

Obligatorisk innlevering 2

Sivert M. Skarning

Mai 2019

1 Oppgave 1

1.1 Oppgavebeskrivelse

Implementer Canny algoritmen slik som gitt i boka/slides. Alle tre stegene av algoritmen skal implementeres på egen hånd, men det er lov å bruke innebygde operasjoner som konvolusjon. Det skal demonstreres at algoritmen fungerer som den skal ved å sammenligne resultatet med det den innebygde implementasjonen i scikit produserer på et lite utvalg av bilder.

1.2 Dokumentasjon

Canny edge detector er en algoritme for å detektere kanter i et bilde. Algoritmen består av tre deler.

- Gaussisk utjevning
- Non-Max suppression
- Hysteresis tresholding

I denn oppgaven har jeg prøvd å implementere disse tre delene av algoritmen. For Non-Max supression delen har jeg brukt en guide som inspirasjon. Hvilken del dette er, er merket i koden. Denne guiden finner du her ¹.

¹<https://towardsdatascience.com/canny-edge-detection-step-by-step-in-python-computer-vision-b49c3a2d8123>

1.3 Resultat

Resultatet er blandet. Får fortsatt ikke samme resultat som jeg gjør med den innebygde implementasjonen. Har prøvd forskjellige standardavvik på Gaussisk utjevning og fot øvre og nedre threshold. Resultatet av dette ser i figur 4, 5 og 6. Som du ser var kantene for tykke, og selve kanten på månen var aldri sammenhengende. Jeg antar at problemet ligger i implementasjonen av Non-Max suppression som ikke klarer å kjenne igjen kantene riktig. Som man ser i figur 7. Jeg prøvde med andre bilder også, men får fortsatt dårlig resultat. Et forsøk på å kantdetektere Lena kan bli observert i figur 8

2 Oppgave 2

2.1 Oppgavebeskrivelse

I denne oppgaven skal dere sammenligne ulike kantdeteksjonsalgoritmer. Det er her lov å ta i bruk innebygde implementasjoner. Dere må teste minst tre ulike algoritmer og dere skal teste med ulike verdier for konstanter slik som sigma og threshold. Resultatet skal vises frem og diskuteres.

2.2 Dokumentasjon

I denne oppgaven valgte jeg tre kantdeteksjons algoritmer.

- Sobel operator
- Prewitt operator
- Canny kantdetektor

For canny algoritmen prøvde jeg med tre verdier for standardavvik på det Gaussiske filteret, 1, 5 og 20. Selvom jeg implementerte canny algoritmen i forrige oppgave valgte jeg å bruke en innebygd versjon fordi jeg ikke var helt fornøyd med resultatet. Den innebygde algoritmen fungerer bra å her enkel å bruke. Det er derimot ikke mulig å endre thresholdene på hysteresis thresholding.

2.3 Resultat

Canny kantdetektor I figur 1 ser dere tre bilder. Disse bildene er resultatet av en Canny kantdetektor. Forskjellen på bildne er standardavvik brukt på den Gaussiske utjevningen. Fra venstre ser dere resultatet med

standardavvik på 1, 5 og 20. Med 20 som standardavvik blir bildet såpass utjevnet at Canny algoritmen ikke finner noen kanter i bilde. Følgelig blir hele blidet svart. På bildet med 1 i standardavvik ser man at alle kanter i bildet fremkommer. Med 5 i standardavvik er det kun de klareste kantene som kommer med. Dette resultatet er ønskelig da det er lett å velge filtrere ut de kantene som man annser som støy. Ved hjelp av å variere utjevningen kan man få fram ulike karakteristikker i bilde etter behov.

Sobel og Prewitt kantdetektor I figur 2 ser man resultatet av sobel operatorene i midten og prewitt operatoren til høyre. I figur 3 ser man at man trenger to sett med operatorer. En som går i x-retning og en som går i y-retning. Hvis man konvolverer et bilde med disse filterene får man kantene i x-retning og kantene i y-retning. Hvis man legger dette sammen får man kantene alle kantene i bilde. Som man ser i figur 2 er det ingen forskjell mellom Sobel og Prewitt operatorene for dette bilde. Det som er hovedforskjellen er at sobel koeffisientene kan endres etter behov, det kan man ikke med prewitt operatorene ²

Canny vs Sobel/Prewitt Forskjellen mellom canny kantdetektoren og Sobel/Prwitt kantdetekteringen er stor. Det første man legger merke til er hvor tynne og hvor tydlig kantene er på figur 1 i forhold til på figur 2. De tynne kantene kommer fra non-max suppression delen i Canny algoritmen. Her går man gjennom alle kantene å sjekker etter lokale maxima. Kantene blir tydligere på grunn av Hysteresis Tresholding. Her går man gjennom kantene og gjør de monokrome, man eliminerer også de pixelene som ikke er en del av en større kant for ikke å få med støy. Alt dette gjør at Canny kantdetektoren generelt er et bedre valg en Sobel/Prewitt avhengig av hvilke karakteristikker man er ute etter.

²<https://www.quora.com/What-is-the-basic-difference-between-Sobel-Operator-and-Prewitt-operator>



Figure 1: Resultatet av Canny kantdetktor

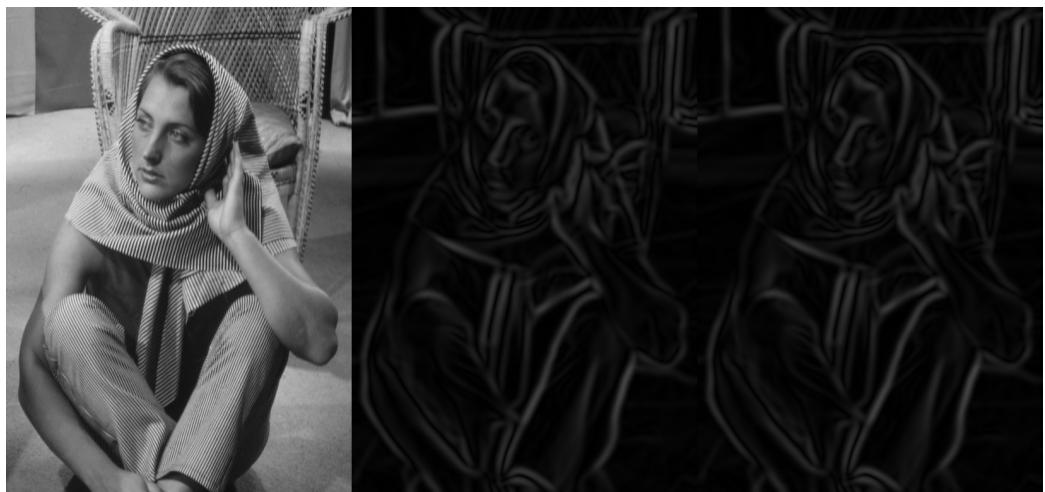


Figure 2: Resultat fra Sobel og Prewitt operator

Roberts								
-1	-1	-1	-1	0	1	-1	0	1
0	0	0	0	-1	0	1	-1	0
1	1	1	1	-1	0	1	-1	0
Prewitt								
-1	-2	-1	-1	0	1	-1	0	2
0	0	0	-2	0	2	-1	0	1
1	2	1	-1	0	1	-1	0	1
Sobel								

Figure 3: Sobel og Prewitt operatorer

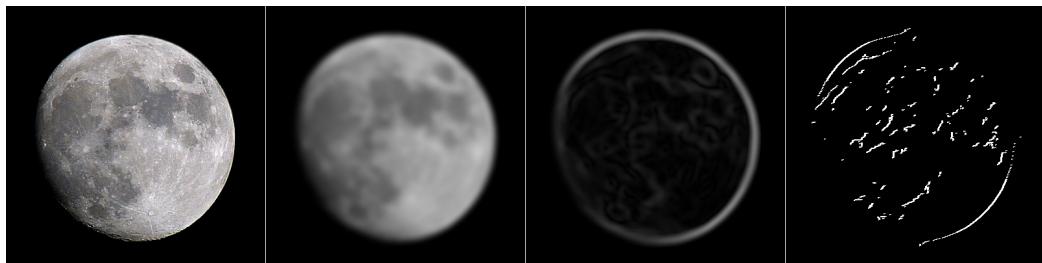


Figure 4: Canny kantdeteksjon, Gaussian(5), Th = 0.05 Tl = 0.6

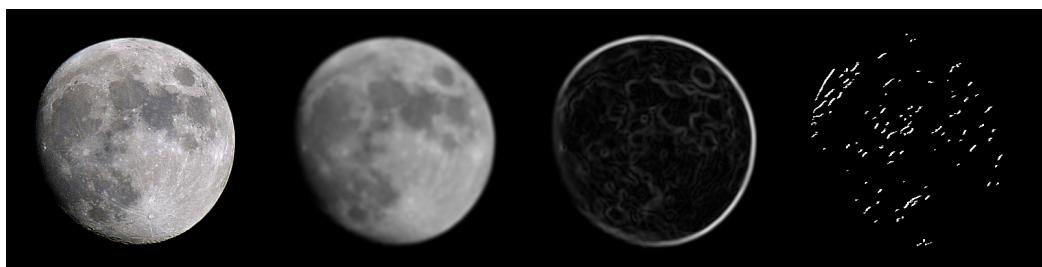


Figure 5: Canny kantdeteksjon, Gaussian(3), Th = 0.1 Tl = 0.5

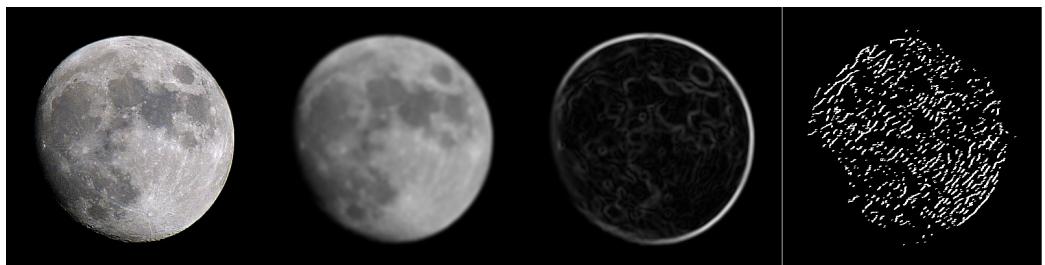


Figure 6: Canny kantdeteksjon, Gaussian(3), Th = 0.01 Tl = 0.3

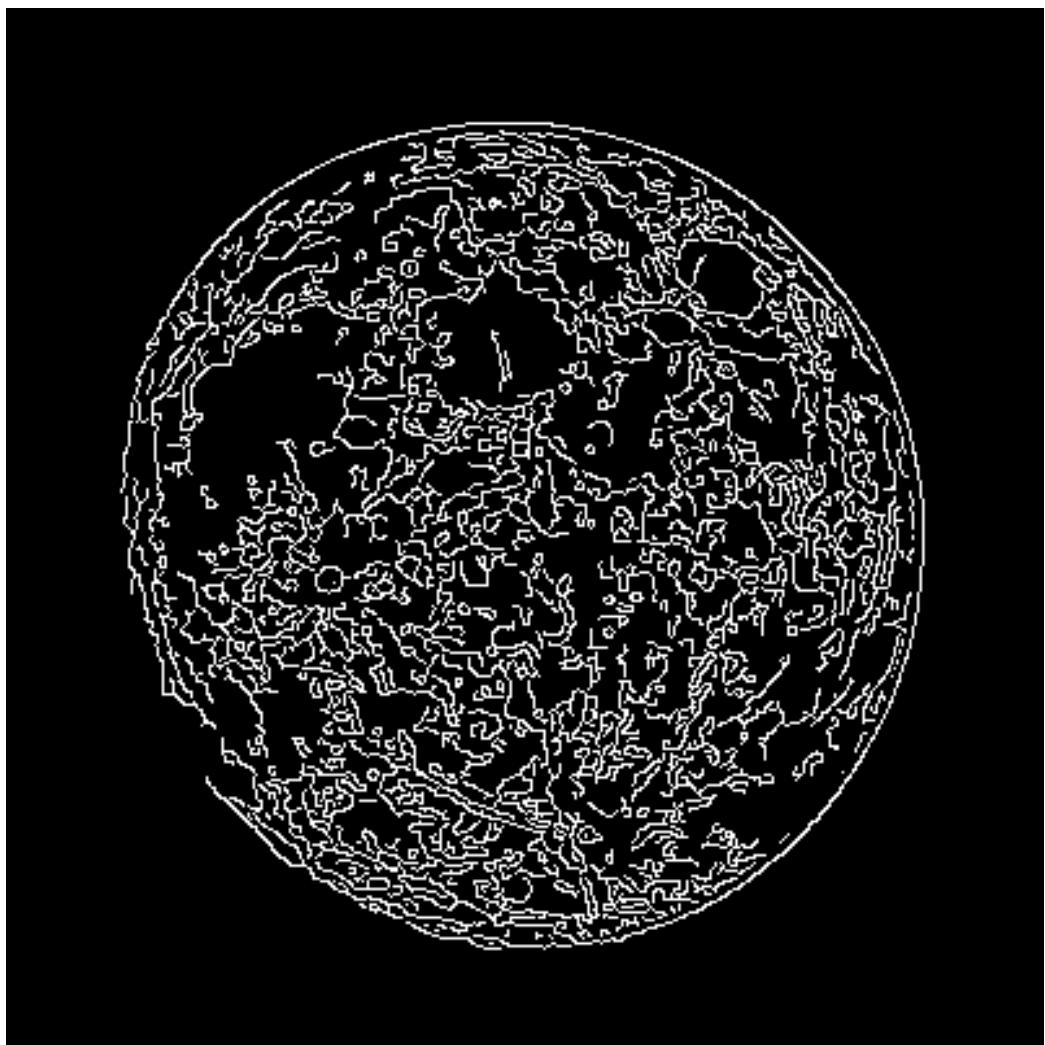


Figure 7: Scikit sin innebygde canny kantdetektor



Figure 8: Lena med canny kantdetektor