PREPOZNAVANJE AKTIVNOSTI ČEBEL NA PANJSKEM ŽRELU S POMOČJO STROJNEGA VIDA IN DRUGIH METOD

Področje: **Računalništvo in informatika**

Vrsta naloge: **Raziskovalna naloga**

Dijaki:

Oskar Rotar, 3.A

Tian Vesel, 3.A

Maks Žnidaršič, 3.A

Mentorica: **Mag. Darja Silan - Gimnazija Jožeta Plečnika Ljubljana**

Somentor: **Dr. Janko Božič - Biotehniška fakulteta Ljubljana**

2023

Gimnazija Jožeta Plečnika Ljubljana

Kazalo

Seznam slik

Seznam tabel

Povzetek

V tem delu smo raziskovali možnost uporabe najnaprednejših tehnologij pri analizi gibanja čebel med vhodom v panj. Čebele imajo ključno vlogo pri opraševanju in so bistvene za ohranjanje uravnoteženega ekosistema, vendar je njihovo proučevanje zaradi majhne velikosti in hitrega gibanja zahtevno. Z uporabo strojnega vida, tehnologije, ki omogoča računalniško interpretacijo in razumevanje vizualnih podatkov ter metod sledenja večim objektom (MOT) nam je uspelo razviti moderno rešitev za njihovo proučevanje. Raziskava je pokazala, da uporaba teh tehnologij ponuja nov, obetaven pristop k zbiranju velikega števila natančnih podatkov o gibanju čebel.

Ključne besede: zaznava čebel, računalniški vid, MOT, YOLO, *Apis mellifera carnica*

Abstract

In this article we researched the possible use of advanced technologies for the analysis of bee movement in front of a beehive. Bees play a key role in the pollination of plants, they have an immense part in keeping our ecosystem balanced. However, due to their small size and fast movements, it is challenging for researchers to study them. With the use of computer vision, a technology that interprets and understands visual information, as well as Multi Object Tracking (MOT), we were able to develop a modern solution to the otherwise difficult task of bee tracking. With this, the study of bees becomes less constrained and far more effective. The research shows that the use of numerous modern technologies offer a promising new approach to the collection of a large number of accurate data on bee movement.

Keywords: bee detection, computer vision, MOT, YOLO, *Apis mellifera carnica*

1 Uvod

V nedavni zgodovini smo bili priča velikim tehnološkim dosežkom in inovacijam. Svet hitro napreduje, vendar nekatera področja še zmeraj ostajajo nespremenjena. V Sloveniji je čebelarstvo velik del naše bogate zgodovine in kulture. Od leta 2022 je celo vpisano na Unescov seznam nesnovne kulturne dediščine. Zaradi podnebnih sprememb, porasta parazitov in bolezni čebel ni bilo čebelarstvo nikoli tako zahtevno, zato želimo z uporabo sodobnih metod in moderne tehnologije prispevati k uspešnejšemu uvajanju informacijske tehnologije za upravljanje čebeljih družin in uspešnejše delo čebelarjev in raziskovalcev.

1.1 Čebele

Čebela ima zelo pomembno vlogo pri opraševanju rastlin[1]. V Sloveniji imamo avtohtono pasmo čebele imenovano kranjska sivka (*Apis mellifera carnica*), ki jo zlahka prepoznamo po njeni rjavo-sivi barvi, z rahlo svetlejšimi lisami. Kranjska sivka je priljubljena zaradi sposobnosti branjenja pred škodljivci, svoje nežne narave in odpornosti pred nekaterimi boleznimi in paraziti, ki bi oslabile druge čebele. Družina kranjske sivke na višku sezone pridela velike količine medu in cvetnega prahu, kar jo naredi privlačno za poklicne čebelarje. To je tudi čebela, ki jo spremljamo v nalogi.

Poleg čebelarjem, bi naša tehnologija lahko dobro služila tudi raziskovalcem. Za globlje razumevanje in ohranitev čebel znanstveniki pogosto potrebujejo velike količine podatkov. Ker je njihovo ročno zbiranje časovno zahtevno in nepraktično, želimo z moderno tehnologijo nuditi boljšo alternativo.

1.2 Gibanje čebel

Čebele za letijo z dvema paroma kril, s katerimi udarjajo okoli 230-krat na sekundo[3]. Letijo hitro, s hitrostjo od 24 do 40 km/h. Njihovo gibanje je na videz kaotično, saj ne letijo neposredno v panj, ampak vijugajo in lebdijo v zraku pred samim panjskim žrelom. To je vhod in izhod v panj za čebeljo družino in je ponavadi obrnjen proti jugu. Poleg hitrega leta pa lahko tudi bliskovito spreminjajo smer. Z opazovanjem vedenja čebel na žrelu so čebelarji v preteklosti, ko niso imeli možnosti brskanja po satju, prepoznali ključna dogajanja v čebelji družini. Spremljali smo čebele pred panjem, kjer se gibljejo rahlo počasneje (okoli 17 km/h), ker letijo obremenjene z nektarjem in cvetnim prahom. Na posnetkih smo opazili, da se čebele pogosto zadržujejo na panjskem žrelu, kar nam je otežilo njihovo označevanje.

Čebele z gibanje tudi sporočajo vrsto, smer in oddaljenost paše. Smer zibanja določa smer paše in sonca, gledano na navpičnico. Kadar je oblačno, čebele ohranijo orientacijo na podlagi spomina. Orientirajo se s pomočjo objektov v okolici panja in na poti do paše, njihova biološka ura pripomore pri iskanju smeri sonca. Oddaljenost paše določa čas zibanja; krajše, kot je zibanje, bližje je paša in obratno[2].

1.3 Obstoječe rešitve

Obstajajo različni načini avtomatizacije sledenja dogajanja v panju, recimo s pomočjo aplikacij[4][5]. Na trgu je tudi nekaj naprav za sledenje zdravja družine s pomočjo zvoka, temperature in vlažnosti v panju kot so BuzzBox[6]. Z uvedbo metode za avtomatizacijo spremljanja dogajanja na vhodu v panj, s pomočjo umetne inteligence, bomo čebelarjem dali orodje, s katerim lahko sledijo ključnim dogajanjem v čebelji družini in njeni okolici. Uporabili bomo umetno inteligenco, saj nepredvidljive značilnosti gibanja čebel otežujejo učinkovito uporabo drugih, analognih ali algoritmičnih, metod zbiranja podatkov.

1.4 Namen

Želeli smo razviti učinkovit, stabilen in posplošen model strojnega učenja za zaznavo čebel, ki vzdrži visoko natančnost tudi v primerih, ko so algoritmične metode neuspešne. Model bi te podatke nato podal naprej algoritmu za sledenje večim objektom, ki bi natančno sledil gibanju čebel. Ves program bi omogočal samodejno zbiranje velikih količin podatkov o gibanju na panjskem žrelu. Hitrost izvajanja celotnega programa bi bila dovolj hitra, da to zbiranje poteka v živo.

Te podatke bi lahko neposredno uporabili znanstveniki za splošno raziskovanje in iskanje vzorcev v gibanju čebel ter njihovo korelacijo s stanjem družine. Informacije o aktivnosti čebel na panjskem žrelu so odraz dogajanja znotraj čebelje družine in stanja okolja zunaj panja. To lahko omogoči razviti sistem za avtomatsko prepoznavo ključnih sprememb v čebelji družini in tudi zgodnje opozarjanje čebelarjev, da ustrezno ukrepajo in preprečijo morebitno oslabitev ali celo smrt čebelje družine. S tem lahko celo uvedejo ukrepe za povečanje pridelave čebeljih pridelkov. Tovrstna tehnologija lahko bistveno pripomore k učinkovitosti in razvoju čebelarjenja.

1.5 Hipoteze

Naš osrednji namen je najprej razviti učinkoviti model za beleženje aktivnosti čebel na panjskem žrelu, kot temelj posrednega spremljanja dogajanja v čebelji družini in okolju. V ta namen smo si postavili sledeče raziskovalne hipoteze:

- Hipoteza 1: Izvajanje sledenja čebel je dovolj hitro, da se izvaja na posnetkih v živo
- Hipoteza 2: Pri hitrosti izvajanja modelov je ključna izbira predusposobljenega modela (glej metodologijo)
- Hipoteza 3: Modeli družine YOLOv7 so hitrejši in bolj natančni kot modeli YOLOv5 (glej metodologijo)
- Hipoteza 4: Sledenje čebel deluje tudi na posnetkih, kjer je razdalja med čebelami majhna
- Hipoteza 5: Zaznava deluje hitreje na posnetkih z manjšim številom čebel

2 Metodologija

2.1 Eksperimentalno okolje

Vso usposabljanje modelov smo izvajali na namiznem računalniku s sledečimi specifikacijami: *Windows 11, AMD Ryzen 7 5800X, 16 GB RAM in GeForce RTX 3070 Ti (8 gb VRAM)*. Grafična kartica je imela največji vpliv na hitrost usposabljanja in zaznave.

2.2 Razvojna orodja in knjižnice

Računalniško kodo smo razvijali z integriranim razvojnim orodjem Visual Studio Code. Za to razvojno orodje smo se odločili, saj smo z njim že imeli izkušnje, prav tako pa ima na voljo veliko število kvalitetnih vtičnikov.

Vse programe smo pisali v programskem jeziku Python 3. Programski jezik se pogosto uporablja za strojno učenje, zato ima bogato izbiro ustreznih knjižnic. Za delo s posnetki in izvajanje modelov smo uporabili knjižnico OpenCV[7], ki je napisana v C++, kar izboljša učinkovitost izvajanja.

Slike smo označevali v orodju Roboflow[8], ki omogoča boljše zbiranje podatkov in predobdelavo.

2.3 Izbira knjižnic za sledenje čebel

Za sledenje čebel smo uporabili knjižnici Norfair[9] in ByteTrack[10], ki implementirata algoritem za MOT (*angl. Multiple Object Tracking*). Zanju smo se odločili na podlagi dobrih rezultatov na številnih merilih uspešnosti[11], ter enostavnosti njune uporabe. Poleg tega pa sta obe knjižnici kompatibilni z zaznavami, ki jih vrnejo modeli YOLOv5 in YOLOv7.

2.4 Izbira modela zaznavanja objektov

V nalogi smo za zaznavo čebel uporabili konvolucijske nevronske mreže. Pri usposabljanju modela zaznave čebel smo uporabili modele dveh družin predusposobljenih nevronskih mrež, namenjenih zaznavi objektov. To sta YOLOv5[12], ki je eden najbolj razširjenih in najboljših modelov za zaznavanje objektov in YOLOv7[13], ki je novejša arhitektura. YOLOv7 domnevno opisuje boljše rezultate, kot YOLOv5. Modele teh dveh družin modelov smo dodatno usposabljali na lastnih podatkih za čebele. Usposabljali smo predusposobljene modele YOLOv5s, YOLOv5m, YOLOv5x in modela YOLOv7 ter YOLOv7x.

2.4.1 Poimenovanje modelov

V testni fazi smo modele arbitrarno poimenovali po opisu modela (npr. bees5s4). To poimenovanje smo pozneje preoblikovali (Tabela 1, stolpec za ime). Vsa imena se začnejo z akronimom BDM (model zaznave čebel - *angl. Bee Detection Model*). Akronimu sledita identifikacijska številka modela in pomišljaj. Za tem je oznaka za vrsto modela. Modeli YOLOv5s imajo oznako s, YOLOv5m oznako m, YOLOv5x oznako x, YOLOv7 oznako 7 in YOLOv7x oznako 7x. Primer poimenovanja modela je **BDM1-s**. To je model z identifikacijsko številko 1, ki je usposobljen na predusposobljenem modelu YOLOv5s.

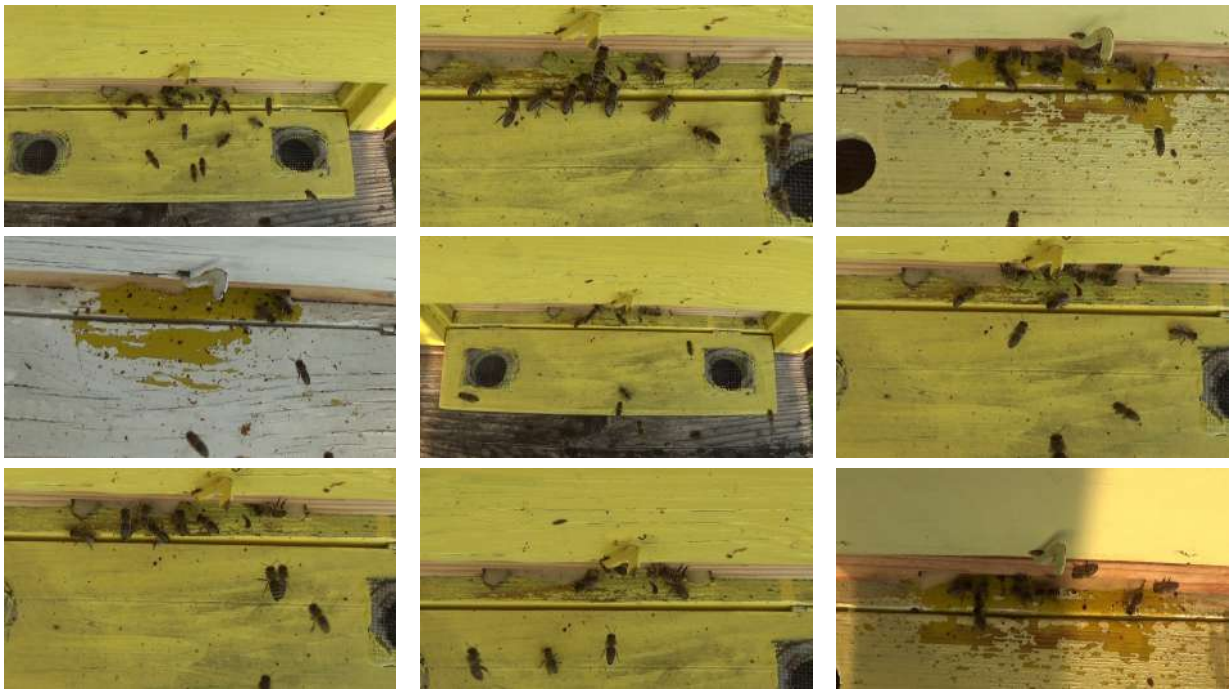
2.5 Obdelava posnetkov

Uporabljali smo 10 različnih posnetkov žrela panja. Izbrali smo zahtevne in raznolike posnetke, da bi dosegli bolj posplošene modele. Posnetki so bili vzeti na Urbanem učnem čebelnjaku botaničnega vrta v Ljubljani[14]. Posnetki so imeli resolucijo 1920x1080 (HD) in bili narejeni v različnih okoliščinah (različna povečava, osvetljenost, barva panja, število čebel) (Slika 1). Zaradi časovne omejitve naloge smo imeli le manjše število posnetkov.

Pri snemanju nekaterih posnetkov se občasno spremeni povečava. Ker bi to negativno vplivalo na analizo premikanja čebel, smo posnetke razdelili na dele z enako povečavo.

Zaradi lažjega dela smo posnetke s programom FFmpeg iz formata MTS spremenili v MP4. Pri tem smo morali biti pazljivi, da smo obdržali enako kvaliteto. S poskušanjem smo ugotovili, da to najlažje dosežemo z naslednjimi parametri:

```
.\ffmpeg.exe -i STARO.MTS -vf yadif=1 NOVO.MP4
```



Slika 1: Prikaz različnih uporabljenih posnetkov

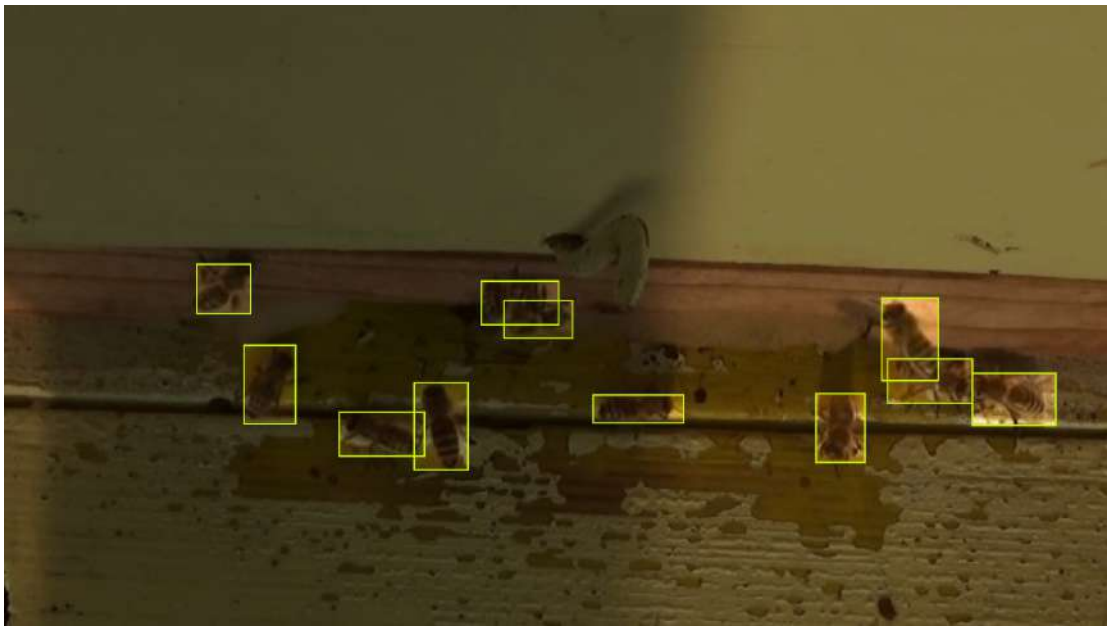
Za nadaljnjo pripravo posnetkov za usposabljanje smo iz njih morali pridobiti posamezne slike. Iz posnetkov smo smo pridobili 2100 slik (Slika 2).

```
import cv2 #openCV
from math import floor
def main():
  TARGET_FRAMES = 2100 # stevilo slik za shraniti
  vidnames = tuple((f"../videosmp4/000{x}.mp4" for x in range(64, 73, 1)))

  #prebere dolzine posnetkov
  lengths = []
  for vid in vidnames:
    v = cv2.VideoCapture(vid)
    lengths.append(int(v.get(cv2.CAP_PROP_FRAME_COUNT)))
    v.release()
  lengths = tuple(lengths)

  th = round(sum(lengths)/TARGET_FRAMES)
  for idx, vidname in enumerate(vidnames):
    vid = cv2.VideoCapture(vidname)
    for frame in range(0, lengths[idx], th):
      success, image = vid.read()
      cv2.imwrite(f"photos/{vidname[13:-3]}{frame}.jpg", image)
      for _ in range(th-1):
        __,__ = vid.read()
      print(f"{frame}/{lengths[idx]} ~ vid {idx+1}")
    vid.release()
if (__name__ == "__main__"):
  main()
```

Slika 2: Program, ki enakomerno pridobi slike iz posnetka



Slika 3: Označevanje čebel z omejevalnimi pravokotniki

2.6 Priprava slik za usposabljanje

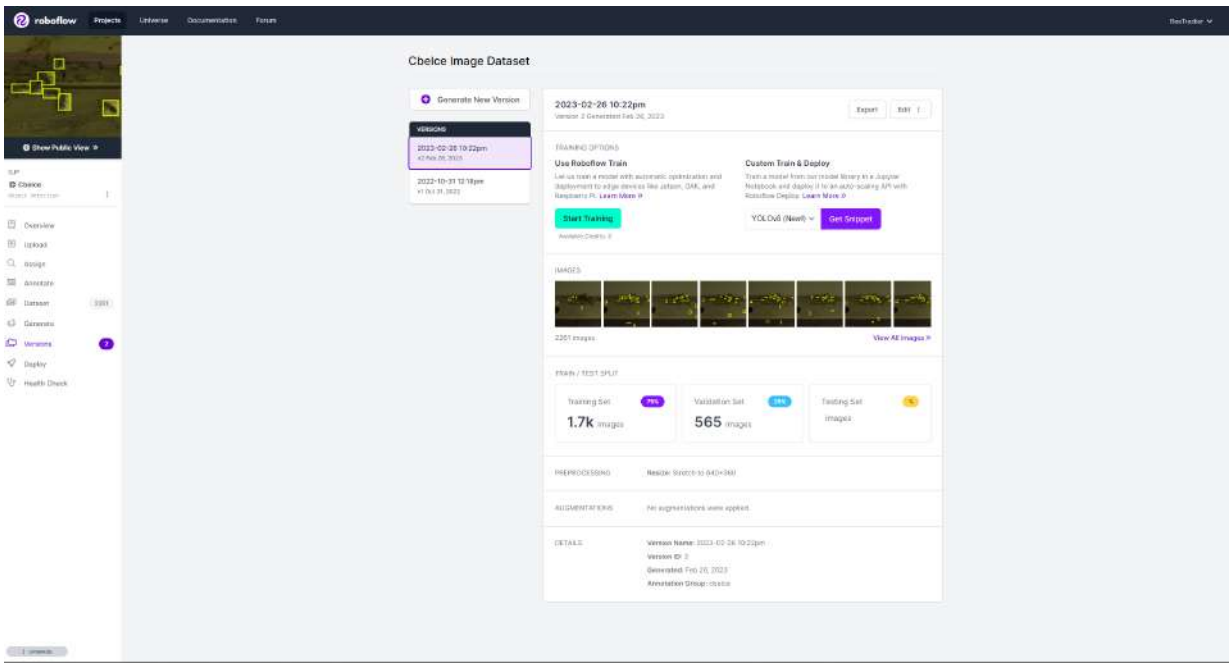
Slike smo označevali z orodjem Roboflow[8] (Slika 5), kar nam je močno olajšalo označevanje čebel. Pri tem smo morali paziti na natančnost, saj bi se napake drugače ponovile pri modelu. Pri označevanju smo imeli težave s slikami z visoko koncentracijo čebel, saj jih je težko označiti natančno (Slika 4).

Orodje Roboflow je pomanjšalo resolucijo slik na 640x360 pixlov, ker je to resolucija modelov.

Roboflow samodejno razdeli vse slike v dve skupini: skupino za usposabljanje in skupino za validacijo modela. Prva se uporablja neposredno za usposabljanje, druga pa med usposabljanjem za testiranje stanja modela. Prva skupina zajema približno 75 % vseh slik. Ta skupina je večja, ker poskušamo modelu podati največje možno število podatkov.

Slike iz teh podatkov Roboflow vrne v naslednji obliki: identifikacijska številka vrste objekta, x, y, širina, višina (Slika 6).

Slika 4: Primer oteženega označevanja slik, zaradi visoke koncentracije čebel



Slika 5: Prikaz orodja Roboflow

```
0 0.684375 0.85 0.0390625 0.05416666666666667
0 0.821875 0.525 0.03359375 0.04722222222222222
0 0.7640625 0.43194444444444446 0.0375 0.03888888888888889
0 0.29375 0.4041666666666667 0.02890625 0.08888888888888889
```

Slika 6: Primer datoteke z označenimi slikami za usposabljanje

2.7 Usposabljanje modelov

Vse modele smo lahko usposabljali na istih podatkih, saj YOLOv5 in YOLOv7 uporabljata isti format za označevanje slik (Slika 6). Da bi prišli do čim boljšega končnega modela smo usposobili več modelov. Pri usposabljanju smo spreminjali naslednje nastavitve modelov:

- **Velikost serije (*Batch size*)** Število primerov slik uporabljenih v eni iteraciji usposabljanja.
- **Število prehodov (*epochov*)** Število prehodov čez vse slike izvedenih med usposabljanjem.
- **Velikost slik** Resolucija slik, na katerih usposabljanje poteka. Izvirni posnetki so imeli resolucijo 1920x1080, vendar smo večino modelov usposabljali na 640x640 (resolucija predusposobljenih modelov).
- **Izbira predusposobljenega modela** Pri usposabljanju smo uporabili različne predusposobljene modele različnih velikosti.

Pri usposabljanju YOLOv5 modelov se nastavi več kot 30 hiperparametrov, ki skupaj vplivajo na proces. Pri spreminjanju posameznih hiperparametrov iz standardnih nastavitev smo ugotovili, da se posledično močno zniža kvaliteta modela. Odločili smo se spreminjati samo velikost serije in število epochov, saj sta to najpogosteje spremenjena hiperparametra. Ostale hiperparametre smo ohranili na standardni nastavitvi.

Usposabljali smo tri YOLOv5 modele, in sicer YOLOv5s, YOLOv5m in YOLOv5x. Usposabljali smo tudi dva YOLOv7 modela, YOLOv7 in YOLOv7-X. Usposobili smo več YOLOv5 modelov,

ime	število epochov	velikost serije	model	resolucija
BDM1-s	100	16	v5s	640
BDM2-s	100	16	v5s	416
BDM3-s	100	16	v5s	640
BDM4-s	174	-1	v5s	640
BDM5-m	268	-1	v5m	640
BDM6-s	174	-1	v5s	640
BDM7-x	27	-1	v5x	640
BDM8-s	20	32	v5s	640
BDM9-s	184	42	v5s	640
BDM10-7x	87	8	v7x	640
BDM11-7	166	10	v7	640
BDM12-7	150	11	v7	640
BDM13-7x	150	11	v7x	640
BDM14-m	135	16	v5m	640
BDM15-m	163	-1	v5m	640

Tabela 1: Specifikacije usposobljenih modelov strojnega učenja. Negativna velikost serije nakazuje označuje samodejno izbiro

saj je njihovo usposabljanje potekalo hitreje, kot v7 modelov. Vse operacije so potekale na grafični kartici, saj bi se na procesorju odvijale bistveno počasnejše. Če bi uporabili procesorsko usposabljanje, bi lahko izbrali rahlo večjo velikost serije, ker bi eksperimentalno okolje imelo več delovnega pomnilnika (RAM), kot grafičnega pomnilnika (VRAM).

Končno število modelov usposobljenih z različnimi nastavitvami je bilo 15. Štirje so bili iz družine YOLOv7 in 11 iz družine YOLOv5 (Tabela 1). Modele smo usposabljali z naslednjima ukazoma:

```
python train.py --img 640 --batch-size 16 --epochs 100 --data --device 0
konfiguracija.yaml --weights yolov5s.pt --name BDM1-s  --cache disk

python train.py --img 640 --batch-size 10 --epochs 300 --data
konfiguracija.yaml --weights yolov7.pt  --name BDM11-7 --device 0
```

V datoteki konfiguracija.yaml je shranjena lokacija skupin slik in oznak za usposabljanje in validacijo.

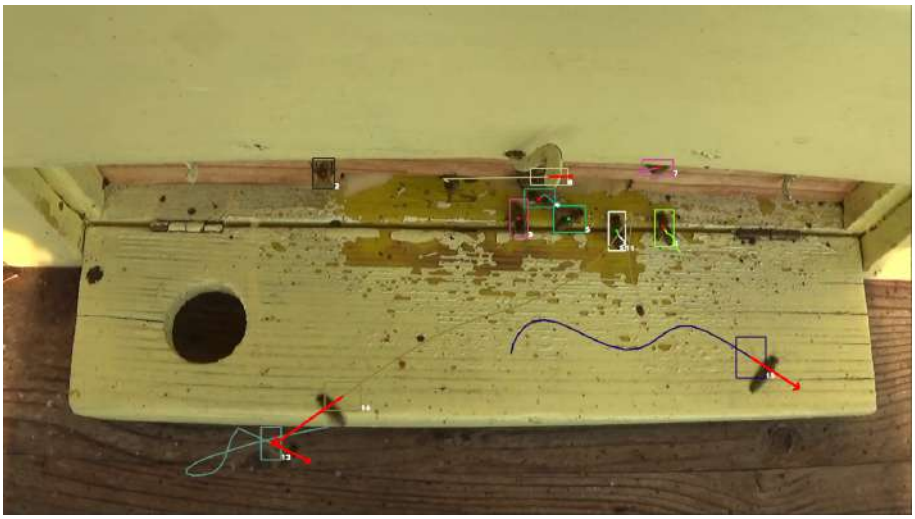
Izvirni posnetki so imeli resolucijo 1920x1080, kar se ob usposabljanju samodejno spremeni v kvadratno sliko. Usposabljanje vedno poteka na resoluciji, v obliki kvadrata. En model smo usposabljali na resoluciji 416x416, ostale pa na resoluciji 640x640. Velikost serije, pri kateri smo lahko usposabljali, je neposredno omejena z velikostjo grafičnega pomnilnika na grafični kartici (8 GB). YOLOv5 ima nastavitev samodejne izbire označene s številko -1. Število epochov je neposredno povezano s časom usposabljanja. Ob usposabljanju manjših modelov (posebej YOLOv5s) se proces samodejno predčasno konča, ko se uspešnost modela ne spreminja, kar omeji pretreniranje. YOLOv7 tega mehanizma nima. Nekatere modele smo ročno predčasno ustavili, ker smo hoteli preveriti njihovo uspešnost v prvih epochih. Pri izdelavi modelov smo imeli dva glavna cilja - hitrost izvajanja in natančnost zaznavanj.

Večinoma smo usposabljali manjše, YOLOv5s modele. Za to smo imeli več razlogov: hitrejše usposabljanje teh modelov, majhno število usposobnih podatkov, samo 1 razred slik (čebela). Po prvih testnih usposabljanjih smo ugotovili, da so modeli že zelo hitro začeli zaznavati velik delež čebel. Kljub temu smo usposabljali tudi nekaj v5m modelov in en v5x model. Usposabljali smo tudi YOLOv7 modele, saj smo upali, da bodo nudili višjo hitrost in kvaliteto.

2.8 Sledenje gibanju čebel

Za sledenje smo uporabili knjižnici Norfair[9] ter ByteTrack[10]. Obe knjižnici uporabljata Kalman filter[15], algoritem, ki zagotavlja natančna predvidevanja stanj pozicij ob upoštevanju negotovosti. Kljub temu se knjižnici razlikujeta, saj porabljata drugačni distančni funkciji (funkciji za izračun podobnosti med trenutno in predhodno lokacijo zaznave). Norfair[9] uporablja evklidsko distančno funkcijo, ki izračuna razdaljo med vektorjema, ByteTrack[10] pa kosinusno $\cos\phi = \frac{\vec{u}\cdot\vec{v}}{|\vec{u}||\vec{v}|}$.

Sledenje gibanja čebel smo izvedli s pomočjo podatkov, ki nam jih je vrnil model. Te podatke smo nato preuredili v kompatibilno strukturo. Pridobljene vrednosti iz modela in sledilca smo



Slika 7: Številka ob čebeli prikazuje njeno identifikacijsko številko, črta za njo pa pot, ki jo je prepotovala v zadnjih nekaj desetink sekunde

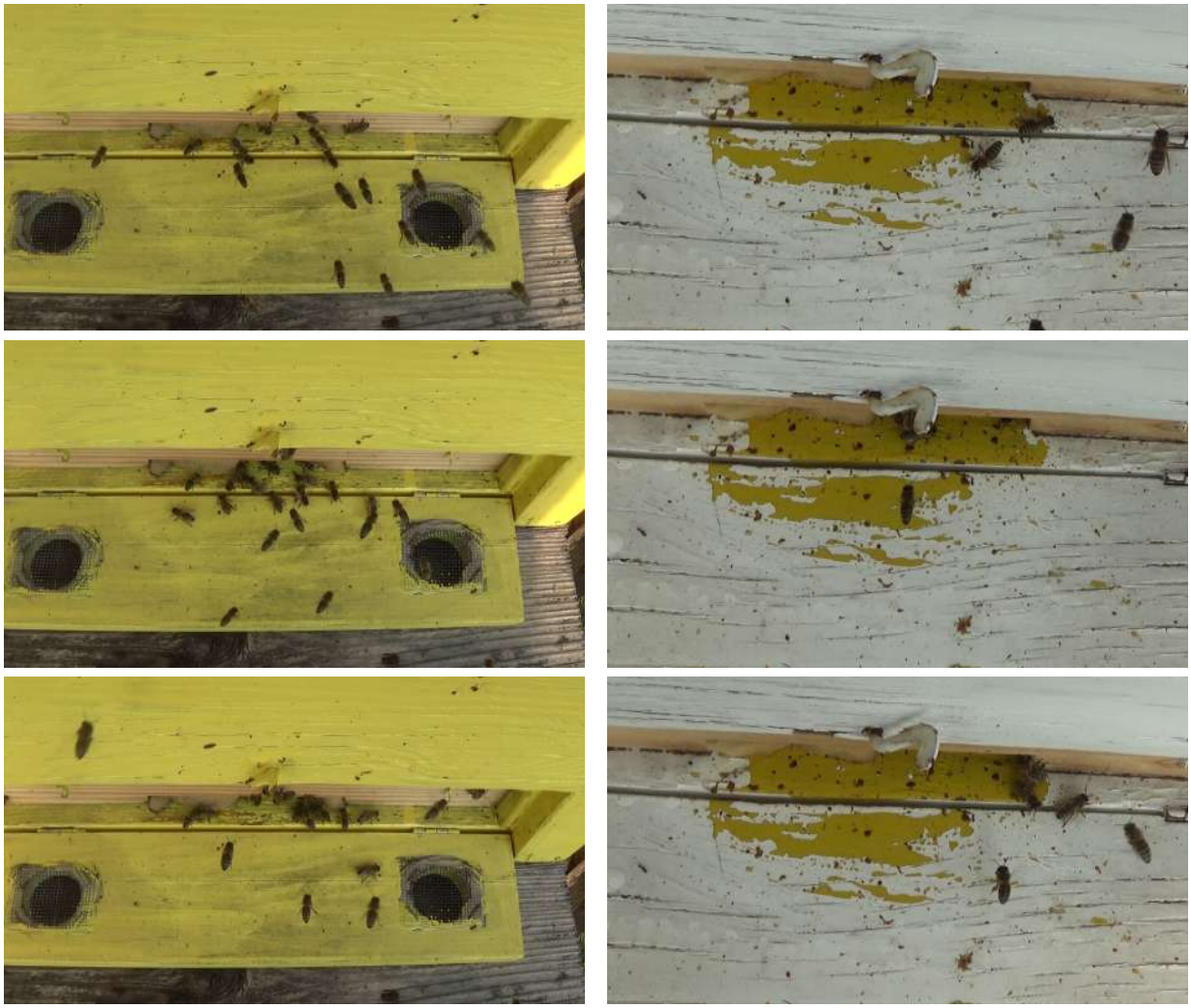
primerjali ter vsakemu objektu določili identifikacijsko številko. Knjižnica Norfair[9] je to opravila sama, pri knjižnici ByteTrack[10] pa smo to dosegli z zunanjo knjižnico Onemetric[16]. Te podatke smo nato prikazali v sliki 7.

2.9 Analiza modelov

Hitrost delovanja modelov smo testirali na dveh posnetkih. Izbrali smo posnetek, ki ima majhno število čebel in počasno gibanje ter posnetek, ki ima veliko čebel s kaotičnim gibanjem (Slika 8). Po usposabljanju nam je program vrnil dve verziji modela - tisto, ki je usposobljena zadnja in tisto z najvišjo natančnostjo. Vedno smo uporabili model z najvišjo natančnostjo, saj smo se s tem izognili preusposobljanju, ko se model primere nauči na pamet in ne črpa iz splošnih vzorcev.

2.9.1 Hitrost zaznavanja modelov

Za merjenje hitrosti smo za znižanje vpliva programa na hitrost uporabili minimalističen program (Slika 9). Pri modelu z resolucijo 416 smo uporabili testni posnetek s pravo resolucijo



Slika 8: Slike posnetkov uporabljenih pri analizi hitrosti modelov (levo posnetek z več čebelami, desno z manj čebelami)

```
from time import perf_counter
def avgTime(func, *args):
    time = 0
    for _ in range(3):
        stime = perf_counter()
        func(*args)
        time += perf_counter() - stime
    return time/3
```

Slika 9: Program za merjenje hitrosti modelov. Trikrat izmeri hitrost funkcije in izračuna povprečje

2.9.2 Natančnost zaznavanja modelov

Pri izračunu natančnosti modela se uporablja več metod in funkcij[17]:

- **Izguba objektnosti (*angl. objectioness loss function*)** Del skupne funkcije izgube. Izguba objektnosti se veča s prepričanjem v zaznave.
- **Regresijska izguba omejevalnega okvira (*angl. bounding box regression loss function*)** Del skupne funkcije izgube. Veča se s podobnostjo zaznanega okvirja in pravega okvirja okoli objekta.
- **Izguba klasifikacije (*angl. classification loss function*)** Del skupne funkcije izgube. Opisuje pravilnost klasifikacije objektov v razred. Za nas nepomembna, saj uporabljamo samo en razred.
- **IoU (*angl. intersection over union*)** Merilo za izračun, kako podobna sta realni omejevalni okvir objekta in omejevalni okvir zaznave. $IoU = \frac{A \cap B}{A \cup B}$
- **Natančnost (*angl. precision*)** Delež pravilnih zaznav od vseh zaznav.
- **Priklic (*angl. recall*)** Delež pravilno zaznanih objektov od vseh dejanskih objektov.
- **Povprečna natančnost (*angl. Average Precision*)** Pogosto uporabljeno merilo za natančnost modelov strojnega vida. Ploščina pod grafom natančnosti in priklica.
- **mAP (povprečje povprečne natančnosti *angl. mean Average Precision*)** Povprečje povprečnih natančnosti čez vse razrede. Enako povprečni natančnosti za en razred.
- **mAP@0.5** Pogosto uporabljena merila za natančnost modelov strojnega vida. Izračunan mAP za zaznave, ki imajo IoU večji, kot 0.5.
- **mAP@[.5:.95]** Pogosto uporabljena merila za natančnost modelov strojnega vida. Izračun mAP za zaznave, ki imajo IoU med 0.5 in 0.95, s korakom 0.05.

Pri usposabljanju model računa svojo uspešnost s funkcijo izgube. Pri YOLOv5 in YOLOv7 modelih je ta funkcija seštevek (pri YOLOv7 obtežen seštevek) funkcije izgube objektnosti, regresijske funkcije izgube omejevalnega okvira in funkcije izgube klasifikacije. Slednji je v našem primeru 0, ker uporabljamo le 1 razred objektov $loss = l_{obj} + l_{box} + l_{cls}$

Modele smo primerjali kvalitativno in kvantitativno. Kvantitativno smo jih primerjali po natančnosti, priklicu, mAP@0.5 in mAP@[0.5:0.95]. Merila smo izračunali na validacijskih slikah, torej slikah na katerih modeli niso bili usposobljeni. Kvalitativno smo jih primerjali vizualno na testnih slikah. Vzeli smo eno validacijsko sliko iz vsakega posnetka in jih označili z vsakih modelom (Slika 12). Četrta in peta slika imata večje število čebel in sta bolj oddaljeni, kar je za modele še posebej zahtevno. Vse teste smo opravili na posnetkih in slikah z resolucijo 640x360, kar je resolucija modela (razen pri modelu z resolucijo 416, kjer smo uporabili 416x234) (Slika 10) (Tabela 1).

(a) 1920x1080

(b) 640x360

Slika 10: Primerjava izvirne resolucije posnetka a in zmanjšane resolucije, v kateri delujejo modeli b

2.10 Analiza metod sledenja premikanja čebel

Pri analizi premikanja čebel smo, kot pri analizi modelov, uporabili dva posnetka (Slika 8). Za zaznavo čebel smo, zaradi visoke hitrosti in natančnosti modela, uporabljali model BDM15-m.

2.10.1 Hitrost sledenja čebel

Pri merjenju hitrosti sledenja smo uporabili isti program, kot pri merjenju hitrosti zaznave (Slika 9).

2.10.2 Ocenjevanje natančnosti sledenja čebel

Pri ocenjevanju natančnosti sledenja smo prešteli število čebel, ki so se približale definiranemu območju ob vhodu v panj. Nato smo že obstoječo kodo opremili s števcem, ki si zapomni vsakič, ko čebela vstopi v to območje. Te podatke smo potem primerjali z ročno preštetimi količinami.

3 Rezultati

3.1 Hitrost izvajanja modelov

Pri primerjavi modelov lahko vidimo, da je razlika v hitrosti izvedbe na posnetku z več in manj čebelami zelo podobna. Pri nekaterih modelih je bil posnetek z manj čebelami celo hitreje procesiran (Slika 11a). Na hitrost je imela bistven vpliv velikost predusposobljenega modela. Najhitrejši so bili vsi modeli predusposobljenega modela YOLOv5s, ki so tudi najmanjši. Najpočasnejši so bili modeli YOLOv7x (Slika 11b), čeprav YOLOv7 modeli niso imeli večje natančnosti.

Vsi modeli so dovolj hitri, da z zmogljivo grafično kartico posnetke procesirajo v živo, ker so vsi modeli zaznavali več kot 30 slik na sekundo, kar je tipična hitrost kamer (Slika 11a). Vsi modeli YOLOv5s in YOLOv5m so celo zaznavali s hitrostjo višjo od 60 slik na sekundo.

3.2 Natančnost modelov

Pri primerjavi modelov po priklicu in natančnosti (Slika 19) (Tabela 2) vidimo, da bistveno izstopata modela BDM1-s in BDM3-s. To sta najslabša modela. Imata priklic pod 0.7 in natančnost pod 0.8, kar nakazuje, da model več kot tretjine slik ne zazna, petina zaznav je pa napačnih. Slabi rezultati so očitni tudi po merilu mAP (Slika 20). Merilo mAP@[.5:.95] (Slika 20b) bolje prikaže dejansko razliko med modeli, kot mAP@0.5 (Slika 20 a).

Poleg BDM1-s in BDM3-s izstopa tudi BDM2-s, ki je še vedno bistveno slabši od ostalih. Zanimivo, slednji ni najslabši, kljub temu da je edini z nižjo resolucijo.

Vsi slabši modeli imajo arhitekturo YOLOv5s, pri tem pa smo opazili, da imajo tudi drugi (BDM4-s, BDM5-s in BDM6-s) dobro natančnost. Ni neposredne povezave med arhitekturo in kvaliteto očitno modela. Najbolj se je izkazal model BDM11-7 z vrednostjo mAP@[.5:.95] 0.617. 8 modelov ima podoben mAP@[.5:.95], malo nad 0.6. Kljub močno počasnejšemu času izvajanja (Slika 11b), je kvaliteta zaznav YOLOv7 primerljiva z modeli YOLOv5. Povprečen model YOLOv7 ima mAP@[.5:.95] 0.599, medtem ko ima povprečen YOLOv5 model 0.515 (0.578 brez najslabših treh modelov).

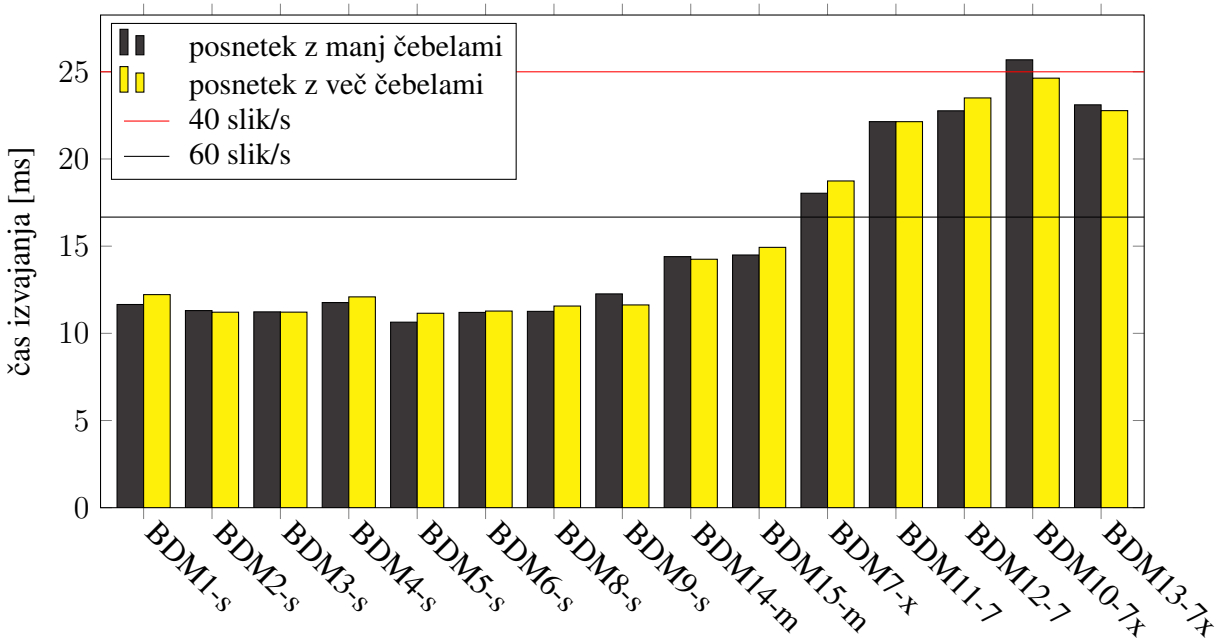
Pri primerjavi kvalitete modelov po zaznavanju na slikah so rezultati rahlo drugačni:

- Pri prvi sliki se dve čebeli skoraj prekrivata (Slika 12a). Nekateri modeli ju v zaznavi ne ločijo (Slika 13).
- Pri drugi sliki se ob robu žrela panja vidi samo majhen del zadka in glave čebele (Slika 12b). Večina modelov, kot pričakovano, teh delov ni zaznala. (Slika 14)
- Tretja slika je preprosta, brez robnih primerov. Velikost čebel je tudi velika (Slika 12c). Edino BDM5-s jo je zaznal napačno, ko je dve čebeli združil v eno (Slika 15).
- Četrta slika ima veliko čebel, ki so bolj razmaknjene (Slika 12d). Veliko modelov jih napačno zaznava (Slika 16).
- Peta slika je podobna četrti (Slika 12e), veliko napak se zgodi pri delno vidnih čebelah pri vhodu v panj (Slika 17).
- Šesta slika ima samo štiri čebele, vendar je ena čebela večinoma skrita ob robu panjskega žrela(Slika 12f), kar povzroča napačne zaznave pri številnih modelih (Slika 18).

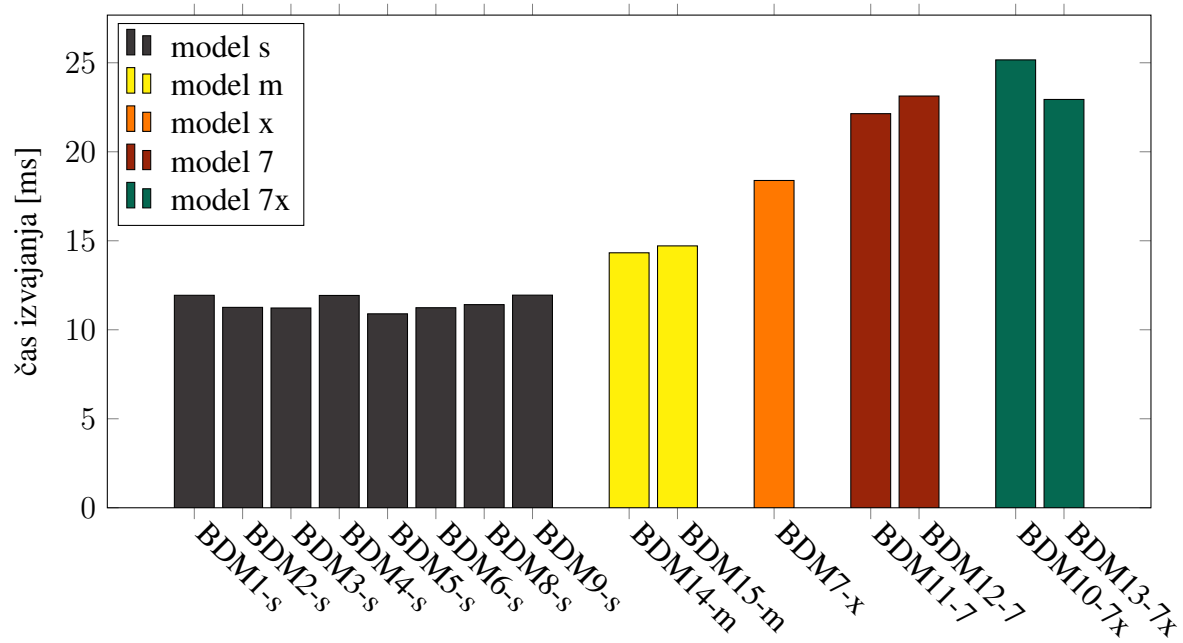
Pri zaznavah je najpogostejša napaka, ki se pojavlja pri slabših modelih združevanje več čebel v eno.

ime	natačnost	priklic	mAP@0.50	mAP@[.5:.95]
BDM1-s	0.765	0.679	0.742	0.316
BDM2-s	0.853	0.753	0.825	0.386
BDM3-s	0.765	0.679	0.742	0.316
BDM4-s	0.948	0.95	0.965	0.636
BDM5-s	0.95	0.944	0.962	0.637
BDM6-s	0.948	0.95	0.965	0.636
BDM7-x	0.916	0.915	0.937	0.574
BDM8-s	0.901	0.881	0.917	0.434
BDM9-s	0.95	0.952	0.966	0.639
BDM10-7x	0.938	0.906	0.936	0.59
BDM11-7	0.944	0.951	0.961	0.64
BDM12-7	0.943	0.944	0.957	0.636
BDM13-7x	0.942	0.919	0.943	0.612
BDM14-m	0.939	0.949	0.961	0.626
BDM15-m	0.959	0.963	0.973	0.659

Tabela 2: Tabela meril vseh modelov strojnega učenja na validacijskem delu slik



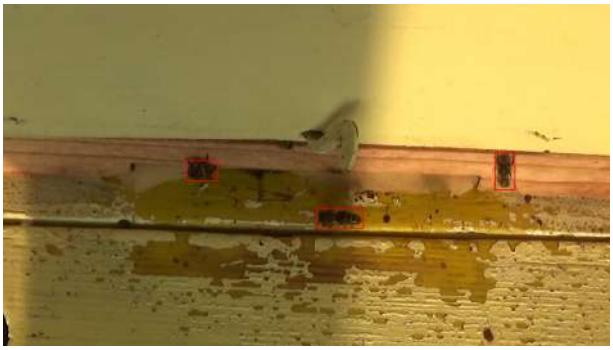
(a) Povprečna hitrost izvajanja modelov na sliki posnetka z več in manj čebelami. Na grafu sta označeni meji za 40 in 60 slik/sekudo



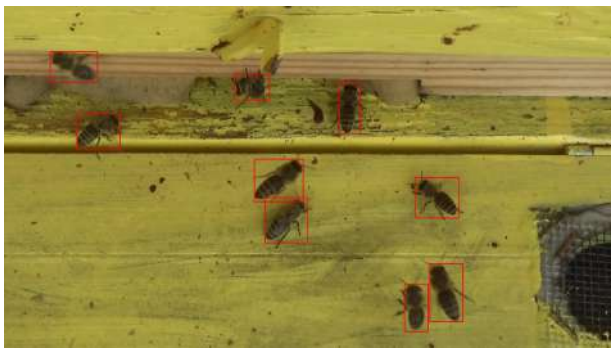
(b) Povprečne hitrosti modelov združene po predusposobljenem modelu, ki je bilo izračunano iz treh testnih pogonov posameznega modela

Slika 11: Hitrosti modelov

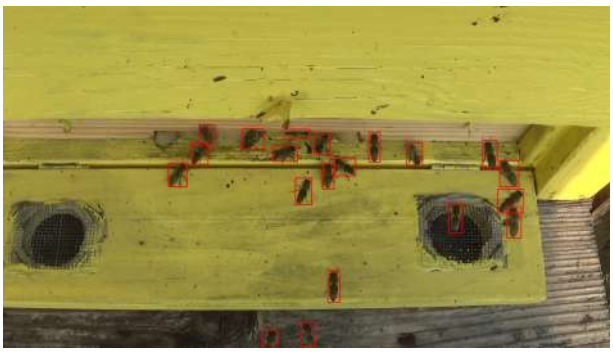
(a) Prva zaznava BDM5-s, pravilno loči čebeli, ki sta blizu



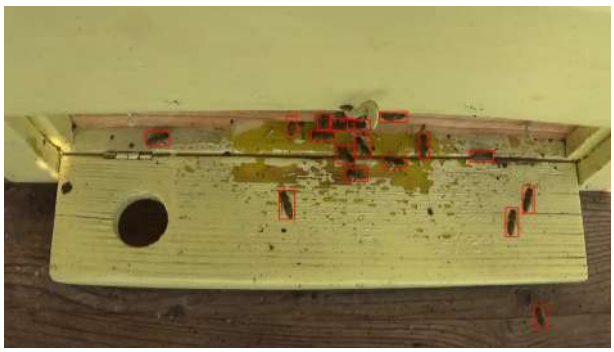
(b) Druga zaznava modela BDM12-7, ne zazna glave in zadka čebel, ki štrlita ven pri ustju žrela



(c) Tretja zaznava modela BDM7-x, pravilne zaznave



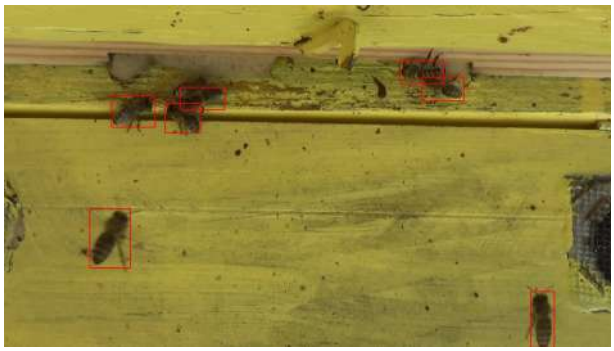
(d) Četrta zaznava modela BDM11-7, pravilne zaznave



(e) Peta zaznava modela BDM6-s, vse zaznave so pravilne, tudi kjer se čebeli prekrivata

(f) Šesta zaznava modela BDM15-m, pravilne zaznave, tudi kjer je čebela napol nevidna pri panju

Slika 12: Primeri najboljših zaznav čebel v ustju panja za vsak posnetek



(a) Zaznava modela BDM10-7x, zazna samo eno čebelo od dveh, ki sta blizu

(b) Zaznava modela BDM1-s, zazna samo eno od čebel, ki sta blizu

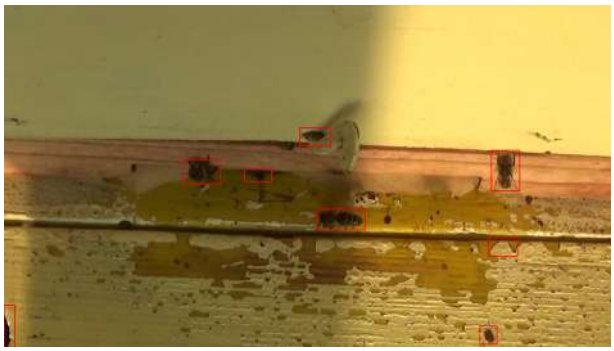
(c) Zaznava modela BDM14-m, zazna dve čebeli kot tri

(d) Zaznava modela BDM8-s, v gruči zazna preveč čebel

(e) Zaznava modela BDM2-s, zazna rob panja kot čebelo

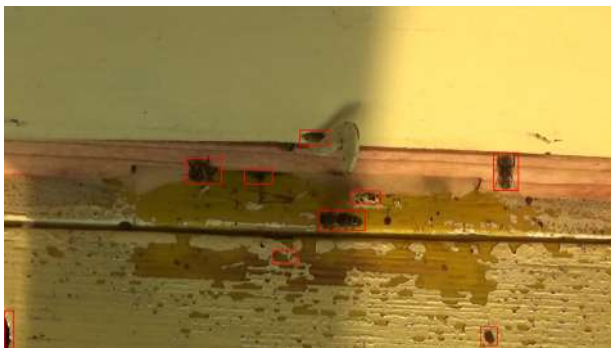
(f) Zaznava modela BDM3-s, zazna samo eno čebelo od dveh, ki sta blizu

Slika 13: Nenatančne zaznave za prvi posnetek

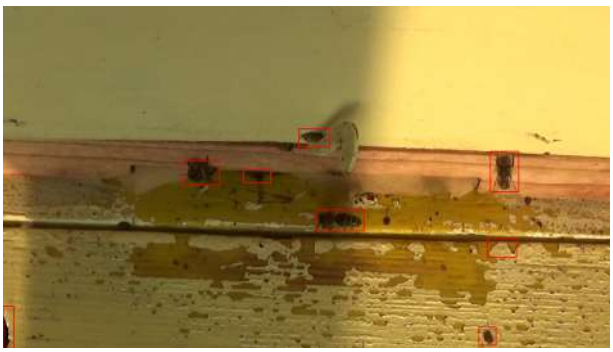


(a) Zaznava modela BDM1-s, zaznava dele panja, ki niso čebela

(b) Zaznava modela BDM8-s, dvojno zazna eno čebelo

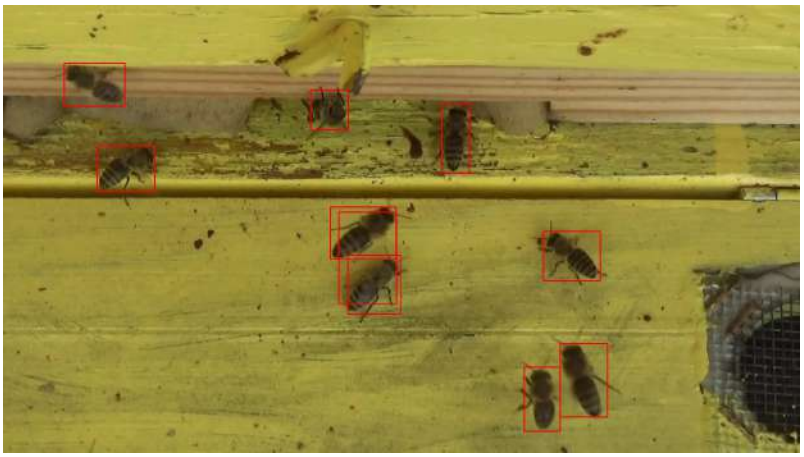


(c) Zaznava modela BDM2-s, zaznava dele panja, ki niso čebela



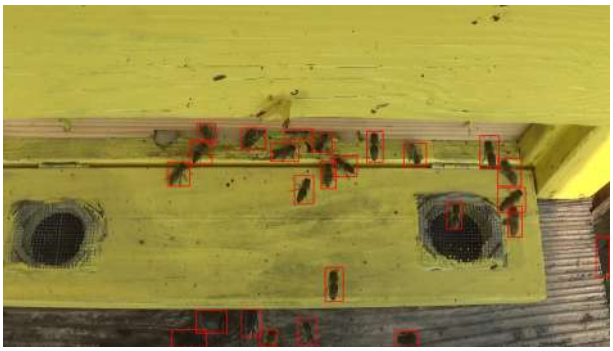
(d) Zaznava modela BDM3-s, zaznava dele panja, ki niso čebela

Slika 14: Nenatančne zaznave za drugi posnetek

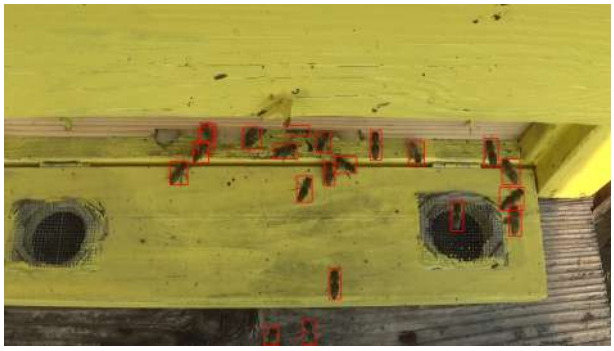


Slika 15: Nenatančna zaznava na tretjen posnetku modela BDM5-s

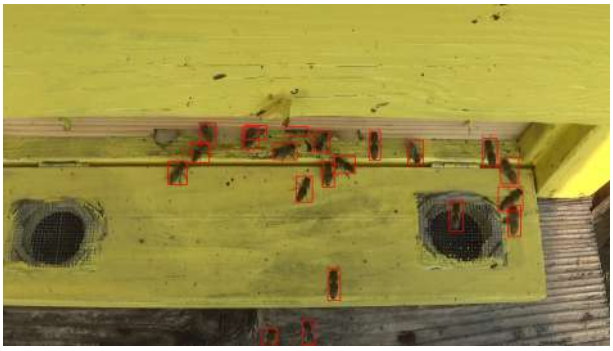
(a) Zaznava modela BDM7-x, čebele zaznane dvojno



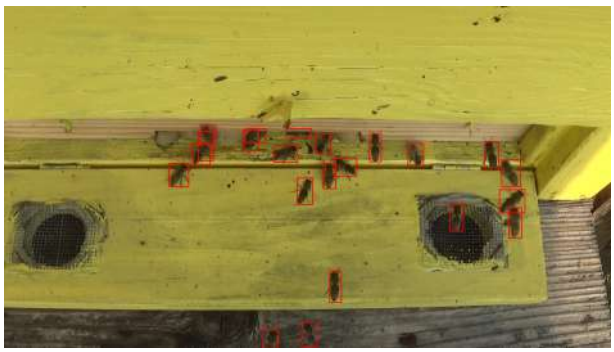
(b) Zaznava modela BDM1-s, vsa temna območja na panju zaznana



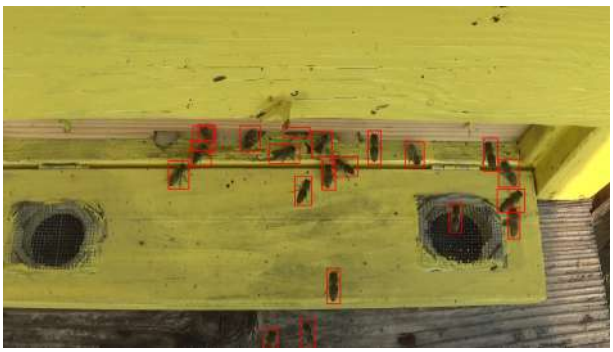
(c) Zaznava modela BDM4-s, čebeli zaznani dvojno



(d) Zaznava modela BDM5-s, čebelo pri žrelu panja zazna dvojno

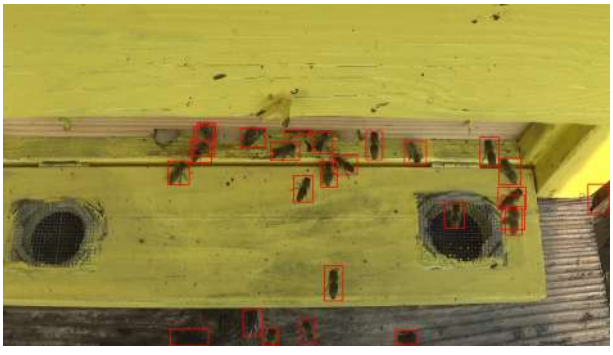


(e) Zaznava modela BDM14-m, čebele zazna dvojno



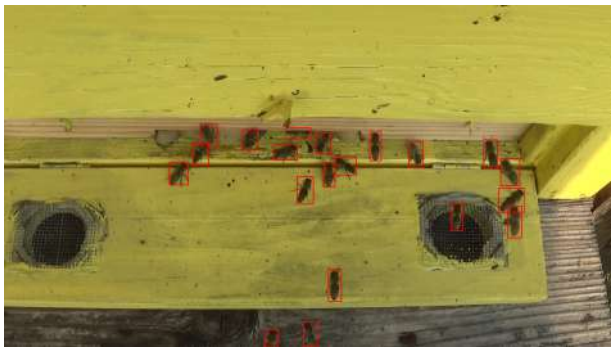
(f) Zaznava modela BDM8-s, zaznava dele panja, ki niso čebela

(g) Zaznava modela BDM8-s, čebeli, ki sta blizu zazna trikrat



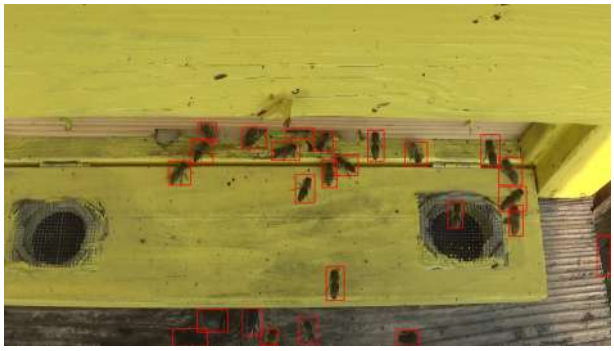
(h) Zaznava modela BDM2-s, zaznava dele panja, ki niso čebela

Slika 16: Premeri nenatančnih zaznav za četrti posnetek

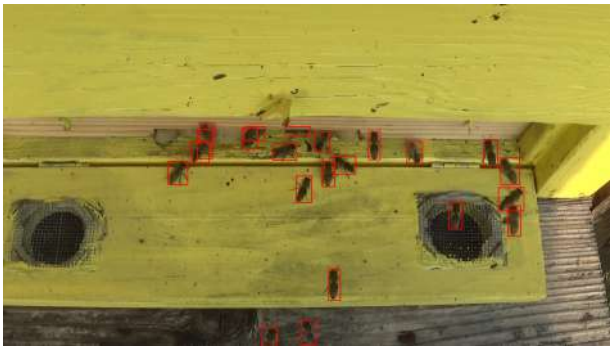


(a) Zaznava modela BDM10-7x, pri vhodu zgreši zaznavo napol prekrite čebele

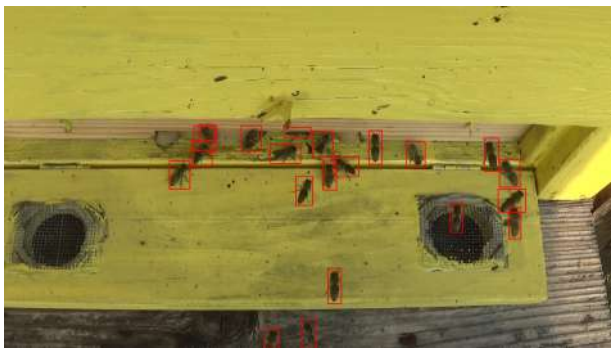
(b) Zaznava modela BDM7-x, napol prekrito čebelo pri vhodu zazna dvakrat



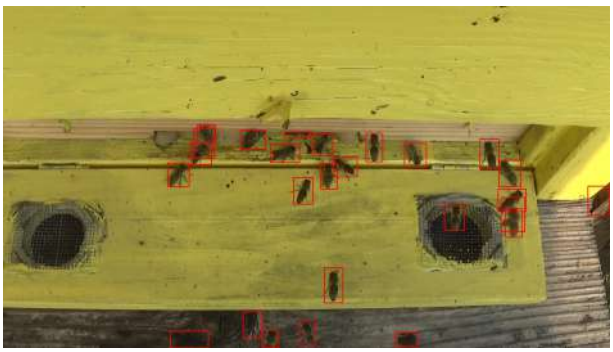
(c) Zaznava modela BDM1-s, vsa temna območja na panju zaznana



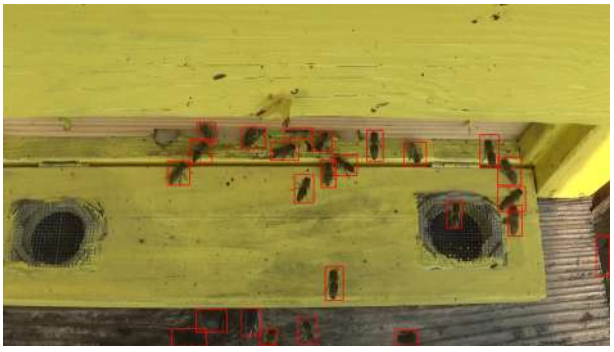
(d) Zaznava modela BDM14-m, napol prekrito čebelo pri vhodu zazna dvakrat



(e) Zaznava modela BDM8-s, gruče čebel zaznava napačno

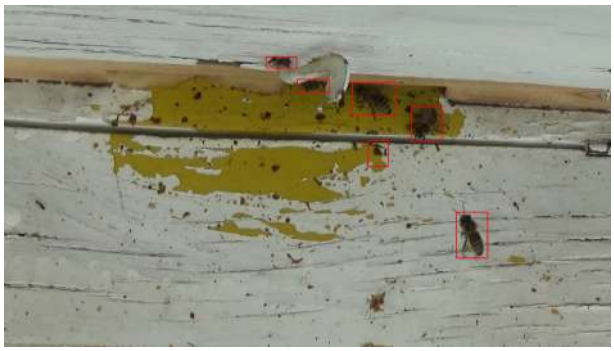


(f) Zaznava modela BDM2-s, gruče čebel zaznava napačno



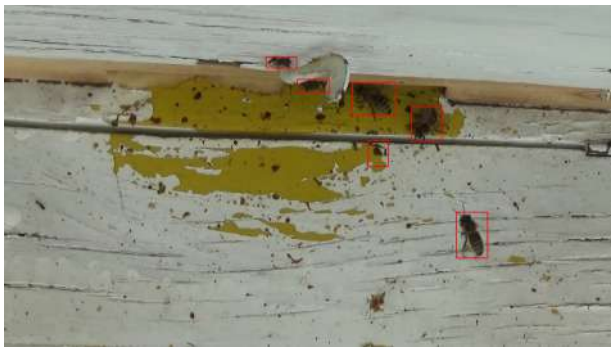
(g) Zaznava modela BDM3-s, gruče čebel zaznava napačno

Slika 17: Nenatančne zaznave za peti posnetek



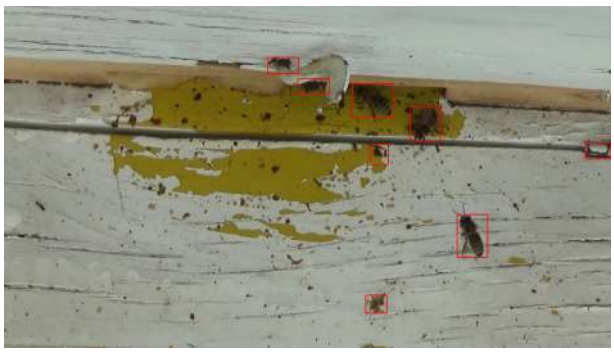
(a) Zaznava modela BDM7-x, dele čebel pri vhodu zazna, kot eno čebelo

(b) Zaznava modela BDM12-7, ne zazna dela čebel pri vhodu

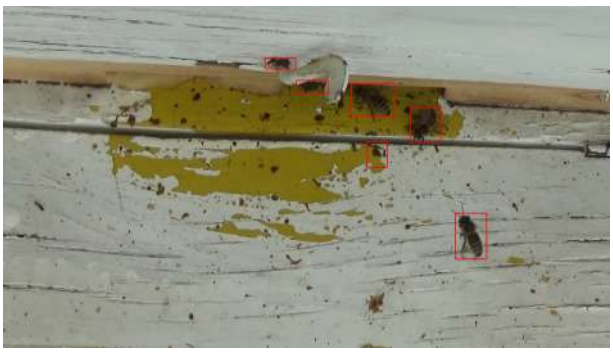


(c) Zaznava modela BDM1-s, zazna vsa temna območja na panju

(d) Zaznava modela BDM8-s, vsa temna območja na panju

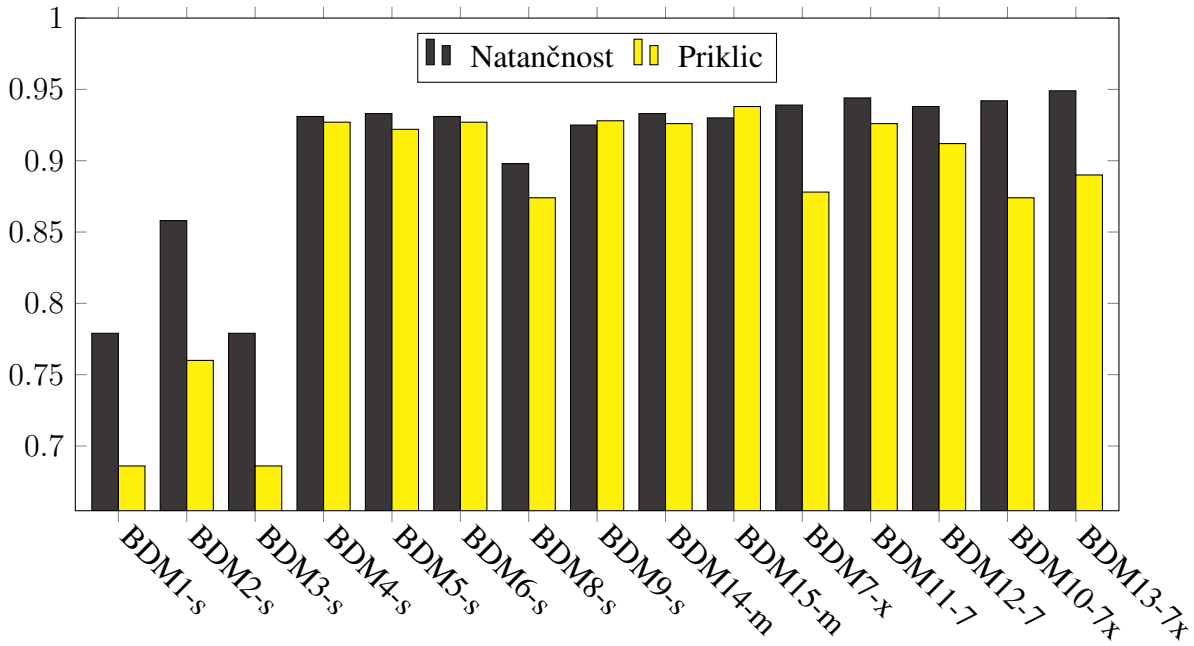


(e) Zaznava modela BDM2-s, vsa temna območja na panju

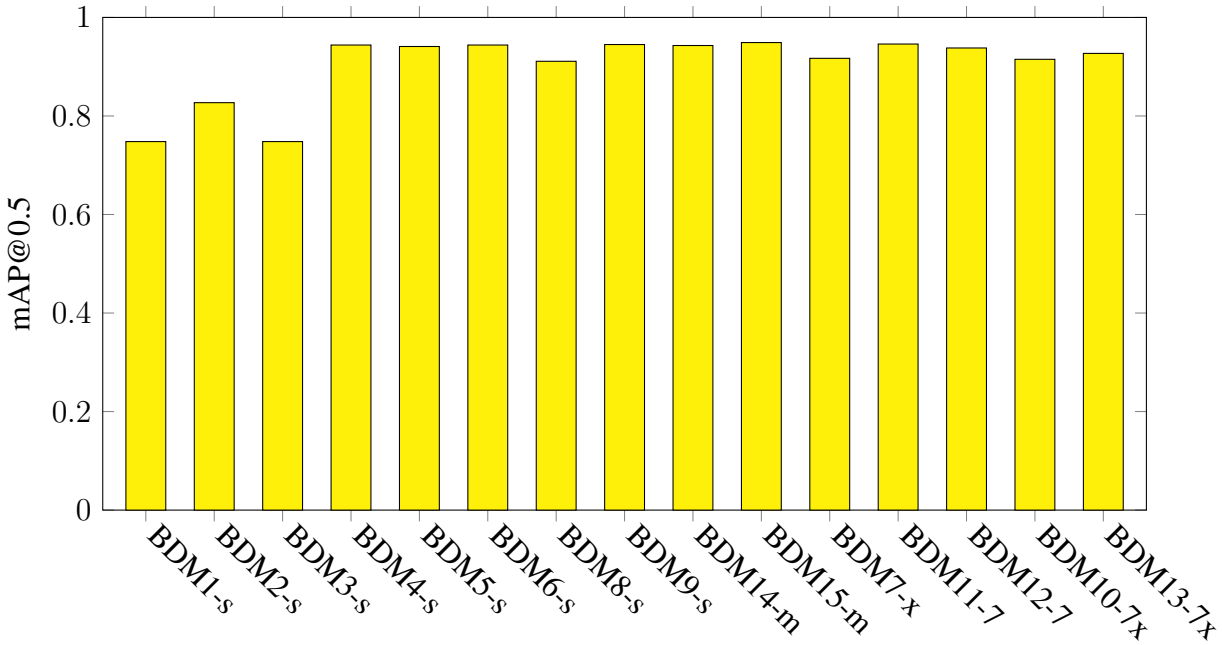


(f) Zaznava modela BDM3-s, vsa temna območja na panju

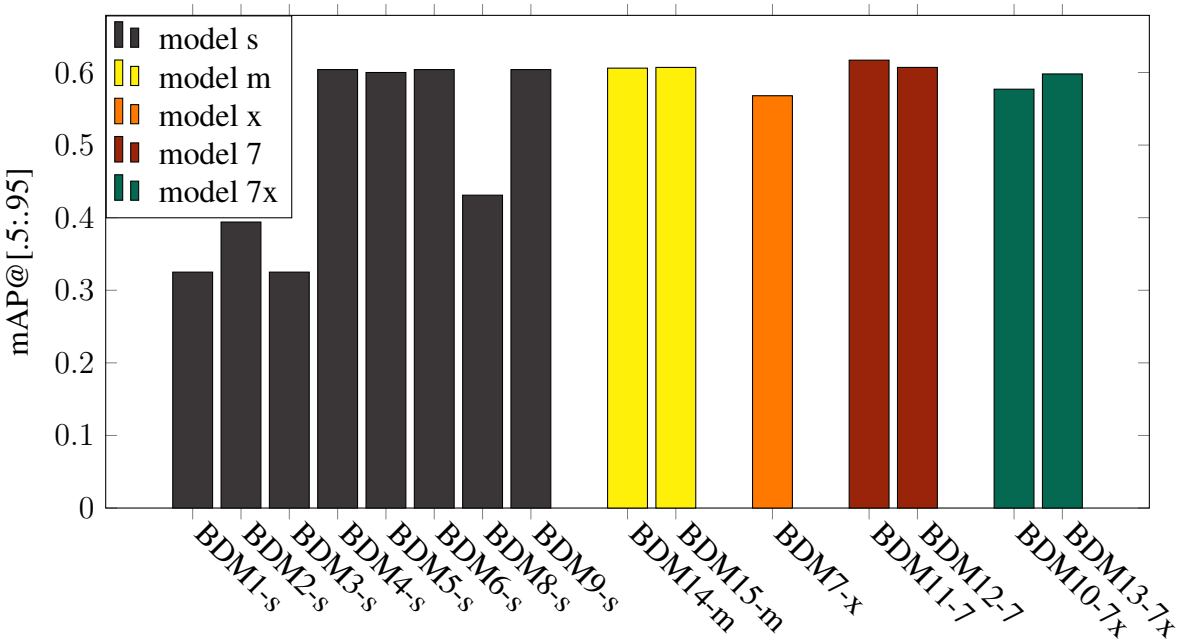
Slika 18: Nenatančne zaznave za šesti posnetek



Slika 19: Natančnost in priklic modelov strojnega učenja



(a) mAP@0.5



(b) mAP@[.5:.95]

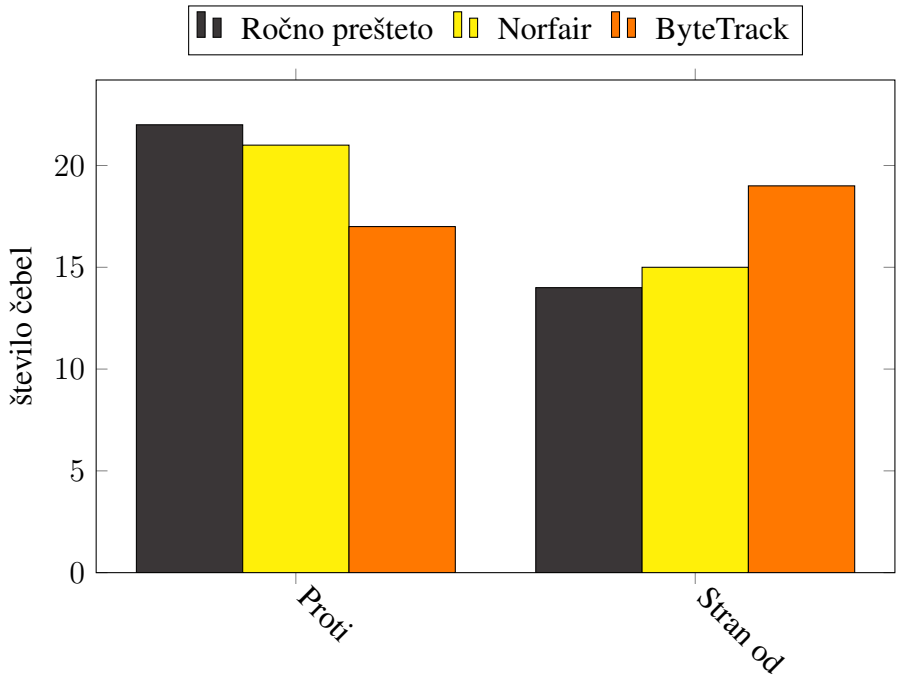
Slika 20: Kvaliteta zaznavanja modelov strojnega učenja

3.3 Natančnost sledenja čebelam

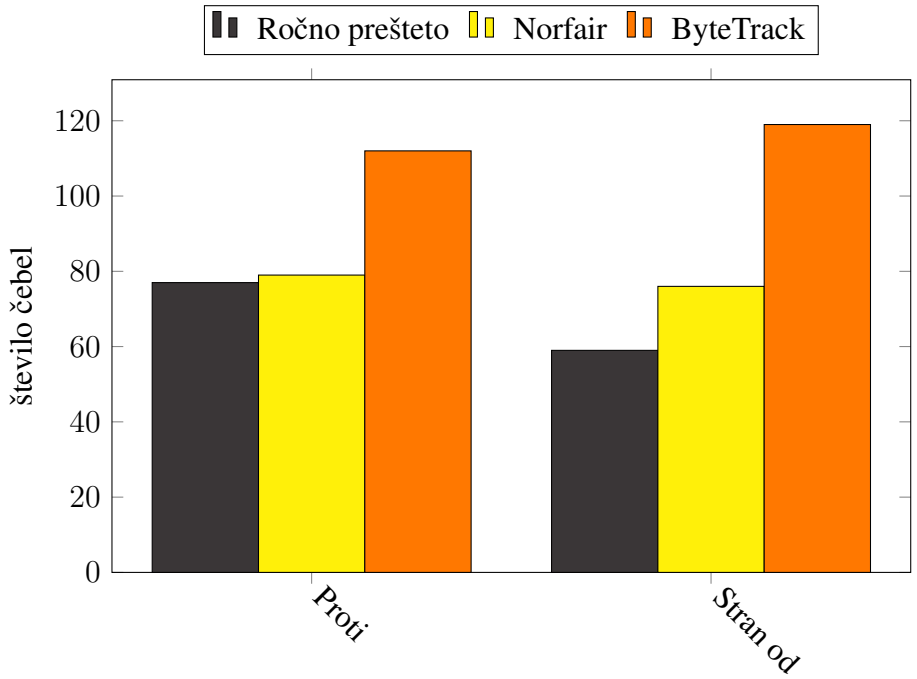
Pri primerjavi natančnosti štetja knjižnic vidimo, da je Norfair bolj natančen od ByteTracka, tako v štetju čebel, ki vstopijo v regijo okoli panja, kot tudi pri tistih, ki izstopajo iz območja (Slika 21). Norfair[9] se je še posebej izkazal pri štetju čebel, ki vstopajo v panj, saj od realnih rezultatov odstopa le za 2,5 % pri posnetku z več čebelami ter za 4,7 % pri posnetku z manj čebelami. ByteTrack[10] je slabši, z odstopanjem nad 40 % pri posnetku z več čebelami ter več kot 20 % pri posnetku z manj čebelami. Obe knjižnici zaznamuje tudi slabše štetje čebel, ko se oddaljujejo od panja. ByteTrack[10] izstopa s kar 101,7 % odstopanjem.

3.4 Hitrost sledenja čebelam

Skupno sledenje je bilo v vseh primerih hitrejše od 50 slik/sekundo (Slika 22a), kar je več kot dovolj za uporabo na posnetkih v živo, ki imajo običajno 25 slik/sekundo. Hitrost sledenja Norfair in ByteTrack je podobna. Primerjava časa, ki je potreben za procesiranja gibanja čebel, če odštejemo trajanje izvajanja modela pokaže, da je Norfair rahlo počasnejši na posnetku z več čebelami in hitrejši na posnetku z manj čebelami (Slika 22b), vendar je ta razlika v povprečju približno samo 0,8 milisekunde.

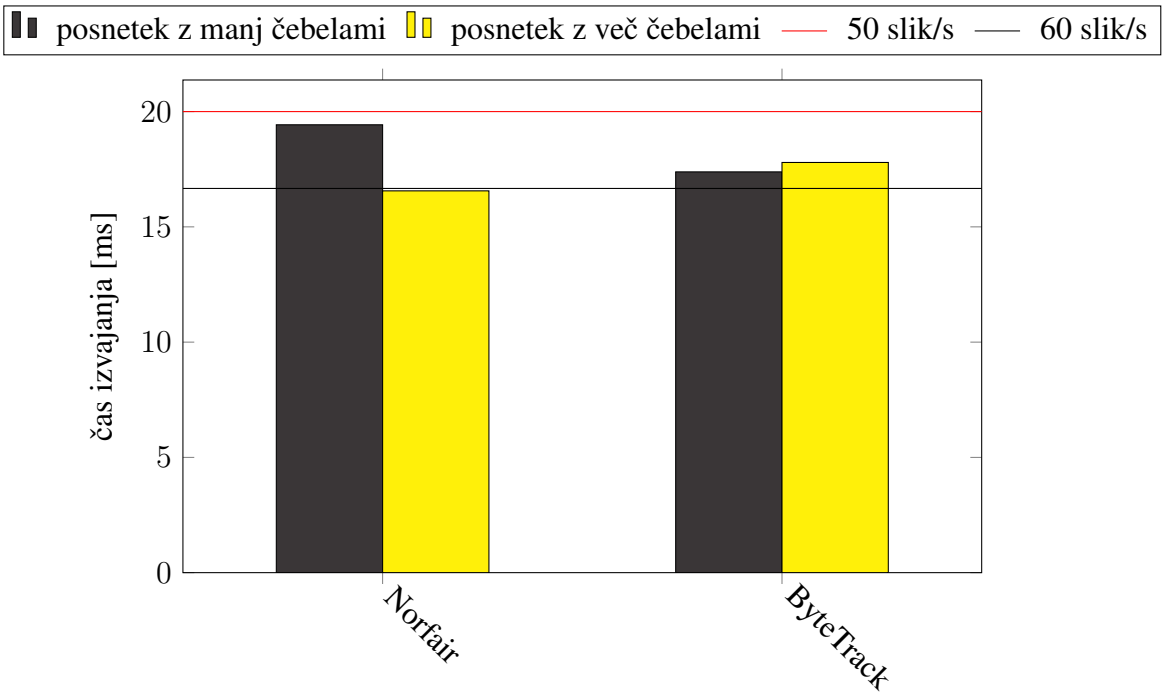


(a) Število letov proti in stran od panja na določenem območju, prešteto s pomočjo programov Norfair in Bytetrack ter ročno, na posnetku z manj čebelami

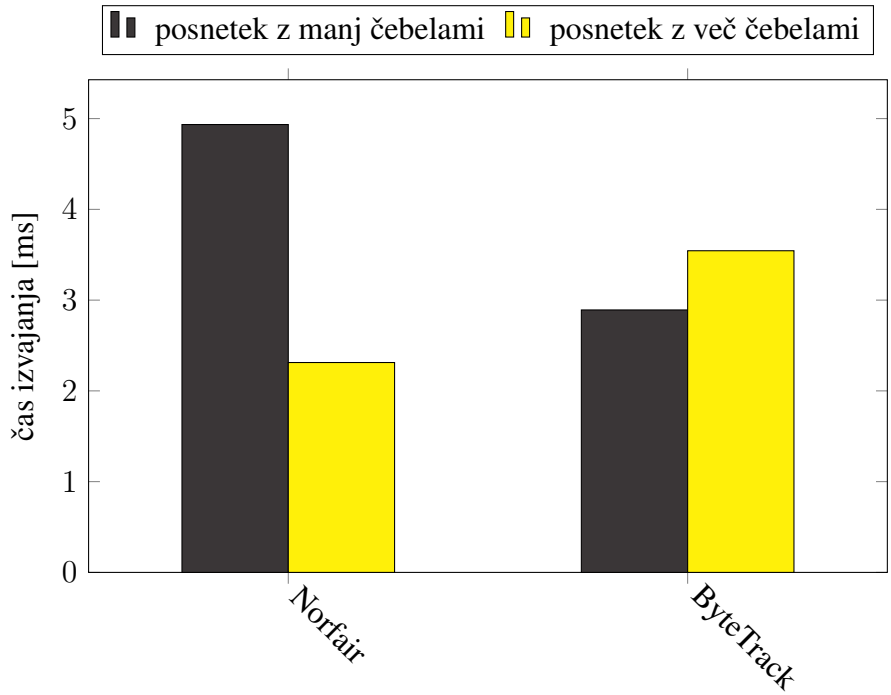


(b) Število letov proti in stran od panja na določenem območju, prešteto s pomočjo programov Norfair in Bytetrack ter ročno, na posnetku z več čebelami

Slika 21: Primerjava ročno preštetega števila letov čebel na določenem območju posnetka s sledenjem ByteTrack in Norfair



(a) Celotna povprečna hitrost sledenja čebelam na sliki posnetka z več in manj čebel. Na grafu sta označeni meji za 50 in 60 slik/sekudo



(b) Povprečna hitrost sledenja čebelam na sliki posnetka z več in manj čebelami z odštetim časom izvajanja zaznavanja

Slika 22: Hitrost izvajanja sledenja čebel

4 Diskusija

Gibanje čebel ni namenjeno le njihovemu premikanju iz kraja v kraj, temveč je povezano tudi z medsebojnim sporazumevanjem čebel. Raziskovalci so od sredine 20. stoletja temu fenomenu posvetili veliko časa, vse skupaj pa definirali kot čebelji jezik. Način premikanja čebel na panjskem žrelu je pri tem zelo pomemben. Čebele, ki se vračajo iz paše sporočajo drugim čebelam iz panja informacije o njeni kvaliteti, sestavi in oddaljenosti. S tem omogočajo večjo učinkovitost in varčnost z energijo za oskrbo družine[18]. Raziskovanje gibanja čebel je torej pomemben del razumevanja vedenja čebel in pomaga čebelarjem, ki se poklicno ukvarjajo z medonosno kranjsko sivko (*Apis mellifera carnica*).

Obstajajo različni načini opazovanja gibanja, od preprostega sistematičnega opazovanja do uporabe modelov strojnega vida. Kljub sorazmerno visoki natančnosti modelov, ki smo jih usposobili menimo, da bi se dalo usposobiti še bolj natančne modele. Večje število podatkov in boljše eksperimentalno okolje (grafična kartica z več grafičnega pomnilnika), bi omogočila še boljše rezultate. Ena od možnosti predstavlja uporaba tudi novejše arhitekture YOLOv8[19], ki je bila objavljena med izdelovanjem naloge in je zato še nismo uporabljali. Obstaja tudi možnost izvajanja programa preprostejši, specializirani strojni opremi, kar bi omogočilo zaznavo na kraju, kjer so bili posnetki pridobljeni[20]. Tudi s sestavo poligona, ki omeji hitrost gibanje čebel na posnetem območju ali samo s postavitvijo kamere bližje panja, bi lahko povečali uspešnost zaznave in sledenja. Z vključitvijo termalne kamere v zaznavo, bi lahko model še izboljšali. Nenazadnje, bi lahko natančnost algoritmičnega sledenja gibanja izboljšali tudi z uporabo umetne inteligence, z usposobitvijo nevronske mreže, ki bi zagotavljala boljšo časovno doslednost sledenja.

Iz zaznav in sledenja posameznih čebel bi se dalo določiti orientacijo telesne osi ali celo iskati cvetni prah na obnožnih koških čebel[21]. Podlaga naših rezultatov za zaznavanje in sledenje čebel, s pomočjo katerih lahko samodejno zberemo veliko število rezultatov, odpira možnosti za nadaljnje raziskave, na primer:

- uporaba posnetkov panja v živo, z označenimi zaznavami čebel in števcem števila čebel
- iskanje kompleksnih vzorcev za ugotavljanje zdravja čebel
- spremljanje aktivnost panja v odvisnosti od različnih spremenljivk (temperatura, vlažnost, letni čas, lokacija, ...)
- iskanje vzorcev v lokalnem gibanju čebel in vpliv dejavnikov okolja
- določitev vloge čebel na podlagi njihovega gibanja okoli panja
- globlje razumevanje medsebojne komunikacije in sodelovanja čebel z gibanjem
- spremljanje drugih živali s pomočjo podobnih modelov, usposobljenih na ustreznih podatkih

5 Zaključek

Čebele so ključne za obstoj življenja, kot ga poznamo danes. Njihova vloga kot opraševalci je nepogrešljiva za razmnoževanje rastlinskih vrst. Posebno pomemben dejavnik so pri opraševanju kulturnih rastlin. Avtomatizirano zbiranje podatkov o čebelah ni uporabno le za čebelarje, ampak lahko tudi podpira mnogo drugih raziskav o vedenju in delovanju teh čudovitih živali. Z uporabo računalniškega vida in metodami sledenja večim objektom (MOT) nam je uspelo dokazati, da je taka avtomatizacija mogoča, natančna in uporabna.

5.1 Pregled hipotez

Med izdelavo naloge smo pregledali vse hipoteze:

✓ **1: Izvajanje sledenja čebel je dovolj hitro, da se izvaja na posnetkih v živo** Potrjena, saj sledenje poteka na hitrosti okoli 50-60 slik/sekundo

✓ **2: Pri hitrosti izvajanja modelov je ključna izbira predusposobljenega modela** Potrjena, saj je neposredna korelacija med hitrostjo modela in izbiro predusposobljenega modela

× **3: Modeli družine YOLOv7 so hitrejši in bolj natančni, kot modeli YOLOv5** Ovržena, saj so modeli YOLOv7 bistveno počasnejši, kot modeli YOLOv5 pri natančnostim ki je podobna boljšim modelom YOLOv5

✓ **4: Sledenje čebel deluje tudi na posnetkih, kjer je razdalja med čebelami majhna** Potrjena, saj zaznava in sledenje na teh posnetkih v veliki večini

× **5: Zaznava deluje hitreje na posnetkih z manjšim številom čebel** Ovržena, saj med posnetkom in hitrostjo zaznave ni korelacije

Literatura

[1] I. Esenko, *Svet čebel: o čebelah in čebelarstvu v Sloveniji*, 1. izd. Družina, 2018, ISBN: 978-961-04-0496-5.

[2] P. Kozmus, B. Noč in K. Kalan, *Brez čebel ne bo življenja*, 1. izd. Beebooks, 2017, 1.500 izv., ISBN: 978-961-94251-0-7.

[3] R. Menzel, U. Greggers, A. Smith in sod., "Honey bees navigate according to a map-like spatial memory," en, *Proceedings of the National Academy of Sciences of the United States of America*, let. 102, št. 8, str. 3040–3045, feb. 2005.

[4] *HiveTracks*. spletni naslov: `https://www.hivetracks.com` (pridobljeno 26. 2. 2023).

[5] *Hivebloom*. spletni naslov: `https://hivebloom.com/` (pridobljeno 26. 2. 2023).

[6] *BuzzBox*. spletni naslov: `https://www.osbeehives.com/` (pridobljeno 26. 2. 2023).

[7] OpenCV, *Open Source Computer Vision Library*. spletni naslov: `https://opencv.org/` (pridobljeno 26. 2. 2023).

[8] B. Dwyer in J. Nelson, ver. 1.0, 2022. spletni naslov: `https://roboflow.com` (pridobljeno 26. 2. 2023).

[9] Tryolabs, *Norfair*. spletni naslov: `https://github.com/tryolabs/norfair` (pridobljeno 26. 2. 2023).

[10] Y. Zhang, P. Sun, Y. Jiang in sod., "ByteTrack: Multi-Object Tracking by Associating Every Detection Box," 2022.

[11] P. Dendorfer, H. Rezatofighi, A. Milan in sod., "MOT20: A benchmark for multi object tracking in crowded scenes," *CoRR*, let. abs/2003.09003, 2020. arXiv: `2003.09003`. spletni naslov: `https://arxiv.org/abs/2003.09003`.

[12] G. Jocher, A. Chaurasia, A. Stoken in sod., *ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation*, nov. 2022. DOI: `10.5281/zenodo.3908559`. spletni naslov: `https://github.com/ultralytics/yolov5` (pridobljeno 26. 2. 2023).

[13] C.-Y. Wang, A. Bochkovskiy in H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," jul. 2022. DOI: 10.48550/ARXIV.2207.02696. spletni naslov: https://github.com/WongKinYiu/yolov7.

[14] *Urbani učni čebelnjak, Botanični vrt Ljubljana*. spletni naslov: http://www.botanicni-vrt.si/urbani-ucni-cebelnjak (pridobljeno 28. 2. 2023).

[15] R. E. Kálmán, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME–Journal of Basic Engineering*, let. 82, št. Series D, str. 35–45, 1960.

[16] P. Skalski, *onemetric*, https://github.com/SkalskiP/onemetric/, 2021. spletni naslov: https://github.com/SkalskiP/onemetric/ (pridobljeno 26. 2. 2023).

[17] M. Everingham in J. Winn, "The pascal visual object classes challenge 2011 (voc2011) development kit," *Pattern Analysis, Statistical Modelling and Computational Learning, Tech. Rep*, let. 8, 2011.

[18] J. Božič, "Dancing with carniolan bee / Plešemo s kranjsko čebelo," *Folia biologica et geologica*, let. 59, str. 5, jan. 2019. DOI: 10.3986/fbg0041.

[19] G. Jocher, A. Chaurasia in J. Qiu, *YOLO by Ultralytics*, ver. 8.0.0, jan. 2023. spletni naslov: https://github.com/ultralytics/ultralytics (pridobljeno 26. 2. 2023).

[20] R. Pilipović, V. Risojević, J. Božič, P. Bulić in U. Lotrič, "An Approximate GEMM Unit for Energy-Efficient Object Detection," *Sensors*, let. 21, št. 12, 2021, ISSN: 1424-8220. DOI: 10.3390/s21124195. spletni naslov: https://www.mdpi.com/1424-8220/21/12/4195.

[21] Z. Babic, R. Pilipovic, V. Risojević in G. Mirjanic, "Pollen Bearing Honey Bee Detection in Hive Entrance Video Recorded by Remote Embedded System for Pollination Monitoring," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, let. III-7, str. 51–57, jun. 2016. DOI: 10.5194/isprs-annals-III-7-51-2016.

[22] V *ROSUS 2022 : računalniška obdelava slik in njena uporaba v Sloveniji 2022 : zbornik 16. strokovne konference*, B. Potočnik, ur., Univerzitetna založba Univerze, apr. 2022. DOI: 10.18690/978-961-286-337-1.

citirano po IEEE standardu