Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.38
DOI: https://doi.org/10.24867/31BE26Goronic

SISTEM ZA IDENTIFIKACIJU BILJAKA I KONTROLU USLOVA OKOLINE POMOĆU RASPBERRY PI PLATFORME
SYSTEM FOR PLANT IDENTIFICATION AND ENVIRONMENTAL CONDITION CONTROL USING RASPBERRY PI PLATFORM

Nikolina Goronić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu je prikazan najidealniji algoritam za detektovanje lista biljke koristeći Raspberry Pi i Raspberry Pi kameru modul 3. Na osnovu uslikanog lista, detektuje se kojoj vrsti sobne biljke je riječ. Uz pomoć senzora DHT11 mjeri se vlažnost i provjerava se sobna temperatura prostorije, te njeno korigovanje ukoliko je potrebno paljenjem LED diode i ventilatora. Podaci se šalju sa uređaja na platformu ThingSpeak, gdje se vrši grafički prikaz vrijednosti, kao i na Node-Red, gdje je izvršena provjera funkcionalnosti.*

Ključne reči: *Raspberry Pi, DHT11 senzor, Node-Red, ThinkSpeak, digitalna slika, OpenCv, NumPi, Canny algoritam, matchShapes() funkcija*

Abstract – *This paper presents the most optimal algorithm for detecting a plant leaf using Raspberry Pi and Raspberry Pi Camera Module 3. Based on the captured leaf, the type of indoor plant is identified. With the help of the DHT11 sensor, humidity is measured and the room temperature is checked, with adjustments made if necessary by turning on the LED light and fan. The data is sent from the device to the ThingSpeak platform, where a graphical representation of the values is displayed, as well as to Node-Red, where the functionality is verified.*

Keywords: *Raspberry Pi, DHT11 sensor, Node-Red, ThingSpeak, digital image, OpenCV, NumPy, Canny algorithm, matchShapes() function*

1. UVOD

Pod pojmom "priroda" podrazumijevamo složen sistem, koji obuhvata sve što postoji, od univerzalnog do subatomskog. U okviru ovog sistema biljke imaju veliku ulogu doprinoseći održanju ravnoteže u prirodi. Jedna od najvažnijih uloga biljaka jeste obezbjeđivanje kiseonika. Posljednjih decenija, sve više ljudi spoznaje vrijednost kućnih biljaka, koje ne samo da oplemenjuju prostor svojom ljepotom, već donose i brojne koristi za zdravlje. Čiste vazduh, povećavaju vlažnost i doprinose stvaranju harmonije. Često ljudi, dobivši biljku na poklon ne znaju koja je najpogodnija sobna temperatura za opstanak iste. Ono do čega još može da dođe jeste da pomiješaju vrste biljaka, neke vrste biljaka tretiraju kao sobne što one zapravo nisu. Kako bi se suzbile ovakve greške postoji mogućnost prepoznavanja vrste biljke njihovim slikanjem, a i napravljen je spisak sobnih biljaka. Biranjem biljke dobija se njena idealna temperatura. Ovo može da zainteresuje ljude za ispravan uzgoj sobnih biljaka, da privuče neke nove potrošače koji ranije nisu imali uspjeha u uzgoju istih, da okušaju ponovo svoju „sreću“, te da podsjeti kupce i na neke zaboravljene vrste biljaka.

U ovom radu će biti prikazani različiti algoritmi i metode realizovane u programskom jeziku *Python*, koje su testirane prilikom istraživanja najadekvatnijeg algoritma za otkrivanje vrste biljke na osnovu lista, kao i prikaz toka u *Node-Red*-u, a grafika na *ThinkSpeak*-u.

2. HARDVERSKI DIO

Komponente koje su korištene prilikom realizacije rada su: *Raspberry Pi*, *DHT11 senzor*, *LED dioda*, ventilator, *Raspberry Pi kamera moduo 3* i *L298N modul*. *Raspberry Pi* je embeded računar, koji je napravljen u skladu sa *von Neumann*-ovom arhitekturom. Posjeduje sistem na čipu *BCM2837* proizvođača *Broadcom*, koji sadrži *ARM Cortex-A53* sa 4 jezgra, grafički procesor *VideoCore IV* i *RAM* memorija. 2 x 20 *GPIO* pinovi omogućuju povezivanje i rad sa drugim komponentama [1]. Dioda je povezana na pin 16 *Raspberry Pi*-ja i korišćena za signalizaciju pada temperature ispod dozvoljenog opsega. *DHT11* se koristi za mjerenje temperature i relativne vlažnosti vazduha u prostoriji. Posjeduje *NTC* termistor i kapacitivni senzor vlage. Komunikacija između modula i mikrokontrolera bazirana je na *One-wire* protokolu, koji predstavlja serijsku komunikaciju između master i jednog ili više slejv uređaja [2]. Ventilator se koristi za hlađenje komponenti, a njegovo pokretanje vrši se uz pomoć *L298N* modula, koji sadrži 11 pinova.

Ventilator je povezan na izlazne priključke OUT1 i OUT2, VCC modula je spojen sa 5V *Raspberry Pi*-ja, dok su VC i GND povezani na generator napona. Na ulazne priključke IN1 i IN2 dovodi se kombinacija visokog i niskog naponskog nivoa i na taj način se bira smer obrtanja, dok je na ulazni priključak EN doveden logički signal (logička nula i logička jedinica), na osnovu kojeg se vrši kontrola paljenja ventilatora. On je povezan sa pinom 33 *Raspberry Pi*-ja.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je dr Vladimir Rajs, van.profesor.

3. OBRADA SLIKE

Digitalna slika je elektronski zabilježena slika predstavljena u binarnom brojevnom sistemu, koja sadrži konačan broj redova i kolona piksela [3]. Pikseli su najmanji pojedinačni elementi slike, koji sadrže vrijednosti osvjetljenja date boje u svakoj specifičnoj tački.

Jedna od najpopularnijih biblioteka korišćenih za obradu slika i videa je *OpenCV*. U programskom jeziku *Python* njegovom interfejsu se pristupa putem *cv2* modula. *NumPi* je biblioteka za rad sa višedimenzionalnim nizovima (matricama) i obavlja različite matematičke operacije. U ovom radu se koristi za kreiranje matrica i za operacije nad konturama (spajanje kontura).

Prvobitna ideja je bila da se list izdvoji iz biljke i na osnovu njegovog oblika utvrdi o kojoj biljci je reč, zbog čega su izvršena testiranja na samo uslikan list ili na čitavu biljku. Algoritmi i metode koje su testirane su:

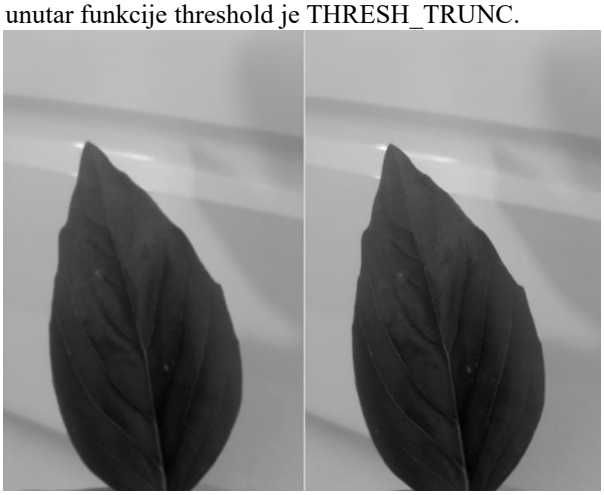
- *LoG* filter (nejasna kontura lista)
- *Sobel* (gubi se detaljnost)
- *GrabCu*t (jasno izdvajanje čitavog objekta)
- *Hough* algoritam (samo za određen oblik)
- *K-means* (neuspješnost kod sličnih boja)
- *Canny* algoritam.

Pored *Canny* algoritma (o njemu će biti više u narednoj glavi), najbolje rezultate je davao *GrabCut* algoritam, jer je dobro izdvajao biljku od sredine. Izdvajanje lista iz biljke je pokušano na osnovu boje, jer nakon djelovanja algoritma objekat se posmatra kao cijeli, pa je to jedini način za izdvajanje konture. Definisane su gornja i donja granica za boju. Koristeći gornju granicu za zelenu boju, kreira se binarna maska u kojoj su pikseli unutar opsega zelene boje označeni kao 1, a ostali kao 0. Ova maska služi za izdvajanje samo zelenih dijelova slike [4]. Ovo nije bilo uspješno, jer nemaju sve biljke istu boju listova, neke imaju svetlije, a neke tamnije. Takođe, postoji cvijeće koje ima zeleni pigment i algoritam će njih izdvojiti. Na svjetlosti i u mraku, pigment lista neće biti isti, pa samim tim ni opseg zelene boje, što znači da je jaka osjetljivost na šum i osvjetljenost.

Postoji još jedan način izolacije, a to je na osnovu oblika konture. Kako svaki list ima eliptičan tj. okrugao oblik, isprobana je izolacija upravo na osnovu toga. Na osnovu dužine i površine konture računa se kružnost, koja prikazuje koliko je oblik sličan krugu. Kružnost ima vrijednost blisku 1 za savršeno kružne objekte, a manju za izdužene ili nepravilne forme. Izračunat je aspektni odnos (odnos visina/širina) i ukoliko konture imaju aspektni odnos veći od 2.5 (odnosno visina je znatno veća od širine) i kružnost manju od 0.4, smatraju se potencijalnim listovima. Rezultat ovoga je da se pojedini listovi ne detektuju kao polukružni, dok oni koji su detektovani, ne detektuje kao cijeli list, već kao dio, pa zbog toga se ovaj algoritam ne čini pogodnim.

4. CANNY ALGORITAM

On je popularan i efikasan metod za detekciju ivica u slikama, razvijen od strane Johna F. Cannyja 1986. godine. Njegova svrha je da pronađe tačke u kojima se intenzitet boje mijenja naglo, što obično predstavlja ivicu objekta na slici. Sastoji se od pet koraka, a to su: Filtriranje šuma (Gaussian Blur), detekcija gradijenta, Non-maximum suppression, double threshold i praćenje ivica putem histerezisa (Edge Tracking by Hysteresis) [5]. Tokom realizacije postoji mogućnost modifikovanja navedenih koraka, kako bi se dobili što precizniji rezultati. Upravo na tim modifikacijama se zasniva konačno rješenje ovog rada. Nakon očitavanja slike, vrši se njena konverzija u sivu skalu pomoću cv2.cvtColor() funkcije i normalizuju se vrednosti piksela cv2.normalize() funkcijom (slika 1). Na ovaj način je slika spremna za prvi korak Canny algoritma, a to je Gaussian Blur. Uklanjanjem šumova potrebno je da se ona threshold-uje, postavljajući odgovarajući prag sa kojim se porede svi pikseli na slici. Izabrana tehnika unutar funkcije threshold je THRESH_TRUNC.



Slika 1. *Slika uslikanog lista nakon blurovanja (lijevo) i normalizacije (desno)*

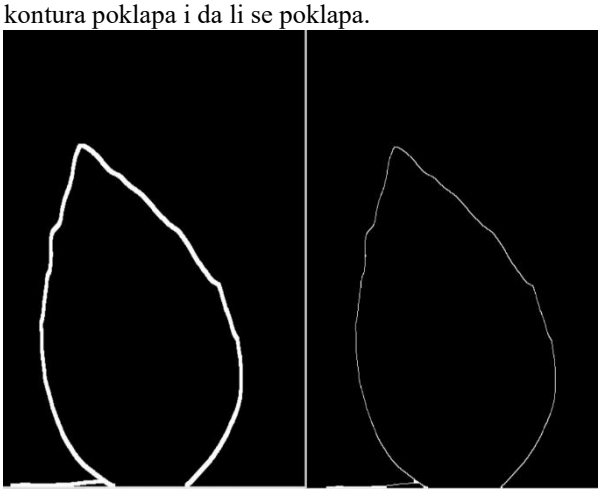
Sljedeći korak je detekcija ivica i crtanje kontura, (slika 2), koje su na nekim listovima isprekidane, dok su na nekim savršene. Ideja je bila da se iz skupa isprekidanih kontura izdvoji najveća i uporedi sa bazom biljaka, međutim bezuspešno.

Slika 2. *Slika uslikanog lista nakon thresh_trunc tehnike (lijevo) i detektovanih ivica (desno)*

Jako su velike greške, kao i razlike između iste vrste listova. Upravo iz tog razloga je izvršeno sabiranje konturica lista i dobijanje jedne konture. Prije nego što se izvrši sabiranje kontura, bilo je potrebno da se izvrši provjera da li je kontura potpuno zatvorena (konture na

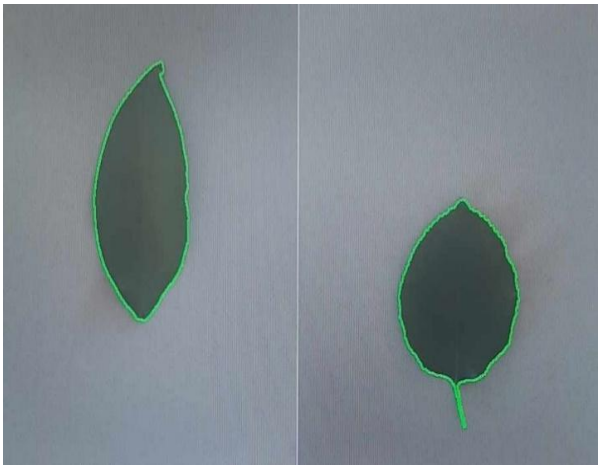
nekim listovima su savršene) ili kada je kontura skoro zatvorena, jer se desi da se uslika pola lista. Posmatrale su se krajnje tačke koje su najbliže i na osnovu toga se izdvajala ta kontura. Ovo nije dalo uspješne rezultate, jer su se pojavili slučajevi kada konture unutar pora imaju manju razliku u tačkama.

Pošto prethodni koraci nisu davali najbolje rezultate, nego prosječne, bilo je potrebno izvršiti poboljšanje ivica. Zbog toga su iskorištene dve funkcije cv2.dilate() i cv2.erode() nakon crtanja ivica, a pre crtanja konture. Dilatacijom su proširene ivice i pomaže u popunjavanju praznina, dok erozija „stanjuje“ ivice. Ovo je doprinijelo poboljšanju strukture i detektovane su bolje konture (slika 3), bez prekida. Kada se odredi oblik konture uslikanog lista, potrebno je da se proveri sa kojim se od kontura poklapa i da li se poklapa.



Slika 3. *Slika uslikanog lista nakon dilatacije (lijevo) i erode funkcije (desno)*

Funkcija cv2.matchShapes() u OpenCV-u služi za upoređivanje oblika, odnosno za mjerenje sličnosti između kontura [6]. Koristi Huove momente, koji su invarijantni na translaciju, skaliranje i rotaciju, što znači da su otporni na promjene u veličini i položaju oblika. Ova funkcija prolazi kroz bazu kontura svih listova i vraća numeričku vrijednost koja predstavlja koliko su konture slične. Manje vrijednosti (bliže nuli) označavaju veću sličnost između kontura, dok veće vrijednosti označavaju da su konture znatno različite. Provjerava se koja je najmanja sličnost između biljaka i da li je manja od 1.



Slika 4. *Konture dva različita lista*

Ukoliko je uslov ispunjen, ispisuje se koja je vrsta biljke. Kao što je rečeno u prethodnom poglavlju pre korišćenja dilatacije i erozije, ova funkcija je davala prosječne rezultate. Većinom je otkrivala ispravnu vrstu biljke, međutim dešavalo se da drastično pogriješi, kao što se može videti na slici 4. Lijevi list se poklapao sa konturom desnog lista 0,48, što je prevelika vrijednost. Nakon uvođenja navedenih funkcija, tačnost je postala bolja i najmanja vrednost se dobija za vrstu biljke kojoj pripada.

5. NODE-RED I THINGSPEAK

Kao što je već rečeno u integrisanom razvoju okruženju, nazvanom *Thonny*, napisan je odgovarajući kod za implementaciju senzora *DHT11* i komunikaciju između odgovarajućih platformi, kao i rad sa kamerom i algoritmom za obradu slike koristeći programski jezik *Python*. Kako bi *DHT11* senzor radio ispravno, potrebno je instalirati biblioteku Adafruit_Python_DHT.

Korišćenje funkcije *Adafruit_DHT.read_retry* čitaju se podaci sa senzora. Vrijednosti sa senzora se šalju na *ThingSpeak* platformu (*IoT* analitička platforma, koja omogućava vizuelizaciju i analizu podataka) i *Node-*RED koristeći *HTTP* protokol, koji se zasniva na principu server-klijent. Korisnik šalje zahtev, a server odgovara na taj zahtev.

Node-RED Flow (tok) se odnosi na povezivanje i sekvenciranje različitih ulaznih, izlaznih i procesnih čvorova unutar *Node-RED* platforme [7]. Svaki čvor unutar toka obavlja jedinstven i specifičan zadatak. Kada se podaci prenose do čvora, čvor ih obrađuje prema svojoj naznačenoj funkciji, prije nego što ih proslijeđuje sljedećem čvoru u toku.

Potrebno je odabrati jedan od dva button-a, koji predstavljaju način na koji korisnik želi da se utiče na uslove u prostoriji, ručno ili automatski. Ukoliko se klikne na taster “Automatski” pojavljuje se dropdown čvor i dobija se lista biljaka.

Klikom na ime biljke koja se nalazi u prostoriji, iskače message (show notrification) sa optimalnim opsegom temperature, koji se dalje šalje na join čvor na koji stiže i temperature sa senzora. Kada pristugnu oba podatka na join čvor, prelazi se na sljedeći čvor, gdje se provjerava koji je podatak prvi stigao, a koji drugi. Nekada se desi da prvo stigne podatak sa senzora, a nekada sa dropdowna. Ukoliko je podatak string, stigla je vrijednost sa *dropdown*-a, a ukoliko je number sa senzora Pristigle vrijednosti se smiještaju u promjenljive temperaturaDropdowna i temperaturaPosta. Potrebno je razdvojiti opseg temperature iz dropdowna, na minimalnu i maksimalnu vrijednost, koje su u dropdownu zapisane u obliku: minTemperatura-maxTemperatura.

Funkcija *split* je korištena za razdvajanje minimalne i maksimalne vrijednosti temperature. Ukoliko je temperatura senzora manja od minimalne vrijednosti dropdowna, pali se diode i gasi se ventilator, a ukoliko je veća od maksimalne vrijednosti temperature, gasi se diode i pali se ventilator. Kako *join* čvor uzima bilo koja dva podatka koja mu stižu, u funkciji je naznačeno da je

to pogrešno i ne uzima se u obzir prilikom upoređivanja temperatura.

Ukoliko je izabran taster “Rucno”, pojavljuje se padajućo meni sa biljkama, njihovim opsezima i poruka da je neophodno uporediti temperature sa senzora i opseg idealne temperature. Ukoliko je neophodno upaliti diodu ili ventilator, moguće je pomoću 2 *switch* čvora, koji kada su on tj.1, pale diodu ili ventilator, a off, gase.

6. ZAKLJUČAK

Uspješno je realizovana tema koristeći *DHT11* senzor za očitavanje temperature i vlažnosti, čije se vrijednosti prikazuju na *ThingSpeak* platformi. Uspješno je primljena vrijednost temperature i u Node- RED-u, kao i njeno poređenje sa opsegom idealne temperature određene biljke i paljenje *LED* diode ili ventilatora u skladu sa zadatim uslovima. Pronađen je odgovarajući algotiram za detekciju oblika lista nakon što je uslikan pomoću Raspberry Pi kamere. Testirane su slike, koje su uslikane kamerom sa mobilnog telefona, čime je potvrđena mogućnost korišćenja algoritma pri različitim uslovima.

Postoje mogućnosti poboljšanja rada, a to su:

- Smanjivanje/povećavanje jačine ventilatora u skladu sa neophodnom temperaturom
- Korišćenje nekih drugih aktuatora za signalizaciju, u slučaju kvara jednog od korišćenih
- Praćenje osvjetljenosti biljke uz pomoć senzora (npr. *BH1750*)
- Neke biljke zavise od godišnjih doba, pa im optimalne temperature variraju zimi i ljeti, pa bi ideja bila da se odabirom godišnjeg doba izlistaju idealne temperature u tom period
- Korišćenje *lm7805*, tranzistora ili mosfetova prilikom stabilizacije napona

Problemi prilikom realizovanja projekta:

- Nemogućnost istovremenog dolaska podatka sa senzora i sa dropdowna, kao i ne čuvanje trenutne vrijednosti, dok ne dođe nova (ne pomažu delay, ni *time.sleep(),* ni if i wait naredbe). Zato je korišten join čvor, ali mora se paziti na vrijeme čekanja
- U slučaju da se više podataka dovodi na drugi čvor, voditi računa koji podatak prvi stiže i uzeti u obzir sve mogućnosti, posebno ako su istog tipa
- Nemogućnost izdvajanja lista iz biljke koristeći algoritme za izdvajanje lista na osnovu boje in a osnovu oblika
- Ukoliko se dva lista drastično razlikuju, sličnost će im biti velika, a ukoliko se manje razlikuju, sličnost im je manja. Samim tim postoji mogućnost greške ukoliko se list biljke ne nalazi u bazi, a sličnost im je ispod 1. Ako se list biljke nalazi u bazi, neće biti greške.
- Nemogućnost prepoznavanja istih listova prilikom različitih skaliranja (ovo je riješeno).

11. LITERATURA

[1] Ivan Mezei (2016) Računarska elektronika, praktikum laboratorijskih vežbi

[2] https://components101.com/sensors/dht11-temperature-sensor (pristupljeno u septembru 2024.).

[3] https://dsp.etfbl.net/multimediji/2017/09_slika.pdf (pristupljeno u septembru 2024.).

[4] https://www.geeksforgeeks.org/filter-color-with-opencv/ (pristupljeno u septembru 2024.).

[5] https://towardsdatascience.com/canny-edge-detection-step-by-step-in-python-computer-vision-b49c3a2d8123 (pristupljeno u septembru 2024.).

[6] https://www.tutorialspoint.com/how-to-match-image-shapes-in-opencv-python*2013* (pristupljeno u septembru 2024.).

[7] https://en.wikipedia.org/wiki/Node-RED (pristupljeno u septembru 2024.)

- **Kratka biografija:**

Nikolina Goronić rođena je u Prijedoru, gdje završava osnovnu i srednju školu. Na Fakultetu tehničkih nauka odbranila je 2023. diplomski rad iz oblasti Embeded sistema i algoritama, na temu: “Projektovanje i verifikacija Linear Congruential Generator hardverskog akceleratora”.

kontakt:
nikolinagoronic@gmail.com