DOCUMENTATIE

Tema 1

Petcu Mircea
Grupa 334
Facultatea de Matematica si Informatica
Concepte si Aplicatii in Vederea Artificiala

1 Task 1

1.1 Extragerea tablei de joc

Am inceput prin a extrage tabla de joc in intregime din fiecare imagine pe care o primesc pentru procesare. M-am folosit de spatiul HSV, aplicand 2 limite de Hue, Value si Saturation ((24,0,0) si (255, 255, 255)), scapand astfel de masa (masa s-a innegrit) pe care era asezata tabla de joc [figura 2]. Am convertit imaginea in grayscale si am aplicat filtrele de medie si Gaussian pentru a scapa de zgomotul din imagine.

Pentru a putea gasi mai bine muchiile, am aplicat un filtru Sobel atat pentru liniile orizontale, cat si pentru cele verticale. Pentru a scapa de zgomotul de tip piper (agresiv) produs de filtrul Sobel, m-am folosit de un filtru bilateral.

Am convertit imaginea in imagine binara [figura 3] si am aplicat Canny (algoritmul lui Canny din OpenCV) pentru a detecta muchiile. M-am folosit de functia din opencv findContours() pentru a gasi toate contururile si am folosit o bucata de cod din laboratorul 6 de CAVA pentru a gasi cel mai mare contur (cel al tablei de joc) si am folosit functiile din OpenCV: getPerspectiveTransform si warpPerspective, pentru a schimba si pentru a aplica perspectiva asupra imaginii (din colturile gasite anterior a celui mai mare contur).

Astfel, dupa toate aceste operatii, in imagine se afla doar tabla de joc. De asemenea, am scapat de piesele de domino desenate pe margininile tablei, decupand imaginea in toate cele 4 muchii cu 100 de pixeli. [figura 1]

1.2 Extragere gridului din tabla de joc

Am colorat in negru toate cele 4 margini ramase din tabla de joc cu 23 de pixeli pentru a fi mai usoara selectarea matricii din tabla de joc. Am extras in aceeasi maniera matricea cu ajutorul spatiului HSV si am aplicat filtru bilateral pentru a scapa de zgomot.

Am convertit imaginea in imagine binara si am aplicat Canny pentru a detecta muchiile, dar cu parametrii diferiti fata de procesul de la extragerea tablei, adaptati contextului. Cu acelasi procedeu ca la extragerea tablei de joc, am extras matricea tablei de joc si am schimbat perspectiva imaginii, astfel incat sa o cuprinda doar pe aceasta. [figura 4]

1.3 Gasirea piesei adaugate la runda curenta

Incep prin a gasi configuratia curenta pe baza matricii obtinute la punctul 1.2. Ma folosesc de spatiul HSV pentru a extrage piesele de domino din matrice. Imaginea obtinuta o convertesc la grayscale si aplic un filtru median pentru a scapa de zgomot. In cele ce urmeaza, o convertesc in imagine binara si aplic operatia de deschidere pe aceasta. Desenez linii peste liniile din gridul tablei de joc si gasesc configuratia plimbandu-ma prin fiecare patch al gridului si marcand cu 'x' daca se afla o piesa in patch-ul respectiv sau cu 'o' in caz contrar.Aceasta decizie o iau pe baza mediei pixelilor din fiecare patch [figura 5]. Daca media este mai mare decat 100 in patch-ul respectiv, inseamna ca am o piesa pe pozitia respectiva, deoarece piesele, dupa aplicarea transformariilor, sunt albe iar restul imaginii negre. Media o selectez ca fiind 100, deoarece se poate intampla sa mai existe zgomot in imagine (cum este in centrul tablei).

Ma folosesc de configuratia care la inceput este goala pentru a gasi piesa nou adaugata. Pozitia unde am piesa nou adaugata din configuratia curenta va fi diferita de pozitia din

configuratia trecuta. In locul in care gasesc diferenta, pastrez pozitiile in forma in care trebuie adnotate. Configuratia trecuta la inceputul jocului va avea peste tot numai 'o', fiind goala.

2 Task 2

2.1 Selectarea celor 2 fete a piesei adaugate la runda curenta

Pentru a gasi numarul de cercuri de pe fiecare parte a piesei care a fost adaugata la runda curenta, am inceput prin a selecta fiecare fata pe rand. Acest procedeu il realizez cu ajutorul task-ului 1. Selectez pozitia din grid indicata la cerinta anterioara, cu o marja de eroare de 3 pixeli in fiecare directie. In acest proces, transform pozitiile din forma adnotata in formatul de numerotoare de la 0 la 15 pentru coloane si pentru linii. Cand selectez patch-ul dorit am grija sa nu ies din limitele gridului din cauza marjei de eroare de 3 pixeli [figura 6, figura 8]. Peste fiecare fata selectata aplic o transformare HSV pentru a scapa de eventuale bucati din tabla.

2.2 Numararea cercurilor din fiecare fata a piesei adaugate

Pentru a numara cercurile din fiecare fata a piesei ma folosesc de Transformata Hough. Transform imaginea in grayscale, iar pentru o numarare mai precisa, o convertesc in imagine binara si aplic blur. Parametrii Transformatei Hough [figura 10] sunt ajustati corespunzator: distanta minima intre cercuri este de 10 pixeli, parametrul responzabil pentru Canny edge detection este setat la 60(param1), numarul minim de voturi pentru fiecare cerc posibil selectat este 10 iar raza cercurilor selectate trebuie sa fie intre 3 si 8 pixeli. [figura 7, figura 9]

3 Task 3

3.1 Declarare variabile globale

Am declarat de la inceput pozitiile pentru fiecare dintre cele 5 tipuri de romburi, deoarece acestea nu se vor schimba pe parcursul jocului. De asemenea, am creat un dictionar care reprezinta numarul de cercuri de pe pozitia pe care se afla.

3.2 Acordarea punctajului

In primul rand, verific al carui jucator este randul (1 sau 2) in functie de fisierul de mutari pentru jocul curent si in functie de runda curenta. In cazul in care am ajuns cu unul dintre pioni pe o piesa, verific daca acesta va primi cele 3 puncte bonus sau nu. Verific acest lucru pentru fiecare dintre jucatori, indiferent de randul lor. Adaug jucatorului curent numarul de puncte pe care il indica rombul sau romburile pe care a asezat piesa, daca a asezat piese pe unul dintre randuri. Dublez doar punctele obtinute cu ajutorul romburilor pentru jucatorul curent daca piesa asezata este dubla.

In variabilele scor_player1 si scor_player2 pastrez scorurile obtinute de jucatori pentru runda curenta. Pentru a calcula noile pozitii a celor 2 jucatori adaug scorurile celor 2 la

cei 2 contori,contor_player1 si contor_player2, pentru fiecare in parte. Iar pentru calculul punctelor bonus pastrez in variabilele, pozitie_player1 si pozitie_player2, numarul de cercuri din pozitia unde se afla fiecare pion cu ajutorul dictionarului amintit la punctul 3.1. La final scriu in fisier conform adnotorilor pozitiile, numarele de cercuri si scorul obtinut de jucatoru curent pentru runda curenta.

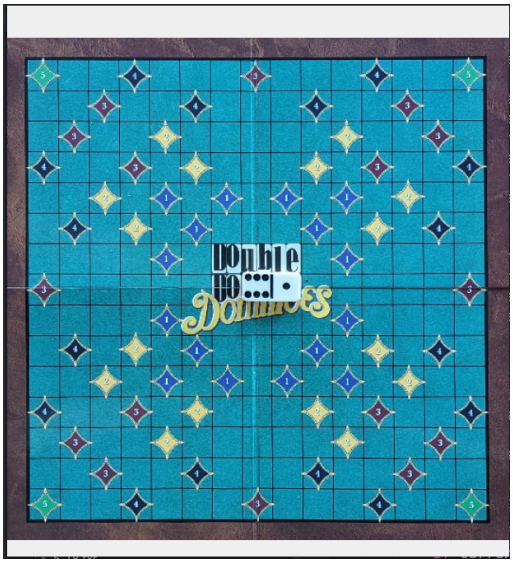

Figure 1: Tabla de joc extrasa

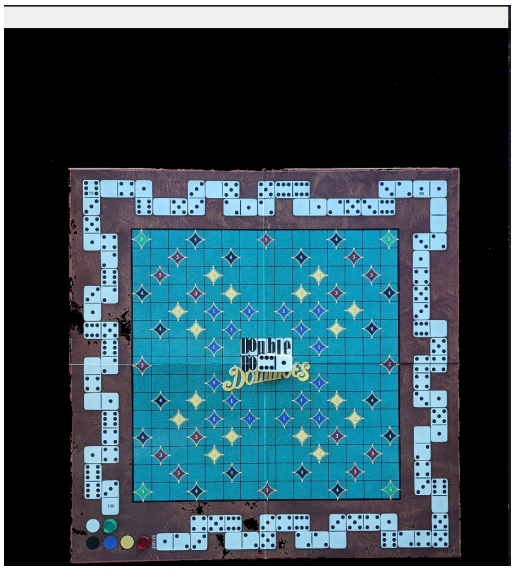

Figure 2: Extragere tabla de joc cu spatiul HSV

Figure 3: Tabla de joc in format binar

Figure 4: Grid-ul tablei de joc extras

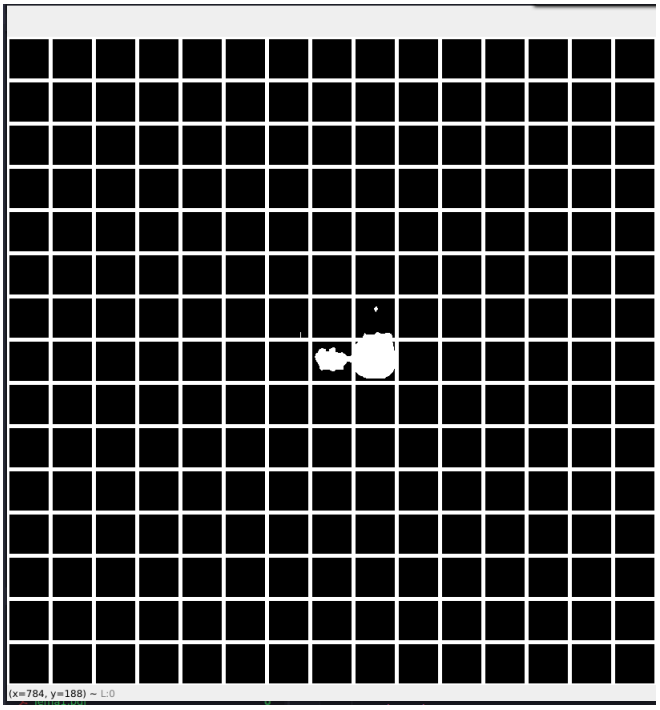

Figure 5: Grid-ul tablei de joc in format binar

Figure 6: Fata de domino extrasa

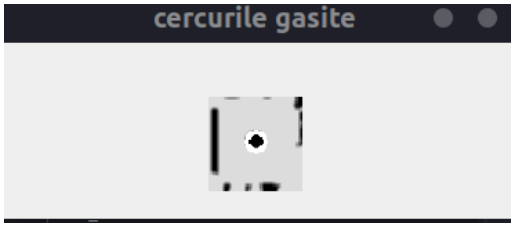

Figure 7: Cercurile detectate

Figure 8: Fata de domino extrasa

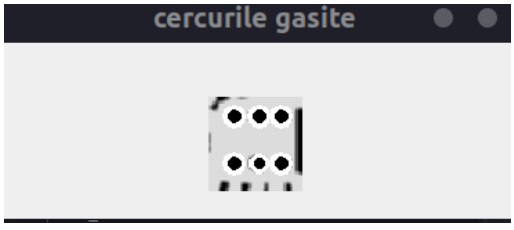

Figure 9: Cercurile detectate

```python
def get_number(candidate):
    candidate = cv.cvtColor(candidate,cv.COLOR_BGR2GRAY)
    # show_image('o fata de domino',candidate)
    _,candidate = cv.threshold(candidate, 90, 220, cv.THRESH_BINARY)
    candidate = cv.blur(candidate,(3,3))

    numar_cercuri = 0
    # 60 param1 --> 96%, minDist = 13
    cercuri = cv.HoughCircles(candidate, cv.HOUGH_GRADIENT, dp=1, minDist=10, param1=60, param2=10, minRadius=3, maxRadius=8)

    # Draw circles on the original image
    if cercuri is not None:
        # Convert the (x, y) coordinates and radius of the circles to integers
        cercuri = np.round(cercuri[0, :]).astype("int")
        for (x, y, r) in cercuri:
            # print('cate un cerc')
            numar_cercuri += 1
            cv.circle(candidate, (x, y), r, (255, 0, 0), 2)

    # show_image('cercurile gasite',candidate)
    return min(6,numar_cercuri),candidate
```

Figure 10: Cod pentru detectarea cercurilor