Poročilo o implementaciji detektorja Cannyjevega detektorja robov

Lucija Fekonja

28. februar 2024

Povzetek

V sklopu druge seminarske naloge se osredotočamo na iskanje robov v medicinskih slikah, pridobljenih iz baze CTMRI. Pri tem smo uporabili Cannyjev detektor robov, ki smo ga sprva implementirali na posameznih slikah. Nadaljnje izboljšave smo dosegli z vpeljavo 24-povezanosti, s katero smo povezali robove iz zaporedja slik. Ta pristop nam je omogočil pridobitev 2D slik robov v prvem delu postopka, medtem ko smo v drugem delu združili zaporedje 2D slik v eno tridimenzionalno sliko.

1 Uvod

V sodobni medicinski diagnostiki so napredki na področju obdelave medicinskih slik ključnega pomena za natančnejše diagnoze in učinkovitejše zdravljenje. V sklopu seminarske naloge se osredotočamo na iskanje robov v medicinskih slikah, pridobljenih iz baze CT/MRI [1], z namenom izboljšanja vizualizacije in analize anatomskih struktur. Pri tem smo uporabili Cannyjev detektor robov, ki smo ga sprva implementirali na posameznih slikah. Osnovna verzija programa je shranjena v `canny.mm`. Nadaljnje izboljšave smo dosegli z vpeljavo 24-povezanosti, s katero smo povezali robove iz zaporedja slik. To smo storili s programom `canny24.mm`. Ta pristop nam je omogočil pridobitev 2D slik robov v prvem delu postopka, medtem ko smo v drugem delu združili zaporedje 2D slik v eno tridimenzionalno sliko. Pri implementaciji smo sledili postopku opisanjem v [2].

2 Metoda

Metoda, ki smo jo uporabili za zaznavanje robov v medicinskih slikah, je temeljila na Cannyjevem detektorju robov in je vključevala štiri ključne korake. Prvi korak je bil filtriranje slike z Gaussovim filtrom, s čimer smo zmanjšali šum. Nato smo z uporabo Prewitt Crossovega operatorja določili velikost in kot gradienta slike. Sledil je non-maximum suppression, ki je izničil piksle šibkejše od svojih sosedov v pozitivni in negativni smeri gradienta. Dodatno smo določili

dva praga, pri čemer smo piksle z vrednostjo pod spodnjo mejo postavili na vrednost 0, vse nad zgornjo mejo pa smo ohranili. Preostale piksle smo pregledali glede na sosednjih 8 pikslov, pri čemer smo piksel obdržali, če je bil vsaj eden od sosedov nad zgornjo mejo. V nadaljevanju je vsak korak opisan nekoliko natančneje.

2.1 Filtracija slik z Gaussovim filtrom

Gaussov filter je nizkoprepustni filter, torej zmanjšuje visokofrekvenčne komponente v sliki, vključno s šumom in detajli. Jedro Gaussovega filtra velikosti $(2k+1) \times (2k+1)$ se izračuna kot:

$$H_{ij} = \frac{1}{2\pi\sigma^2} e^{-\frac{(i-(k+1))^2+(j-(k+1))^2}{2\sigma^2}} \qquad \text{za } i,j \in \{1,\dots(2k+1)\}.$$

V našem detektorju smo uporabljali σ med 1.4 in 2.5. Velikost jedra smo izračunali kot šestkratnik σ, kar je v našem primeru dalo matrike velikosti med 11×11 in 15×15.

2.2 Intenziteta in smer gradienta slike

Pri izračunu gradienta slike smo uporabili Prewitt Crossov operator. Prvi odvod po x, G_x, smo izračunali s konvolucijo

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \times I,$$

medtem ko smo prvi odvod po y, G_y, pridobili s konvolucijo

$$G_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \times I,$$

kjer je I prebrana slika. Intenziteto posameznega piksla smo nato določili z uporabo formule

$$mag(i,j) = \sqrt{G_x(i,j)^2 + G_y(i,j)^2},$$

smer pa z

$$ang(i,j) = \arctan 2(G_y, G_x).$$

2.3 Zatiranje ne-maksimuma

Za izboljšanje natančnosti pri detekciji robov smo robove dodatno ožili z uporabo postopka zatiranja pikslov. Ta postopek zmanjša število detektiranih robov, tako da ohrani le piksle z maksimalno intenziteto v smeri gradienta. To zatiranje se izvaja v štirih smereh, pri čemer se upošteva kot gradienta, ki se zaokroži na določene stopinje.

1. V smeri vzhod - zahod, kjer je kot zaokrožen na 0°, se piksel ohrani, če je njegova magnituda večja kot magnituda pikslov levo in desno od njega. V nasprotnem primeru se vrednost piksla postavi na 0.

2. V smeri sever - jug, kjer je kot zaokrožen na 90°, se piksel ohrani, če je njegova magnituda večja kot magnituda pikslov gor in dol od njega. V nasprotnem primeru se vrednost piksla postavi na 0.

3. V smeri severozahod - jugovzhod, kjer je kot zaokrožen na 135°, se piksel ohrani, če je njegova magnituda večja kot magnituda pikslov levo gor in desno dol od njega. V nasprotnem primeru se vrednost piksla postavi na 0.

4. V smeri severovzhod - jugozahod, kjer je kot zaokrožen na 45°, se piksel ohrani, če je njegova magnituda večja kot magnituda pikslov desno gor in levo dol od njega. V nasprotnem primeru se vrednost piksla postavi na 0.

2.4 Histereza

Z namenom dodatnega zmanjšanja števila detektiranih robnih pikslov smo uporabili dvojni prag. Za določitev praga smo vrednosti pikslov uredili glede na njihovo intenziteto. Urejene vrednosti smo razdelili na pet enakovrednih delov, pri čemer smo za spodnji prag vzeli vrednosti na petini urejene vrste, za zgornji prag pa vrednosti na štirih petinah. V naslednjem koraku smo vse piksle obdelane slike, katerih intenziteta je bila manjša od spodnjega praga, postavili na vrednost 0, medtem ko smo vse piksle z intenziteto nad zgornjim pragom ohranili. Piksle med spodnjim in zgornjim pragom smo obdržali, če je bil kateri od sosednjih pikslov z intenziteto večjo od zgornjega praga.

2.5 Iskanje robov zaporedja slik

Z zgoraj opisanim postopkom smo filtrirali posamezne medicinske slike. Kadar pa imamo na vhodu zaporedje slik, si lahko pri iskanju robov pomagamo s sosednjimi slikami zaporedja. Tako določanje robov temelji na 24-povezanosti.

V procesu obdelave slik smo najprej uporabili Cannyjev detektor robov, kot smo ga predstavili zgoraj. S tem postopkom smo pridobili zaporedje slik, kjer so robovi jasno izpostavljeni. Nadalje smo se osredotočili na analizo dveh zaporednih slik, označenih kot n-ta in $(n+1)$-ta slika. Za vsak piksel na mestu (i, j) na obeh slikah smo izvedli naslednje preverjanje. Če sta piksla izpolnjevala pogoje spodaj, smo v n-to sliko dodali vse piksle na najkrajši poti od (i, j) do tistih pikslov, ki so imeli na $(n+1)$-vi sliki v 5×5 okolici opazovanega piksla vrednost 1. V nasprotnem primeru smo se vrnili na prvi korak oziroma nadaljevali z naslednjima zaporednima slikama, če smo v n-ti pregledali vse piksle.

1. Na n-ti sliki izberemo piksel z vrednostjo 1.

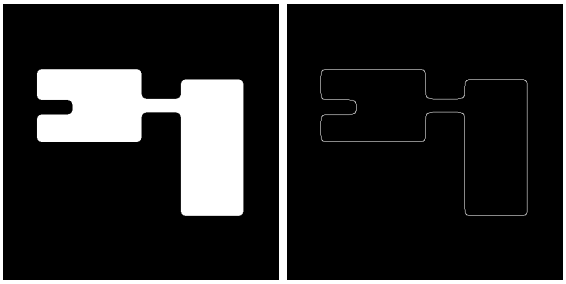
2. Soležeč piksel na $(n+1)$-vi sliki ima vrednost 0.

3. Na $(n+1)$-vi sliki v 3×3 okolici opazovanega piksla ne obstaja piksel z vrednostjo 1.

4. Ne n-ti sliki v 5×5 okolici opazovanega piksla obstaja piksel, ki nima vrednosti 0.

3 Rezultati

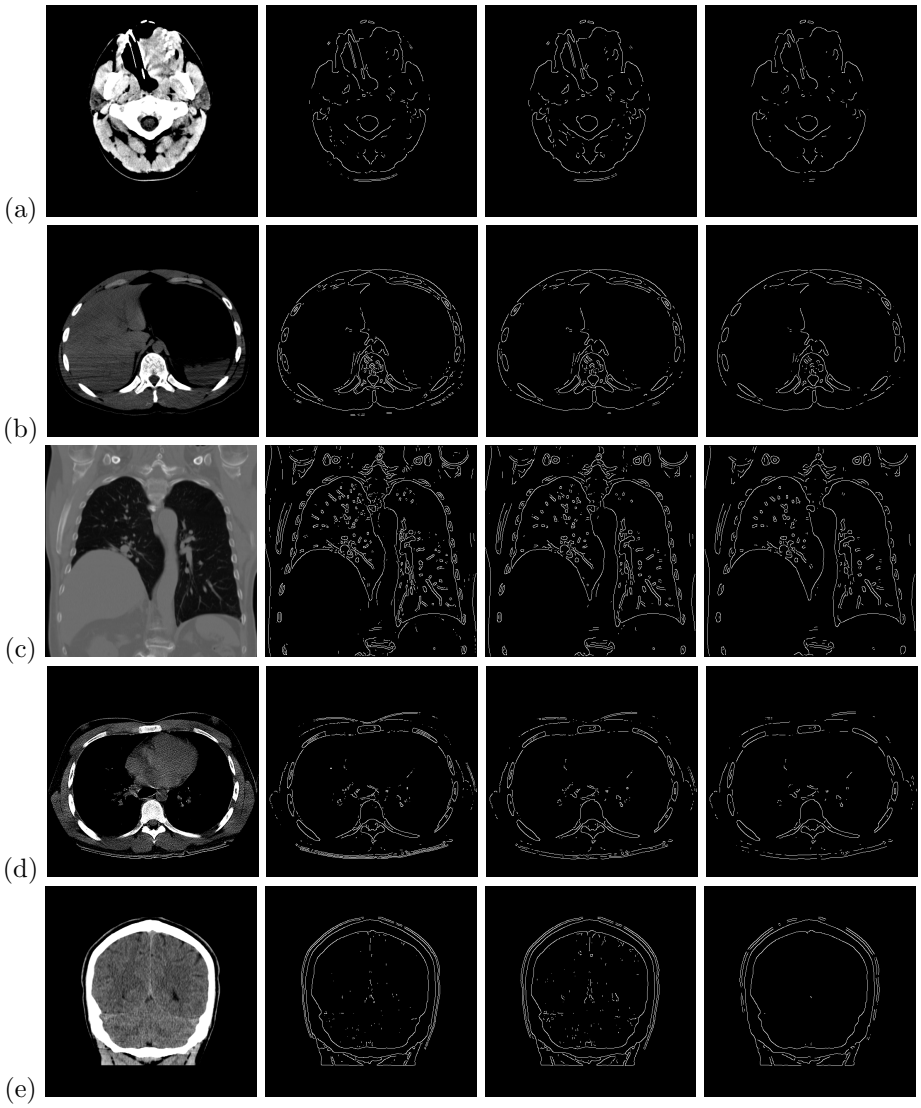
Preverimo najprej delovanja osnovnega Cannyjevega detektorja na preprostem testu.



Slika 1: Cannyjev detektor uporabljen na preprosti obliki.

Opazimo, da detektor na testu deluje dobro za $\sigma \in \{1.3, 1.4, \dots\}$.

Oglejmo si še Cannyjev detektor na nekaj medicinskih slikah. Opazimo, da Cannyjev detektor v večini primerov najbolje deluje pri $\sigma = 1.4$, zato bomo tega uporabljali v nadaljevanju. Filter je namreč odstranil nekaj šuma, ki je še razviden pri $\sigma = 1$, sploh v primeru (e), vseeno pa je ohranil zadostno količino robov, ki pri $\sigma = 1.8$ manjkajo.

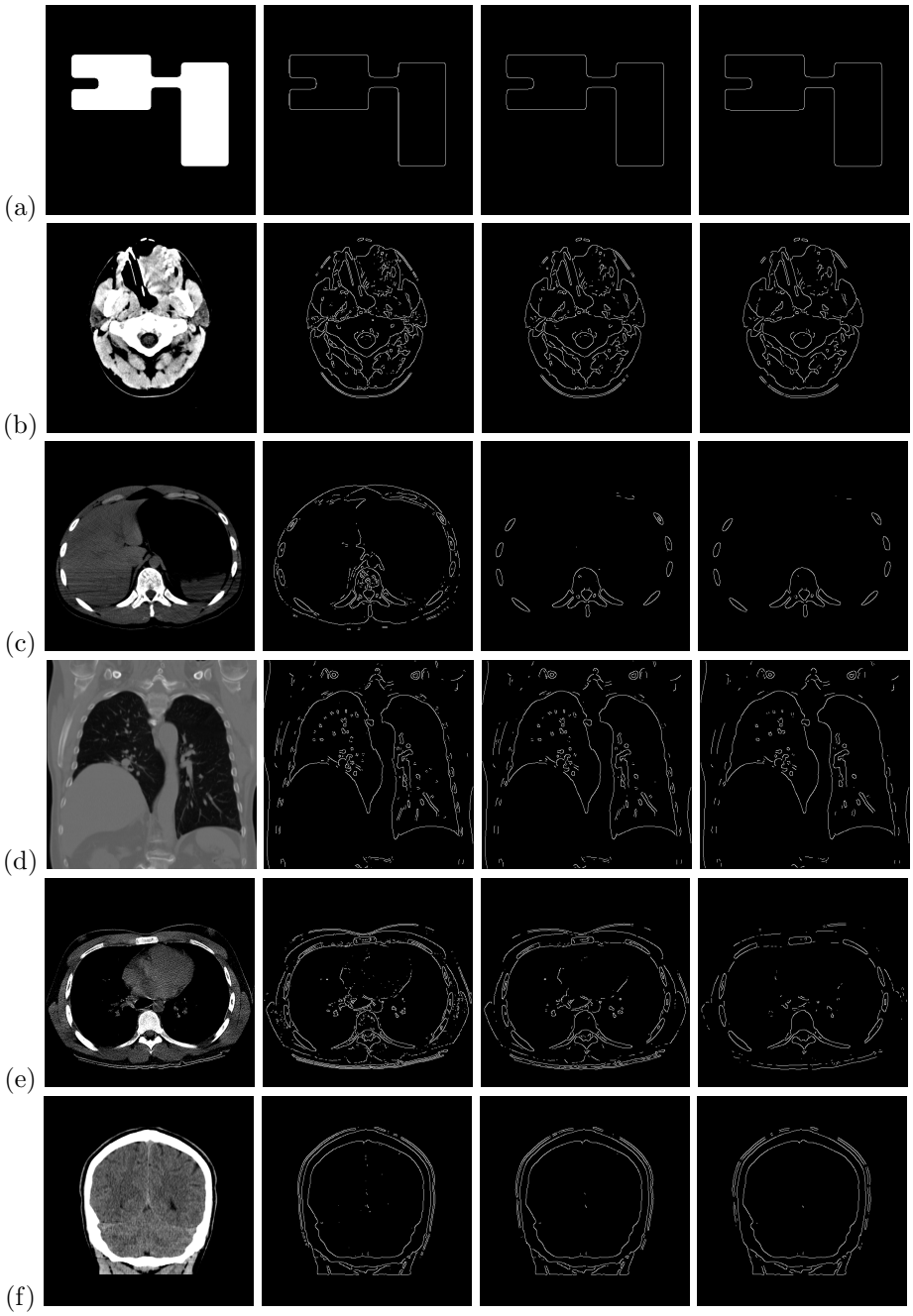


Slika 2: Cannyjev detektor uporabljen na medicinskih slikah pri $\sigma = 1$ (levo), $\sigma = 1.4$ (sredina) in $\sigma = 1.8$ (desno).

Nadgradnja

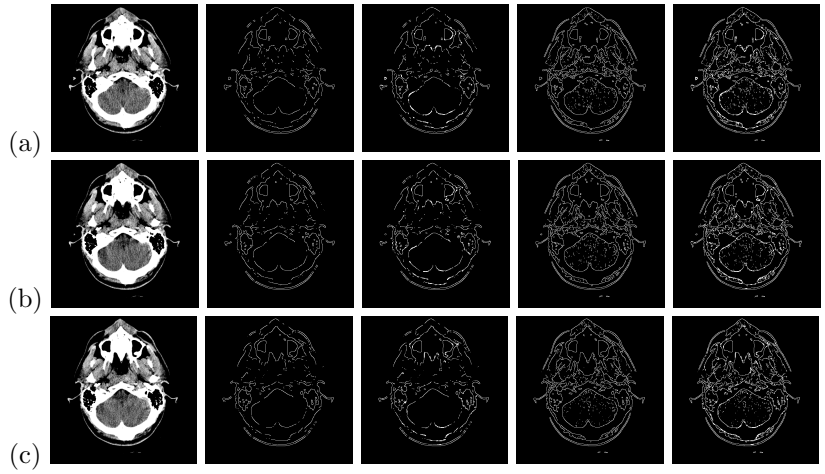
Do zdaj smo dvojni prag iz zadnjega koraka Cannyjevega detektorja določali tako, da smo neničelne vrednosti postavili v urejeno vrsto. Obstajajo pa že dobri algoritemi za določanje pragov. Otsujeva metoda je lahko uporabljena direktno na sliki, iz točke 3 v opisu metode. Z njo dobimo vrednost zgornjega praga. Za vrednost spodnjega vzamemo polovično vrednost zgornjega. Oglejmo

si cannyjev detektor s prilagojenim dvojnim pragom na istih primerih kot zgoraj. Na testnem primeru metoda deluje za vse $\sigma > 0$, kar je precej bolje prvotne verzije. Opazimo, da metoda najbolje deluje pri $\sigma = 1$, saj večja jedra Gaussovega filtra ne zaznajo marsikaterega pomembnega robu.



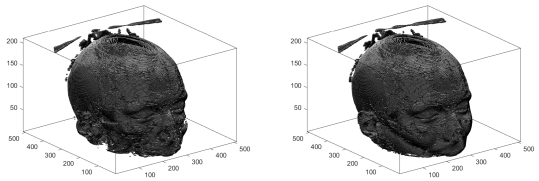
Slika 3: Cannyjev detektor z Otsujevim določanjem dvojnega praga uporabljen na testnem primeru in medicinskih slikah pri $\sigma = 1$ (levo), $\sigma = 1.4$ (sredina) in $\sigma = 1.8$ (desno).

V nadaljevanju si oglejmo še detektor robov zaporedja slik iz baze CT/MRI. V tem delu smo dodali povezovanje med dvema zaporednima slikama. Najprej primerjajmo nekaj zaporednih slik brez tega povezovanja in z njim.



Slika 4: Cannyjev detektor uporabljen na slikah 0034.png (a), 0035.png (b) in 0036.png (c) iz zaporedja slik 4 pacienta $Patient-240163-01$. Zaporedoma si sledijo originalna slika, osnovni Cannyjev detektor, osnovni Cannyjev detektor z 24-povezanostjo, Cannyjev detektor z Otsujevim dvojnim pragom ter Cannyjev detektor z Otsujevim dvojnim pragom in 24-povezanostjo.

V tem primeru je z 24 - povezanostjo dodanih približno 1500 pikslov na sliko. Če zaporedje slik narišemo eno nad drugo, dobimo nasledjo tridimenzionalno sliko.



Slika 5: Cannyjev detektor z 24 - povezanostjo uporabljen na zaporedju slik 4 pacienta $Patient-240163-01$. Na desni sliki je uporabljena Otsujeva metoda za določanje pragov.

Primeri so shranjeni v mapi `Slike`.

4 Zaključek

Implementirali smo Cannyjev detektor robov za izboljšano obdelavo medicinskih slik iz baze CTMRI. Z dodatkom Otsujeve metode določanja pragov smo povečali natančnost detekcije robov, kar je izboljšalo rezultate našega algoritma. Nadalje smo razširili funkcionalnost programa z implementacijo 24-povezanosti, ki je omogočila boljšo povezavo robov med zaporednimi slikami.

Program detekcije robov smo pognali na celotni bazi CTMRI in si v tem poročilu ogledali nekaj primerov njegovega delovanja. Hkrati smo se trudili izboljšati filtre, zlasti z zamenjavo Gaussovega filtra s prilagodljivim filtrom. Naš poskus implementacije prilagodljivega filtra, ki bi nadomestil Gaussov, žal ni bil uspešen. Pri implementaciji smo sledili postopku opisanjem v članku [3]. Bralec si ga lahko ogleda v `adaptiveFilterKernel.png`

Literatura

[1] Taddei A., Jager F., Smrdel A., Bezlaj K., Zadnikar M. (2016) *CTMRI*, Univerza v Ljubljani, Fakulteta ra računalništvo in informatiko, Ljubljana, `https://lbcsi.fri.uni-lj.si/OBSS/Data/CTMRI/`

[2] Wikipedia contributors (2023) *Canny edge detector*, Wikipedia, The Free Encyclopedia, `https://en.wikipedia.org/w/index.php?title=Canny_edge_detector&oldid=1188845015`

[3] Isaack Kamanga (2015) *An Adaptive Approach to Improve Canny Method for Edge Detection*, Department of Electronic Engineering, Tianjin University of Technology and Education, Tianjin, P.R, China, `https://www.researchgate.net/profile/Isaack-Kamanga/publication/332571402_An_Adaptive_Approach_to_Improve_Canny_Method_for_Edge_Detection/links/5cbeca3292851c8d22feac0d/An-Adaptive-Approach-to-Improve-Canny-Method-for-Edge-Detection.pdf`