Proiect CAVA - 2023

462, Ghergu Nicolae-Marius
CTI

Scopul acestui proiect a fost calcularea scorului automat in jocul Double Double Domino care ia ca input niste imagini ce contin o table de Double Double Domino si calculeaza pozitia pieselor au fost adaugate pe tabla la fiecare mutare, valoarea pozitiei fiecarui parte a domino-ului adaugat si scor-ul fiecarui jucator la mutarea respectiva.
In arhiva se afla:
1. un fisier README cu instructiuni despre cum trebuie sa fie rulat script-ul si ce librarii sunt folosite in script
2. un fisier solutie.py care reprezinta script-ul respectiv
3. un fisier evaluare_solutie.py care l-am folosit la proiect, dar nu e necesar rularii (se specifica mai jos cum l-am folosit)

Aceasta documentatie prezinta solutia propusa de mine pentru a atinge scopul acesui proiect si pasii cum am ajuns la aceasta solutie.

Pasul 1: Cum incep?

Initial, am incercat sa inteleg proiectul per total. Astfel, am urmat fiecare documentare din video-ul de pe youtube si din prezentarea temei si am luat-o pas cu pas.
Astfel, per total m-am gandit: “trebuie sa aflu pozitia” -> asta inseamna ca trebuie sa gasesc o solutie sa analizez poza respectiva si sa vad cum incep.

Am inceput folosind laboratorul 6, cel de scrabble. Am incercat sa urmaresc iarasi solutia propusa si cum s-ar mula aceasta pe problema mea. Initial, aceeasi pasi i-am urmarit, am incercat sa identific careul de joc.

Pasul 2: Ce am de rezolvat si cum ma mulez ce am aflat pe problema mea?

Initial, folosind facand poza gri, aplicand niste blur si dupa aceea Canny, nu am reusit, nu se detecta nimic. Urmatorul pas, a fost sa decupez imaginea. Dupa aceea, am vazut ca nu functioneaza acest lucru si am zis sa incerc si alte metode. A doua a fost cu o masca hsv folosind din laboratorul 7 script-ul de filtrare al culorilor si am reusit.

Astfel, primul pas a fost rezolvat prin taierea imaginii si folosind o masca hsv cu dimensiunile ((80, 255), (70, 255), (0, 255)) (low yellow - (80, 70, 0), high yellow - (255, 255, 255)).

Pasul 3: Identificarea pieselor

Aici, m-am gandit la multe variante si eram pierdut, nici una nu mergea. Initial, am incercat folosind aceeasi metoda ca si in proiectul scrabble si anume sa impart imaginea in patch-uri si sa iterez peste fiecare patch. Totul bun, insa pana la detectia pieselor. Am incercat la fel ca si la scrabble, adica sa folosesc niste gri, sa aplic niste blur si sa folosesc canny, iar dupa aceea niste template-uri, dar nu a functionat. (Poate) Una dintre probleme era ca template-urile nu aveau aceeasi marime si am incercat sa extrag intr-un fel “miezul” (adica sa extrag mai putin din piesa), dar fara ajutor. A doua problema, la care m-am gandit, a fost poate ca template-ul trebuia rotit, dar dupa cautari pe internet si multe incercari, n-am reusit deloc cu template-uri.
A doua posibilitate, a fost folosind media per fiecare patch (media pixelilor albi mai exact), dar fara reusinta si cu acest lucru, problema fiind ca mi se detecta si centrul unde scria “double double domino”, nu doar piesele. Dupa multe incercari, am abandonat si aceasta idee.
Dupa ceva zile, mi-a venit o idee si aceea care am si folosit-o si anume sa detectez cercuri. Am luat fiecare patch, iar dupa aceea am incercat sa detectez cercurile din fiecare patch, iar daca se gaseau cercuri, inseamna ca era o piesa de domino. Patch-ul este format dintr-o piesa de domino, iar

acel domino ar fi “valid” daca ar exista cercuri per fiecare jumatate din domino, dar problema ar fi fost in felul urmator: piesele care nu au cercuri si domino-urile care au cercuri pe jumatate din piesa si pe cealalta jumatate nu.
Aceste probleme le-am rezolvat ulterior.

Pasul 4: Detectia cercurilor

Detectarea cercurilor am incercat sa o aflu folosind un google search pe site-ul de la opencv si am gasit urmatorul lucru: https://docs.opencv.org/4.x/da/d53/tutorial_py_houghcircles.html , anume metoda HoughCircles care m-a ajutat la detectarea cercurilor. Parametrii din documentatie nu m-au ajutat foarte mult, insa facand debug si ajustand valorile, am incercat sa ajung la o solutie decenta.

Pasul 5: Aflarea numerelor de pe domino

Detectia cercurilor de mai sus m-a ajutat foarte usor si sa detectez care sunt valorile de pe fiecare patch (dupa ce l-am impartit in doua) si acest lucru l-am si folosit.

Pasul 6: Problemele din trecut
Una dintre probleme a fost si faptul ca atunci cand am taiat tabla, am observat ca sunt 15 linii si 15 coloane, iar pentru a fi sigur ca am iterat peste toata tabla, a trebuit sa iterez cu 0 si 100 pixeli de la start pe verticala si orizontala (pentru ca asa puteau fi plasati domino-uri).

Solutia curenta era una care parea promitatoare si am incercat sa avansez cu ea. Am incercat sa ma gandesc la cum as putea rezolva urmatoarele cazuri: ce fac daca o piesa de domino nu are cercuri si una dintre solutii pe care am folosit-o: calculeaza media pixelilor albi (patch > 200, concret) si vezi care este procentajul de alb din imagine. Eu personal, am ales 70% si acest lucru a functionat.

A doua problema, a fost cand un domino avea cercuri si cand nu avea cercuri (o jumatate de patch avea, iar cealalta nu). Una dintre idei, care la fel dupa multe incercari de altele, a fost sa fac o masca hsv, dar n-a functionat. Si dupa aceea am incercat sa folosesc aceeasi metoda ca cea cand as avea un domino fara cercuri si anume sa fac media pixelilor albi pe masca hsv ((90, 255), (0, 100), (190, 255)) low yellow - (90, 0, 190) si high yellow - (255, 100, 255) si sa iau un procentaj de 60% (aleator, nu cu un motiv anume) si am vazut ca functioneaza, deci am considerat o solutie buna.

Pasul 7: Solutie promitatoare, curiozitatea cat de bine functioneaza
Desi parea promitator, am zis sa nu stau sa verific de fiecare data fiecare folder sa vad daca raspunsurile din datele de antrenare corespund cu cele din output-ul meu, asa ca am facut un script de “evaluare solutie” care verifica corectitudinea solutiei mele cu cele din datele de antrenare. Acest script m-a ajutat foarte mult si la ajustarea valorilor la metoda care foloseste HoughCircles, jucandu-ma cu valorile (destul de mult), pana cand am ajuns la un rezultat (neasteptat) mai promitator, pe care l-am atasat in arhiva .zip.

Pasul 8: Calculul scorului
Calculul scorului nu a fost unul dificil deoarece depinde mai mult de cat de bine functioneaza ceilalti pasi, singura parte dificila fiind cum as putea sa tin cont de rezultatele anterioare. Desi posibil solutia mea sa nu fie cea mai buna (overall), am incercat sa fac ceva care functioneaza pentru acest pas si anume sa deschid fisierul si-l inchid cu diverse permiuni (cel generat de catre mine cu output-ul) astfel incat sa-l modific pe cel generat principal pana cand sa obtin datele necesare calcularii scorului. (Am citit fisierul initial, am luat pozitiile si valorile domino-ului si am calculat scorul, dupa aceea am facut append la rezultatul obtinut in fisierul generat initial ca output).
Pentru a afla unde se afla piesa (daca a fost pusa pe un punctaj sau nu) am folosit o matrice unde am hardcodat unde se afla fiecare punct in careu si pentru a afla care este ‘valoarea’ (traseul din afara careului) pe care este domino-ul , am hardcodat valorile folosind un vector).