Calculator automat de scor pentru Scrabble

- documentație -

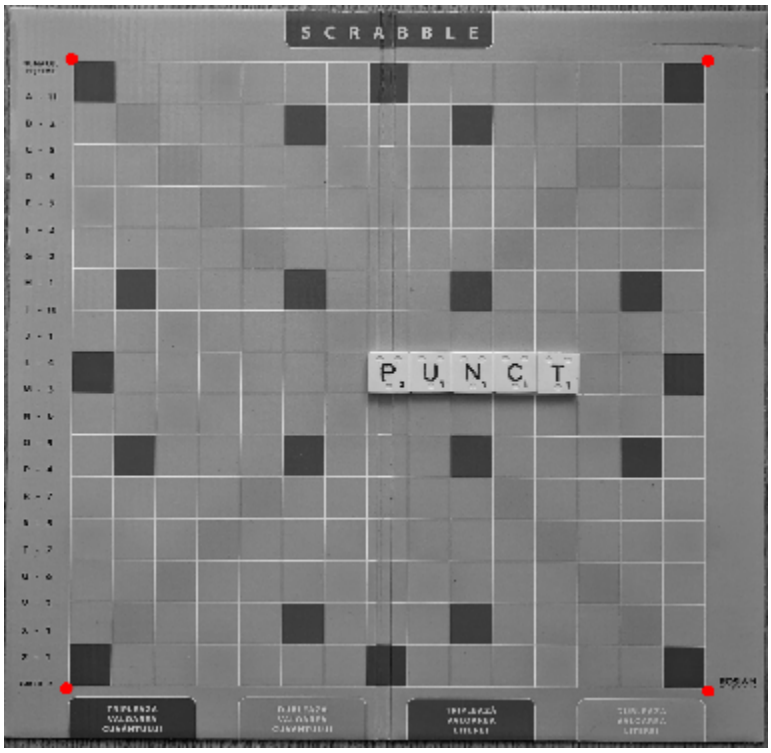
Task 1 - detectarea pozițiilor pe care sunt depuse literele în fiecare rundă

Pentru realizarea acestui task a fost mai întâi necesară crearea unei funcții de detectare a tablei de joc. Imaginea primită ca input este mai întâi convertită în grayscale, deoarece culoarea nu este relevantă pentru detectarea tablei, apoi sunt folosite o serie de funcții de procesare a imaginii care au ca rol scoaterea în evidență a careului de joc:

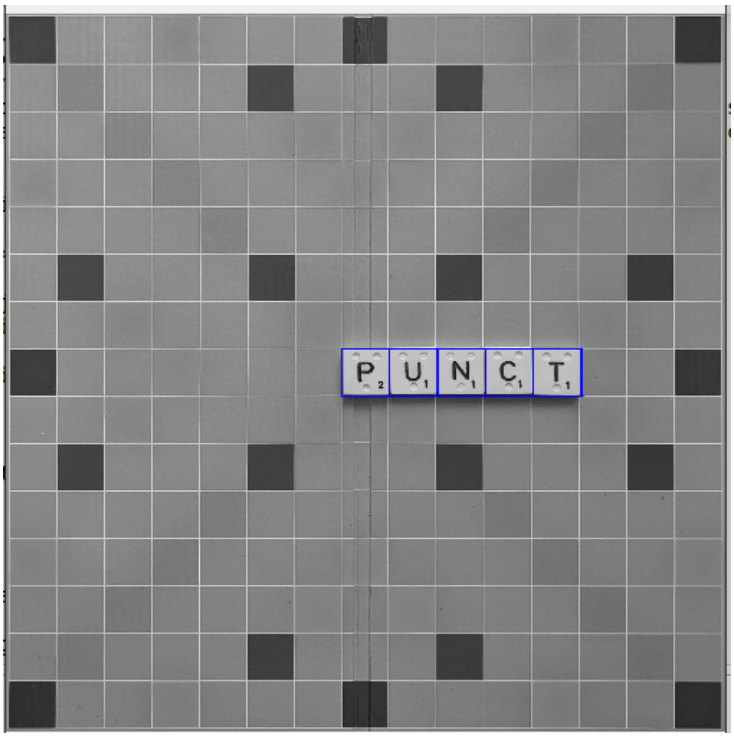
```
image = cv.cvtColor(image,cv.COLOR_BGR2GRAY)
image_m_blur = cv.medianBlur(image,3)
image_g_blur = cv.GaussianBlur(image_m_blur, (0, 0), 3)
image_sharpened = cv.addWeighted(image_m_blur, 1.47, image_g_blur, -0.9, 0)
#show_image('image_sharpened',image_sharpened)
_, thresh = cv.threshold(image_sharpened, 115, 255, cv.THRESH_BINARY)

kernel = np.ones((8, 8), np.uint8)
thresh = cv.dilate(thresh, kernel)
#show_image('image_thresholded',thresh)

edges =  cv.Canny(thresh ,200,400)
edges = cv.dilate(edges, kernel)
#show_image('edges',edges)
contours, _ = cv.findContours(edges,  cv.RETR_EXTERNAL, cv.CHAIN_APPROX_SIMPLE)
```

Este aplicată o funcție de sharpen, ce reduce detaliile inutile precum umbre și reflexii, un threshold ce face toți pixelii ce au intensitatea mai mare de un prag albi și pe restul negri pentru a evidenția careul care este desenat cu o culoare deschisă, un dilate care îngroașă zonele de pixeli albi, îngroșând astfel liniile careului, funcția Canny pentru a găsii muchiile din imagine și

încă un dilate pentru a face muchiile mai definite, și findContours care caută toate contururile din imagine.
După sunt alese colțurile conturului cel mai din exterior, adică al întregului careu de joc și este aplicată funcția getPerspectiveTransform pe poza neprocesată pentru a obține imaginea dintre cele patru puncte alese sub forma unui pătrat perfect ce poate fi împărțit într-un grid de 15 x 15.



Detectarea pozițiilor pe care se află piese se face prin aplicarea unui threshold pe imaginea returnată de funcția de extragere a tablei de joc. Deoarece piesele au o culoare deschisă, aplicarea unui threshold înalt va face ca toți pixelii care nu fac parte din piesă sau careul desenat pe tablă să devină negri. Pentru fiecare poziție de pe tablă fac media culorii, iar dacă aceasta nu este 0 sau aproape 0 înseamnă ca pe poziția respectivă se află o piesă.



Pentru a determina care piese sunt noi adăugate se compară matricea de poziții curente cu cea anterioară.

Task 2 - recunoașterea literelor de pe piese

Recunoașterea literelor de pe tablă a fost realizată prin template matching.
Template-urile au fost obținute prin aplicarea funcției de extragere a careului pe o imagine cu o tablă de joc ce avea câte un exemplar din fiecare piesă pe o poziție și memorarea imaginilor cu piesele într-o listă.
Pentru fiecare poziție de pe tablă pe care a fost detectată o piesă este folosită funcția matchTemplate pentru a compara piesa curentă cu toate template-urile din listă și reține litera pentru care este returnată cea mai mare valoare.

Task 3 - calcularea scorului

Pentru a calcula scorul la fiecare etapă a jocului se compară matricea cu literele curente cu cea anterioară (matricea goală în cazul primei runde) și sunt salvate pozițiile pieselor noi într-o listă. Este verificată coordonata comună a pieselor pentru a afla dacă cuvântul este pe orizontală sau pe verticală. Pentru fiecare literă nou adăugată se verifică dacă formează cuvânt și pe cealaltă direcție pentru a îl aduna la scorul final, precum și dacă se află pe o poziție ce multiplică scorul. Multiplicarea pe cuvânt este aplicată după parcurgerea tuturor literelor pentru a obține scorul corect.