Prepoznavanje i rješavanje labirinta sa slike

Irhad Fejzić, *TKN, Odsjek za matematiku, PMF Sarajevo*

***Abstract*—U ovom radu opisana je tehnika detekcije i rješavanja labirinta sa slika koristeći metode koje vrše perspektivnu transformaciju slike, detekciju početka i kraja labirinta i pronalazak puta koristeći algoritme BFS i Djikstra. Nakon digitalizacije slike labirinta, vrše se tehnike procesiranja slike za analizu zidova labirinta i linija. Koristi se prag za konverziju sivoskalirane slike i zidovi se pronalaze metodom erozije i dilatacije. Najkraći put se zatim iscrtava natrag na sliku korištenjem algoritama teorije grafova.**

***Ključne riječi*—labirint, procesiranje slike, djikstra, detekcija i segmentacija**

I. Uvod

Porast autononomnih vozila i robota je evidentno prateći današnje trendove mašinskog učenja i vještačke inteligencije. Pored velikog doprinosa u razvijanju autonomnih vozila i robota za svakodnevni život, njihova korist dodatno raste u industriji gdje je potreban brz i precizan transport dijelova i robe sa jednog mjesta na drugo. Koriste se i u svakodnevnom životu za navigaciju i čišćenje kuća kao autonomni roboti, ali i ozbiljnije poslove u vojsci, zdravstvu i policiji. Za većinu ovih poslova planiranje samog puta je ozbiljna tema koja se nastoji daljnje unaprijediti modifikovanjem poznatih algoritama poput DFS, BFS, Flood fill i mnogih ostalih. Glavni problem koji se susreću roboti prilikom pronalaska destinacije u labirintu je što pronađeni put u mnogim slučajevima nije i najoptimalniji. Dodatno, postupak pokušaja i pogreške (engl. *trial and error*) je vremenski dugotrajna tehnika koja može dovesti robota do beskonačne petlje ukoliko je labirint dovoljno kompleksan. Robot se u ovom slučaju oslanja na lokalnu paradigmu gdje ne zahtjeva veliki broj informacija u prostor i efikasno reaguje na nepredviđene prepreke u okruženju, ali ne garantuje optimalnost najkraćeg puta što je ranije navedeno. Tehnika korištenjem procesovane slike labirinta i generisanja puta pomoću algoritma vještačke inteligencije omogućuje pronalazak najkraćeg puta gdje je i potrebno vrijeme pronalaska manje jer eliminiše statičke prepreke generisanjem predefinisanog puta prije samog pokreta robota.

II. Pregled literature

Labirint se smatra slagalicom ili puzlom koja se sastoji od konfuznih mreža međusobno povezanih puteva koje robot mora da riješi pronalazeći put od početne tačke do krajnje. Proces pronalaženja jednog takvog puta se naziva rješavanje labirinta (engl. *maze solving*). Rješavanje labirinta može se izvršiti postavljanjem robota koji će bez prethodnog znanja o detaljima labirinta pronaći put prolazom kroz moguće puteve ili korištenjem kamere koja će prvobitno uslikati čitav labirint i vršenjem analize i pretprocesiranja u veoma brzom vremenu naći korektan i rješenje. Cilj procesa planiranja puta u mobilnim robotima je pronalazak izlaza iz labirinta sa prvobitnog stanja gdje robot mora da izbjegne koliziju sa preprekama unutar labirinta. U radu [1] navodi se da korištenjem slike čitavog labirinta uslikano kamerom i analizom algoritma tehnikama vještačke inteligencije zajedno sa procesiranjem slike dolazi do inteligentnog algoritma koji efikasnije pronalazi kraći put od tradicionalnih metoda. Predloženi algoritam funkcioniše korištenjem kamere postavljene iznad labirinta koja periodično šalje slike računaru koji je povezan sa robotom. Na ovaj način eliminiše se tradicionalna tehnika postavljanja robota bez prijašnjeg znanja i oslanjajući se na nepotpuni algoritam da riješi labirint. Efikasan *Flood-fill* algoritam je predložen u radu [2] sa modifikacijom preskakanja koraka ukoliko su bespotrebni u nekim trenucima. *Flood-fill* tehnika koristi koncept vode koja teće sa viših visina na niže i korištenjem takvog koncepta postavljaju se vrijednosti čelija labirinta zavisno koliko su udaljeni od robota. Algoritam u svakom trenutku ažurira pronađene zidove oko robota, dodaje nove vrijednosti procesom "poplave" puteva, (engl. *flood*) unutar labirinta i određuje koji je korak najefikasniji sa novim informacijama. U radu [3] prezentuje se rješenje labirinta korištenjem wireless navigacije na robotu za planiranje puta unutar labirinta. Sistem naviacije sadrži kameru koja šalje slike vidljivog dijela labirinta u realnom vremenu. Korištenjem kamere na robotu algoritam je u mogućnosti naći trenutnu poziciju robota i njegovu orijentaciju zavisno korištenjem detekcije markera trenutnog prostora. Rezultat se pokazao efektivnim kod implementacije sistema direktno povezanog sa robotom. Dodatno, primjenom pretprocesiranja slike prije pokretanja algoritma koji će naći traženi put dovodi do korektnijeg rezultata i efikasnijeg algoritma pretrage. U radu [4] navodi se da transformacija slike korištenjem korekcije ptčije perspektive dovodi da sve linije i zidovi labirinta budu horizontalne i vertikalne direkcije što smanjuje dodatno računanje. Vršenje transformacije RGB slike u sivoskaliranu i korištenje Gaussovog filtera za uklanjanje šuma su opcije koje se mogu iskoristiti za poboljšano detektovanje linija i zidova unutar slike. Sa istaknutim zidovima labirinta slika se dodatno pretvara u binarnu sliku gdje se sa 1 definišu slobodni putevi labirinta dok elementi 0 unutar matrice označavaju zidove i dijelove gdje put ne može proći. Procesirana binarna slika (primjer na slici 1) se finalno poziva unutar algoritma teorije grafova za pronalazak puta zajedno sa koordinatom početka i cilja unutar labirinta. Za labirinte koji se sastoje isključivo od

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

Slika 1: Binarna slika labirinta

vertikalnih i horizontalnih zidova moguće je primjeniti Canny-evu metodu za detekciju ivica na slici i dodatno Houghovu transformaciju za potvrdu koje ivice predstavljaju linije zida. Ovim postupkom moguće je popuniti zidove koji su se izgubili prilikom primjene praga na slici ukoliko je slika labirinta parcijalno osvijetljena ili podaci na slici nedostaju.

III. Opis projekta

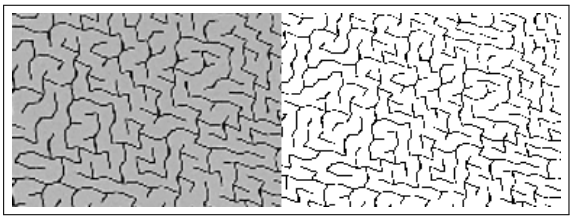
Tehnika nalaženja puta na labirintu započinje slikanjem čitavog labirinta i spremanjem slike na računar. Bitno je da se čitav labirint vidi na slici kao i da postoji ulaz i izlaz naglašen crvenom tačkom. Dodavanje ulaza i izlaza može se izvršiti markerom ili tokom pretprocesiranja slike. Ova metoda ne koristi tehniku koja se oslanja na robota i njegov algoritam pronalaska puta već koristi tehniku pretprocesiranja slike prije pokretanja nekih od poznatih algoritama poput BFS, Djikstra, A* za nalaženje najraćeg puta u veoma efikasnom vremenu.

A. Pretprocesiranje slike labirinta

Jedan od najvažnijih koraka uključuje učitavanje slike kao i korištenje raznih tehnika za pretvaranje slike u adekvatnu matricu koja se može poslati algoritmu za nalaženje puta. Ovaj korak uključuje filtriranje slike, nalaženje početne i krajnje destinacije i perspektivne transformacije za bolje čitanje vrijednosti na slici. Za projekat se koristi PyCharm, OpenCV biblioteka za učitavanje slike i prethodno navedenih metoda i numpy biblioteka koja se koristi za manipulaciju nizova i matrice. Informacije glavnih boja u slici su sadržane u tri posebna R, G i B kanala koja nisu potrebna za rješavanja labirinta i samim tim vrši se konverzija slike u sivoskaliranu korištenjem formule

$$siva = 0.2989R + 0.5870G + 0.1140B$$

za prirodniju percepciju zavisno od osjetljivosti ljudskog oka [5]. Nakon konverzije dolazi do mijenjanja dimenzija slike za lakše procesiranje i dodatno pretvaranje slike u binarnu postavljanjem praga. Binarna slika sadrži informacije zidova labirinta od ostatka elemenata na slici koji će se kasnije proslijediti algoritmu teorije grafova za pronalazak puta. Slika 2 prikazuje poboljšanje ivica i njihovo izdvajanje sa papira korištenjem jednostavne metode praga.



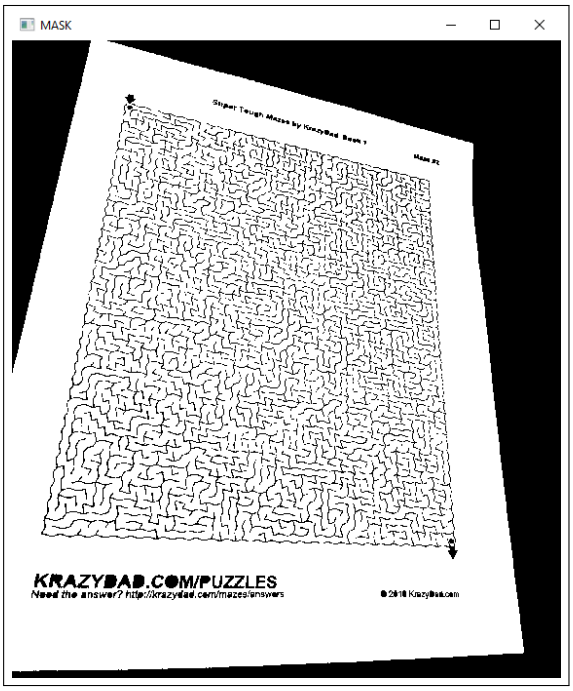
Slika 2: Slika prije i poslije postavljanja praga

Dilatacija i erozija su dvije morfološke operacije nad binarnim slikama koje čestu primjenu imaju u detakciji objekata na videozapisu. Dilatacija je transformacija koja koristi marker sliku (u ovom slučaju mali isječak centralnog dijela labirinta) za identifikaciju objekata na maski koje je potrebno istaći. Za labirint erozija uklanja neravnine oko granica/zidova dok dilatacija popunjava rupe oko tih istih granica, ukoliko postoje. Marker na slici 3 tokom procesa

Slika 3: Dilatacija - *Seed* ili Marker za rekonstrukciju

će proći kroz dilataciju, tj. proširenje gdje tokom procesa vrijednosti piksela neće prevazići odgovarajuće vrijednosti na maski. Ovom metodom dolazi do povećanja granica regiona piksela sa ciljem nalaženja samo labirinta i eliminacije drugih nepotrebnih objekata koji mogu dovesti do teže detekcije puta. Druga operacija, erozija, se koristi za eliminaciju granica u regionu piksela što dovodi do slike gdje je labirint u centralnom fokusu i time ostali objekti izbačeni. Na slici 4 korištenjem erozije dolazi do uklanjanja crne pozadine radi lakšeg pronalaska granica labirinta. Dobijanje korektnih ivica i granica labirinta zavisi da li je struktura labirinta četverougao ili neki drugi oblik. Ukoliko se radi o klasičnoj strukturi četverougla algoritam kroz dvije petlje matrice slike prolazi kroz potencijalne piksele ivica i bilježi ukoliko postoji bolji kandidat za ivicu. Oblik labirinta pronalazi se pomoću njegovih kontura. Konture posmatramo kao krive koje povezuju kontinuirane tačke na slici gledajući boje i intenzitet tih tačaka. Prije pronalaska kontura vrši se nalaženje praga slike i dodatno primjenjuje detekcija ivica (koristeći Canny-evu transformaciju) ukoliko znamo unaprijed da je labirint četverougao. U radu se ne koristi detekcija ivica pošto se posmatraju labirinti raznih struktura, ali detekcija ivica znatno poboljšava pronalazak i popounjavanje zidova labirinta. Ukoliko struktura labirinta nije četverougao nakon detekcije njegovih kontura, tada preko njihove aproksimacije vrši se detekcija oblika labirinta na slici. Ovo je moguće korištenjem openCV funkcije *approxPolyDP* koja vraća tačke ivica prethodno pronađene konture labirinta preko kojih se nakon toga može aproksimirati četverougao koji obuhvaća labirint. Drugi način definisanja oblika, koji ne zahtjeva biblioteku, je pomoću pronalaska konveksnog omotača bijele regije na slici. Posmatranjem bijelih piksela regije labirinta kao skup 2D tačaka daje mogućnost definisanja najmanjeg konveksnog poligona koji obuhvata sve tačke bijelog intenziteta i samim tim i konveknsi omotač. Na slici 6 pronađen je put za labirint koji je drugačijeg oblika od standardnog četverougla što je omogućeno detekcijom oblika labirinta prije perspektivne transformacije. Dodatno poboljšanje slike labirinta dobija se postavljanjem drugačijih vrijednosti za adaptivni prag ovisno o tipu slike (labirint na papiru ili botanički labirint), kao i podebljanjem ili povezivanjem zidova labirinta ukoliko pragom dolazi do njihovog gubljenja. Bitno je navesti da ove vrijednosti nisu fiksne pošto u nekim slučajevima podebljanjem je moguće dovesti do zatvaranja puteva koji su prethodno bili dostupni tako da se proces odabira vrijednosti treba razmatrati posmatranjem originalne slike. Sa dodatnim informacijama korisnik može mijenjati vrijednosti prilikom postavljanja praga, dilatacije, erozije i detekcije rubova, ali

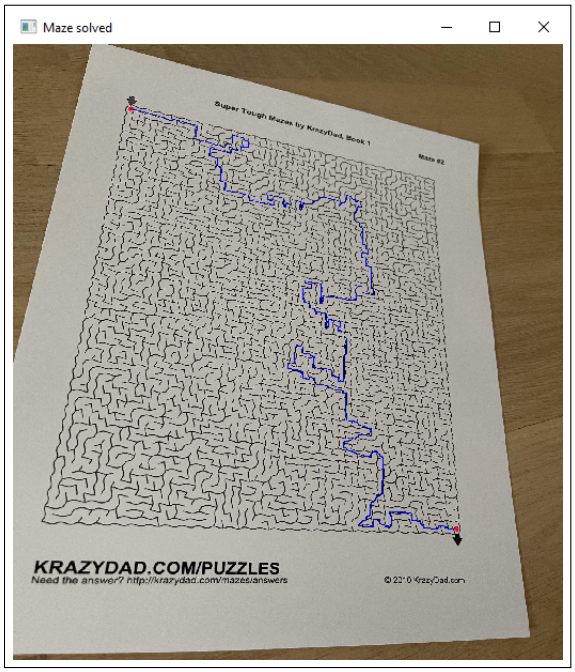
sami automatizirani proces ovog postupka nije trivijalan.



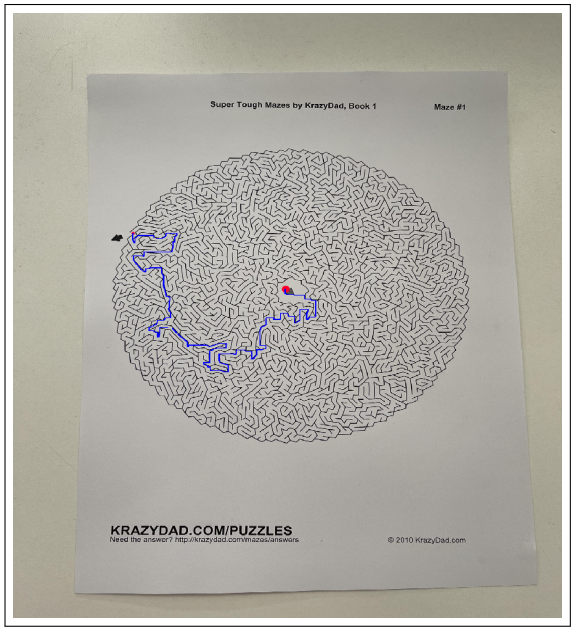
Slika 4: Labirint nakon dilatacije

B. Nalaženje najkraćeg puta u labirintu

Rješavanje labirinta i nalaženje najkraćeg puta između početne i krajnje tačke zahtjeva binarnu sliku labirinta u kojoj su istaknuti zidovi crnih piksela dok ostatak, tj. slobodno bolje označeno bijelom bojom. Proces pronalaska puta zahtjeva korištenje popularnih algoritama i pokretanje tih algoritama nad matricu binarne slike gdje elementi označeni nulom predstavljaju puteve koje robot može da prođe prilikom kretanja. Algoritmi koji se koriste za nalaženje najkraćeg puta su Djikstra, A*, Flood-fill i modificirani Flood-fill algoritam. Djikstra i A* algoritam se ističu radi svoje efektivnosti, ali i nedostatka pronalaska bržeg rješenja zbog iscrpne pretrage labirinta. Flood-fill algoritam zahtjeva ažuriranje svih susjednih čelija robota prilikom kretanja iako nije potrebno. Ovaj problem se eliminiše modifikovanom verzijom algoritma, ali ostaje ranije spomenuta mana gdje pronađeni put nekad nije i najkraći [6]. U radu labirint je rješen korištenjem Djikstrinog algoritma koji vrši "relaksaciju" (ažuriranje cijene svakog čvora povezanog sa trenutnim čvorom v) ukoliko se cijena puta može minimizirati. Drugi implementirani algoritam za pretragu grafova je BFS (engl. *Breath-first search*). Zadatak ovog algoritma, kao i kod Djisktrinog, je da nađe put do krajnjeg čvora (izlaza) ukoliko kreće od nekog početnog čvora (ulaza) labirinta. Od ulaza posmatra i pretražuje sve susjedne čelije prije nego što krene na sljedeći "nivo" susjeda. Glavna razlika ova dva algoritma je što Djikstrin algoritam se koristi i za grafove gdje grane sadrže određene cijene, dok BFS algoritam posmatra sve grane kao da imaju težinu/cijenu 1. Riješeni put od početka do kraja labirinta očekuje se da će biti u potpunosti isti za oba algoritma.



Slika 5: Riješeni labirint

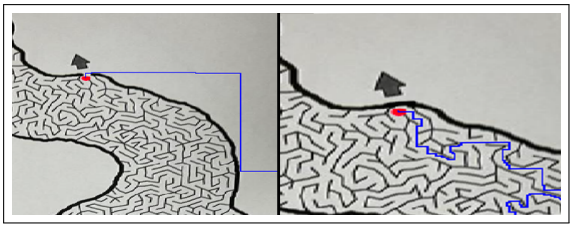


Slika 6: Riješeni "okrugli" labirint

IV. Diskusija

U radu je opisana tehnika procesiranja slike labirinta kako bi se izdvojili zidovi za bolji rezultat nakon primjene algoritama teorije grafova. Prilikom procesiranja, tip labirinta kao i njegovo okruženje imali su veliki uticaj za finalni procesirani rezultat. Labirint vraća puno bolje i realističnije puteve ukoliko je njegov oblik bio čeverougao, iako i za ostale oblike imao je iznenađujuće rezultate. Neobičan problem sa kojim se algoritam susretao su otvoreni ulazi i izlazi koji bi sa lošom procjenom kontura vraćao put koji bi zaobilazio čitav labirint. Na slici 7 prikazan je metod ručnog zatvaranja labirinta kako

bi se riješio problem pronalaska puta zaobilaženjem labirinta. Ovaj problem se dodatno može popraviti postavljanjem kontura oko labirinta koje oponašaju nove granice. Ovakav metod je i implementiran u radu iako za loše pronađene konture vraća nepoželjne rezultate. Dolazak do nepreciznih kontura javlja se ukoliko se proslijedi loše "uslikan" labirint gdje se javljaju neravnomjerno osvijetljene regije slike kao i slikanjem pod oštrim uglom gdje se detalji labirinta počinju gubiti. Za ostale probleme prilikom procesiranja, jednostavno podešavanje praga slike kao i izbacivanjem dodatne erozije dovelo je do korektnijeg prikaza i segmentacije labirinta od ostatka slike.



Slika 7: Slika prije i poslije ograničavanja otvorenih prostora labirinta

V. Zaključak

U ovom projektu opisana je metoda rješavanja labirinta korištenjem postupka pretprocesiranja slike prije pokretanja algoritama teorije grafova za nalaženje najkraćeg puta. Za pronalazak takvog puta čitav labirint se slika kamerom i daljnje procesira i analizira programom korištenjem biblioteke OpenCV. Dobijeni put je tada moguće poslati robotu ili uređaju koji u tom trenutku ima detaljno opisane korake koji ga vode do kraja labirinta. Ovo odstupa od tradicionalne metode pronalaska puta koristeći robota i njegove sposobnosti da izađe bez prethodnog znanja o samom labirintu. Nedostatak korištenja metode procesiranja slike u radu je što za veće dimenzije slike postaje teže i naći put u razumnom vremenu. Pored problema sa kojim se susreću algoritmi za pronalazak puta, pretprocesiranje ima svoj izazov segmentacije labirinta ukoliko je slika labirinta nepotpuna ili zidovi nisu jednaki. Za budući rad grupisanjem slobodnih piksela (piksela koji ne označavaju zidove) minimizirala bi se pretraga algoritma iz teorije grafova i znatno ubrzalo vrijeme vraćanja puta.

Reference

[1] M. O. Aqel, A. Issa, M. Khdair, M. ElHabbash, M. AbuBaker, and M. Massoud, "Intelligent maze solving robot based on image processing and graph theory algorithms," *2017 International Conference on Promising Electronic Technologies (ICPET)*, 2017.

[2] H. Dang, J. Song, and Q. Guo, "An efficient algorithm for robot maze-solving," in *2010 Second International Conference on Intelligent Human-Machine Systems and Cybernetics*, vol. 2. IEEE, 2010, pp. 79–82.

[3] K. Lutvica, J. Velagic, N. Kadic, N. Osmic, G. Dzampo, and H. Muminovic, "Remote path planning and motion control of mobile robot within indoor maze environment," *2014 IEEE International Symposium on Intelligent Control (ISIC)*, 2014.

[4] O. Aki and A. Gullu, *OBTAINING PATH DATA FROM MAZE IMAGE USING IMAGE PROCESSING TECHNIQUES*, Nov 2016.

[5] Matplotlib. (2016) Image tutorial - startup commands.

[6] G. Z. Ye and D.-K. Kang, "Maze solving algorithm," in *Proceedings of the Korean Institute of Information and Commucation Sciences Conference*. The Korea Institute of Information and Commucation Engineering, 2011, pp. 188–191.