát RGB nebo stupně šedé). Snímky by měly být oříznuté na přibližně stejný výřez (bez přehnaně širokých okrajů). Mohou být různě pootočené (viz ukázky).

Hromadné zpracování:

Program **falsevisir_batch.py** je určený pro hromadné zpracování většího počtu obrázků. Program načte snímky ze zvolených složek a zpracuje páry souborů podle ID souborů, které musí být uvedeno na začátku názvu následované podtržítkem. Zpracuje např. soubory

"a001_vis_image.jpg" a "a001_ir_image.jpg".

```
python falsevisir_batch.py "samples/vis_samples/" "samples/ir_samples/"
```

Jupyter notebook

Program je také možné spustit interaktivně v prostředí Jupyter notebook, to je výhodné v případě, kdy je potřeba sledovat jednotlivé kroky zpracování obrazů, např. při úpravě nastavení konfigurace programu.

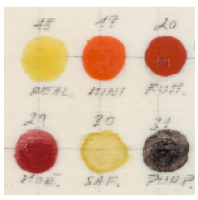
```
jupyter notebook falsevisir_jupyter.ipynb
```

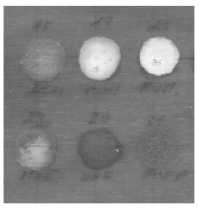
Změna konfigurace v prostředí Jupyter notebook:

In [3]:

```
from falsevisir import *
from pprint import pprint

CFG['preprocess_images']['blur_sigma'] = 1
CFG['preprocess_images']['normalize'] = 0

CFG['downsize'] = 400

CFG['extract_features']['patch_size'] = CFG['downsize'] // 8

# pprint(CFG)

im_paths = "samples/vis_samples/a002_palette.jpg", "samples/ir_samples/a002_s002__TL.jpg"
vi_path, ir_path = [Path(fp) for fp in im_paths]

#%% Load images
vi_image, ir_image = [load_image(fp) for fp in (vi_path, ir_path)]

print(info(vi_image))
print(info(ir_image))
#show_images((vi_image, ir_image))

#%% Resize to same height
vi_image, ir_image = resize_images((vi_image, ir_image))

print(info(vi_image))
print(info(ir_image))
show_images((vi_image, ir_image))
```

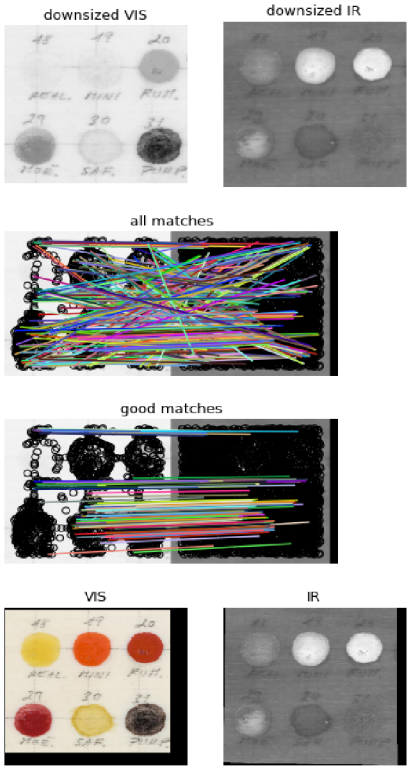
```
shape: (582, 600, 3) dtype: float64 min--max: 0.000--1.000
shape: (1311, 1287, 3) dtype: float64 min--max: 0.220--1.000
shape: (582, 600, 3) dtype: float64 min--max: 0.000--1.000
shape: (582, 571, 3) dtype: float64 min--max: 0.271--0.998
```





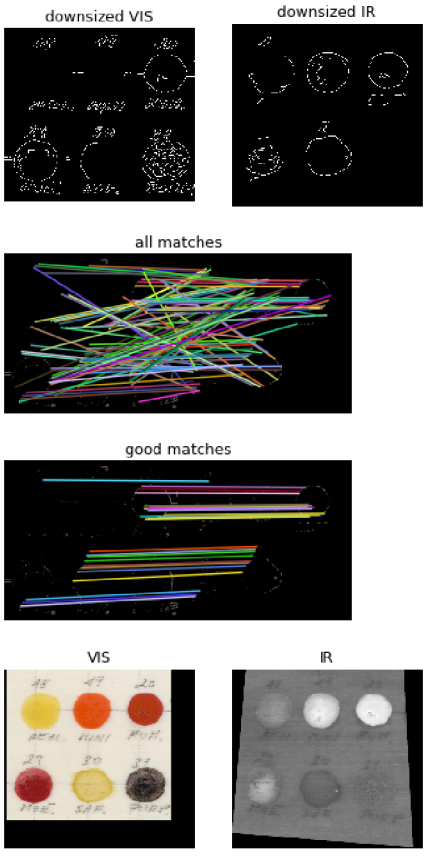
Kontrola průběhu transformace:

In [2]:
```
#%% Warp images
vi_image_w, ir_image_w = warp_images(vi_image, ir_image, show=True)
```



Nezdařená transformace při nastavení parametru preprocess_images - edge:

In [10]:
```
CFG['preprocess_images']['edge'] = 1

#%% Warp images
vi_image_w, ir_image_w = warp_images(vi_image, ir_image, show=True)
```



Funkce programu:

- Změna velikosti obrázků na stejnou výšku
- Transformace snímků pro přesné překrytí - oprava rozdílů v natočení, perspektivním zkreslení, zkreslení různých objektivů apod. (obrázky musí mít podobné rysy, jinak může selhat) [3]
- Spojení obrazů do falešných barev (IRR-R-G)
- Prolnutí obrazů (50% IRR, 50% VIS)
- Uložení výsledků

Konfigurace:

Parametry jsou uloženy v globální proměnné CFG. Jejich úprava může být užitečná, pokud program nenajde správnou transformaci.

- *downsize*: výška zmenšeného obrázku v pixelech. Zmenšené snímky program používá pro urychlení výpočtu transformace. Změna velikosti může pomoci, když transformace selže. Vyšší hodnota vede k pomalejšímu výpočtu, výchozí hodnota: 500 pix.
- *preprocess_images*: Před výpočtem transformace je lze provést úpravu jasu a kontrastu ("normalize"), equalizaci histogramu ("equalize") nebo detekci hran ("edge")
- *extract_features*: parametry funkce pro výběr bodů
- *ransac*: parametry Ransac algoritmu použitého pro výběr odpovídajících dvojic bodů
- *match*: parametry match algoritmu

Výchozí konfigurace:

```
CFG = {
'downsize': 500,
'extract_features': {'method': 'HARRIS',
                     'min_distance': 1,
                     'patch_size': 59,
                     'threshold_rel': 1e-07},
'irr_weight': 0.5,
'match': {'max_distance': 200},
'model_robust_param_limits': [[[-10, -1, -100], [-1, -2, -100], [-0.1, -0.02, 0]],
                              [[10, 1, 100], [1, 2, 100], [0.1, 0.02, 2]]],
'preprocess_images': {'blur_sigma': 2,
                      'edge': False,
                      'edge_high_threshold': 0.1,
                      'edge_low_threshold': 0.05,
                      'edge_sigma': 2,
                      'equalize': False,
                      'normalize': False},
 'ransac': {'max_trials': 10000, 'min_samples': 5, 'residual_threshold': 10}
}
```

Výstup:

- Snímek ve falešných barvách
- Snímek s překrytými snímky v IR a VIS světle (50 % IR + 50 % VIS)
- Transformované snímky v IR a VIS světle (se snímky lze dále pracovat, např. pro vytvoření prolnutí s jinými parametry)

Ukázka:

Zdrojové obrázky:

Výsledek:

Obraz ve falešných barvách

Prolnutí

Transformované snímky

Odkazy:

[1] https://en.wikipedia.org/wiki/False_color

[2] Cosentino, A. Identification of pigments by multispectral imaging; a flowchart method. herit sci 2, 8 (2014). https://doi.org/10.1186/2050-7445-2-8

[3] https://en.wikipedia.org/wiki/Affine_transformation

Software je výsledkem projektu NAKI II: Neinvazivní výzkum portrétních miniatur pro účely jejich datace, autentikace, prezentace a ochrany, číslo projektu: DG18P02OVV034