# Semestrální práce ZDO

Detekce kleštíků

Jiří Valeš

28. 5. 2021

# Data

Fotografie z anotací, stažena ve formátu jpg.

Data (fotografie) byla rozdělena na tři části:

- Učící (hledal jsem na nich optimální prahy, rozsahy hodnot jednotlivých popisů, apod.)
- Zkoušecí/optimalizační (zkoušení metod, zda dávají/nedávají smysl a hází rozumné výsledky, jejich optimalizování)
- Testovací (20 nepoužitých fotografií v předešlé tvorbě) - na github ve složce mytest

# Předpoklady

Detekce kleštíků stojí především na těchto jejich vlastnostech:

- velmi tmavé zbarvení v porovnání s ostatními objekty
- eliptický tvar
- všechny kleštíci podobně velcí

# Použité metody

## Prahování

Bylo použito jednoduché prahování ze šedotónového obrazu. Práh 0.2 byl nastaven podle "učícího datasetu" několika fotografií. Původní záměr byl zkusit prahovat více kanálů či skrz HSV, vytěžit vlastnost, že všichni kleštíci jsou tmavě hnědí. Nakonec ale lepší výsledky poskytovalo jen prahování ve stupních šedi.

## Vyplnění děr

Řada kleštíků se často leskne, především ve své prostřední části. Při prahování, tak dochází k tomu, že střed je mimo daný interval a tudíž je potřeba vyplnění těchto děr ke zpětnému zrekonstruování celé oblasti kleštíka.
Méně častěji dochází k lesku kleštíků i na jejich stranách, či různých překryvům kleštíků ostatním materiálem z úlu. Snažil jsem se takové případy řešit například morfologickými operace (dilatací, erozí), ale výsledky dané metody nezlepšili, tak jsem od nich ve výsledné verzi upustil.

## Popisy objektu

Po řadě zkoušení nalezení objektů pomocí detekce hran a jejich eliptické vlastnosti. Jsem přešel na čisté popisy vyprahovaných objektů. Na 20 zkušebních učících kleštících jsem stanovil detekci kleštíků na základě těchto vlastností a jejich rozsahů:

### Minimální rozdíl mezi plochou a konvexní plochou

Tvar kleštíka má menší tento faktor než většina ostatních objektů. Konkrétně počítáno pomocí vzorce (convexarea - area)/((convexarea+area)/2) a definován rozsah do 0.1

### Kompaktnost

Tvar kleštíka je velmi kompaktní. Nekompaktnost spočítána pomocí vzorce: (perimeter*perimeter)/area. Rozsah podle učících dat byl stanoven 13 - 21.

### Poměr hlavní a vedlejší osy

Kleštík má eliptický tvar. Pomocí vzorce pomer os = hlavní osa / vedlejší osa je počítán vztah těchto dvou os. Rosah této hodnoty byl určen 1.2-1.6

### Velikost (obsah)

Tři předchozí popisy velmi dobře popíší tvar kleštíka a vyselektují drtivou většinu neodpovídajících objektů. Zůstane však kromě kleštíků i pár drobných úlomků z úlu, které mají shodou okolností stejný tvar a jas jako kleštíci. Proto je nutné ještě provést selekci na základě velikosti objektu. Rozsah velikosti (hodnota popisu area) z učící množiny stanoven na 100 - 200. Počítá s fotografiemi v plus mínus podobné vzdálenosti (s rezervou), stejné velikosti, rozlišení. V případě, že by vstupní fotografie měly jiné parametry / jinou velikost hledaných kleštíků je zapotřebí tuto hodnotu přenastavit, jinak algoritmus selže (neautomatizováno, na základě anotací předpokládáno, že data jsou velmi podobná).

Pokoušel jsem se najít ještě další metody, kterými by se bezpečně určili kleštíci, například využít toho, že mají “tykadla” . Ale zjistil jsem, že například tato vlastnost není moc použitelná, neboť u řady těchto objektů nejsou na fotografiích ani pořádně patrná lidským okem, natož detekovatelná algoritmem. Navíc jsem nenašel metodu, jak to rozumně detekovat.
Také snaha využít homogenity (stejného celého povrchu) objektu selhávala z důvodu lesku. Nakonec jsem tedy zůstal u výše zmíněných metod a podmínil jejich parametry, že se jedná o kleštíka.

# Použité knihovny

- Skimage
- Numpy

- Matplotlib
- Cv2

# Testování

Pro testování bylo použito 20 nepoužitých náhodně vybraných fotografií. Protože na všech byly kleštíci, tak na 10 z nich byly manuálně umazány části s kleštíky, tak aby byla 10 členná množina fotografií bez kleštíků. Ve složce mytest fotografie 1-10 jsou s kleštíky, 11-20 jsou bez kleštíků. Na těchto 20 fotografiích proběhl test.

Výsledky:
True positive = 6
True negative = 6
False positive = 4
False negative = 4
**F1 skóre rovno 0.6**

Chybovost je velmi vysoká. Dala by se pravděpodobně snížit například prahem, kolik detekovaných kleštíků na fotografii musí být minimálně, aby byl posouzen jako “s kleštíky”. 3 ze 4 falešně pozitivních fotografií detekovaly údajného 1 kleštíka - ve skutečnosti to byl vždy úlomek velmi špatně rozeznatelný od kleštíka.
Vzhledem ke zvolené detekci na základě velmi jednoduchých popisů obecně větší chybovost bude mít algoritmus při větším překrytu prvků, většímu množství tmavých úlomků a naopak dobře bude detekovat kleštíky v řídčeji rozmístěném materiálu.

# Použité prostředí

Pro vývoj jsem používal jupyter notebook. Výsledný algoritmus jsem poté překopíroval na github na https://github.com/JiriVales/zdo2021-vales/.
Do souboru *main* jsem vložil metodu *varoadetektor* (vstupní parametr: adresa fotografie), která detekuje kleštíky na jedné fotografii a metodu *predict* se vstupními parametry: číslo prvního obrázku (od kterého se začíná), název adresáře, kde jsou fotografie, a pole data obsahující podle požadavků počet fotografií, kanály, výšku, šířku. Metoda *predict* projde všechny fotografie a vrátí kolik z nich je s kleštíky a zda více než 50% fotografií je pozitivních (s kleštíky).
V souboru *init* jsou pak vstupní parametry a volání metody *predict*.
Složka *mytest* obsahuje vytvořená testovací data.

# Limity

- Nerobustní
    - dost vztažené na vlastnosti konkrétních fotografií (při odlišném rozlišení/vzdálenosti/velikosti/osvětlení fotografie než v anotovaném souboru nutnost změnit i parametry, kterými se selektují kleštíci
- Závislé na “kompozici”

    - algoritmus neumí detekovat kleštíky překryté jinými úlomky, spolehá se především na dobře detekovatelné osamocenější objekty, což může být při větší koncentraci jiných objektů problém
- Malý statistický základ
    - ať už testování, tak i množiny, ze kterých byly určovány parametry nejsou tak velké, aby je šlo považovat za spolehlivé ze statistického hlediska, také nebylo pracováno s pravděpodobnostími rozděleními a statistickými metodami, které by algoritmu dali větší exaktnost
- Málo sofistikované
    - i přes snahu hledat větší množství metod k detekci kleštíků nakonec stejně ve výsledku stojí především na popisech objektu a možnosti prahování, nejsou zde žádné alternativní podpůrné metody

# Diskuze + Závěr

Výše jsou popsány metody použité ve finálním algoritmu včetně jejich limitací. Kromě těchto metod jsem v procesu vytváření algoritmu zkoušel i další jako morfologické operace, detekce hran, jiné druhy prahování, ale většinou jsem od nich upustil, neboť jsem je vyhodnotil jako slepou cestu. Vzhledem k charakteru dat objektů se ukázali popisy objektů jako celkem dobrá cesta, která lze dále vylepšovat pro větší flexibilitu algoritmu a větší úspěšnost. Zajímavé by bylo srovnání s neuronovými sítěmi či detekce například vzorů v obraze. Původně jsem se k nim chtěl také dopracovat a porovnat je s mými “primitivnějšími” metodami, ale nakonec jsem se k nim z časových důvodů nedostal a dosyta si užil popisů objektů. Algoritmus jsem především cílil na rozpoznávání evidentních osamocených kleštíků, rozhodl jsem se zanedbat schované kleštíky v mělu.
Nemohl jsem najít fotografie bez kleštíků, tak jsem si je pro otestování vytvořil překrytím varroa objektů bílými oblastmi.

Přikládám ukázku z algoritmu - výstup z testování (z jupyter notebook, ale na githubu v pythonu by mělo fungovat stejně):

```
In [6]: adresar = "mytest"
        startobr = 11
        pocetobr = 10
        vyska = 0 #nepracoval jsem s tim v algoritmu, predpokladam datasety s podobnymi vlastnosti (rozliseni, vzdalenost, velikost obr,
        sirka = 0 #nepracoval jsem s tim v algoritmu, predpokladam datasety s podobnymi vlastnosti (rozliseni, vzdalenost, velikost obr,
        barevne_kanaly = 3 #nepracoval jsem s tim v algoritmu, predpokladam datasety s podobnymi vlastnosti (rozliseni, vzdalenost, velil
        data = [pocetobr,vyska,sirka,barevne_kanaly]
        result = predict(startobr, adresar, data)
        print(result)

        ([10, 0, 0], 0, 4)
```

*Nalezené 4 falešně pozitivní z testovacích fotografií 11-20 ze složky mytest*

# Github

https://github.com/JiriVales/zdo2021-vales/, ve složce ZDO2021

# Zdroje

- přednášky/cvičení předmětu ZDO
- dokumentace uvedených knihoven
- https://realpython.com/ - některé články k Image Segmentation
- anotovaná data
- https://machinelearningmastery.com/ - výpočet f1 score