Tema 1
Hirica Ioan Alexandru

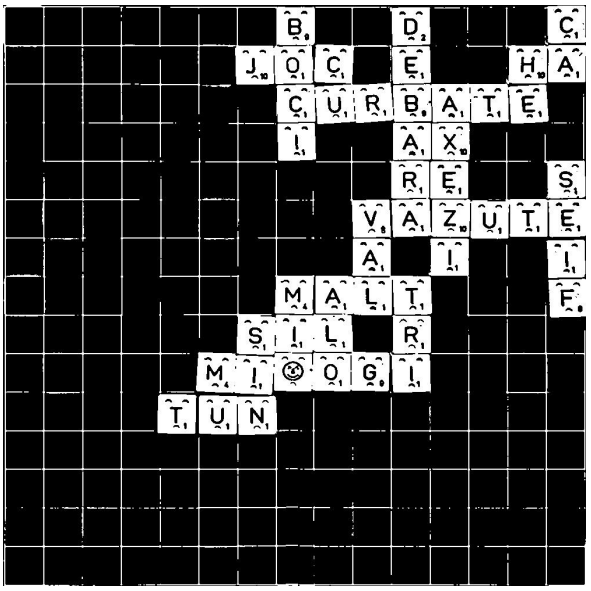
Etapa 1, recunoasterea tablei de joc:

Pentru a recunoaste tabla primul lucru pe care l-am facut a fost sa aplic un sharpen imaginii folosind exemplul din laborator, si anume am aplicat un blur median, apoi un blur gaussian si am combinat aceste doua imagini.
Dupa acest pas, am dilatat imaginea rezultata de doua ori folosind un kernel de 5x5, pentru a ingrosa liniile careului.
Am aplicat un threshold binar imaginii rezultante, care a fost transformata in alb negru: `ret,thresh1 = cv.threshold(imgray,80,255,cv.THRESH_BINARY)`
Pentru a gasi careul de joc am folosit functia din opencv, “findContours” si am selectat conturul care are aria maxima. Pentru a gasi ariile contururilor am folosit functia `cv.contourArea(cnt)` unde cnt este un contur.
Dupa aceea am facut transformarea de perspectiva folosind conturul gasit intr-o imagine de 1000x1000, alegerea colturilor fiind la fel ca in exemplul din laborator.

Etapa 2, gasirea pozitiilor ocupate de catre piese:

Pentru a gasi pozitiile ocupate de catre piese am folosit imaginea alb negru a zonei de joc careia i-a fost aplicata transformarea de perspectiva si algoritmul de sharpen.
Am folosit un threshold binar ca mai sus avand valorile (70, 255).

Am observat ca in imaginea transformata o casuta are aproximativ dimensiunile de 65x65 si astfel am reusit sa iau fiecare casuta de joc. Deoarece imaginea a fost binarizata valorile acesteia erau fie de 255 (pixel alb), fie de 0 (pixel negru). Pentru fiecare pozitie am obtinut patratul si i-am facut suma pixelilor. Aceasta suma am impartit-o la 255 pentru a obtine numarul de pixeli albi. Daca exista mai mult de 1000 de pixeli albi am considerat ca in aceasta locatie se afla o litera si astfel am adaugat pozitia intr-un vector de pozitii.



Imaginea care este folosita pentru a gasi pozitiile

Etapa 3, identificarea pieselor:

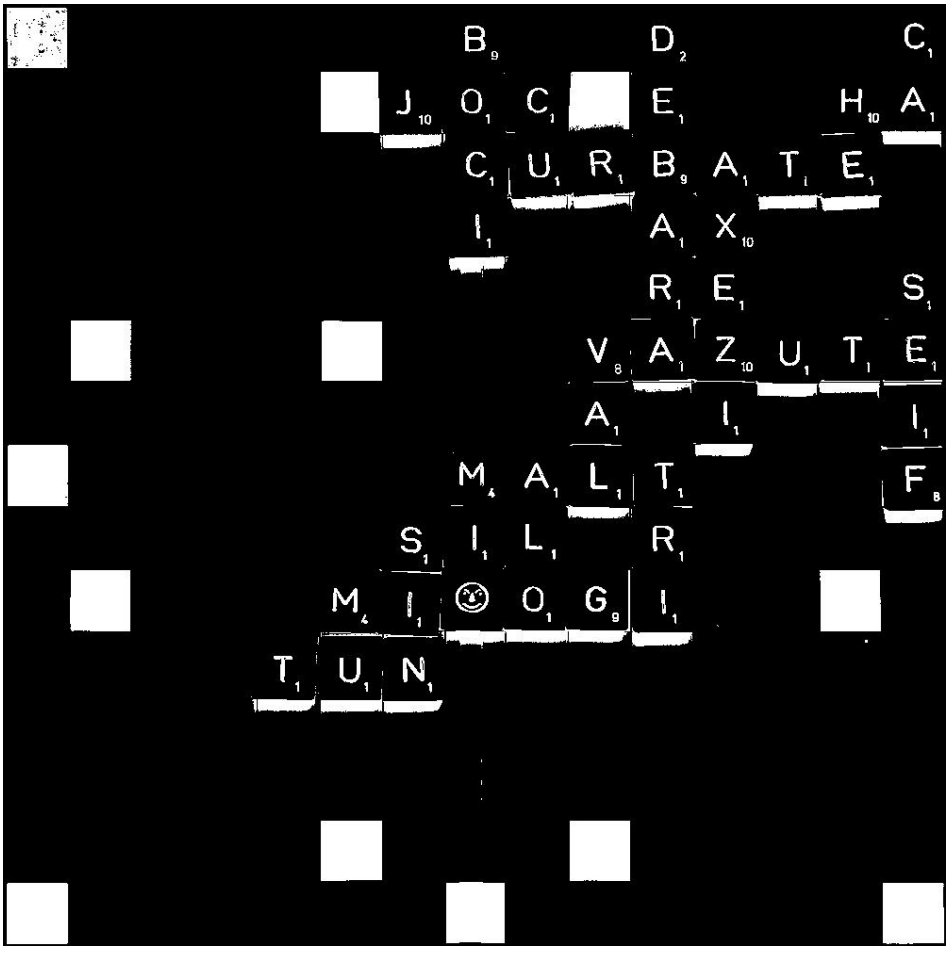
Pentru a identifica piesele am facut manual poze cu toate literele, fiecare sa aiba o dimensiune de 32x32, cuprinzand doar litera

Exemplu litera template

Pentru fiecare imagine template, aceasta a fost augumentata folosindu-se de rotatiile sale. Am adaugat un nou set de imagini astfel incat sa existe si rotatiile de la [-5, 5] grade folosind un pas de 0.5.

Apoi pentru fiecare pozitie gasita ca fiind ocupata am facut template matching, si am gasit valoarea maxima de corespondenta. Daca valoarea aceasta depaseste valoarea 0.6 atunci returnez litera pe acea pozitie.

In cazul in care nu se gaseste o litera in aceasta pozitie, atunci pozitia nu o sa mai fie marcata ca fiind ocupata.



Imaginea folosita pentru gasirea valorilor pieselor.

Eliminarea casutelor albe nu este necesara intrucat ele nu sunt marcate ca fiind ocupate in etapa precedenta.

Etapa 4, calculul scorului:

Pentru fiecare pozitie noua descoperita la pasul i, i-am atribuit un tuplu avand valorile (Fals, Fals). Primul fals reprezinta faptul ca pozitia nu a fost luata considerare in calculul scorului pe verticala, iar celalalt reprezentand calculul orizontal. Se parcurge fiecare pozitie noua iar in cazul in care piesa nu a fost calculata pe verticala, algoritmul va gasi inceputul cuvantului din care face parte pe verticala, parcurgand in “sus” fiecare casuta. Se va returna pozitia de unde se incepe cuvantul, iar apoi algoritmul va porni in “jos” calculand scorul conform enuntului. In aceasta parcurgere in cazul ca este intalnita o litera noua, in cadrul acelui tuplu prima valoare va fi atribuita cu True.
Acelasi lucru se intampla si cazul parcurgerii pe orizontala.