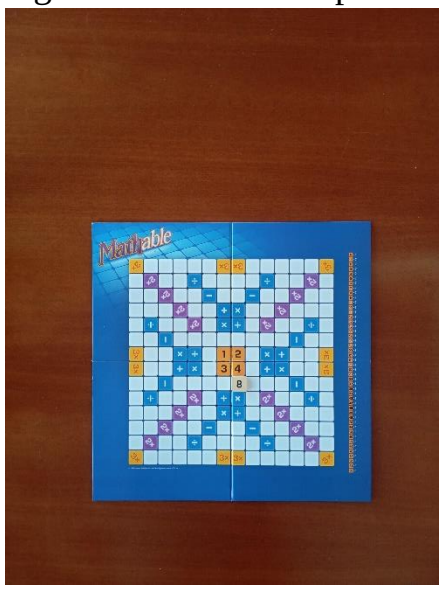
Tema 1

Primul script: Decuparea initiala a tablei de joc folosind HSV

Scriptul se focalizeaza pe imaginile initiale din fisierul cu cele 200 de imagini Figura(1), scopul este de a extrage noi imagini cu tabla de joc folosindu-se de pe segmentarea culorilor pentru a obtine marginea tablei si a salva noile imagi



Imagine initiala (Figura 1)

Tabla de joc (Figura 2)

Input:

-path-ul catre cele 200 de imagini.

-valorile HSV low si high pentru detectarea ramelor albastre in jurul tablei.

Aspecte ale codului:

-Citirea imaginii plus conversia HSV.

```
img = cv2.imread(image_path)
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
```

-Crearea mastilor pentru segmentele albastre.

```
blue_mask = cv2.inRange(hsv, np.array([100, 100, 50]), np.array([140, 255, 255]
```

-Detectarea si extragerea conturului care este ce l mai mare.

```
contours, _ = cv2.findContours(blue_mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
if not contours:
    print(f"Nu s-a gasit tabla {image_path}")
    return
```

Output:

-Imaginile de test cropuit astfel incat sa fie prezenta doar tabla de joc (Figura 2.)

Al doilea script: Decuparea logo-ului si al marginii din imaginile noi.

Scriptul selecteaza imaginiile noi prin alegerea manuala a unor parametrii x si y, Se standardizeaza si elimina artefacte. Noua ne trebuie doar grid-ul de 14x14 pentru a crea o matrice patratica pe care putem opera, astfel putem imparti in patrate egale corespunzatoare cu pozitiile pieselor care se vor adauga la fiecare mutare.

Input:

-Imaginile prelucrate dupa primul script.

-Coordonatele alese manual.

Aspecte ale codului:

-Decupare folosind coordonatele noi, acestea sunt fixe:

```
CROP_X_START = 136
CROP_Y_START = 140
CROP_X_END = 885
CROP_Y_END = 890
```

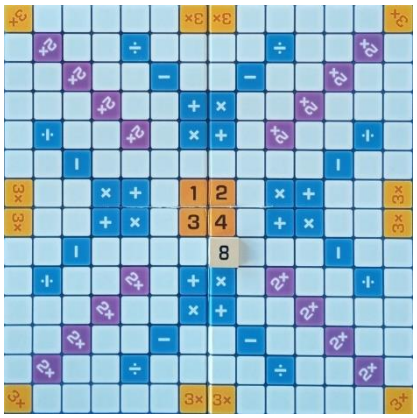
```
cropped = img[CROP_Y_START:CROP_Y_END, CROP_X_START:CROP_X_END]
```

-Salvarea imaginilor noi in alt folder.

```
  cv2.imwrite(output_path, cropped)
  print(f"Noua tabla de joc a fost salvata {output_path}")
```

Output:

Imaginile taiate finale cu grid-ul de 14x14 care ne interesa. (Figura 3)



Grid 14x14 (Figura 3)

Al treilea script: Generarea Mastilor Binare care sunt o urmare a diferentlor intre imagini.

In urma ultimului laborator la CAVA, am fost sfatuiti ca o buna practica, sa privim imaginile ca niste frame-uri cu background constant, unde obiectele “care

se msica” sa fie chiar pisele de joc adaugate la fiecare frame. Astfel se calculeaza diferentele intre starile tablei de joc si apoi genereaza mastile binare.

Input:

-Imaginile din starile prcedente, pentru prima imagine se foloseste starea tablii de joc goala din imaginii_auxiliare, bineinteles, prelucrata.

Aspecte ale codului:

-Calcularea diferentei absolute.

```
diff = cv2.absdiff(img1, img2)
```

-Crearea mastii binare cu threshold.

```
_, mask = cv2.threshold(diff, threshold, 255, cv2.THRESH_BINARY)
```

-Salavarea mastilor.

Output:

-Mastile binare salvate in alt path. (Figura 4,5)

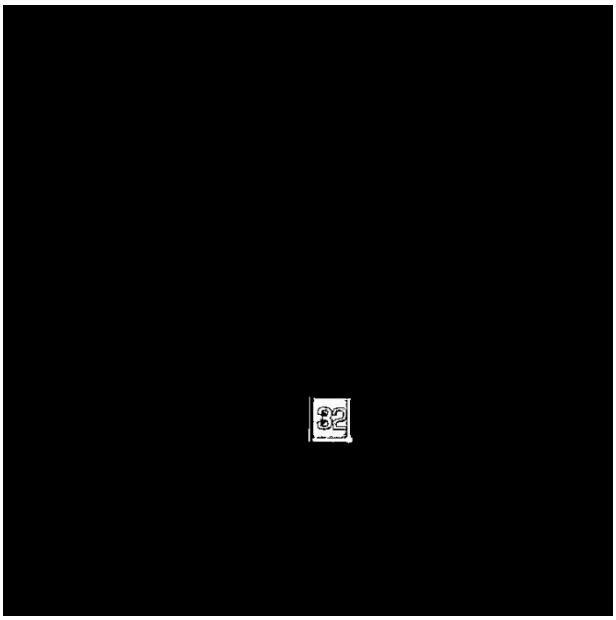


Figura 4. Masca binara

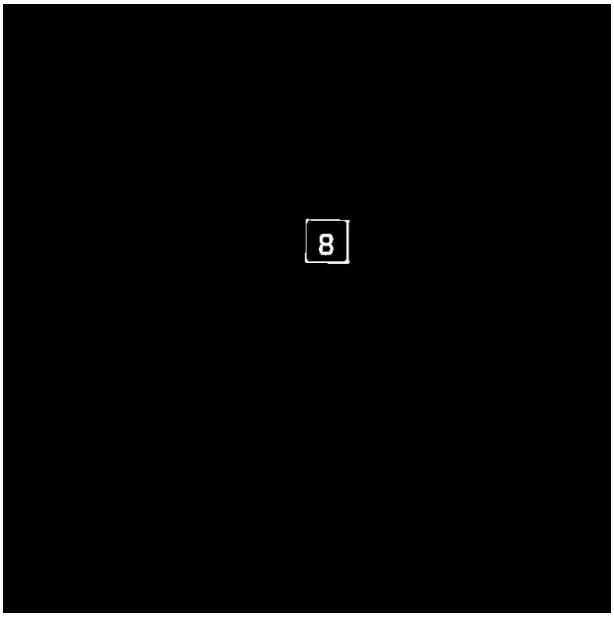


Figura 5. Masca binara

Al patrulea script: Detectare pozitiei si cropuirea patratelor din grid cel mai mult modificate.

Se analizeaza care sunt patratele cel mai mult modificate si se decupeaza in alt path, dar se calculeaza si pozitia din grid a acestora.

Input:

- Mastile calculate precedent.

Aspecte ale codului:

- Impartirea tablei de joc in 14x14.

```
grid_size = 14  # 14x14 grid

def crop_most_affected_tile(mask):


    mask_height, mask_width = mask.shape

    tile_height = mask_height // grid_size
    tile_width = mask_width // grid_size
```

- Se numara ce patrat are cei mai multi pixeli albi, acestia denumesc pixelii schimbati de la un frame la altul.

```
white_pixel_count = np.sum(tile == 255)
```

- Salvarea patratului cel mai mult modificat.

Output:

-Patratele cel mai mult modificate (Figura 6,7).

-Fisier txt pentru fiecare poza cu pozitia patratului cel mai mult modificat. (Figura 8)


Figura 6 Masca unui numar adaugat.


Figura 7 Masca unui numar adaugat.

FIgura 8 Pozitia numarului adaugat in fisierul txt.

Al 5-lea script: OCR pe noile patrate si modificare txt-urilor cu pozitiile.

Se foloseste OCR pe patratele decupate si se adauga la finalul txt-urilor cu pozitiie numarul intrepretat

Input:

-Imaginili cu patratele

-Fisierele cu pozitiile patratelor in grid

Aspecte ale codului:

-Preprocesare pentru OCR

```
_, thresh = cv2.threshold(img, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)
resized = cv2.resize(thresh, (100, 100), interpolation=cv2.INTER_CUBIC)
```

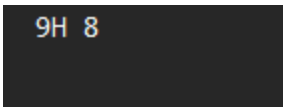
-Extractie text folosind easyOCR

```
result = reader.readtext(preprocessed_image, detail=0)
```

-Adaugarea textului cu numarul intrepretat la finalul txt-ului cu pozitia.

Output:

-Txt-uri cu pozitia si numarul adaugat la fiecare mutare a jucatorului. (Figura 9)



Pozitia unui numar adaugat alaturi de intrepretarea numarului in documentlui txt.