

Calculator automat de scor pentru jocul Qwirkle

Dragomir Daria Nicoleta
Grupa 352
daria-nicoleta.dragomir@s.unibuc.ro

1 Metodologia de Procesare a Imaginilor

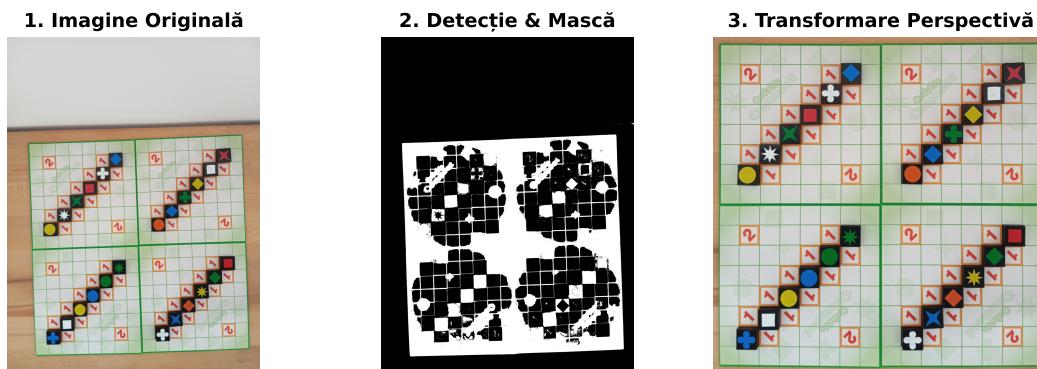
Scopul acestui proiect este dezvoltarea unui sistem automat care calculează scorul jocului Qwirkle, utilizând concepte de Vedere Artificială. Algoritmul de procesare este împărțit în mai multe etape distincte, de la preluarea imaginii brute până la identificarea piesei și calculul scorului.

1.1 Extragerea Tablei de Joc

Pentru a analiza tabla de joc, aceasta trebuie izolată din imaginea originală și adusă într-o perspectivă "top-down" (vedere de sus).

Primul pas este convertirea imaginii în spațiul de culoare HSV. Se aplică o filtrare HSV pentru a separa tabla de fundal, urmată de operatori morfologici (eroziune, dilatare) și un blur median pentru eliminarea zgromotului. Apoi, se identifică conturul maxim care aproximează forma tablei.

În final, se calculează matricea de proiecție folosind cele patru colțuri ale conturului detectat. Pentru a evita tăierea pieselor aflate la margine, proiecția tablei de joc va conține și o mică parte din masa pe care se află (buffer de 32 pixeli în fiecare direcție).



1.2 Identificarea Configurării Tablei

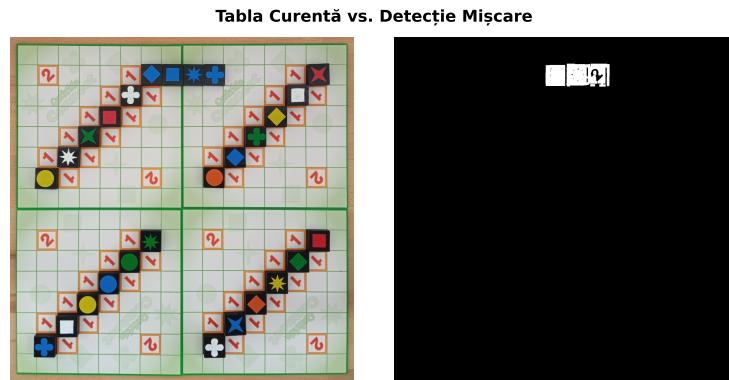
Deoarece fiecare joc poate avea o configurație initială diferită, sistemul trebuie să detecteze orientarea corectă a grilei. Astfel, vom considera tabla formată din 4 sferturi, unde fiecare sfert poate avea 2 orientări posibile (piesele se află pe diagonala principală (1) sau secundară (0)).

Pentru a identifica orientarea fiecarui sfert de tablă, se verifică poziționarea pătratelor de bonus (2 puncte) cu ajutorul template matching-ului cu template-urile pentru cifra 2. Piese sunt poziționate pe diagonala opusă pătratelor de bonus. Pentru rezultate mai bune, am verificat doar pătratele în care s-ar fi putut afla cifra 2 pe tablă.

1.3 Detectarea Pieselor Adăugate

Pentru a identifica pozițiile în care s-au adăugat piese noi, comparăm starea curentă a tablei cu cea anterioară. Prin calcularea diferenței absolute (framediff) între imaginea tablei curente și cea anterioară. Diferența este apoi procesată printr-un filtru Gaussian și unul Median, urmate de operații morfologice pentru a umple goliurile din interiorul pieselor detectate.

În continuare, imaginea obținută este împărțită în grila de 16x16 a jocului. Pentru fiecare celulă a grilei, se numără pixelii activi (voteForSquare()). Dacă o celulă conține peste 3000 de pixeli activi, se consideră că acolo a fost amplasată o piesă nouă



1.4 Clasificarea Pieselor

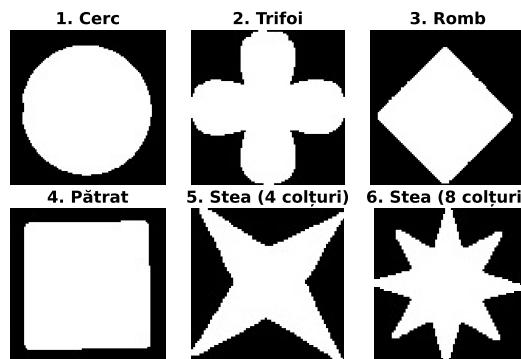
Odată ce o piesă nouă este localizată, aceasta trebuie clasificată. Acest proces se desfășoară în doi pași:

1.4.1 Identificarea Colorii

- Se extrage pătratul în care a fost amplasată o nouă piesă.
- Se aplică succesiv 6 măști HSV, corespunzătoare culorilor din joc (Roșu, Albastru, Verde, Galben, Portocaliu, Alb).
- Filtrul care produce cei mai mulți pixeli albi în masca binară este desemnat ca fiind culoarea corectă.

1.4.2 Identificarea Formei

- Se utilizează masca binară rezultată din pasul anterior.
- Se aplică Template Matching pe această mască folosind săabloanele celor 6 forme posibile.
- Se caută cea mai bună potrivire (scor maxim) la mai multe scale ratios = [0.8, 0.85, 0.9, 0.95] pentru a trata variațiile mici de dimensiune.



2 Calculul Scorului

Logica de joc se află în clasa QwirkleScorer. Aceasta ține în memorie o matrice 16x16 (board_state) care este actualizată la fiecare tură.

Pentru fiecare piesă nouă, se identifică liniile formate pe orizontală și verticală prin utilizarea unui set(), pentru a ne asigura că o linie nu este punctată de două ori în aceeași tură.

Pentru fiecare piesă din liniile noi formate sau prelungite, se acordă 1 punct (funcția getLineCoordinates()). Dacă o piesă amplasată în această tură se află pe un pătrat bonus, se adună și acest punctaj. În final, dacă lungimea liniei este 6, se acordă un bonus suplimentar de 6 puncte.