Obligatorisk oppgave 1

IN2070 - høst 2023

Dette oppgavesettet er på 3 sider, og består av 2 bildebehandlingsoppgaver som skal besvares.

Besvarelsen av denne og neste obligatoriske oppgave må være godkjent for at du skal få anledning til å gå opp til endelig skriftlig eksamen i kurset. Besvarelsene kan utarbeides i smågrupper på opp-til to studenter, men det er ikke noe i veien for å arbeide alene. Studenter i samme smågruppe kan levere identisk besvarelse, men samarbeidet må framgå av navnene på forsiden av besvarelsen.

Av forsiden skal det fremgå hvem som har utarbeidet besvarelsen.

Det forventes at arbeidet er et resultat av egen innsats. Å utgi andres arbeid for sitt eget er uetisk og kan medføre sterke reaksjoner fra IFIs side. Se `http://www.uio.no/studier/admin/obligatoriske-aktiviteter/mn-ifi-oblig.html`.

Den skriftlige rapporten leveres primært som en PDF-fil som inneholder hele besvarelsen, med figurer og bilder. Kode skal leveres i tillegg til PDF-en. Besvarelsen skal leveres via `https://devilry.ifi.uio.no`.

Legg merke til følgende:

- Alle filene må lastes opp hver gang man skal levere.
- PDF-en skal ha følgende navn: `in2070-oblig1-brukernavn.pdf`, der `brukernavn` byttes ut med ditt eget sådan.
- Oppgavene skal kunne kjøres med ">>python oppgave1.py" osv. (evt. oppgave1a.py osv.).
- Spørsmål angående innlevering: **mathime@ifi.uio.no**

Bildene det refereres til vil være å finne under: `https://www.uio.no/studier/emner/matnat/ifi/IN2070/h23/undervisningsmateriale/bilder/`.

Oppgaven utleveres onsdag 27. september 2023. Innleveringsfrist er **fredag 13. oktober 2023**.

Lykke til!

1 - Preprosessering av portrett for ansiktsgjennkjenning

Viktig: Dere skal her programmere transformene "fra bunnen av", altså ikke bruke ferdige programpakker!

Denne oppgaven går ut på å klargjøre (preprosessere) portrettbilder slik at det senere kan sendes til en ansiktsgjenkjenningsalgoritme.

Bildet vi skal jobbe med er `portrett.png`. Som vi ser er bildet både skjevt og det har lav kontrast. For å lettere kunne sammenligne dette bildet med andre bilder i en database, vil vi her standardisere både geometrien og kontrasten. I denne oppgaven skal vi gjøre dette på følgende måte:

1. Standardisere kontrasten ved å benytte en lineær gråtonetransform spesifisert slik at resultatbildet får en middelverdi på rundt 127 og et standardavvik på rundt 64. Bildet skal kunne lagres med 8 bit (uint8, altså ha verdier mellom 0 og 255.)
2. Standardisere geometrien ved å utføre en affin transform slik at øyne og munn passer over en forhåndsdefinert maske. Masken er gitt ved filen `geometrimaske.png`.

Lag en implementasjon av både forlengs- og baklengstransformasjon. Ved baklengstranformasjonen, prøv ut både nærmeste nabo og bilineær interpolasjon. Ved forlengstransform kan dere benytte den noe "naive" fremgangsmåten med å dytte de transformerte pikslene inn i nærmeste ut-piksel.

Dere skal altså ende opp med et resultatbilde som er like stort som maske-bildet (512×600 piksler), har god kontrast, og hvor øyne og munn finnes på de samme pikselkoordinatene som i maske-bildet.

NB! Dere kan benytte ferdige funksjoner til å lese/skrive fra/til fil. Gråtonetransformen og den geometriske transformen MÅ dere implementere selv.

Hva skal leveres:

1. Mellom-resultat-bildet etter gråtonetransformen, samt en kort forklaring pår hvordan du fant koeffisientene i den lineære transformen (a- og b-parametrene).
2. Forklaring på hvordan dere fant koeffisientene til den affine transformen. Samt hvordan dere fant koeffisientene til den inverse transformen som kreves i baklengstransformen.
3. Resultat-bilder for både forlengs- og baklengstranformasjonen. Sistnevnte for både nærmeste nabo og den bilineære interpolasjonen.
4. Diskusjon/forklaringer på eventuelle forskjeller i resultatene ved forlengs- og baklengsmapping, og ved nærmeste nabo og bilineær interpolasjon.
5. Programkode.

2 - Detektering av cellekjerner

Viktig: Dere skal her programmere hele Cannys algoritme fra "fra bunnen av", altså ikke bruke andres implementasjon for verken Cannys algoritme eller noen av dets bestanddeler, inkludert utvidelsen av inn-bildet, alle filtreringene, tynningen av gradientmagnitudebildet og hysterese-tersklingen.

Vi skal i denne oppgaven detektere kantene til cellekjernene i bildet `cellekjerner.png`. Dette bildet er et lysmikroskopibilde av DNA-fargede cellekjerner i et tynt snitt av svulsten til en prostatakreftpasient. Deteksjon av cellekjernene er nødvendig for å kunne separat analysere cellekjernene for å f.eks. avgjøre aggressiviteten til kreftsvulsten, noe som er viktig for å kunne bestemme hvor kraftig etterbehandling pasienten bør bli utsatt for.

a) Programmér en generell implementasjon av konvolusjon av et bilde med et konvolusjonsfilter. Du kan anta at naboskapet er rektangulært med odde lengder og har origo i senterpikselen. Ut-bildet skal ha samme størrelse som inn-bildet og bilderandproblemet skal bli behandlet ved å utvide inn-bildet med nærmeste pikselverdi.

b) Implementér Cannys algoritme. Bruk et ekte Gauss-filter gitt ved

$$h(x, y) = A \exp\left(- \frac{x^2 + y^2}{2\sigma^2} \right).$$

hvor A settes slik at summen av vektene blir 1.Bruk den symmetriske 1D-operatoren til å estimere gradientene og benytt 8-tilkobling i hysterese-tersklingen. For å tynne gradientmagnituden skal du benytte metoden i notatet til den andre filtreringsforelesningen og som også er beskrevet på side 720-722 i lærebokas 3. utgave, side 730-731 i 4. utgave, d.v.s. at man går igjennom hver piksel i gradientmagnitudebildet og setter verdien til 0 dersom minst én av de to 8-naboene i eller mot gradientretningen har større gradientmagnitude.

Tips:

1. For å tilpasse Gauss-filterets størrelse til parameteren `sigma` kan du sette hver lengde av filteret som én pluss 8 ganger `sigma` rundet opp til nærmeste heltall.
2. Benytt implementasjon din fra a) til Gauss-filtrering og estimering av gradientene.
3. Det er selvsagt tillatt å benytte implementasjoner for avrunding og beregning av kvadratrot og tangens invers, f.eks. numpy-funksjonene `ceil`, `sqrt` og `atan2`.

c) Anvend din implementasjon av Cannys algoritme på bildet `cellekjerner.png`. Eksperimenter med parametrene `sigma`, T_h og T_l, til du har funnet noen verdier som gir et resultatbilde med flest mulig av kantene til cellekjernene og færrest mulig andre kanter.

Bildet `detekterte_kanter.png` viser et mulig resultatbilde, men det er på ingen måte en fasit – det er godt mulig du vil få et annet resultat (men lignende) ved å bruke andre parameterverdier!

Tips: For å lettere sammenligne forskjellige parameterverdier er det lurt å normalisere gradientmagnitudebilde slik at maksimalverdien er 1.

Husk at du kan benytte andres implementasjoner for å lese/skrive fra/til fil.

Hva skal leveres:

6. Tekstlig beskrivelse av implementeringene i a) og b).
7. Resultatbildet med tilhørende parameterverdier i c).
8. Drøfting av resultatbildet, inkludert hva som kjennetegner det metoden er bra på og det metoden ikke er bra på, samt hvordan dette blir påvirket av å endre parameterverdiene.
9. Programkode.