E5ADSB
Avanceret Digital Signalbehandling
Læsning af nummerplader

5. semester 2020

Science and Technology, Ingeniørhøjskolen, Aarhus Universitet, EDE, Herning

Deltagere

Studienummer	Navn	Initialer
201808044	Lasse Greve Rasmussen	LGR
201808886	Rikke Rothmann Pedersen	RRP

Underviser

Kristian Peter Lomholdt

Indhold

Forord

Denne rapport er dokumentation for det selvvalgte projekt der er arbejdet med under faget 'Avanceret Digital Signalbehandling'. Projektet tager udgangspunkt i den del af faget der omhandler digital billedbehandling.

Hvem er vi:

Rikke Rothmann Pedersen

Lasse Greve Rasmussen

Uddannelsessted: Aarhus Universitet, Herning

Studieretning: Ingeniør - Elektronik

Semester: 5. semester

Underviser: Kristian Peter Lomholdt

Afleveringsdato: 11 december 2020

Bedømmelsesdato: 12 januar 2021

Begreber og forkortelser

Begreb	Definition
nummerpladetegn	'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F','G','H','J','K','L','M','N','P','R','S','T','U','V','W','X','Y' og 'Z' 'O', 'I', 'Q','Æ','Ø' og 'Å' er ikke med da 'O' er meget sjælden, og resten ikke indgår i almindelige danske nummerplader[09]
fejl	En værdi der definerer afvigelsen af en måling fra et mål
Parameter	En værdi der siger noget om et objekt. Disse parametre bruges som features i 'feature extraction'
Matrix	En tabel af elementer, der indeholder billedets data
Objekt	Matrix der indeholder et billede eller udsnit fra et billede
Pixel	Et billede består af pixels i en matrix. Efter form kan en pixel have en bolsk værdi, være en float, eller have flere værdier for farvesammensætning
SE	Struktur Element
Krydskorrelation	En måling af hvorledes to elementer ligner hinanden

Navne

Forkortelse	Betydning
RRP	Rikke Rothmann Pedersen
LGR	Lasse Greve Rasmussen

Indledning

I denne rapport bliver der gennemgået hvordan der er fundet en metode til at tolke tegnene på nummerplader, som automatisk findes ud fra billeder af biler.

Gennem processen vil der blive arbejdet med preprocessering, segmentering, feature extraction og objekt klassifikation.

Projektet har været opdelt i to dele:

- Rikke har arbejdet med preprocessering og segmentering, så 7 objekter af nummerpladetegn automatisk udskæres fra et billede af en bil. Objekterne sendes videre til feature extraction.
- Lasse har arbejdet med feature extraction og klassifikation, hvor der udlæses parametre for de modtagne objekter, og nummerpladetegnene klassificeres ud fra disse.

Grænsefladen mellem de to dele er altså et objekt der indeholder et udskåret nummerpladetegn.

Figur 1 - overordnet diagram over systemet

Problemformulering

Hvordan er det muligt at skabe binære billeder ud fra de enkelte karakterer fra en nummerplade på et farvebillede taget af forenden eller bagenden af en bil? og i hvor høj grad er det muligt at automatisere denne proces?

Hvordan kan man tolke disse binære billeder som specifikke nummerpladetegn? Og hvor stor sikkerhed giver de anvendte metoder?

Afgrænsning

For at overskueliggøre opgaven, er der lavet nogle afgrænsninger af emnet.

Der tages udgangspunkt i almindelige hvide danske nummerplader. Med det menes nummerplader:

- bestående af først 2 bogstaver, og dernæst 5 tal.
- Med hvid baggrund, og ikke hverken helt eller delvist gul.

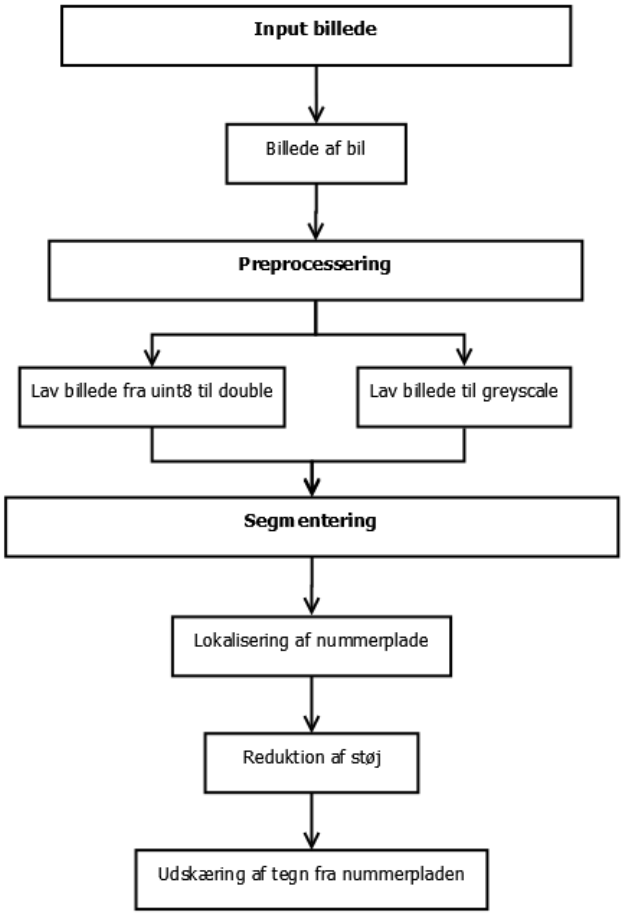
Der anvendes som udgangspunkt billeder taget lige ud for nummerpladen, med omtrent samme afstand, og samme opløsning for kameraet.

Preprocessering og segmentering (RRP)

I denne del vil der blive beskrevet hvordan en nummerplade automatisk kan findes på et billede af en bil, således de 7 tegn kan udskæres og analyseres.

Metode

Her vil der blive beskrevet de metoder der er brugt, for at løse første del af projektet. En overordnede oversigt over metoderne, kan ses på Figur 2.



Figur 2 - Oversigt over preprocessering og segmentering

Input billede

Input billederne er af biler med en nummerplade. Disse billeder er hovedsageligt taget med en FairPhone2 med opløsning på 4096 x 3072, og få billeder er taget med en OnePlus med opløsningen 4608 x 2176. Billederne er af front og bagende af bilen, samt forskellige biler med forskellige farver. Derudover er der forskel på motivet omkring bilen, hvilket giver forskellige farveniveauer på billederne. Dette betyder at der både er generelt lyse og mørke billeder samt lyse og mørke biler. Derudover er nogle billeder taget lidt på skrå, for at gøre det endnu sværere at aflæse.

Preprocessing

Første step er at lave billederne fra RGB til greyscale. Ved at lave billedet til greyscale, bliver det 2-dimensionelt, i stedet for 3-dimensionelt. Dette betyder at billedet fylder mindre, hvilket gør at det er hurtigere at arbejde med. Flere dimensioner betyder mere data, som i dette tilfælde bare er støj. Billedet kan laves greyscale ved at bruge funktionen rgb2grey().

Billedet er ofte i uint8, og laves til double for at holde niveauerne inden for 0-1. Derefter laves et histogram for at finde fordelingen af lysniveauet, således der kan vælges en tærskelværdi. Denne tærskelværdi bruges til at lave billedet binært. For at lave billedet fra uint8 til double, bruges funktionen im2double().

Segmentering

I segmenteringen skal nummerpladen lokaliseres og udskæres fra billedet. Derefter skal tegnene findes, og udskæres til enkelte elementer.

Nummerplade lokalisering

For at finde nummerpladen, bruges der normaliseret krydskorrelation.

Krydskorrelation kan findes på formlen:

$$r(x,y) = \sum_s \sum_t w(s,t) I(x+s, y+t)$$

Hvor w er template og I er billedet.

Der bruges ikke standard krydskorrelation, da denne form er meget følsom overfor lys. Billederne er ikke taget i kontrollerede miljøer, hvilket giver meget forskel på lys niveauet.

Normaliserede krydskorrelation kan findes på formlen[08]

$$\frac{\sum s \sum t (w(s,t) - w_{mid}) \sum s \sum t \left(I(x+s, y+t) - I_{xymid}\right)}{\sqrt{\sum s \sum t (w(s,t) - w_{mid})^2 \sum s \sum t \left(I(x+s, y+t) - I_{xymid}\right)^2}}$$

Hvor w er template, I er billedet, men her bliver middelværdierne trukket fra indenfor det lille område der laves korrelation. Der bliver divideret med variansen, for at normalisere i forhold til energien.

Normaliseret krydskorrelation er dog meget følsom overfor rotation, hvilket kan give lidt problemer når der tages billeder af biler. Dog er algoritmen effektiv og simpel, og kræver ikke megen preprocessering. I matlab hedder funktionen normxcorr2(). Denne korrelation bruger en template, som i dette tilfælde bliver en generisk dansk nummerplade[06]. Denne nummerplade er en dansk EU nummerplade, hvor EU-delen er skåret væk. Alle biler har den hvide del, men ikke alle er EU-nummerplader, så dette er valgt, for at bruge en mere generisk nummerplade der virker på flere biler.

Denne template skal laves binær, og kan skaleres op eller ned i størrelse efter behov. Til sidst skal templaten korreleres med det binære billede af bilen.

Efter korrelationen fås en matrix hvor høj korrelation giver et højt tal, og en lav korrelation giver et lavt tal. Derfor kan man tage højeste punkter fra korrelationen, som bruge det til at lave et threshold på matrixen. Så ender man med et billede med et punkt hvor der er højest korrelation. Dette punkt vil være i det øverste venstre hjørne af nummerpladen, som skal bruges til at udskære nummerpladen.

For at finde det nederste højre hjørne af nummerpladen, tages størrelsen af template [m,n], som kan lægges til øverste venstre hjørne. Når man har begge hjørner af nummerpladen, kan der bruges imcrop til at udskære billedet.

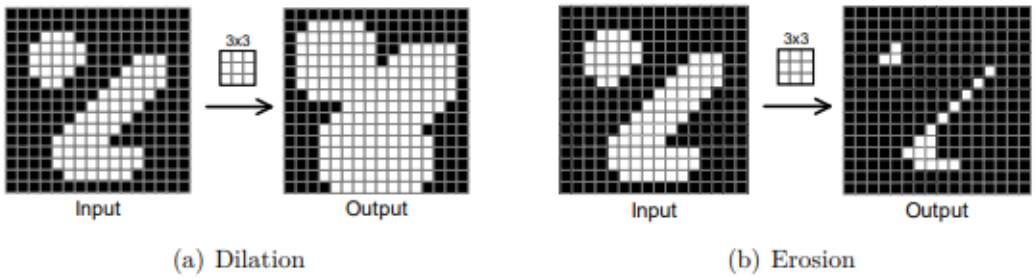
Reduktion af støj

Nummerpladerne kan ofte indeholde støj, det kan være prikker eller muterede tegn, som skal fjernes.

Figur 3 - Nummerplade med støj øverst på pladen, som hænger sammen med tegnene. Dette kan give nogle tegn med ekstra dele, og kan gøre dem svære at genkende.

Figur 4 - Et andet eksempel på en nummerplade med støj.

Ved hjælp af binære morfologiske operationer, kan man nemt fjerne støj. Dette kræver at man har et strukturelement, som er et vindue hvor der bliver udført logiske operationer i. Vindue størrelsen kan man selv bestemme, men det skal være et ulige tal. De logiske operationer man kan udføre, er AND og OR. Når der laves en AND operation, kaldes det også erosion, som krymper hvide pixels omkring figurer. Derudover er der en OR operation, som kaldes dilation, og modsat erosion, udvider hvide pixels omkring figurer.



Figur 5 - Eksempel på dilation og erosion [02]

For at rengøre nummerpladerne, eksperimenteres med et passende SE, som skal passe til billede størrelsen og området der skal behandles i. For at få de reneste nummerpladetegn, bruges der opening efterfuldt af closing. Open betyder at der først laves en erosion for at fjerne potentielle prikker og mindske sorte områder der ikke ønskes, og derefter laves dilaion for at lappe de huller i tegnene der kan være forekommet. Efter der laves en open laves close, som modsat open laver dilation først og derefter erosion. En ekstra dilation sikre at alle huller i tegene, som ikke skal være der, bliver fyldt ud. Dette gør også at omkredsen på tegnene

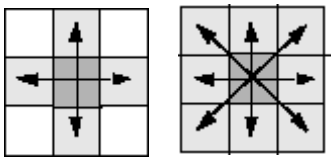
bliver større, derfor laves der erosion igen for at lave omkredsen mindre, så man ender med en nummerplade hvor tegnene bliver mere tydelige.

Segmentering af tegn på nummerpladen

Når nummerpladen er udskåret og rengjort, skal hver enkelt tegn tages ud fra billedet, således tegnet kan blive analyseret.

Det første step er at lave billedet negativ, for at gøre baggrunden sort, og gøre det element vi er interesseret i til hvid. Dette betyder at matrixen for billedet har 1 hvor der et element og 0 hvor der er baggrund.

For at finde elementerne i matrixen, bruges funktionen bwlabel(). Denne funktion kigger de steder, hvor de hvide pixels er i kontakt med sorte pixels. Der findes to forskellige måder at afskille elementer, og det er med 4-connectivity og 8-connectivity. Med 4 betyder det at der kigges på pixels ovenover, nedenunder, til højre og venstre for den nuværende analyseret pixel. De 8 betyder at der kigges på alle pixels rundt om den nuværende analyseret pixel, og et eksempel kan ses på Figur 6.



Figur 6 - bwlabel() for 4(venstre billed) og 8(højre billed) connectivity[05]

Der er valgt at gå med 4-connectivity, da en skygge på nummerpladen kan få tegnene til at flyde sammen, derfor bruges den grove model.

Når elementerne er fundet og tildelt et tal, kan det skæres ud således man kun har et tegn per matrix. For at udskære tegnene, bruges der imcrop(), som kan tage et rektangulært argument som skal udskæres. Denne rektangel findes ved at finde det sted man møder første hvide pixel fra alle sider(top, bund, højre og venstre).

Implementering

I dette afsnit kan man se implementering af de metoder som blev fremstillet i metodeafsnittet. Dette afsnit omhandler kun input billede, processering og segmentering. Koden kan findes på github [07].

Input billede

Inputbilledet er af en Suzuki Alto, i en garage. Der er forskellige lydniveauer i garagen, samt flere steder med rektangulære former.

Figur 7 - Original billede før preprocessering

Dette billede vil blive brugt som eksempel, gennem implementering af preprocessering og segmentering.

Preprocessering

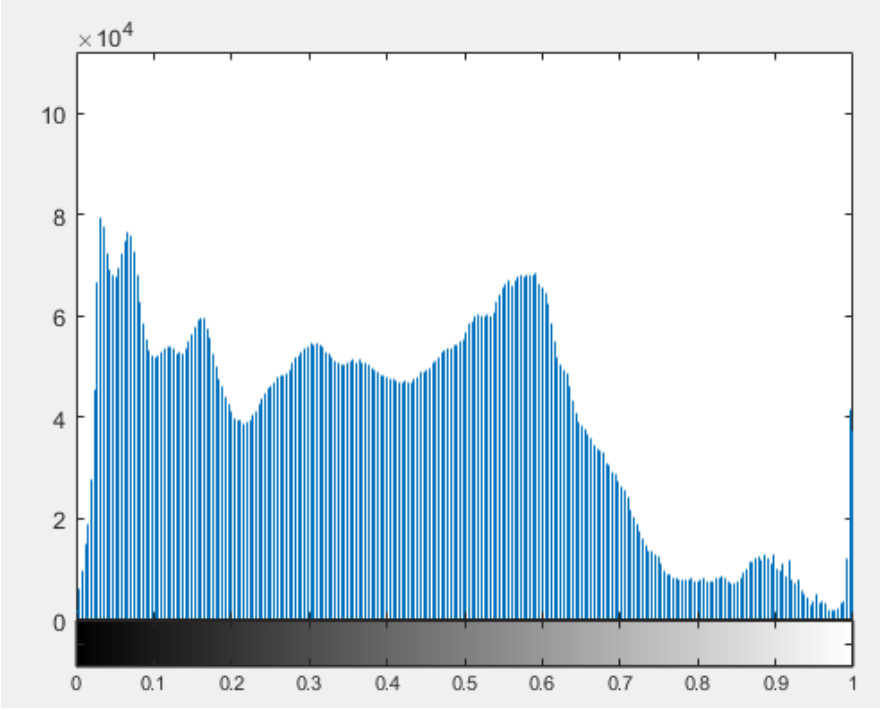
Billedet laves fra rgb til greyscale, og fra uint8 til double.

Resultatet kan ses på Figur 8.

Figur 8 - Bil i greyscale

Histogrammet kan nu aflæses, for at analysere niveauerne.

På Figur 9 ses fordelingen af lysstyrken, og dette er den sværeste del at gøre automatisk. Ikke alle billeder har den samme fordeling, hvilket betyder nogle billeder kræver et threshold på 0,30 hvor andre kræver 0,70.



Figur 9 - Histogram over greyscale billede af bil

Dette billede, Figur 8, kræver et threshold på 0,72 for at få en klar nummerplade uden skygger, og uden huller i tegnene.

For at finde en formel til at lave automatisk passende threshold, laves der en lille analyse.

I Tabel 1 ses der udregnede værdier for hver bil-billede under "Udregnet værdier for billederne", som potentielt kan bruges til at finde en formel.

Det ønskede threshold er under "Threshold" i tabellen, hvor man kan se en spredning fra 0,34 til 0,72.

Under "Udregnet threshold værdier med forskellige formler" ses grønne og røde bokse med tal, som skal vise hvor godt den respektive formel passer på det ønskede threshold. Stærk rød betyder at den er langt fra, og stærk grøn betyder den at tæt på.

Når man kigger på Tabel 1 ville det være oplagt at vælge formelen "max – mean", da den indeholder mest grøn, men den der passer bedst er i virkeligheden "max / 2,5"(som er markeret med fed skrifttype i tabellen).

Formlen "max-mean" passer på to biler, **_tyr1** og **_white1**, og en tredje bil mangler 1 af de 7 tegn, som er **_1**. Derimod passer "max / 2.5" på 3 biler, om er **_1**, **_2** og **_red1**.

Karakterer udlæst	Bil billede	Udregnet værdier for billederne			Threshold	Udregnet threshold værdier med forskellige formler				
		max	mean	median	ønsket	**max/2,5**	max-mean	max-med*2	2-all	mean+med
ja	_1	0,8706	0,3059	0,2431	0,50	0,34824	0,5647	0,3844	0,5804	0,5490
ja	_2	0,8863	0,4350	0,4157	0,34	0,35452	0,4513	0,0549	0,2630	0,8507
ja	_red1	0,9882	0,3395	0,3176	0,40	0,39528	0,6487	0,353	0,3547	0,6571
nej	_red2	0,9961	0,3878	0,3882	0,55	0,39844	0,6083	0,2197	0,2279	0,7760
nej	_tyr1	1,0000	0,3860	0,3804	0,72	0,4	0,6140	0,2392	0,2336	0,7664
nej	_tyr2	1,0000	0,3843	0,3647	0,72	0,4	0,6157	0,2706	0,2510	0,7490
nej	_white1	1,0000	0,3857	0,3059	0,60	0,40	0,61	0,39	0,31	0,69

Tabel 1 - Oversigt over forsøg med forskellige formler til at automatisk finde en threshold værdi

Der er valgt at bruge formelen "max / 2,5" til at automatisere programmet, men der fås det bedste resultat hvis der vælges den specifikke threshold der passer. Der kan ses en oversigt over de brugte bilbilleder herunder:

_1

_2

_red1

_red2

_tyr1

_tyr2

_white1

Der vælges at gå videre med billedet fra _tyr1, derfor vælges den specifikke værdi på 0,72 til threshold, i stedet for formelen der bruges til automatieringen. Dette giver bedste resultat, og formlen virkede ikke på dette billede. Bliver der brugt værdien 0,61, bliver nummerplade-karaktererne for lyse, og de får huller.

Figur 10 - Bil efter threshold på 0,72

På Figur 10 ses bilen, hvor nummerpladen er tydelig.

For at sammenligne de forskellige billeder gennem preprocesseringen, er de opstillet på Figur 11.

Figur 11 - Overblik over original og de 3 operationer

Segmentering

I segmenteringen skal nummerpladen lokaliseres og hvert tegn skal udskæres. For at finde positionen af nummerpladen, bruger normaliseret korrelation, som bruger en template. Denne template er en EU nummerplade, men EU delen fjernes, og der laves en sort bar i venstre side. Dette betyder at både normale og EU nummerplader kan detekteres, da begge nummerplader har en isoleret hvid baggrund med sorte tegn.

Figur 12 - Generisk dansk nummerplade[06]

Efter korrelationen af template på Figur 12 og billede af bil efter threshold på Figur 10, kan man se hvor nummerpladen kan være. På Figur 13 kan man se hvide pixels, hvor der er høj korrelation.

Figur 13 - Korrelation af billede efter threshold og template

Det kan godt være lidt svært at se de hvide pixels på Figur 13, derfor er sat ring om de tydelige steder på Figur 14.

Figur 14 - Korrelation af billede efter threshold og template, med røde cirkler omkring de tydeligste områder

Her på Figur 14 ses fire punkter med de tydeligste hvide pixels, hvor den lille røde ring til venstre er på den hvide bil, den store røde ring i midten er fronten af Alto'en. Den lille røde ring i midten er lygten på Alto'en, og den større røde ring til højre er en rejsepalle.

For at finde det sted med højest korrelation, tages max-værdien af billedet på Figur 13, og trukket lidt fra. Dette giver et lille område, hvor der er højest korrelation, i stedet for den ene pixel med højest korrelation.

```
>> th = max(out,[],'all')-0.001

th =

    0.1938
```

Figur 15 - Resultat af korrelation med 0,001 trukket fra

Ud fra Figur 15 kan man aflæse, at denne værdi er 0,1938. Hvis alt er gået som planlagt, vil området være i øverste venstre hjørne af nummerpladen.

Figur 16 - Det punkt med højest korrelation samt modsat hjørne

På Figur 16 ses nummerpladen samt to grønne punkter. De to grønne punkter er markeret med en rød cirkel på Figur 17.

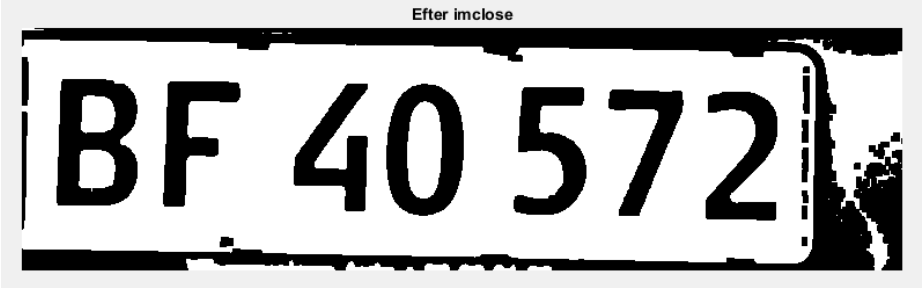
Resultatet på Figur 17 betyder at det punkt med højest korrelation er i øverste venstre hjørne, og den prik i nederste højre hjørne er fundet ud fra template's dimensioner. Det vil sige at templaten's størrelse har stor betydning, når nummerpladen skal findes. Er template for lille vil den ikke få hele nummerpladen med, og er den for står kommer der et for stort område med. Derudover skal templaten helst passe godt med størrelsen af bilens nummerplade, for at korrelationen bliver høj.

Figur 17 - Zoom ind på de to punkter fundet fra korrelation

Som man kan se på Figur 17, går template ud over nummerpladen på Alto'en, hvilket betyder at der fås lidt ekstra bil med.

Figur 18 - Udskæring af nummerplade med de to punkter modtaget fra korrelation

De to punkter fra korrelationen bruges til at udskære nummerpladen. På Figur 18 kan man se resultatet af udskæringen. Udskæringen indeholder meget støj, som skal fjernes. På Figur 19 ses resultatet efter der laves en close.



Figur 19 - Fjernelse af støj med imclose()

Der er nu fjernet de små sorte pixels, og tegnene står stadig klart. Der laves nu en open, for at reparere nummerpladen igen. Man kan se på Figur 20 at de mange prikker fra Figur 18 nu er blevet til en stor enhed.

Figur 20 - Fjernelse af støj med imopen()

Nummerpladen er nu klar til at blive inddelt i elementer, og sorteres således tegnene kan udskæres. Først laves nummerpladen negativ, for at få tegnene til at være 1 i matrixen.

Figur 21 - Billedet gøres negativt

På Figur 21 ses nummerpladen, hvor baggrunden er 0, og forgrund er 1. Der kan nu bruges bwlabel, til at inddele nummerpladen i elementer. På Figur 22 ses de 11 elementer, som skal sorteres, således kun tegnene bliver udskåret.

Figur 22 - bwlabel() med 4 connectivity

For at sortere store og små elementer fra, som ikke er tegn, bliver der kigget på summen af matrixen. Er den langt over eller under et gennemsnitlig tegn, bliver det smidt væk. Derudover bliver der også tjekket på dimensionen af elementerne, så de rigtige store elementer, såsom nummerpladerammen, ikke kommer med. Til sidst er der kun tegn tilbage, som hver bliver skåret ind fra alle sider, indtil der er en række/kolonne der indeholder 1.

Figur 23 - Udskæring af de enkelte tegn

Der kan nu udskæres tegn fra en nummerplade, som man kan se på Figur 23.
Hver tegn får sin egen matrix, så den kan sendes videre til en anden funktion.

Feature extraction og klassifikation (LGR)

Dette er anden del, hvor matrixen fra hvert nummerpladetegn skal klassificeres som et af de 33 nummerpladetegn. For at kunne bestemme et nummerpladetegn, skal der først udlæses egenskaber for objektet i matrixen (feature extraction), og dernæst skal det sammenlignes med templates for hvert nummerpladetegn.

Metode

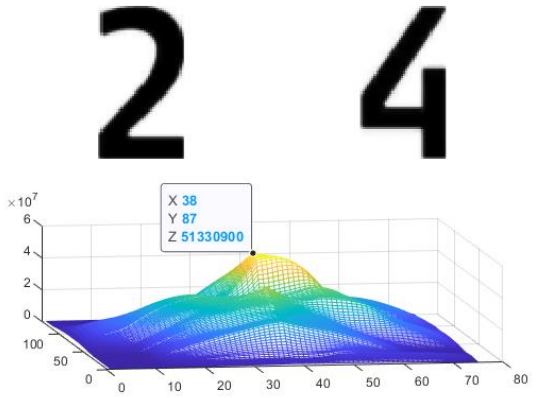
Til feature extraction og klassifikation er 3 metoder overvejet. Disse beskrives i de efterfølgende 3 afsnit.

Krydskorrelation

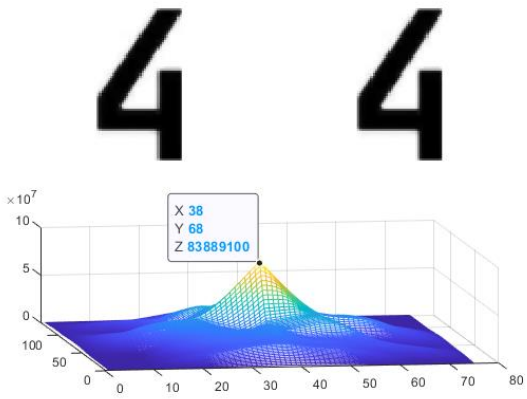
Der kan anvendes todimensionel krydskorrelation. Som det også ses i forbindelse med 'Nummerplade lokalisering', er dette er en simpel men meget effektiv metode til at se om der er sammenfald mellem objekter. Denne metode kan, i modsætning til andre metoder, anvendes på billeder i grayscale, og denne mulighed gør metoden effektiv og fleksibel.

Metoden kræver objekter af samme opløsning/skalering. Da krydskorrelation ellers ikke giver nogen mening. Det er dog muligt at tilpasse størrelsen på de objekter der skal analyseres, så metoden kan anvendes.

I Figur 24 og Figur 25 ses eksempler hvordan en krydskorrelation af to ens tegn giver en markant høj værdi midt i billedet, hvorimod et forkert tegn giver flere mindre sammenfald tilfældige steder i billedet.



Figur 24 - eksempel på krydskorrelation mellem '4' og '2'



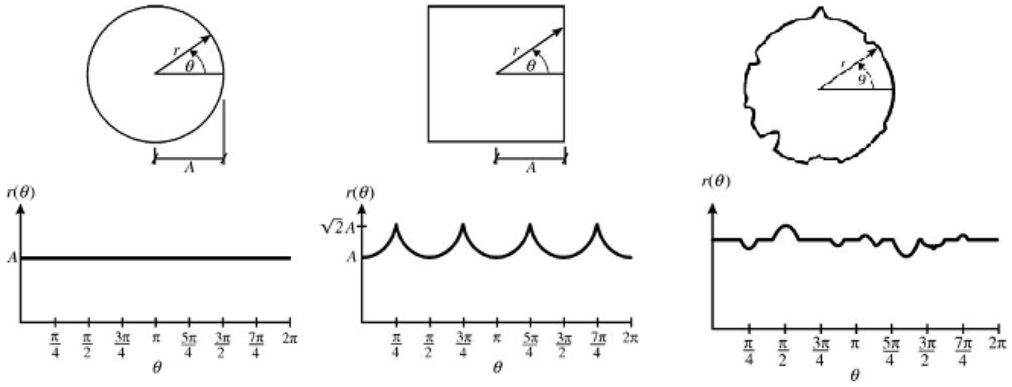
Figur 25 - eksempel på krydskorrelation mellem '4' og '4'

Signatur

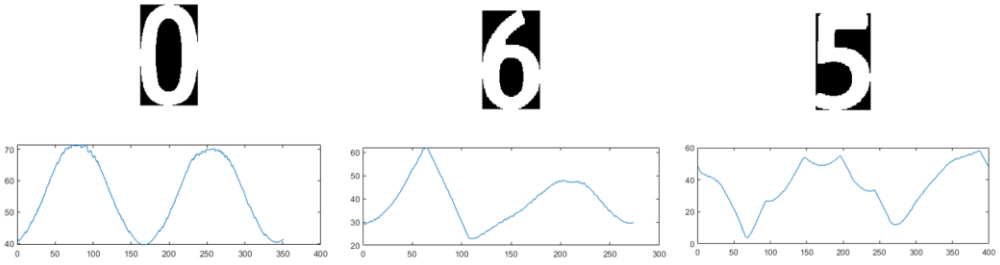
I forbindelse med billedbehandling er en signatur en metode til at lave en éndimensionel fremstilling af et todimensionelt objekt på et billede. Signaturen består af en graf der viser afstanden til objektet ydre kant målt fra objekets massemidtpunkt. Der er anvendt et eksempel til at udlæse signaturer [01]. Eksempler på signaturer fra MARQUES [03] ses i Figur 26, og for udvalgte nummerpladetegntegn ses egne eksempler i Figur 27.

Når der er udlæst en signatur for et objekt kan man krydskorrelere denne med signaturer fra templates. Denne metode er ikke afhængig af skalering, og det er også muligt at man kan genkende tegn der hælder til den ene eller den anden side. Dette vil blot give et offset i grader, men der kan stadig være en høj grad af sammenfald.

Teorien om signaturer er beskrevet i MARQUES afsnit 18.4.2



Figur 26 - eksempler på signaturer fra MARQUES



Figur 27 - eksempler på signaturer

Udvalgte parametre

Den sidste metode er at kigge på udvalgte parametre som massefordeling, areal og omkreds. Ud fra en template findes der for hvert nummerpladetegn et sæt parametre. Når et objekt skal tolkes, sammenlignes dets parametre med parametrene for hvert nummerpladetegn. Objektet tolkes som nummerpladetegnet der giver den mindste fejl.

Af interesse for at undersøge potentialet af denne metode nærmere, er den valgt til det videre arbejde.

Introduktion til parametre

For at kunne klassificere et objekt som et nummerpladetegn, er der fundet forskellige parametre taget ud fra templates af nummerpladetegn. Under klassificeringen af et objekt bliver objektets tilsvarende parametre sammenlignet med hver template. Ud fra hvilken sammenligning af template der har den laveste fejl, vælges det nummerpladetegn objektet tolkes som.

Der er i klassifikationen lagt vægt på at udvikle metoder fra bunden. I matlab kan funktionen regionprops() anvendes til at finde frem til mange forskellige karakteristika for et billede. For at få en dybere forståelse for processerne, er denne funktion med vilje ikke anvendt.

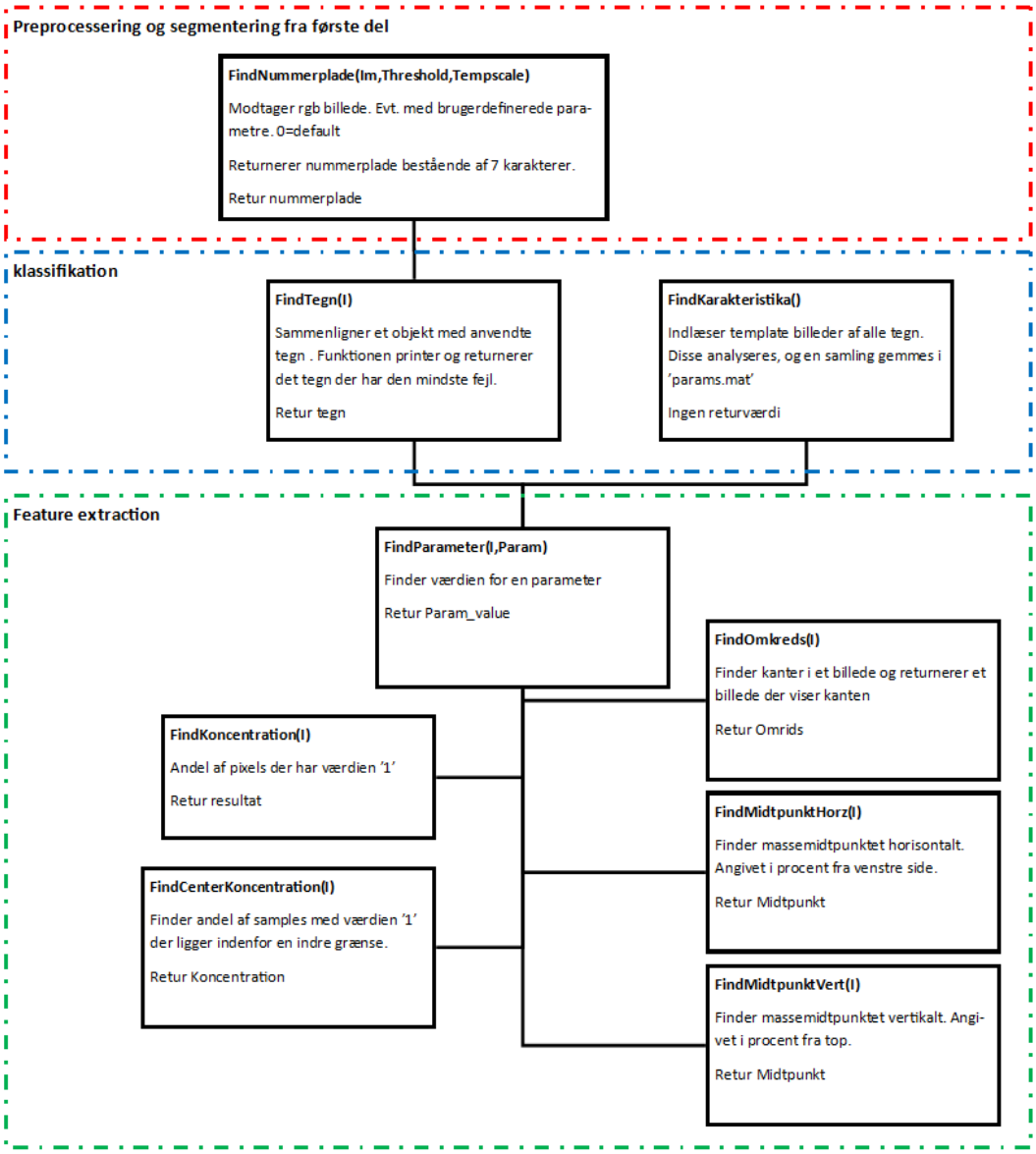
Desuden er der taget tage udgangspunkt i at parametre ikke skal være afhængige af skalering. Derfor er de fleste parametre forhold mellem én del og en anden. Et eksempel kunne være at der ikke kigges enkeltvist på højden eller bredden, men forholdet mellem de to, da dette vil være det samme uanset opløsning på billedet og afstand til nummerplade. De anvendte templates er af denne grund heller ikke tilpasset i størrelse, da dette ikke har nogen betydning.

Parametrene er fundet gennem en kreativ proces, som har bygget på egne idéer til karakteristikker der kunne have betydning. Der er derfor ikke kilder på de hvorfor netop disse parametre er valgt. Men resultatet og valgene vurderes efterfølgende.

Implementering

Figur 28 viser et blokdiagram der kort beskriver hvordan de forskellige funktioner i systemet er implementeret. Her ses det at FindKarakteristika() kan bruges til at finde parametre for alle templates. På den måde får vi trænet en model som vi senere kan bruge til sammenligning af andre objekter ved at bruge FindTegn().

I de efterfølgende afsnit vil hver funktion blive gennemgået.



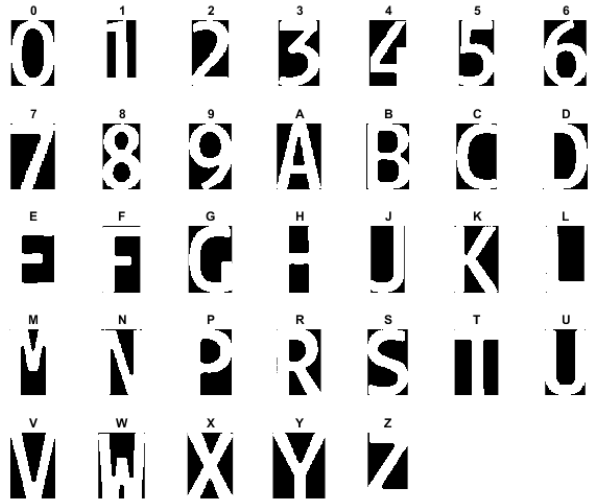
Figur 28 - blokdiagram over funktionerne til at udlæse og sammenligne parametre

FindKarakteristika()

Denne funktion bruges til at 'træne' vores system. Den resulterer i at vi får gemt en matrix med parametre for alle de tegn vi ønsker at analysere. Binære billeder af nummerpladetegn bruges som templates. Disse analyseres, og resultatet af dette der gemmes i params.mat.

Hvert tegn gemmes som ACII værdi i første kolonne. De følgende indeholder hver sin parameters værdi for dette tegn. De fleste af tegnene er fundet ved at tage billeder af tilfældige nummerplader på parkeringspladsen. Alle tegn kan ses i Figur 29.

Implementeringen kan ses i ' FindKarakteristika.m' på github [07].



Figur 29 - alle tegn I binær form

Find Parameter()

Der er udvalgt 6 parametre som hvert nummerpladetegn vurderes ud fra. Argumentet 'param' bestemmer hvilket parameter der udregnes og returneres.

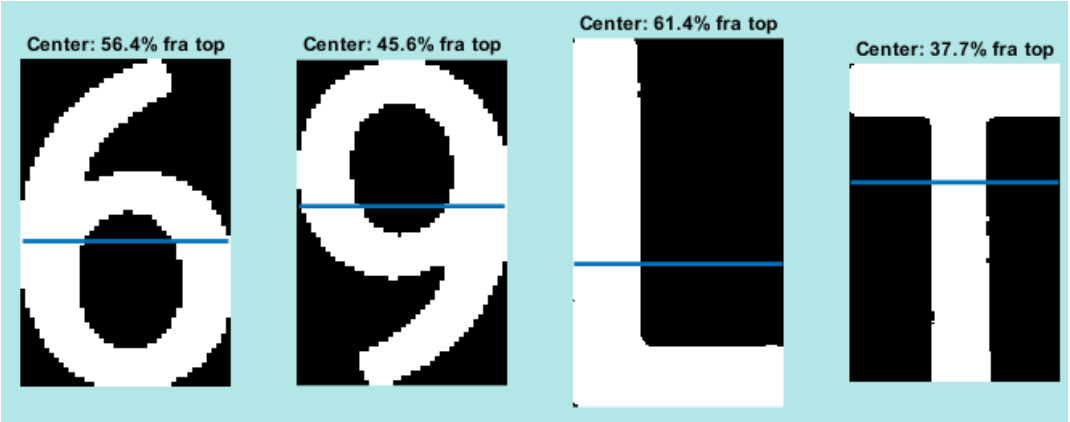
indeks	Param2	Param3	Param 4	Param5	Param6	Param7
Nummerpladetegn i ASCII	Massemidtpunkt vertikalt	Højde/omkreds	Euler forhold	Mætning i billede	Massemidtpunkt horisontalt	Koncentration omkring midte

Den primære funktion af FindParameter() er at forbinde til den efterspurgte parameter.

Implementeringen kan ses i ' FindParameter.m' på github [07].

FindMidtpunktVert()

Det vertikale massemidtpunkt findes ved at tage summen af indholdet fra hver række. Der findes en middelværdi for hvilken række der repræsenterer midtpunktet. Denne række er demonstreret med den blå linje i Figur 30, hvor det også ses at dette er en god parameter til at kende forskel på '6' og '9', som ellers er ens i de fleste andre parametre.



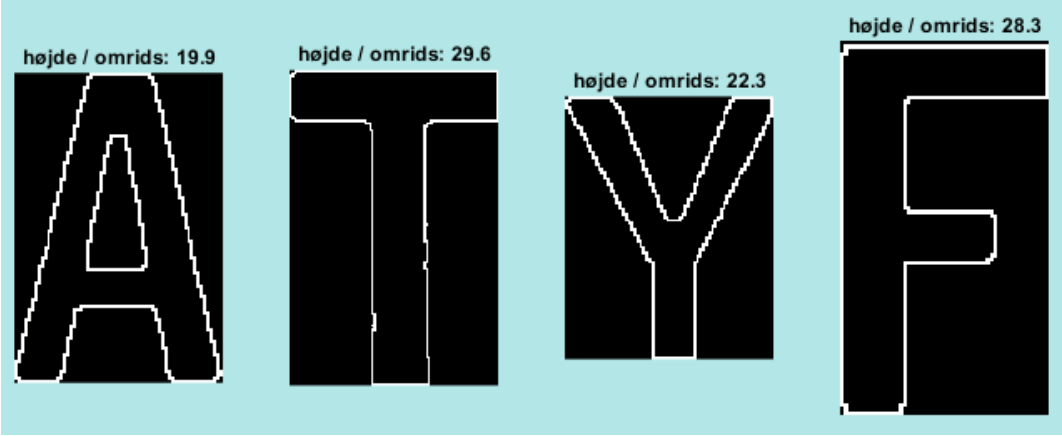
Figur 30 - eksempel på parameter2 - massemidtpunkt vertikalt

Koden er bygget op om en dobbelt løkke der finder summen af hver række. Der findes en middelværdi og denne skaleres efter højden.

Implementeringen kan ses i ' FindMidtpunktVert.m' på github [07].

FindOmkreds()

Som noget særligt returnerer FindOmkreds() faktisk et nyt image til FindParameter(). Dette gøres for at det bl.a. er muligt at analysere dette omrids som en del af både fejlfinding og dokumentation. Den sidste del af behandlingen foregår med en enkelt linje i FindParameter() hvor summen af indholdet fra det returnerede billede findes.



Figur 31 - eksempel på parameter3 - relation mellem højde og omrids

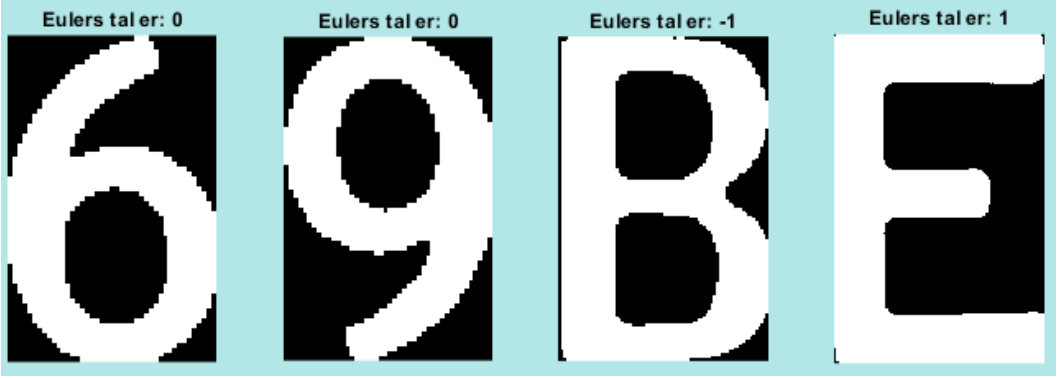
Omridset findes ved at lave pixels der er '1', men som grænser op til et '0' forblive '1'. Alle andre pixels bliver '0'. Dette resulterer i et billede som de der vises i Figur 31.

Implementeringen kan ses i ' FindOmkreds.m' på github [07].

Bweuler()

Alle andre udregninger foregår i funktioner som er konstrueret særligt til denne opgave. Der er dog én enkelt undtagelse i parameter 4, der findes ved funktionen bweuler(). Denne funktion modtager et sort/hvidt billede, og returnerer antallet af objekter fratrukket antallet af huller. Eksempler på euler forholdet ses i Figur 32. '6' og '9' giver euler forholdet 0, da der er lige så mange huller som der er objekter. 'B' har to huller og et objekt euler relationen bliver altså 1-2 = -1. 'E' har et objekt og ingen huller og eulerforholdet bliver 1.

Dette er en stærk parameter da den kategoriserer nummerpladetegnene i tre grupper. Euler forholdet er enten eller, og der er altså ikke tvivl om hvilken kategori et objekt tilhører.

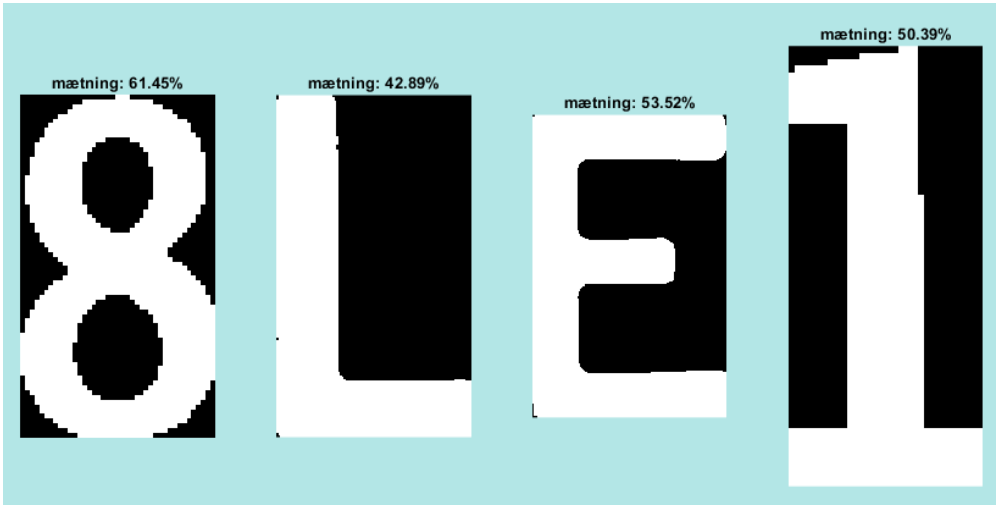


Figur 32 - eksempel på parameter4 - euler forhold

FindKoncentration()

Med koncentrationen eller mætningen menes forholdet mellem pixels der indeholder værdien '1' og dem der indeholder værdien '0'. Dette er en meget simpel men også effektiv funktion.

I Figur 33 ses eksempler på hvordan mætningen er udregnet for fire tegn.



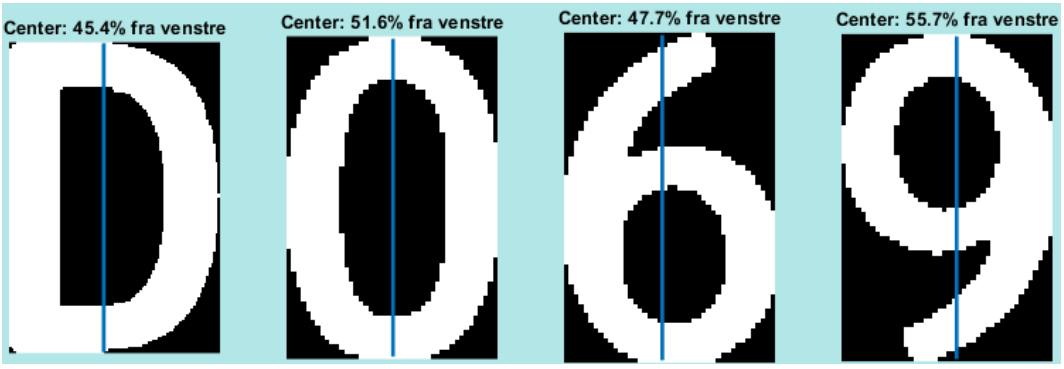
Figur 33 - eksempel på parameter5 - koncentration i billedet

Implementeringen kan ses i ' FindKoncentration.m' på github [07].

FindMidtpunktHorz()

På samme måde som det vertikale massemidtpunkt findes, findes et tilsvarende horisontalt massemidtpunkt.

Dette kan som det ses i Figur 34 være effektivt til at skelne mellem 'D' og '0' samt '6' og '9'. Disse tegn ligger tæt for de fleste andre parametre.



Figur 34 - eksempel på parameter4 - massemidtpunkt horisontalt

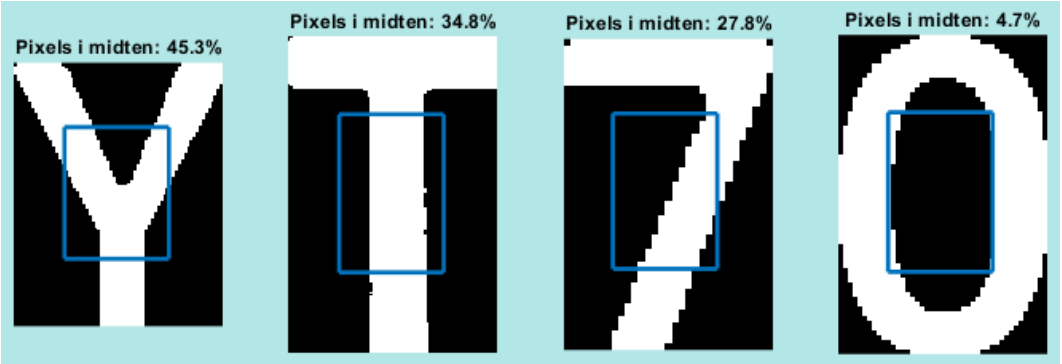
Metoden er den samme som 'FindMidtpunktVert()' blot på kolonner i stedet for rækker.

Implementeringen kan ses i ' FindMidtpunktHorz.m' på github [07].

FindCenterKoncentration()

Center koncentration returnerer forholdet mellem samples med værdien'1' inden for en ramme, og antallet af samples udenfor rammen, som har værdien '1'.

Rammen er defineret ved at have en afstand på 25% af bredden fra hver side, og en afstand på 25% af højden fra top og bund. Dette er er vist med de blå rammer i Figur 35.



Figur 35 - koncentration i midten af billedet

Koncentration i midten billedet har vist sig at være effektiv til nogle af de tegn som de andre det har været svært at skelne på baggrund af de andre parametre. 'Y', '7' og 'T' blev ofte fejltolket før denne 6. parameter blev tilføjet. I sammenligning baseret på de andre parametre lå disse tre tegn meget tæt, men som det ses af eksemplet i Figur 35 er der for denne parameter stor forskel.

Implementeringen kan ses i ' FindCenterKoncentration.m' på github [07].

FindTegn()

I denne funktion findes parametre for det aktuelle objekt, så dette kan sammenlignes med hvert nummerpladetegn.

Inden sammenligningen skal alle parametre skaleres og vægtes. Dette gøres for at have muligheden for af give nogle parametre større betydning end andre. For at effektivisere koden kunne dette være gjort på forhånd, og et par ekstra celler for hver parameter kunne definere hvordan de forskellige parametre for det nye objekt skulle skaleres og vægtes.

Efter skalering og vægtning, holdes de nye objekt op imod hvert tegn. For hvert tegn sammenlignes alle parametre. Ifølge MARQUES kapitel 19.2 [03] er der tre metoder til at måle afstanden mellem vektorer i flere plan.

Den første er Manhattan afstand. Denne udregnes ved at tage summen af den absolutte difference for hver parameter. Man kan sige at den finder den samlede afstand hvis man skal forbinde alle knudepunkter. Som hvis man skal fra et sted i til et andet i Manhattan, hvor man må gå i lige linjer mellem blokkene. Ved afprøvning har denne metode fejl i sine udlæsninger.

$$d_{\mathrm{M}} = \sum_{i=1}^{n} |a_i - b_i|$$

Figur 36 - formel for Manhattan afstand

I matlab implementeres manhattan således:

```
%% Manhattan
errors = zeros(Nsigns,Nparams+1);
for i = 1:Nsigns
    %error = 0;
    for p = 1:Nparams
        errors(i,p) = abs(I_params(1,p) - params(i,p+1));
    end
    errors (i,end) = sum(errors(i,1:end-1),'all');
end
```

Den næste metode er euclidean afstand som i et plan kan sammenlignes med den klassiske pythagoras. Denne metode finder korteste afstand mellem to punkter. Ved afprøvning viser denne metode gode resultater, men den er grundet brug af kvadratrod mere procestung.

$$d_{\mathrm{E}} = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2}$$

Figur 37 - formel for euclidean afstand

I matlab implementeres euclidean således:

```
%% euclidean error
errors = zeros(Nsigns,Nparams+1);
for i = 1:Nsigns
    for p = 1:Nparams
        errors(i,p) = abs(I_params(1,p) - params(i,p+1));
    end
    errors (i,end) = sqrt(sum((errors(i,1:end-1)).^2 ,'all'));
end
```

Den sidste metode er Minkowski afstand. Denne metode kan implementeres i forskellige grader defineret ved 'r'. For r=1 svarer denne metode til Manhattan, og for r=2 til Euclidean. Der er ikke fundet nogle værdier for 'r' der giver bedre resultater eller mere effektiv kode end de andre metoder.

$$d_{\mathrm{M}} = \left[\sum_{i=1}^{n} |a_i - b_i|^r \right]^{1/r}$$

Figur 38 - formel for Minkowski afstand

I matlab implementeres minkowski således:

```
%% Minkowski
r = 4;
errors = zeros(Nsigns,Nparams+1);
for i = 1:Nsigns
    for p = 1:Nparams
        errors(i,p) = abs(I_params(1,p) - params(i,p+1));
    end
    errors (i,end) = nthroot( sum( (errors(i,1:end-1)).^r,'all') ,r);
end
```

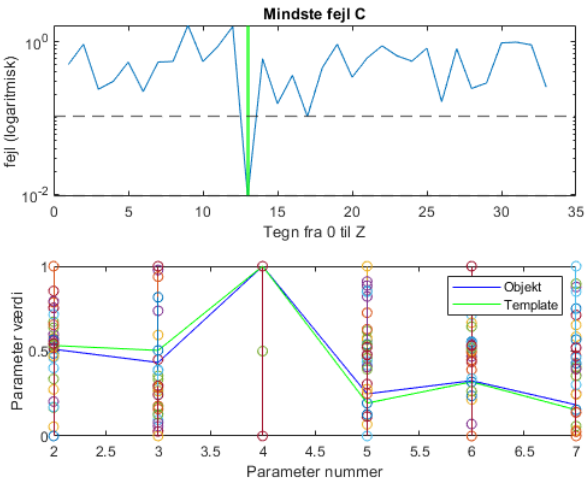
Det vurderes at euclidean afstanden vil give det bedste resultat til videre implementering, og denne anvendes til at finde fejlen for hver parameter.

Fejlene gemmes i en matrix, og til sidst findes placeringen for den mindste fejl i matrixen. Denne placering svarer til placeringen i første kolonne af 'params', og tegnet på denne placering returneres.

Resultatet kan illustreres ved graferne i Figur 39. Af den øverste graf er det let at se at fejlen for sammenligningen med C (illustreret med den grønne linje) er meget lavere end de andre fejl. Grafen er plottet med logaritmisk y akse da nogle

af fejlene bliver meget høje, og ellers ville gøre det svært at sammenligne de lave værdier.

På den nederste graf i Figur 39 ses sammenligning af parametrene for det indlæste objekt og karakteren 'C'. Cirklerne indikerer hvordan de andre tegn er placeret. Sammenligningen viser at der for nogle parametre er andre nummerpladetegn der minder mere om det indlæste objekt, men for den samlede fejl er der dog ikke noget at være i tvivl om.



Figur 39 - graf1: sammenligning med alle tegn – graf2: præcision for hver parameter

Implementeringen kan ses i ' FindTegn.m' på github [07].

Verifikation (LGR)

For at teste hvor effektivt systemet er laves et script hvor seks billeder testes. For de fire første billeder findes threshold og skalering automatisk, hvor det for de sidste to billeder har været nødvendigt at sætte de to argumenter manuelt. skaleringen skyldes at størrelsen på nummerpladen målt I pixels, har været for langt fra templaten til at lokalisere nummerpladen. At threshold skal sættes manuelt kan i det ene tilfælde skyldes at der solen giver genskin i selve nummerpladen, og i det andet tilfælde er bilen helt hvid hvilket forstyrrer det automatiske threshold, og dermed fremstår nummerpladen mørk i forhold til resten af billedet.

I scriptet defineres titlen på 6 billeder af biler, indeholdende en nummerplade.

Dernæst sættes et array med treshold værdier, og et med værdier til scale. I begge tilfælde betyder værdien '0' default.

I en løkke indlæses og analyseres hvert billede, og alle billeder plottes samlet sammen med en tolkning af nummerpladen

Kode til test findes i ' Test_Af_System.m' på github [07], og det er muligt at hente filerne ned, og teste systemet lokalt.

Resultat

De seks billeder er testet, og resultatet er meget overbevisende. Ud af de 42 nummerpladetegn er alle tolket korrekt. Resultatet ses i Figur 40.

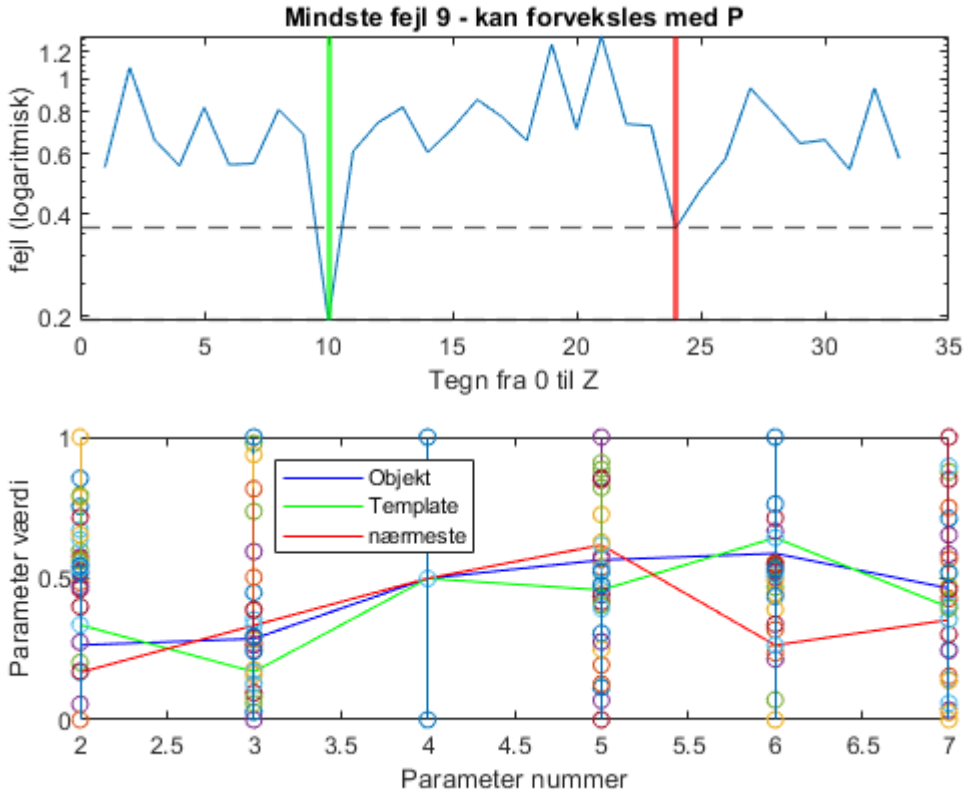
Figur 40 - resultat af test

Vurdering af parametre

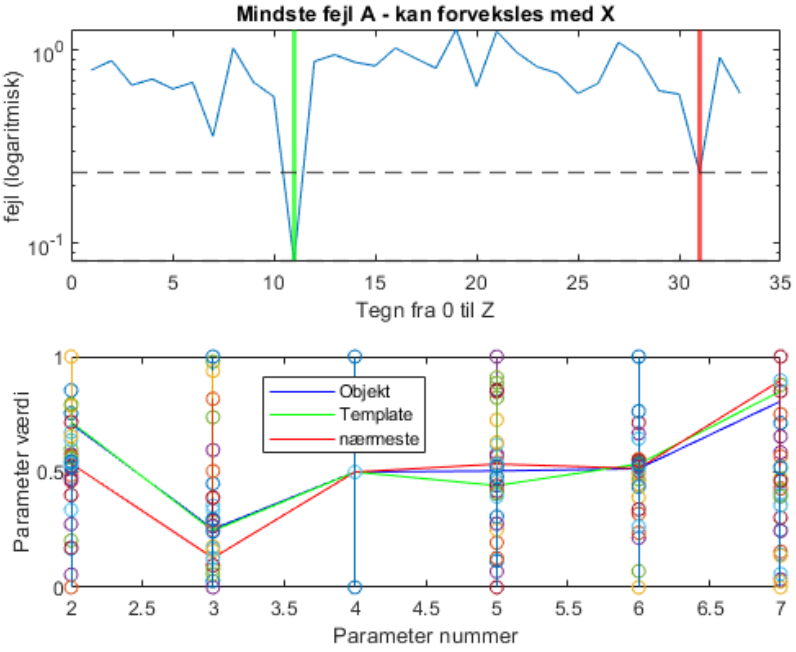
For at kigge nærmere på sikkerheden i udlæsningerne, er der lavet et plot af fejlen for hvert udlæst tegn. Disse er vist i Figur 41, Figur 42 og Figur 43. Graferne viser med grøn det tegn der er tolket, og med rød det tegn der minder næst mest om. Cirklerne viser spredningen på værdierne for de enkelte parametre.

Her ses det at der er forskel på hvilke parametre der er vigtige for tolkningen af de enkelte tegn. For '9' og 'P' kan de to første parametre være med til at give en fejllæsning, men parameter 6 (horisontalt massemidtpunkt) er meget stærk, og sikrer den korrekte tolkning. Modsat har parameter 6 ingen betydning mellem 'A' og 'X', men det har til gengæld de to første parametre (vertikalt massemidtpunkt og højde i forhold til omkreds). For '5' og 'S' er det parameter 3 og 5 der er vigtigst.

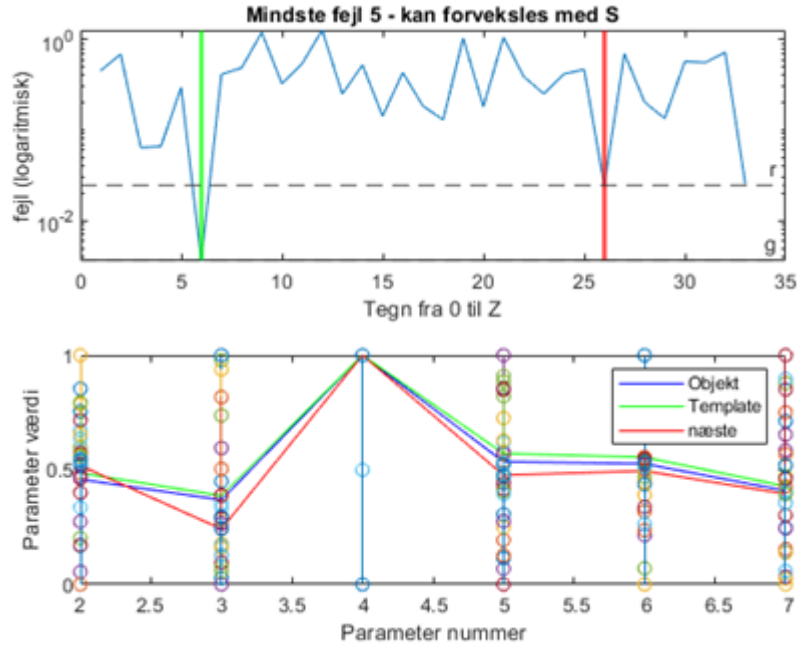
På den måde har alle parametre en vigtig rolle, og der er ikke fundet noget som ville give mening at skære fra.



Figur 41 - udlæsning af '9' der kan forveksles med 'P'



Figur 42 - udlæsning af 'A' der kan forveksles med 'X'



Figur 43 - udlæsning af '5' der kan forveksles med 'S'

Konklusion

Vi har gennem denne opgave er der arbejdet med forskellige metoder til at analysere, detektere og klassificere objekter.

Billederne af bilerne er ikke taget i et kontrolleret miljø, hvilket giver nogle problemer i forhold til lysniveau i billedet samt rotation og størrelse af nummerplade.

Lydniveauet er meget forskelligt, da nogle biler er taget udenfor og andre i en garage. Målet var at få en nummerplade der er synlig, og resten er fjernet, for at sikre korrelationen bliver høj ved nummerpladen. Det var dog ikke muligt at finde en formel, der virker på alle biler, der sikre at nummerpladen er synlig.

Derudover bruger den normaliserede krydskorrelation en template, som helst skal passe med nummerpladen på billedet. Her kunne der bruges en anden metode såsom Connected Component Analysis, CCA, sammen med Ratio Analysis. CCA finder komponenter der hænger sammen, og derefter kan man eliminere det fleste elementer ved at kigge efter samme størrelsesforhold som nummerpladen.

Resultatet af nummerpladelokaliseringen er tilfredsstillende, da det er muligt at automatisk finde og udskære nogle nummerplader, og alle andre biler fungerer også med en specifik threshold værdi og template størrelse.

Desuden har vi oplevet hvor effektivt det kan være at sammenligne objekter på simple håndgribelige parametre, som vi selv har defineret.

Dette giver en anderledes tilgang i sammenligning med machinelearning. Algoritmer i machinelearning med f.eks. neurale netværk får ofte en kompleksitet der gør, at man som designer ikke længere har mulighed for at forstå hvilke parametre der vægtes, i hvilken grad.

I testen hvor 42 nummerpladetegn sammenlignes med 33 templates på seks parametre, har det designede system succes med at ramme alle tegn. Vi er derfor meget tilfredse med løsningen.

Referenceliste for samlet dokumentation

Id	Reference
[01]	https://se.mathworks.com/matlabcentral/answers/115659-how-to-find-signature-of-the-object-for-the-given-binary-image
[02]	http://electronics.aau.dk/wp-content/uploads/2015/03/15gr416_Billedbehandling.pdf
[03]	PracticalImage and Video Processing Using MATLAB, OGE MARQUES
[04]	https://github.com/spejderlasse/E5ADSB
[05]	https://se.mathworks.com/help/images/ref/bwlabel.html
[06]	https://denstoredanske.lex.dk/nummerplader
[07]	https://github.com/spejderlasse/E5ADSB/blob/main/scripts/FindNummerplade.m
[08]	matchingBy2DCorrelation_Gonzalez4ThEd.pdf fra Gonzalez & Woods, Digital Image Processing 4th ed.
[09]	http://www.nrpl.dk/comb-1966.php

Bilag

Matlab scripts, funktioner og testbilleder kan findes på:

https://github.com/spejderlasse/E5ADSB