Universitetet i Stavanger

DET TEKNISK-NATURVITENSKAPELIGE FAKULTET

MASTEROPPGAVE

Studieprogram/spesialisering: Kybernetikk	Vårsemesteret, 2011 Åpen / ~~Konfidensiell~~
Forfatter: Andreas Kristiansen	…… (signatur forfatter)
Fagansvarlig: Ivar Austvoll Veileder(e): Ivar Austvoll	
Tittel på masteroppgaven: Kamerakalibrering med 1-dimensjonalt kalibreringsobjekt Engelsk tittel: Camera calibration with 1-dimensional calibration object	
Studiepoeng: 30	
Emneord: Kamerakalibrering, MATLAB, kalibreringsobjekt, maskinsyn, bildebehandling.	Sidetall: 45 + vedlegg/annet: 18+cd Stavanger, 15. Juni 2011 dato/år

Universitetet
i Stavanger

Teknisk-naturvitenskapelig fakultet

Institutt for data- og elektroteknikk

Masteroppgave

Kamerakalibrering med 1-dimensjonalt kalibreringsobjekt

Andreas Kristiansen

15. Juni 2011

Forord

Denne oppgaven er skrevet som avslutning på et mastergradsstudium i kybernetikk ved Universitetet i Stavanger. Oppgaven er på 30 studiepoeng, tilsvarende et halvt års arbeid, og er gitt av universitetet.

Jeg ønsker å takke fagansvarlig og veileder Ivar Austvoll for et godt og motiverende samarbeid gjennom semesteret.

Andreas Kristiansen
Stavanger, 15. Juni 2011

Sammendrag

Kamerakalibrering er en viktig forutsetning for å kunne rekonstruere en scene fra et bilde. Dette er nyttig om en ønsker å bruke kamera som sensor i automatiske systemer.

Denne oppgaven fokuserer på en metode for kamerakalibrering med 1-dimensjonalt kalibreringsobjekt, presentert av Zhengyou Zhang i 2004. En ønsker å undersøke om denne metoden kan tilby fordeler som gjør at den er å foretrekke framfor andre metoder for kamerakalibrering som i hovedsak baserer seg på 3- og 2-dimensjonale kalibreringsobjekter. Dette er gjort eksperimentelt ved å implementere Zhangs algoritme i Matlab og sammenligne resultater for kalibrering med 1-dimensjonalt kalibreringsobjekt med kalibrering av samme kamera under like forhold, med den samme Zhangs mye brukte metode for 2-dimensjonalt kalibreringsobjekt.

Det er også gjort forsøk på å forbedre resultatene fra Zhangs metode henholdsvis ved å normalisere bildedataene før kalibrering, og med normalisering underveis i kalibreringen. En har fått resultater som bekrefter at normalisering før kalibrering forbedrer kalibreringsresultatene, mens resultatene for normalisering underveis ikke gir like god forbedring.

Underveis har det blitt avdekket svakheter ved algoritmen som ikke tidligere har vært problematisert, spredningen i kalibreringsresultatene viser seg å være større enn antatt. Det konkluderes derfor med at kamerakalibrering med 1-dimensjonalt kalibreringsobjekt ikke tilbyr fordeler som gjør det til et reelt alternativ til de metodene som allerede benyttes. Unntaket er for spesialtilfellet der en ønsker å kalibrere et nettverk av kameraer og det stilles krav til at alle må se kalibreringsobjektet samtidig.

Innhold

Figurer

1 Introduksjon

Et kamera er et system bestående av et bildeplan og en linse som overfører punkter fra et 3-dimensjonalt scenerom til et 2-dimensjonalt bildeplan. I denne prosessen går informasjon om scenen tapt. For å kunne dra ut informasjon om scenen fra bildene som er tatt av den, er det nødvendig å fastslå så nøyaktig som mulig hva som skjer i transformasjonen mellom scene og bilde. Å kunne beskrive denne transformasjonen nøyaktig er en viktig forutsetning for å kunne bruke kameraer til å løse oppgaver innen maskinsyn. På grunn av distorsjon kan ikke transformasjonen beskrives eksakt. Transformasjonen mellom scene og bilde representeres ved hjelp av en kameramodell. Kamerakalibrering vil si å beregne verdien til parametrene i denne modellen. En skiller mellom intrinsikke og ekstrinsikke parametre i kameramodellen, hvor de intrinsikke beskriver kameraets egenskaper som fokallengde og senterpunkt, mens de ekstrinsikke beskriver kameraets plassering og orientering (rotasjon) i forhold til scenen.

1.1 Tidligere arbeid

Forskningen på området bærer preg av å være drevet fra to forskjellige fagfelt, datasyn og fotogrammetri, med sine forskjellige tilnærminger og målsetninger. Fotogrammetri er læren om å bestemme objekters geometriske egenskaper fra bilder. Fagfeltet er nesten like gammelt som fotografiet. Den mest elementære formen for fotogrammetri er å omregne størrelser i bildet til virkelige størrelser ved hjelp av en skaleringsfaktor. Etter hvert som kameraene og optikken ble bedre, samt at flyet ble oppfunnet, så en mulighet for å bruke flyfoto til å lage kart[1]. Utviklingen er beskrevet av Remondino og Fraser i [9] og av Clarke og Fryer i [10]. I begynnelsen ble det ikke tatt hensyn til distorsjon, men etter første verdenskrig ble det klart at en trengte metoder for å kalibrere linsesystemene for å oppnå større nøyaktighet i stereoskopiske fotogrammetriske målinger. Militær kartlegging og overvåkning var den største drivkraften bak forskningen. Mye av arbeidet var unntatt offentligheten, noe som begrenset informasjonsflyt og samarbeid på tvers av landegrenser. På slutten av 1940-tallet ble det jobbet for å utvikle internasjonale standarder for kamerakalibreringsteknikker. Det eksisterte mange teknikker, blant annet å ta bilde av stjernehimmelen og bruke stjerneavstander til å estimere linsens distorsjon. Det kom et gjennombrudd i forskningen da Brown fra 1965 og utover presenterte arbeidet sitt fra etterkrigstiden, som fram til da hadde vært unntatt offentligheten [4] [5]. Han innførte en metode kalt "bundle adjustment" for samtidig korreksjon av distorsjon forårsaket av imperfekte linser

[1]Kartlegging vha. flyfoto gjøres ved å ta overlappende bilder av landskapet som i etterkant settes sammen, en får da en 3-dimensjonal effekt som brukes til å beregne høydekurver. Dette ligner mye på metoden som i dag brukes for å lage 3-dimensjonal video, hvor en setter sammen bilder fra to overlappende kameraer og på den måten får dybdefølelse i det sammensatte bildet.

og beregning av kameraplassering, som blant annet bygde på Magills arbeid med variasjon i distorsjon ved ulik fokus [18].

Innen datasyn og maskinsyn[2], som er relativt nye fagfelt, har en sett mange tilnærminger basert på å bruke kalibreringsobjekter med kjente kontrollpunkter. Både 3-dimensjonale og 2-dimensjonale objekter har blitt brukt. Den enkleste og mest brukte metoden er Tsais metode [24] som bestemmer parametrene i en modell som beskriver kameraets interne og eksterne orientering, radiell distorsjon og skalering ved hjelp av n kjente kontrollpunkter i hvert bilde ($n > 8$). Metoden antar at noen av kameraparametrene er oppgitt av kameraprodusenten. Det kan benyttes både 2-dimensjonale og 3-dimensjonale kalibreringsobjekter, men det kreves at en kjenner kalibreringsobjektets nøyaktige plassering i scenen. Metoden til Tsai har inspirert mange og utviklingen har gått mot mer nøyaktig og/eller fleksibel kalibrering. Andre viktige kalibreringsmetoder for bruk med 2-dimensjonalt kalibreringsobjekt er presentert av Heikkila og Silven [14] og Zhang [27]. Heikkila og Silvens teknikk fungerer både med 2-dimensjonale og 3-dimensjonale kalibreringobjekter, mens Zhangs metode benytter hjørnepunkter i et rutemønster plassert i forskjellige orienteringer for å finne kameraets parametre. Begge metodene inneholder ledd for å beregne linsens distorsjon. Zhang presenterte i 2004 et arbeid som introduserte kamerakalibrering med 1-dimensjonalt kalibreringsobjekt [28], senere har også andre presentert lignende metoder, blant annet Zhao og Liu i [29]. Zhang viser at en med 1-dimensjonalt kalibreringsobjekt kan oppnå gode resultater, dette bekreftes av Franca et. al. i [8]. Det har også blitt vist metoder som eliminerer bruk av kalibreringsobjekt. Et eksempel er metoden til Isard et. al. [16], der det vises at kamerakalibrering kan utføres ved å bruke bilder av menneskeskapte miljøer, der rette linjer brukes til å beregne forsvinningspunkter. Metodene som ikke benytter kalibreringsobjekt krever at bildene de brukes på har spesielle egenskaper, noe som gjør dem mindre fleksible. Det er ikke funnet dokumentasjon på at nøyaktigheten som oppnås er konkurransedyktig med resultatene fra metoder basert på 2- og 3-dimensjonale kalibreringsobjekt.

1.2 Mål

Med innføring av 2-dimensjonalt kalibreringsobjekt for bruk i kamerakalibrering viser det seg at en ved å benytte et enklere kalibreringsobjekt kan få betydelig økt fleksibilitet uten å gå for mye på akkord med nøyaktigheten i forhold til metoder basert på 3-dimensjonale kalibreringsobjekter. Formålet med oppgaven er å undersøke om en på samme måte kan oppnå fordeler ved å benytte 1-dimensjonalt istedenfor 2-dimensjonalt kalibreringobjekt. Det vil i starten av kapittel 2 bli

[2]Det skilles mellom datasyn og maskinsyn ved at maskinsyn er applikasjonsorientert, mens datasyn fokuserer mer på metode. Fagfeltene overlapper hverandre likevel endel, og betegnelsene brukes ofte om hverandre.

redegjort for teorien bak, og matematikken som kreves for å forstå kalibreringsmetodene som presenteres fra midten av kapittelet. I kapittel 3 er nødvendige eksperimenter og resultater beskrevet, mens disse blir diskutert i kapittel 4. Konklusjonen og svar på problemstillingen er presentert i kapittel 5.

2 Teori

Dette kapittelet gjennomgår den teoretiske bakgrunnen som behøves for å følge gjennomføringen av eksperimentene i kapittel 3.

2.1 Geometri og matematiske forutsetninger for kamerakalibrering

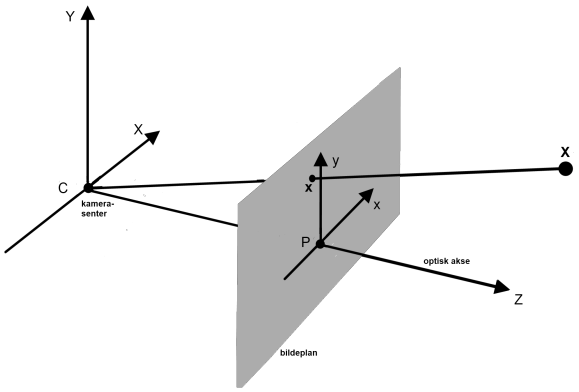
Dette kapittelet tar for seg matematikken som ligger til grunn for kalibreringsmetodene som presenteres i oppgaven.

2.1.1 Projektiv geometri og homogene koordinater

For å kunne beskrive dybdeinformasjon og perspektiv i bilder matematisk tar en i bruk projektiv geometri. I projektiv geometri benyttes homogene koordinater som fungerer omtrent som kartesiske koordinater gjør for euklidisk geometri, men med mulighet for at parallelle linjer og plan kan ha skjæringspunkter som ligger uendelig langt unna. Et euklidisk punkt i et plan $\mathbb{R}^2$ kan som kjent beskrives med koordinatene (x, y). Hvis en ser på $\mathbb{R}^2$ som et vektorrom blir koordinatparet (x, y) en vektor, heretter brukes kolonnevektorer for å beskrive geometriske enheter. Et punkt i det euklidiske planet, (x, y), kan representeres med de homogene koordinatene $(w \cdot x, w \cdot y, w)^T$, hvor $w \neq 0$. Dersom $w = 0$ ligger punktet uendelig langt unna. Dersom man multipliserer de homogene koordinatene med en felles faktor $k \neq 0$ vil de representere det samme kartesiske punktet. Et enkelt punkt kan derfor representeres av uendelig mange homogene koordinater. Generelt kreves derfor en koordinat mer enn dimensjonen til det projektive rommet som beskrives. Et projektivt plan, $\mathbb{P}^2$, beskrives med 3 koordinater, mens et projektivt 3-dimensjonalt rom, $\mathbb{P}^3$, beskrives med 4 koordinater.

2.2 Kameramodellering

Det finnes flere metoder for å modellere kameraer. Nøyaktigheten til modellene avgjøres ofte av hvilke forenklinger som blir gjort i modelleringen. Forenklinger gir modeller som er lettere å bruke, men mindre nøyaktige. Valg av modell må tilpasses bruksområdet. De fleste kameraer følger sentralprojeksjonsprinsippet, og



Figur 1: Nålehullskamerageometri. C er kamerasenter og P er bildets hovedpunkt.

de mest brukte modeller bruker også dette som utgangspunkt. Sentralprojeksjon vil si at det går en tenkt rett linje fra et punkt i scenen, via kameraobjektivets senter, og videre til avbildingsflaten. Den enkleste og mest brukte modellen er nålehullsmodellen. Parametrene i, og strukturen på kameramatrisen bestemmes av modellen.

2.2.1 Kameramodell basert på nålehullskameramodellen

Modellen beskriver transformasjonen mellom verdenskoordinater og bildekoordinater for et ideelt nålehullskamera. Blenderåpningen beskrives som et punkt (uendelig lite) og det brukes ikke objektiv/linser for å fokusere lys. Modellen tar heller ikke hensyn til distorsjon, uskarphet eller det faktum at digitale kameraer har diskrete bildepunkter. Modellnøyaktigheten vil være avhengig av kameraet, men normalt sett størst nær bildesentrum og gradvis avta mot bildets kanter fordi objektivet tilfører en viss distorsjon. Noen av svakhetene ved modellen kan kompenseres for og andre kan være små nok til å neglisjeres, slik at nålehullsmodellen likevel kan være brukbar i mange tilfeller. Nålehullskameramodellen gir en kameramatrise bestående av 5 intrinsikke parametre.

Et 3-dimensjonalt punkt $X_w = (X, Y, Z)^T$ blir overført til et 2-dimensjonalt punkt, x, i bildeplanet der en tenkt linje mellom X_w og kamerasenter C krysser bildeplanet. Avstanden mellom bildeplanets hovedpunkt (fokalpunkt) og kamerasenter kalles fokallengde og har betegnelsen *f*. En grafisk framstilling av modellen er gitt i figur 1. Figuren og notasjonen er i hovedsak basert på [13]. Overgangen blir da fra verdenskoordinater til bildekoordinater:

$$(X, Y, Z)^T \rightarrow (fX/Z, fY/Z)^T \tag{2.1}$$

Denne modellen kan utvides noe, dersom en benytter homogene koordinater kan overgangen skrives som en matrisemultiplikasjon:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.2}$$

I utrykket i ligning 2.1 tas det utgangspunkt i at bildets origo er i hovedpunktet (P i figur 1). Dette trenger ikke alltid være tilfelle, og en kan derfor generalisere uttrykket og ta med hovedpunktets (P) koordinater $(p_x, p_y)^T$ i bildeplanet:

$$(X, Y, Z)^T \rightarrow (fX/Z + p_x, fY/Z + p_y)^T \tag{2.3}$$

og med homogene koordinater:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.4}$$

Videre kan en skrive

$$K = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} \tag{2.5}$$

Der K kalles kameramatrisen og inneholder kameraets intrinsikke parametre. En får da i en mer kompakt form

$$x = K[I|0]X_w \tag{2.6}$$

I ligning 2.4 antas det at kameraet er plassert i origo av et euklidisk koordinatsystem med kameraets hovedakse langs Z-aksen (Som i figur 1). Modellen kan videre utvides for å ta hensyn til andre kameraplasseringer ved å innføre kamerakoordinater og verdenskoordinater som to ulike euklidiske koordinatsystemer som kan beskrives i forhold til hverandre ved hjelp av rotasjon og translasjon.

Dersom $\tilde{X}$ er en vektor $(X_w, Y_w, Z_w)^T$ som refererer til et punkt i verdenskoordinater, og $\tilde{X}_{cam}$ er en vektor som referer til det samme punktet, men i kamerakoordinater, kan dette skrives

$$\tilde{X}_{cam} = R(\tilde{X} - \tilde{C}) \tag{2.7}$$

der $\tilde{C}$ er koordinatene til kamerasenteret representert i verdenskoordinater og R er en 3×3 matrise som beskriver orienteringen til kameraets koordinatsytem. Dette kan med homogene koordinater skrives:

$$X_{cam} = \begin{bmatrix} R & -R\tilde{C} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} R & -R\tilde{C} \\ 0 & 1 \end{bmatrix} X \tag{2.8}$$

Kompakt har en da

$$x = KR[I| - \tilde{C}]X \tag{2.9}$$

som er den generelle overgangen mellom scenepunkter og bildepunkter ved bruk av et nålehullskamera. Den generelle nålehullskameramodellen, $P = KR[I| - \tilde{C}]$, har 9 frihetsgrader, (uavhengige variable). 3 i matrisen K (fokallengden f, samt hovedpunktets koordinater u_0 og v_0), 3 i R og 3 i $\tilde{\mathbf{C}}$. En kan erstatte hovedpunktets koordinater (p_x og p_y) med en translasjonsvektor $\mathbf{t} = -\mathrm{R}\tilde{\mathbf{C}}$ og få:

$$\mathbf{x} = \mathrm{K}[\mathrm{R}|\mathbf{t}]\mathbf{X} \tag{2.10}$$

Slik får en transformasjonen fra verdenskoordinater til bildekoordinater som $\tilde{\mathbf{X}}_{\mathbf{cam}} = \mathrm{R}\tilde{\mathbf{X}} + \mathbf{t}$. Og kameramatrisen:

$$\mathrm{P} = \mathrm{K}[\mathrm{R}|\mathbf{t}] \tag{2.11}$$

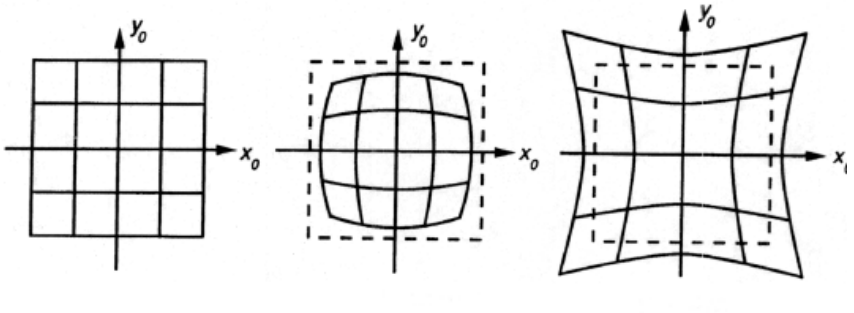
Ved bruk av digitale kameraer (CCD) må en også ta hensyn til at pikslene på bildebrikken ikke nødvendigvis er kvadratiske. En introduserer derfor ulike skaleringsfaktorer i bildebrikkens vertikale og horisontale retning. Slik at antall piksler per distanseenhet i bildekoordinater er m_x og m_y, i henholdsvis x- og y-retning. Det tas også med en skråstillingsparameter, s, som tar hensyn til at bildebrikken kan være plassert skrått i forhold til kameraet. Denne vil for de fleste kameraer være 0, men kan være viktig i situasjoner der det for eksempel tas bilde av et bilde [13]. En får da den endelige kameramatrisen:

$$\mathrm{K} = \begin{bmatrix} \alpha_x & s & u_0 \\ & \alpha_y & v_0 \\ & & 1 \end{bmatrix} \tag{2.12}$$

der $\alpha_x = fm_x$ og $\alpha_y = fm_y$. u_0 og v_0 er hovedpunktet gitt i piksler, med koordinater $u_0 = m_x p_x$ og $v_0 = m_y p_y$

2.2.2 Distorsjon (forvregning)

Distorsjon kan beskrives som en ikke-uniform strekking av bildet, forårsaket av at linsen har ulik forstørrelse over linseflaten. Distorsjon er et avvik fra rektilineær projekson, de rette linjene blir buet i bildet. Dette er spesielt lett å se ved bruk av vidvinkellinser, men en vil finne elementer av distorsjon i de fleste kamerasystemer. Noen høykvalitets vidvinkellinser konstrueres med optisk rektilineær



Figur 2: Venstre: Rutenett uten distorsjon. Midten: Rutenett med tønnedistorsjon. Høyre: Rutenett med nåleputedistorsjon

korreksjon, men det er for mange linser aktuelt å utføre en programvarebasert korreksjon for distorsjonen. Distorsjon deles vanligvis i radielle og tangentielle komponenter. De mest fundamentale formene for radiell distorsjon, tønne- og nåleputedistorsjon, er vist på et rutenett i figur 2. En ser at de i utgangspunktet rette linjene blir buet av turen gjennom linsen. Effekten er størst i kantene, mens linjene oppfattes som rette i nærheten av sentrum. Tønnedistorsjon er typisk for vidvinkellinser, da linsen forstørrer mer i sentrum, mens nåleputedistorsjon er motsatt effekt, mindre forstørrelse i sentrum. Distorsjon opptrer også som en blanding av de to, dette kalles bartedistorsjon eller kompleks distorsjon. Tangentiell distorsjon skyldes som nevnt tidligere at bildesensoren i kameraet ikke er montert eksakt parallellt med linsen. Det finnes flere algoritmer for å korrigere for distorsjon [20]. Mest brukt er Browns distorsjonsmodell [5], som også benyttes av Tsai [24]. Radiell distorsjon modelleres med et skift i bildekoordinater. Avstanden til en piksel endres i forhold til bildesentrum. La $(x_u, y_u)^T$ beskrive uforvrengte bildekoordinater og $(x_d, y_d)^T$ beskrive forvrengte bildekoordinater, mens $(x_c, y_c)^T$ beskriver bildets senterpunkt. Transformasjonen fra $(x_d, y_d)^T$ til $(x_u, y_u)^T$ kan approksimeres ved en Taylorrekkeutvikling:

$$(x_u, y_u)^T = \begin{matrix} x_d + (x_d - x_c)(k_1 r^2 + k_2 r^4 + k_3 r^6 + \dots) \\ y_d + (y_d - y_c)(k_1 r^2 + k_2 r^4 + k_3 r^6 + \dots) \end{matrix} \tag{2.13}$$

$r = \sqrt{(x_d - xc)^2 + (y_d - y_c)^2}$ og k_i er radielle distorsjonskoeffisienter som er blant parametrene som finnes ved kalibrering.
Tangentiell distorsjon kan korrigeres med:

$$(x_u, y_u)^T = \begin{matrix} x_d + (2p_1 y_d + p_2(r^2 + 2x_d^2)) \\ y_d + (p_1(r^2 + 2y_d^2) + 2p_2 x_d) \end{matrix} \tag{2.14}$$

Hvor p_i er tangentiell distorsjonskoeffisient og må kalibreres. En kan da ved å sette sammen ligning 2.13 og ligning 2.14 få Browns distorsjonsmodell som kan korrigere for både radiell og tangentiell forvregning [4]:

$$x_u = x_d + (x_d - xc)(k_1 r^2 + k_2 r^4 + \dots) + (p_1(r^2 + 2(x_d - xc)^2) + 2p_2(x_d - xc)(y_d - y_c)(1 + p_3 r^2 + \dots) \tag{2.15}$$

$$y_u = y_d + (y_d - y_c)(k_1 r^2 + k_2 r^4 + \dots) + (p_2(r^2 + 2(y_d - yc)^2) + 2p_1(x_d - x_c)(y_d - y_c)(1 + p_3 r^2 + \dots) \tag{2.16}$$

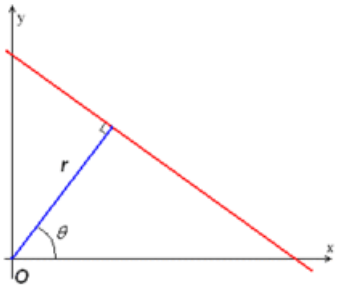
Tangentiell distorsjon kan i de fleste praktiske applikasjoner neglisjeres, og for radiell distorsjon er det tilstrekkelig å inkludere det første leddet, da flere ledd ikke øker nøyaktigheten og vil føre til numerisk ustabilitet [24].En ender da opp med en nokså enkel modell:

$$(x_u, y_u)^T = \begin{matrix} x_d(1 + k_1 r^2) \\ y_d(1 + k_1 r^2) \end{matrix} \tag{2.17}$$

Det kan likevel være aktuelt å ta med et 2. og evt. 3. ledd ved kalibrering av vidvinkellinser med mye radiell distorsjon [20][21].

2.3 Avlesning av fokuspunktenes bildekoordinater

De fleste kamerakalibreringsalgoritmene krever bildekoordinatene til kalibreringsobjektenes fokuspunkter som inngang. Registreringen av disse er derfor viktig og for nøyaktig kalibrering kreves en stor grad av avlesningsnøyaktighet. Den mest intuitive metoden å lese av fokuspunktene på, er å klikke på de med musepekeren. Dette kan gi nøyaktighet ned mot 1 piksel dersom oppløsningen i bildet er vesentlig lavere enn skjermens oppløsning, eller det benyttes zoom. Hvis en bruker kalibreringsobjekter med mange fokuspunkter, eller bruker mange observasjoner (bilder) er dette en tidkrevende prosess. Det er derfor vanlig å automatisere innhentingen av bildepunkter. Dersom kalibreringsobjektet er plant og består av et rutenett kan det benyttes en hjørnedetektor. Brukeren må gjerne identifisere mønstrets ytre hjørner, og en algoritme finner de resterende. Med en modifisert Harris hjørnedetektor [11] kan en få en nøyaktighet under $\frac{1}{10}$ piksel, uavhenging av skjermoppløsning. Dette er brukt i [3]. Metoden for å avlese fokuspunkter er selvsagt avhengig av kalibreringsobjektets utforming, og det kan være lurt å tilpasse kalibreringsobjektet til eksisterende metoder enn å tilpasse metodene etter kalibreringsobjektet. Ved 1-dimensjonal kalibrering benyttes fokuspunkter på en rett linje. Dette kan være kuler tredd på en pinne, eller rett og slett markeringer på en linjal. Ved bruk av kuler kan det benyttes en sirkeldetektor, da kuler avbildet på et 2-dimensjonalt plan vil få sirkelform. Ved å benytte markeringene på en linjal kan en gradientbasert metode være hensiktsmessig, der en detekterer markeringene langs linjalens ytterkant.



Figur 3: Viser hvordan parametrene r og θ beskriver en rett linje for bruk med Hough-transform

2.3.1 Sirkeldeteksjon basert på Hough-transform

Hough-transform er en teknikk beregnet for å finne forekomster av en spesiell form i et bilde. Formen som skal finnes må kunne beskrives på parameterform. Metoden bygger på en patent tilhørende Paul Hough fra 1962 [15]. I sin mest brukte form [22] kan Hough-transformen finne linjer, sirkler og ellipser, former som enkelt kan beskrives med en matematisk ligning. Den finnes også i en generalisert form som kan finne vilkårlige objekter og former i bilder, kravet er at de vilkårlige objektene kan modelleres med en analytisk ligning på formen $f(\mathbf{x}, \mathbf{a}) = 0$, hvor $\mathbf{x}$ betegner et bildepunkt og $\mathbf{a}$ er en parametervektor [2]. Den generaliserte formen krever stor beregnings- og lagringskapasitet. Med n parametre i $\mathbf{a}$, med M diskrete verdier i hver, blir tidsbruken på algoritmen $O(M^{m-2})$. Denne har senere blitt videreutviklet, og flere, mer beregnigseffektive metoder har blitt foreslått [1]. Betegnelsen Hough-transform vil i resten av oppgaven brukes på metoden beskrevet i [22]. En rett linje kan beskrives med ligningen $y = mx + b$. Hvis en ser for seg at den rette linjen plottes i et binært bilde, sort på hvit bakgrunn, vil en piksel (x, y) da enten være en del av linjen eller ikke. Ideen bak Hough-transformen er å identifisere parametrene i funksjonen, m og b i dette tilfellet, som beskriver linjen istedenfor å identifisere den som bildepunkter. For å gjøre dette benytter en seg av en en akkumulatortabell, dimensjonen på denne samsvarer med antall ukjente i ligningen som beskriver den formen som skal finnes. For den rette linjen får den 2 dimensjoner. Siden horisontale og vertikale linjer vil gjøre at stigningstallet går mot henholdsvis 0 og ∞, verdier som kan gi vanskeligheter ved implementering, benyttes andre parametre. For en rett linje benyttes vektoren r som er den korteste avstanden fra bildets origo til linjen, og vinkelen θ som er vinkelen til r, dette er vist i figur 3. Akkumulatortabellen kan sees på som en bokhylle. Hvis en sorterer bøkene på forfatterens etternavn og allokerer en hylle per forbokstav vil en ha en 29 etasjer høy bokhylle. Etter å ha sortert bøker kan en se for seg at noen hyller har mer innhold enn andre. Hough-transformen prosesserer hver piksel og dens nabopiksler i bildet, den avgjør så om pikselen inneholder en kant, og hvis det er tilfelle beregner den parametrene til den ønskede kurven (formen) som skal detekteres. Disse parametrene blir bøkene i analogien med bokhyllen.

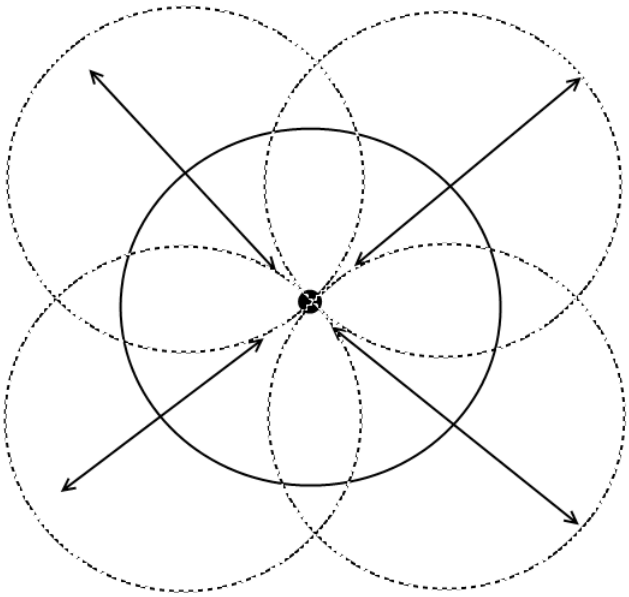
Når parametrene er beregnet kvantiseres de, og en øker verdien på akkumulatortabellen for den kvantiserte verdien. Dette kan sees på som å lese forfatterens navn, finne forbokstaven i etternavnet og sette boken i riktig hylle. Etter prosesseringen finner en toppunkter i akkumulatortabellen (hyllene med flest bøker). Disse indikerer parametrene til de mest sannsynlige linjene i bildet. Det er viktig å finne en hensiktsmessig størrelse på vinduene i akkumulatortabellen, slik at en får nok forekomster til å se tendensene. For å dra analogien videre; dersom en har 30 bøker vil en om bøkene er tilfeldig utvalgt ha 29 mer eller mindre tomme hyller. Om man derimot har 1000 bøker vil enkelte hyller være smekkfulle, mens andre vil ha færre bøker. Med få bøker kunne man da brukt færre hyller, mens man med mange kanskje må utvide. For å finne sirkler i bildet med Hough-tansform er fremgangsmåten typisk følgende:

1. Beregn gradienten til bildet.
2. For hvert element i gradientbildet som ikke er 0, øk verdien i akkumulatortabellen for hvert punkt som er lengden av oppgitt radius ifra elementet. Dette fører til at ethvert punkt på sirkelen vil øke verdien på senterpunktet. Prinsippet er vist i figur 4.

2.4 Kamerakalibrering

Kamerakalibrering defineres som sagt her som å finne parametrene i en gitt kameramodell. De fleste metodene benytter bildepunktene til et kalibreringsobjekt som inngang. Disse brukes i et ligningssystem, der løsningen gir et estimat på parametrene i kameramodellen. Dette estimatet kan være meget følsomt for støy fra avlesningen av bildepunktene, derfor forbedres estimatet ofte ved å bruke linjære eller ulinjære optimeringsalgoritmer. Ligningssystemene løses ved hjelp av matriseinvertering. Dette er en metode som setter krav til skalering på tallverdiene på matrisen. Dersom verdiene spriker mye, altså at en har en blanding av veldig store og veldig små verdier, vil matriseinvertering gi en unøyaktig løsning på ligningssystemet. Dette er som vi skal se senere ofte tilfelle. For å forbedre resultatet normaliseres derfor ofte verdiene i matrisen. Kamerakalibreringsmetodene kan klassifiseres på ulike måter [9], her benyttes antall dimensjoner på kalibreringsobjektet, som er lik klassifiseringen til Zhang [27], [28].

Kalibreringsmetodene som diskuteres i denne oppgaven forutsetter at optikken til kameraet er fast. Likevel finnes det metoder for å modellere og kalibrere kameraer med variabel optikk (eks. zoom, autofokus) [26].



Figur 4: Viser hvordan Hough-transform fungerer for sirkeldeteksjon. Alle verdiene langs de stiplede sirklene vil inkrementeres, noe som resulterer i en markant høyere verdi i sentrum av sirkelen. (Markert med et punkt)

Figur 5: Viser et 3-dimensjonalt kalibreringsobjekt bestående av to plan vinkelrett på hverandre. Planene kalles ofte "Tsai grid".

2.4.1 3D kalibreringsobjekt

Det benyttes et 3-dimensjonalt kalibreringsobjekt, "jig", med kjente dimensjoner. Objektet består ofte av to plan stående vinkelrett på hverandre, men kan være et hvilket som helst 3-dimensjonalt objekt, så lenge en kjenner dets nøyaktige geometri [23]. Eksempel på 3-dimensjonalt kalibreringsobjekt er vist i figur 5. Det kan også benyttes et 2-dimensjonalt objekt som gjennomgår en presist kjent forflytting i rommet [25]. Kalibrering med 3-dimensjonale objekter gir generelt bedre nøyaktighet enn andre metoder, men medfører høyere kostnader i form av utstyr.

2.4.2 2D kalibreringsobjekt

Kalibreringsmetodene i dette kapitlet bruker bilder tatt av et 2-dimensjonalt plan påtegnet et mønster, som oftest et rutenett, for å bestemme kameraparametrene. Det finnes flere tilnærminger, de tre viktigste er utviklet av Roger Y. Tsai [24], Heikkila og Silven [14] og Zhengyou Zhang [27]. Det sees i dette kapitlet nærmere på metodene til Tsai og Zhang.

Tsais metode

Denne metoden er en av de mest brukte til kamerakalibrering, utviklet av Roger Y. Tsai [25]. Metoden er todelt, først beregnes de intrinsikke parametrene, deretter brukes disse til å finne de ekstrinsikke. Det tas utgangspunkt i nålehullskameramodellen beskrevet i kapittel 2.2.1. Parametrene som skal kalibreres er dermed:

- Intrinsikke parametre:
  - fokallengde,f
  - hovedpunkt, u_0, v_0
  - skaleringsfaktor, d_x, d_y
  - distorsjonsfaktor, k_1
  - skaleringsforhold, τ_1
- Ekstrinsikke parametre:
  - translasjonsvektor,
$$t = [t_x, t_y, t_z]^T \tag{2.18}$$
  - rotasjonsmatrise,
$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.19}$$

Rotasjonsmatrisen inneholder kun 3 uavhengige parametre, rotasjonsvinklene om hver akse (R_x, R_y, R_z). Oppbygningen til matrisen R er:

$$\begin{aligned}
r_{11} &= \cos(R_y)\cos(R_z) \\
r_{12} &= \cos(R_z)\sin(R_x)\sin(R_y) - \cos(R_x)\sin(R_z) \\
r_{13} &= \sin(R_x)\sin(R_z) + \cos(R_x)\cos(R_z)\sin(R_y) \\
r_{21} &= \cos(R_y)\sin(R_z) \\
r_{22} &= \sin(R_x)\sin(R_y)\sin(R_z) + \cos(R_x)\cos(R_z) \\
r_{23} &= \cos(R_x)\sin(R_y)\sin(R_z) - \cos(R_z)\sin(R_x) \\
r_{31} &= -\sin(R_y) \\
r_{32} &= \cos(R_y)\sin(R_x) \\
r_{33} &= \cos(R_x)\cos(R_y)
\end{aligned}$$

Det benyttes en flate påtegnet et mønster, vanligvis et rutenett eller sirkler. Sentrum i sirklene eller hjørnene i rutenettet benyttes som referansepunkter. Disse

kan enten finnes manuelt eller automatisk ved hjelp av for eksempel hjørnedeteksjon. Mønsteret påmonteres typisk en skinne og er bevegelig langs den optiske aksen, kameraposisjonen holdes fast, og en avbilder mønsteret med ulik avstand fra kameraet. Kameraets fokus må holdes konstant over alle bildene og en kan derfor ha et noe begrenset område å variere avstanden på, da avbildningen av mønsteret må være noenlunde skarpt.

Zhangs metode (2D)

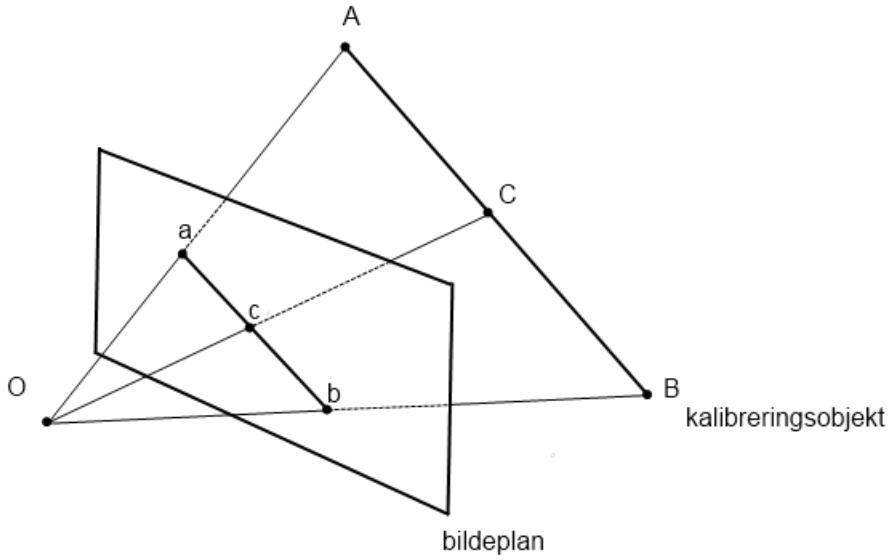
Zhang beskriver i [27] en metode for å kalibrere et kamera gjennom minst to observasjoner av et plan. Metoden krever at enten planet eller kameraet beveger seg i rommet mellom observasjonene. Bevegelsen trenger ikke være kjent. Metoden er mer fleksibel en andre tilnærminger til 2-dimensjonal kalibrering og metoder med 3-dimensjonale objekter, men gir likevel meget gode resultater. Kalibreringsobjektet kan være så enkelt som en utskrift av et rutemønster limt på en skive papp eller en annen forholdsvis stiv flate. Algoritmen bruker bildekoordinatene til hjørnene i rutemønsteret til å beregne en projektivtransformasjon, eller homografi, mellom bildepunktene i de ulike observasjonene i bildet.

2.4.3 1D kalibreringsobjekt

Det å bruke 1-dimensjonale kalibreringsobjekt er innen kamerakalibrering en forholdsvis ny teknikk, flere metoder er publisert. Zhang presenterte i 2004 en metode som baserer seg på 6 observasjoner av et 1-dimensjonalt objekt der objektets dimensjoner er nøyaktig kjent [28]. Cao og Foroosh presenterte i 2004 en metode for å utføre kamerakalibrering med 4 observasjoner av et 1-dimensjonalt objekt av ukjent størrelse, der objektets posisjon er konstant, mens kameraets posisjon varierer for hver observasjon [7].

Zhangs metode (1D)

Det tas utgangspunkt i et kalibreringsobjekt bestående av punkter på en rett linje, f.eks en snor med påtredde kuler hengende fra et tak. Bruksområdet foreslås til kalibrering av oppsett med flere kameraer, der en for å få avdekket de geometriske sammengengene mellom kameraene er avhengig av at flere kameraer kan ta bilde av det samme kalibreringsobjektet fra forskjellige posisjoner. Den foreslås også som aktuell å bruke istedenfor 3-dimensjonale eller 2-dimensjonale objekter i andre tilfeller. Det blir i [28] vist at kamerakalibrering ikke er mulig for fritt bevegelige 1-dimensjonale kalibreringsobjekter. For at kamerakalibrering skal være mulig er det vist at kalibreringsobjektets bevegelse må begrenses omkring et fast punkt, og en trenger minst 6 observasjoner med 3 punkter med kjente avstander. Det er i [28] ikke vist hva som skjer hvis man øker antall punkter på



Figur 6: 1-dimensjonalt kalibreringsobjekt, punkt A eller B må være fast.

kalibreringsobjektet, men det forventes at en oppnår større nøyaktighet, da mer data vil forbedre signal/støyforholdet.

Det tas utgangspunkt i nålehullskameramodellen som tidligere, kameramodellen er da gitt i ligning 2.12. Det 1-dimensjonale kalibreringsobjektets dimensjoner forutsettes kjent. I figur 6 vises et objekt med 3 referansepunkter. Objektets lengde, L, er gitt ved:

$$\|B - A\| = L \tag{2.20}$$

Siden referansepunktene skal ha lik avstand mellom hverandre kan punktet C beskrives:

$$C = \lambda_A A + \lambda_B B \tag{2.21}$$

og λ_A og λ_B er kjent. I tilfellet i figur 2 med 3 punkter hvor C er midtpunktet blir $\lambda_A = \lambda_B = 0.5$. En lar kameraets koordinatsystem definere kalibreringsobjektet. Dette fører til at $\mathbf{R} = \mathbf{I}$ og $\mathbf{t} = \mathbf{0}$ i ligning 2.10. Videre kaller en dybden (avstanden fra origo til punktet langs den optiske aksen) til punktene A, B og C for z_A, z_B og z_C. Punktene kan ved å bruke kameramatrisen i ligning 2.12 defineres,

$$A = z_A \mathbf{K^{-1}} \tilde{\mathbf{a}} \tag{2.22}$$

$$B = z_B \mathbf{K^{-1}} \tilde{\mathbf{b}} \tag{2.23}$$

$$C = z_C \mathbf{K^{-1}} \tilde{\mathbf{c}} \tag{2.24}$$

hvor $\tilde{\mathbf{a}}$, $\tilde{\mathbf{b}}$ og $\tilde{\mathbf{c}}$ er de respektive punktenes plassering i bildeplanet beskrevet med homogene bildekoordinater. Ved å sette inn i ligning 2.21 får en:

$$z_C\tilde{\mathbf{c}} = z_A\lambda_A\tilde{\mathbf{a}} + z_B\lambda_B\tilde{\mathbf{b}} \tag{2.25}$$

Ved å finne kryssproduktet med vektoren $\tilde{\mathbf{c}}$ og ligning 2.25,

$$z_A\lambda_A(\tilde{\mathbf{a}} \times \tilde{\mathbf{c}}) + z_B\lambda B(\tilde{\mathbf{b}} \times \tilde{\mathbf{c}}) = \mathbf{0} \tag{2.26}$$

får en:

$$z_B = -z_A\frac{\lambda_A(\tilde{\mathbf{a}} \times \tilde{\mathbf{c}}) \cdot (\tilde{\mathbf{b}} \times \tilde{\mathbf{c}})}{\lambda_B(\tilde{\mathbf{b}} \times \tilde{\mathbf{c}}) \cdot (\tilde{\mathbf{b}} \times \tilde{\mathbf{c}})} \tag{2.27}$$

En har fra ligning 2.20 at,

$$\|\mathbf{K}^{-1}(z_B\tilde{\mathbf{b}} - z_A\tilde{\mathbf{a}})\| = L \tag{2.28}$$

Ved å sette inn for z_B fra ligning 2.27 får en,

$$z_A\|\mathbf{K}^{-1}\left(\tilde{\mathbf{a}} + \frac{\lambda_A(\tilde{\mathbf{a}} \times \tilde{\mathbf{c}}) \cdot (\tilde{\mathbf{b}} \times \tilde{\mathbf{c}})}{\lambda_B(\tilde{\mathbf{b}} \times \tilde{\mathbf{c}}) \cdot (\tilde{\mathbf{b}} \times \tilde{\mathbf{c}})}\tilde{\mathbf{b}}\right)\| = L \tag{2.29}$$

Dette kan skrives,

$$z_A^2\mathbf{h}^T\mathbf{K}^{-T}\mathbf{K}^{-1}\mathbf{h} = L^2 \tag{2.30}$$

hvor,

$$\mathbf{h} = \tilde{\mathbf{a}} + \frac{\lambda_A(\tilde{\mathbf{a}} \times \tilde{\mathbf{c}}) \cdot (\tilde{\mathbf{b}} \times \tilde{\mathbf{c}})}{\lambda_B(\tilde{\mathbf{b}} \times \tilde{\mathbf{c}}) \cdot (\tilde{\mathbf{b}} \times \tilde{\mathbf{c}})}\tilde{\mathbf{b}} \tag{2.31}$$

ligning 2.30 inneholder og former basisen for å finne parametrene i kameramatrisen K og den ukjente dybden langs den optiske aksen til kalibreringsobjektets faste punkt, z_A. Vektoren $\mathbf{h}$ består av bildepunkter og de kjente avstandene mellom punktene på kalibreringsobjektet. En sitter da med 6 ukjente, z_A, samt $\alpha_x, \alpha_y, u_0, v_0$ og s fra matrisen $\mathbf{K}$. For å kunne løse kalibreringen trenger en derfor minimum 6 observasjoner av kalibreringsobjektet. Løsningen finnes ved å definere en ny matrise, $\mathbf{B}$.

$$\mathbf{B} = \mathbf{K}^{-T}\mathbf{K}^{-1} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix} \tag{2.32}$$

$$\mathbf{B} = \begin{bmatrix} \frac{1}{\alpha_x^2} & -\frac{s}{\alpha_x^2\alpha_y} & \frac{v_0 s - u_0\alpha_y}{\alpha_x^2\alpha_y} \\ -\frac{s}{\alpha_x^2\alpha_y} & \frac{s^2}{\alpha_x^2\alpha_y^2} + \frac{1}{\alpha_y^2} & -\frac{s(v_0 s - u_0\alpha_y)}{\alpha_x^2\alpha_y^2} - \frac{v_0}{\alpha_y^2} \\ \frac{v_0 s - u_0\alpha_y}{\alpha_x^2\alpha_y} & -\frac{s(v_0 s - u_0\alpha_y)}{\alpha_x^2\alpha_y^2} - \frac{v_0}{\alpha_y^2} & \frac{(v_0 s - u_0\alpha_y)^2}{\alpha_x^2\alpha_y^2} + \frac{v_0^2}{\alpha_y^2} + 1 \end{bmatrix} \tag{2.33}$$

Matrisen $\mathbf{B}$ er symmetrisk og kan defineres med en 6D vektor

$$\mathbf{b} = [B_{11}, B_{12}, B_{22}, B_{13}, B_{23}, B_{33}]^T \tag{2.34}$$

Ved å la $\mathbf{h} = [h_1, h_2, h_3]^T$ og $\mathbf{x} = z_A^2\mathbf{b}$ får en fra ligning 2.30 at

$$\mathbf{v}^T\mathbf{x} = L^2 \tag{2.35}$$

hvor $\mathbf{v} = [h_1^2, 2h_1h_2, h_2^2, 2h_1h_3, 2h_2h_3, h_3^2]$. Med N observasjoner av kalibreringsobjektet, får en et ligningssystem bestående av n ligninger som ligning 2.35

$$\mathbf{V}\mathbf{x} = L^2\mathbf{1} \tag{2.36}$$

med $\mathbf{V} = [\mathbf{v}_1, \ldots \mathbf{v}_N]^T$ og $\mathbf{1} = [1, \ldots, 1]^T$. Minstekvadratsløsningen er da gitt ved

$$\mathbf{x} = L^2(\mathbf{V}^T\mathbf{V})^{-1}\mathbf{V}^T\mathbf{1} \tag{2.37}$$

Hvis det faste punktet, A, er kjent i bildekoordinater, $\mathbf{a}$, har en nå nok informasjon til å beregne verdiene i vektoren $\mathbf{x} = [x_1, x_2, \ldots, x_N]$, $N \geq 6$. En kan da finne de intrinsikke kameraparametrene i matrisen, K, og dybden til det faste punktet, z_A.

$$v_0 = \frac{x_2x_4 - x_1x_5}{x_1x_3 - x_2^2} \tag{2.38}$$

$$z_A = \sqrt{x_6 - \left[x_4^2 + v_0\left(x_2x_4 - x_1x_5\right)\right]/x_1} \tag{2.39}$$

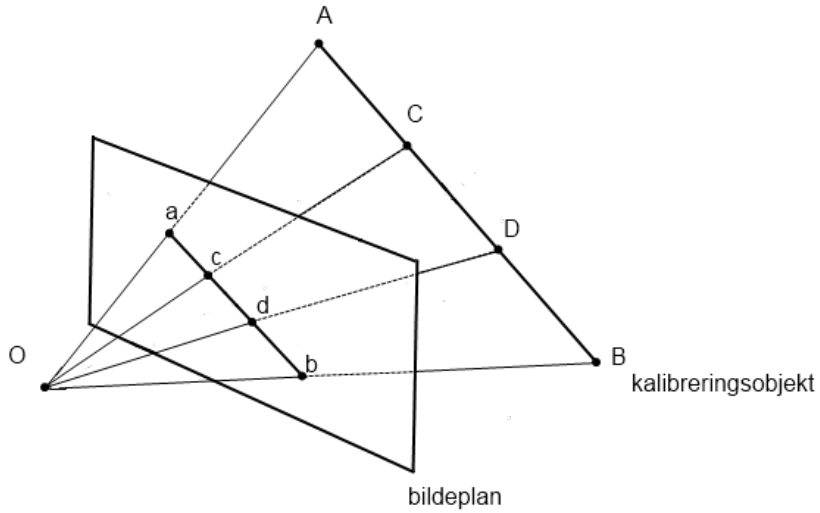
$$\alpha_x = \sqrt{z_A/x_1} \tag{2.40}$$

$$\alpha_y = \sqrt{z_Ax_1/(x_1x_3 - x_2^2} \tag{2.41}$$

$$s = \frac{-x_2\alpha_x^2\alpha_y}{z_A} \tag{2.42}$$

$$u_0 = \frac{sv_0}{\alpha_x} - \frac{x_4\alpha_x^2}{z_A} \tag{2.43}$$

Videre kan dybden, z_B, til punktet B finnes fra ligning 2.27. En er da i stand til å beregne punktene A, B og C fra ligning 2.22, ligning 2.23 og ligning 2.21 henholdsvis. En har da løsning for kamerakalibrering med 1-dimensjonalt objekt med 3 punkter som i figur 6.



Figur 7: 1-dimensjonalt kalibreringsobjekt med 4 referansepunkter.

2.4.4 Utvidelse av Zhangs metode for å gjelde 1-dimensjonalt kalibreringsobjekt med flere enn 3 referansepunkter

For å utvide Zhangs metode til å gjelde kalibreringsobjekter med flere enn 3 referansepunkter er en nødt til å utvide utledningene fra forrige kapittel. Kalibreringsobjektets lengde er fremdeles avstanden mellom endepunktene og gitt i ligning 2.20. Ethvert referansepunkt kan beregnes fra A og B på samme måte som i ligning 2.21. λ_A og λ_B er en vekting som bestemmes av referansepunktets avstand til endepunktene og er kjent. For N referansepunkter blir

$$\lambda_{Ak} = \frac{N-k}{N-1}, k = [2, \ldots, N-1] \tag{2.44}$$

$$\lambda_{Bk} = 1 - \lambda_{Ak} \tag{2.45}$$

Et eksempel med 4 punkter er gitt i figur 7. Indeksen k er referansepunktets plassering fra A. A: $k = 1$, C: $k = 2$, D: $k = 3$ og B: $k = 4$. I dette tilfellet er

$$C = \lambda_{AC}A + \lambda_{BC}B = \frac{2}{3}A + \frac{1}{3}B \tag{2.46}$$

$$D = \lambda_{AD}A + \lambda_{BD}B = \frac{1}{3}A + \frac{2}{3}B \tag{2.47}$$

Et vilkårlig referansepunkt M kan da beskrives,

$$M = z_M \mathbf{K}^{-1}\tilde{\mathbf{m}} \tag{2.48}$$

og,

$$z_M \tilde{\mathbf{m}} = z_A \lambda_{AM} \tilde{\mathbf{a}} + z_B \lambda_{BM} \tilde{\mathbf{b}} \tag{2.49}$$

Ved å utføre kryssprodukt med $\tilde{\mathbf{m}}$,

$$z_M \left(\tilde{\mathbf{m}} \times \tilde{\mathbf{m}}\right) = z_A \lambda_{AM} \left(\tilde{\mathbf{a}} \times \tilde{\mathbf{m}}\right) + z_B \lambda_{BM} \left(\tilde{\mathbf{b}} \times \tilde{\mathbf{m}}\right) \tag{2.50}$$

får en for et generelt referansepunkt

$$z_A \lambda_{AM} \left(\tilde{\mathbf{a}} \times \tilde{\mathbf{m}}\right) + z_B \lambda_{BM} \left(\tilde{\mathbf{b}} \times \tilde{\mathbf{m}}\right) = \mathbf{0} \tag{2.51}$$

Dybden til B, z_B, kan finnes ved bruk av ethvert referansepunkt

$$z_B = -z_A \frac{\lambda_{AM}(\tilde{\mathbf{a}} \times \tilde{\mathbf{m}}) \cdot (\tilde{\mathbf{b}} \times \tilde{\mathbf{m}})}{\lambda_{BM}(\tilde{\mathbf{b}} \times \tilde{\mathbf{m}}) \cdot (\tilde{\mathbf{b}} \times \tilde{\mathbf{m}})} \tag{2.52}$$

Med flere punkter på kalibreringsobjektet får en flere **h**-vektorer. For eksempelet fra figur 7 med 4 referansepunkter, får en to vektorer **h** for hver observasjon av kalibreringsobjektet:

$$\mathbf{h_{11}} = \tilde{\mathbf{a}} + \left[\frac{\lambda_{AC}(\tilde{\mathbf{a}} \times \tilde{\mathbf{c}}) \cdot (\tilde{\mathbf{b}} \times \tilde{\mathbf{c}})}{\lambda_{BC}(\tilde{\mathbf{b}} \times \tilde{\mathbf{c}}) \cdot (\tilde{\mathbf{b}} \times \tilde{\mathbf{c}})}\right] \tilde{\mathbf{b}} \tag{2.53}$$

og

$$\mathbf{h_{12}} = \tilde{\mathbf{a}} + \left[\frac{\lambda_{AD}(\tilde{\mathbf{a}} \times \tilde{\mathbf{d}}) \cdot (\tilde{\mathbf{b}} \times \tilde{\mathbf{d}})}{\lambda_{BD}(\tilde{\mathbf{b}} \times \tilde{\mathbf{d}}) \cdot (\tilde{\mathbf{b}} \times \tilde{\mathbf{d}})}\right] \tilde{\mathbf{b}} \tag{2.54}$$

For N observasjoner av et kalibreringsobjekt med i punkter får en dermed $N \cdot (i-2)$ ligninger som ligning 2.35, dette burde gjøre kalibreringen mer robust for feildetektering av referansepunktene (støy), noe som vil bli verifisert eksperimentelt. Merk at kun en av **h**-vektorene er lineært uavhengig per observasjon. Dette gjør at kravet til antall observasjoner for å løse ligningssystemet blir det samme.

2.4.5 Forbedring av kalibreringsresultatene

Normalisering

Zhangs metode for kalibrering med 1-dimensjonalt kalibreringsobjekt benytter seg av matriseinvertering for å komme fram til løsningen, vektoren **x**. Denne finnes ved hjelp av den pseudoinverse til matrisen **V** som er bygd opp av vektorene **v**, som igjen kommer fra kombinasjoner av vektorene **h**. Siden bildekoordinatene er oppgitt i homogene koordinater, kan et bildepunkt typisk ha koordinatene $\tilde{\mathbf{x}} = [100, 100, 1]^T$. Dette fører til at **h**-vektoren får omtrent samme størrelsesorden. Dersom $\mathbf{h_{1-3}} = [100, 100, 1]^T$ vil **v** bli $\mathbf{v} = [10^4, 2 \cdot 10^4, 10^4, 10^2, 10^2, 1]$. Det store

spriket i numeriske verdier gjør matrisen V dårlig egnet for invertering. For å omgå dette problemet benyttes normalisering. Franca et. al. [8] benytter en projektiv transform utledet fra [12]. Denne benyttes på bildepunktene og er konstruert slik at sentrum i settet med bildepunkter ligger i origo, og at gjennomsnittsavstanden mellom origo og et punkt er $\sqrt{2}$. Det er vist at dette reduserer feilen i kalibreringen til en 10-del sammenlignet med å ikke normalisere. Økningen i beregningskompleksitet med normalisering er liten, og ubetydelig hvis en ser på bidraget til feilredusering. Det er også mulig å normalisere matrisen $\mathbf{V}$ direkte over et intervall (typisk $[0, 1]$), dette gjør $\mathbf{V}$ numerisk sett bedre egnet for invertering. Det er ikke funnet dokumentasjon på bruk av denne metoden, men det antas at den kan gi en forbedring i kalibreringsresultatet ved å redusere forekomsten av numeriske feil knyttet til matriseinverteringen i ligning 2.37. Begge metodene er her implementert og testet eksperimentelt i kapittel 3.

Optimering

En kan bruke prinsippet om maksimal sannsynlighet til å øke presisjonen til parametrene i kameramodellen ved støy. Zhang viser at man ved å minimere funksjonen over antall observasjoner (bilder), N

$$\sum_{i=1}^{N}(\|\mathbf{a}_i - \phi(\mathbf{K}, A)\|^2 + \|\mathbf{b}_i - \phi(\mathbf{K}, B_i)\|^2 + \|\mathbf{c}_i - \phi(\mathbf{K}, C_i\|^2 \dots) \tag{2.55}$$

hvor K er estimatet fra den lineære løsningen og $\phi(\mathbf{K}, M)(M \in A, B_i, C_i, \dots)$ er projeksonen av punktet M på bildet, kan få et maksimalt sannsynlighetsestimat for punktene. Punktet A kan, dersom det er avbildet beregnes som gjennomsnittet av alle observasjoner, eller estimeres dersom det ikke er med i bildet [28]. Punktet B beregnes ved hjelp av estimatet for A, den kjente lengden på kalibreringsobjektet L og vinklene θ og ϕ som beskriver kalibreringsobjektets orientering i rommet, $\mathbb{R}^3$.

$$B = A + L \begin{bmatrix} \sin\theta\cos\phi \\ \sin\theta\sin\phi \\ \cos\theta \end{bmatrix} \tag{2.56}$$

Minimering av ligning 2.55 er et ulineært minimeringsproblem. Dette kan løses med Levenberg-Marquardt algoritmen [13].

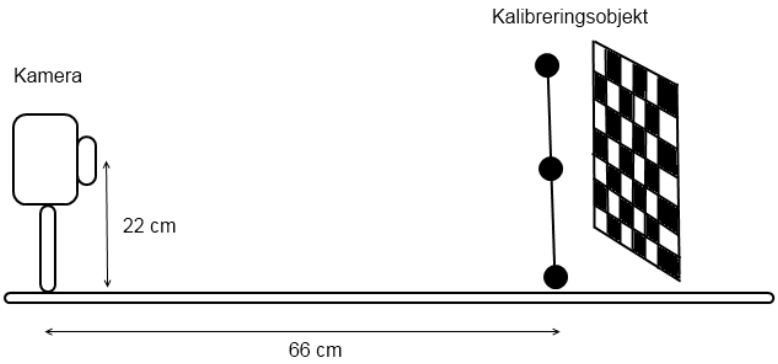
2.4.6 Kalibrering uten kalibreringsobjekt (0D)

Kalibrering uten kalibreringsobjekt kalles ofte autokalibrering. Metodene som eksisterer estimerer kameraparametre og sceneinformasjon fra et sett "ukalibrerte" bilder. Dette kan enten gjøres ved å benytte informasjon om kamerabevegelse [17] eller kreve at bildet har spesielle egenskaper [16].

3 Eksperimenter og resultater

Dette kapittelet inneholder beskrivelse av laboratorieoppsettet, beskrivelse og hensikt med eksperimentene som er utført, samt resultatet av disse.

3.1 Fysisk lab-oppsett

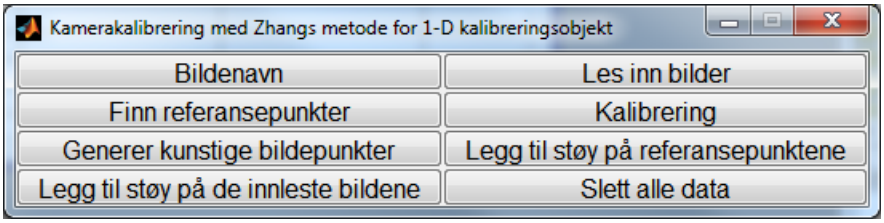


Figur 8: Skisse av det fysiske oppsettet brukt i eksperiment 3.

Eksperiment 3 som medfører bruk av fysiske kalibreringsobjekt er utført på bildebehandlingslaboratoriet ved UIS. Kameraet som er brukt i kalibreringen er et Panasonic DMC-LX3. Dette er et kompaktkamera med mulighet for manuell fokus, noe som er påkrevd for at kalibreringen skal kunne gjennomføres med metodene som er brukt her. Oppløsning kan velges manuelt i forskjellige steg opp til 10.1 megapiksler. Kameraet er fastmontert i en rigg ment for kamerakalibrering. Dette gir en mer stabil montering enn et vanlig kamerastativ som en kan risikere at flytter seg på underlaget når det tas flere bilder. Bildene er tatt med selvutløser for å unngå å berøre kameraet under fotografering, noe som kan gi forskyvning.

3.2 Implementering

Zhangs 1-dimensjonale metode [28] er implementert i Matlab. Implementeringen er basert på "Camera calibration toolbox for Matlab", utviklet ved California In-



Figur 9: Kontrollpanelet til Matlab-programmet

stitute of Technology [3]. Programmet er bygd opp av mindre skript som utfører forskjellige oppgaver, det er utviklet et enkelt grafisk grensesnitt, et kontrollpanel med trykknapper hvor skriptene kjøres fra, se figur 9. En kalibrering av et reelt kamera utføres ved at en først laster inn bildene som skal behandles, deretter markeres kalibreringsobjektets referansepunkter manuelt for hvert enkelt bilde ved å klikke med musepekeren. Det er også laget en rutine for å finne referansepunktene automatisk. Denne er basert på en eksisterende implementering av sirkulær Hough-transform [19]. Metoden som er brukt er i utgangspunktet følsom for støy og finner en stor andel falske og uønskede sirkler. Falske sirkler er punkter i bildet som gjenkjennes som en sirkel uten å være det, mens uønskede sirkler er sirkler som detekteres i bildet utenfor kalibreringsobjektet. Støyfølsomheten kan begrenses ved å være nøyaktig i valget av radiusvindu og filterparametre i filtrene som er innebygd i transformen, men dette er lite hensiktsmessig i praktisk bruk, da det kompliserer og øker tidsbruken på datainnsamlingen betydelig for brukeren. For å omgå dette, samt få rekkefølgen på fokuspunktene riktig, sammenlignes koordinatene til museklikkene med de automatisk detekterte sirklene. Det sirkelsentrumet som er nærmest hvert klikk brukes som bildekoordinater i kalibreringen. Punktet som skal brukes i kalibreringen plottes i bildet, og brukeren godkjenner dette. I alle eksperimentene benyttes Matlab R2010b 64-bit versjon kjørende på 64-bit Windows 7. Programvaren er installert på en bærbar pc med 4GB internminne og Intel Core i5 2,4 GHz CPU. Innledende tester av rutinen ga gode resultater på bilder med lav oppløsning (800×600), men for større bilder tok prosesseringen lang tid, 5-6 minutter for bilder med oppløsning (3648×2736), antakeligvis fordi internminnet ble overbelastet. For store bilder er derfor tilnærmingen litt annerledes. Brukeren markerer et område rundt kulen på kalibreringsobjektet og Hough-transformen behandler kun dette området, uformelle tester av denne framgangsmåten viser prosesseringstid på rundt 0.1 sekund per kule på kalibreringsobjektet (avhengig av størrelsen på området som markeres). Et overfladisk flytskjema over hvordan programmet er lagt opp er vist i figur 10, mer detaljerte flytskjema finnes i Appendix A og er vist med Matlab-kode i Appendix B.

3.3 Eksperimenter

Det er utført totalt 5 eksperimenter. Det første, som tester sirkeldetektorens egenskaper, har som mål å avdekke om implementeringen med Hough-transform som er gjort her gir tilstrekkelig nøyaktighet i avlesningen av kulenes senterpunkt. Dette er viktig for å ha et mål på kvaliteten til dataene som benyttes i kalibreringsalgoritmene.

Eksperiment 2 sammenligner Zhangs metode uten noen form for forbedring (metode 1) mot den samme metoden med normalisering av matrisen $\mathbf{V}$ for å gjøre denne bedre egnet for invertering (metode 2) og den samme metoden med normalisering av bildepunktene etter Hartleys modell [12], som vist i kapittel 2.4.5, før kalibrering.

Figur 10: Overfladisk flytskjema som viser bruken av programmet

Eksperiment 3 sammenligner de tre metodene fra eksperiment 2 med Zhangs kalibreringsalgoritme med 2-dimensjonalt kalibreringsobjekt. Dette er gjort for å gi et mål på kvaliteten en kan forvente med 1-dimensjonalt kalibreringsobjekt.

Formålet med eksperiment 4 er å avdekke hva som skjer når en utvider kalibreringsobjektet utover de 3 kulene Zhang benytter i presentasjonen av algoritmen.

Eksperiment 5 er tatt med for å avdekke hvordan antall bilder som benyttes i kalibreringen påvirker resultatet. Det kreves minimum 6 bilder og det antas at flere bilder øker presisjonen og stabiliteten på kalibreringsresultatet.

3.3.1 Eksperiment 1: Sirkeldetektorens egenskaper

En tester her sirkeldetektorens nøyaktighet. Parametrene som måles er antall sirkler som detekteres i bildet og den utvalgte sirkelens posisjon i bildekoordinater. For at sirkelens posisjon i bildekoordinater skal gi mening må den sammenlignes med en referanse. Et bilde tatt av et kalibreringsobjekt kan her ikke brukes, da en ikke kan finne sirkelsentrum eksakt, og et rent syntetisk bilde kan ikke brukes, da implementeringen krever intensitetsvariasjoner i bildet for å fungere stabilt. Her benyttes derfor et gråtonefoto av månen med omliggende stjerner se figur 11. Oppløsningen på bildet er (800×843). Det brukes to varianter av det samme bildet. Tre like store, hvite sirkler med radius på henholdsvis 20 og 40 piksler for de to bildene, genereres med sentrum i $[100, 700]$, $[100, 100]$ og $[700, 100]$. Det legges Gaussisk støy med middelverdi 0 og standardavvik σ fra 0.01 til 0.2 i steg på 0.1 til bildet. Hough-transformen kjøres med standardverdier på filterparametrene og radiusvinduet settes til henholdsvis $[15, 25]$ og $[35, 45]$. Merk at radiusvinduet er større enn nødvendig, dette er gjort for å simulere en reell situasjon der en ikke kjenner radius eksakt, og ønsker å være sikker på å få med alle relevante sirkler. På grunn av testbildets overkommelige oppløsning benyttes varianten der

brukeren klikker på den interessante sirkelen i bildet. Prosessen er gjentatt 10 ganger og resultatet er middelverdien av disse. Siden sentrum i sirkelen er eksakt kjent er avviket beregnet fra dette.

Bilde 1

3 sirkler med radius 20, radiusvindu $[15, 25]$
Rutinen fungerer bra fram til σ passerer 0.09. For større σ finner den i flere

σ	Sirkel 1: $[100, 700]$	Sirkel 2: $[100, 100]$	Sirkel 3: $[700, 100]$	Avvik
0	$[100.04, 699.99]$	$[100.16, 100.11]$	$[700.08, 99.88]$	-0.02%
0.01	$[100.04, 700.00]$	$[100.26, 100.12]$	$[700.35, 100.02]$	-0.04%
0.02	$[99.89, 700.08]$	$[100.09, 99.98]$	$[700.53, 99.74]$	-0.02%
0.03	$[99.78, 700.11]$	$[99.90, 99.91]$	$[700.06, 100.14]$	0.01%
0.04	$[99.76, 700.33]$	$[100.43, 100.29]$	$[699.81, 99.98]$	-0.03%
0.05	$[100.66, 700.21]$	$[99.97, 99.78]$	$[700.36, 99.64]$	-0.03%
0.06	$[100.31, 700.13]$	$[99.51, 99.60]$	$[699.32, 100.22]$	0.05%
0.07	$[100.02, 699.82]$	$[99.71, 99.66]$	$[699.51, 100.81]$	0.03%
0.08	$[101.38, 700.46]$	$[100.02, 99.84]$	$[699.83, 99.54]$	-0.06%

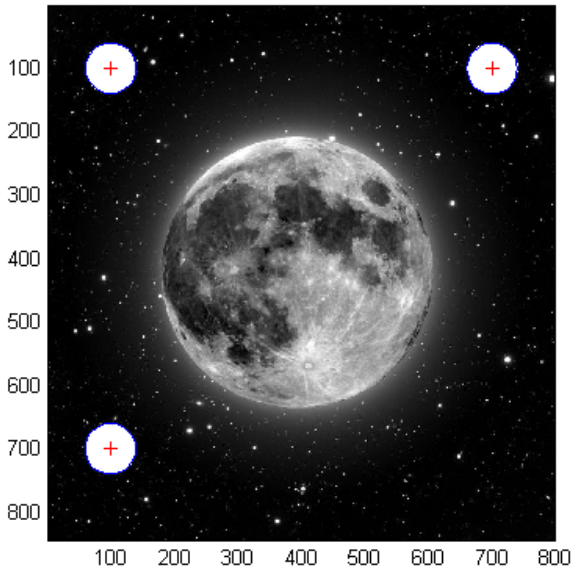
tilfeller ikke alle sirklene, eller bommer helt på sirkelen.

Bilde 2

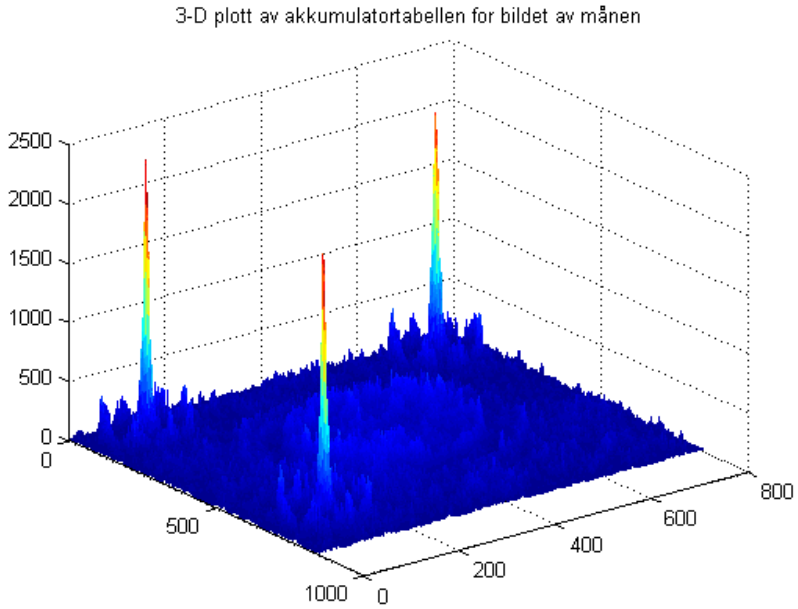
3 sirkler med radius 40, radiusvindu $[35, 45]$

σ	Sirkel 1: $[100, 700]$	Sirkel 2: $[100, 100]$	Sirkel 3: $[700, 100]$	Avvik
0	$[99.40, 699.70]$	$[100.02, 99.97]$	$[700.02, 100.39]$	0.03%
0.01	$[100.04, 700.00]$	$[100.26, 100.12]$	$[700.35, 100.02]$	0.02%
0.02	$[99.27, 699.88]$	$[99.88, 100.54]$	$[699.92, 100.15]$	0.09%
0.03	$[99.43, 699.72]$	$[99.77, 99.83]$	$[699.78, 99.77]$	-0.04%
0.04	$[99.19700.73]$	$[99.6899.85]$	$[700.55100.73]$	0.02%
0.05	$[99.20700.24]$	$[99.0299.93]$	$[699.80100.05]$	-0.1%
0.06	$[100.02699.87]$	$[99.02100.44]$	$[700.12100.17]$	0.02%
0.07	$[100.40700.77]$	$[100.11100.38]$	$[701.23100.83]$	-0.21%
0.08	$[98.91699.76]$	$[99.9799.20]$	$[700.4098.82]$	0.16%

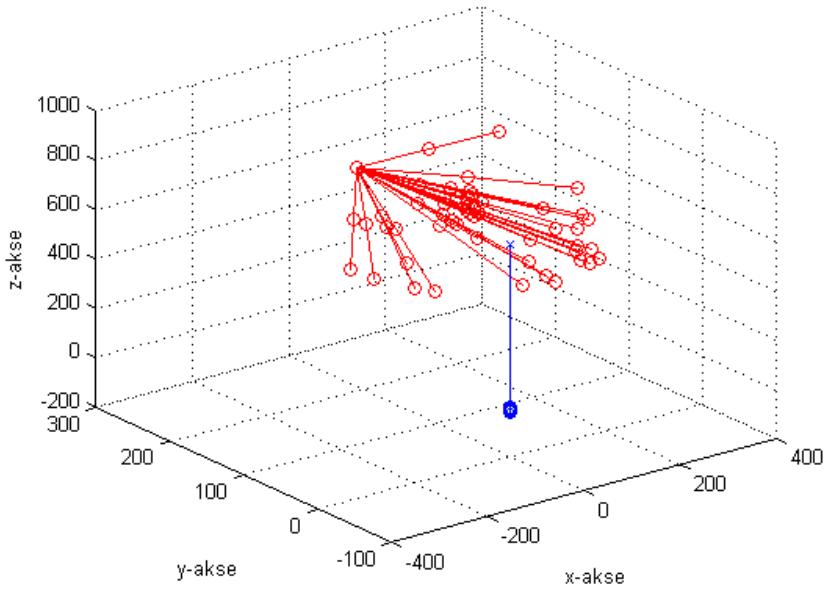
Datagrunnlag er litt tynt for å trekke konklusjoner, men eksperimentet viser at rutinen tåler ca. 6% mer støy ved å doble sirkelstørrelsen. Dette er som forventet.



Figur 11: Testbildet i eksperiment 1, her med radius 40 på de genererte sirklene. Estimert sentrum og estimert radius er plottet av den automatiske sirkeldeteksjonsrutinen.



Figur 12: 3-dimensjonalt plott av akkumulatortabellen til Hough-transformen på bildet i figuren over. Toppene er sentrum til de tre sirklene.



Figur 13: Plott av de genererte kalibreringsobjektene i eksperiment 2 , kameraet med optisk akse er markert i blått.

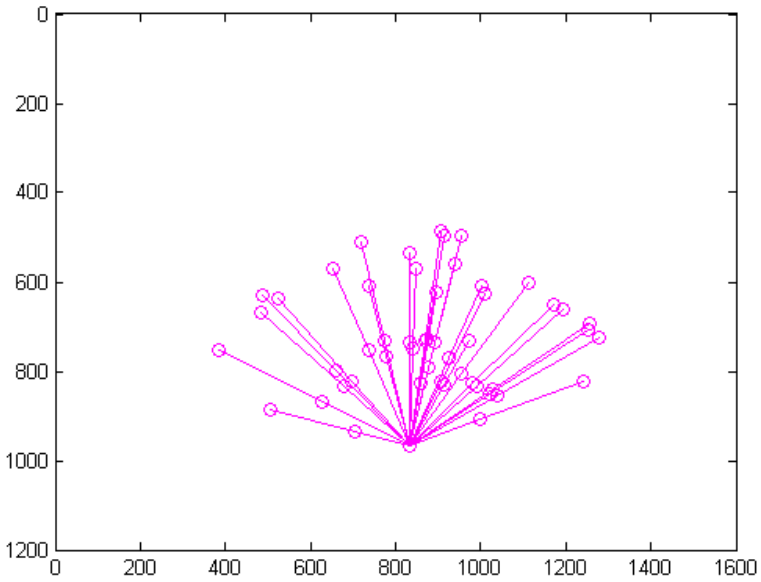
3.3.2 Eksperiment 2: Sammenligning av resultater med og uten normalisering

Forsøket er gjort med syntetiske data. Et sett med 25 observasjoner av et 1-dimensjonalt kalibreringsobjekt med 3 punkter genereres. Punktet A har koordinatene $[20, 220, 660]$, og kalibreringsobjektet genereres ved å sample vinklene θ og ϕ tilfeldig i områdene $[\pi/6, 5\pi/6]$ og $[\pi/6, 5\pi/6]$ henholdsvis. Se figur 13. Dette er gjort for å være sikker på at alle punktene havner i kameraets synsfelt. Kameraet er simulert med kameramatrisen

$$K = \begin{bmatrix} 1100 & 0 & 800 \\ 0 & 1100 & 600 \\ 0 & 0 & 1 \end{bmatrix}$$

og kalibreringsobjektet har lengde 290 mm. Oppløsningen på bildet er 1600×1200, de simulerte bildene er vist i figur 14. Parametrene som er valgt ligger tett opp mot verdiene på det fysiske oppsettet. Det legges til Gaussisk støy med gjennomsnitt 0 og σ standardavvik på det genererte datasettet. Støyen varierer i skritt på 1.5 piksler i området $[0, 30]$ piksler. For hver verdi av σ gjøres kalibrering 250 ganger. Det kalibreres på tre forskjellige måter:

1. Uten normalisering

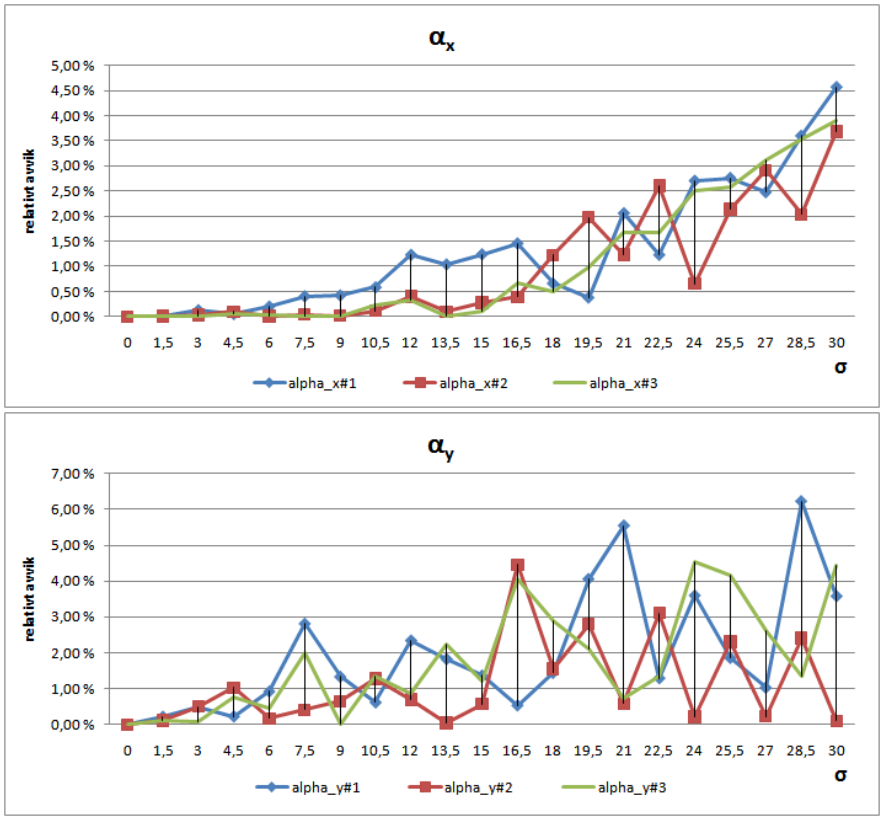


Figur 14: De resulterende bildene av de simulerte kalibreringsobjektet.

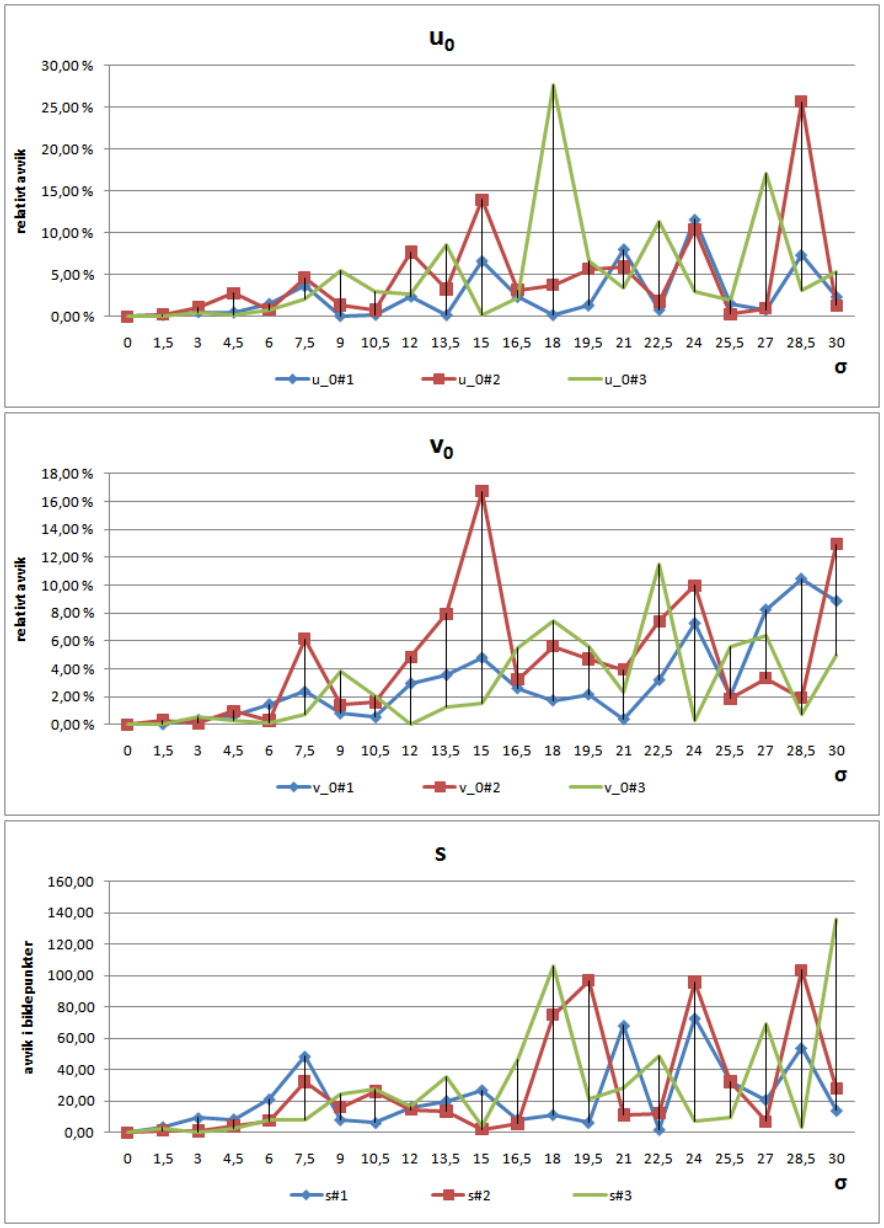
2. Normalisering av $\mathbf{V}$-matrisen i intervallet $[0, 1]$

3. Translasjon og skalering av bildepunktene som beskrevet i kapittel 2.4.5

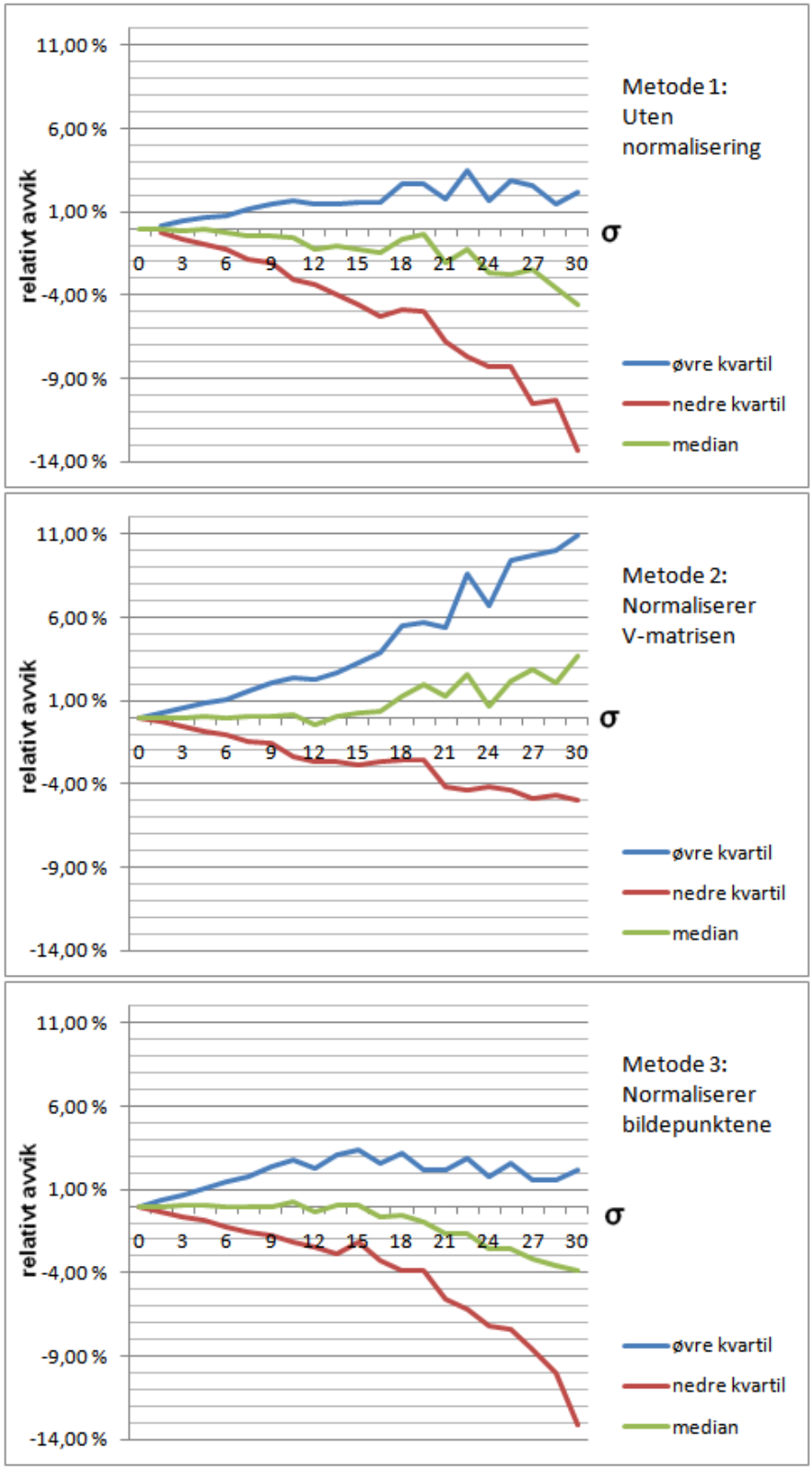
Heretter referert til som metode 1, metode 2 og metode 3. Resultatet av de 250 gjennomkjøringene sorteres etter verdien på α_x, som foreslått av Triggs [6], og det tellende resultatet beregnes som medianen av de sorterte resultatene. På denne måten får en et reelt kalibreringsresultat som tellende, og ikke gjennomsnitt av parametrene, som kan være forskjøvet i forhold til hverandre og ikke mulig å oppnå i praksis. Framgangsmåten for eksperimentet er den samme som er brukt i Franca et. al. [8]. Det beregnes relativt avvik fra resultatet til de kjente kameraparametrene som ble benyttet i genereringen av datasettet for parametrene α_x, α_y, u_0 og v_0. Skråstillingsparameteren s gjengis i bildepunkter, da den i utgangspunktet er 0. På bakgrunn av resultatet, som er vist i figur 15 og figur 16, kan en ikke fastslå store forskjeller på de tre metodene. Det er derfor også utarbeidet en annen framstilling som gir et inntrykk av spredningen for resultatene, medianen er i figur 17 plottet mellom øvre og nedre kvartil for parameteren α_x. Som en ser øker spredningen på kalibreringsresultatene tilnærmet lineært, men det er ikke nevneverdige forskjeller mellom normaliseringsmetodene målt i relativt avvik. Dette er ikke etter forventningene, da det i Franca et. al. [8] fremlegges resultater som tilsier at metode 3 har signifikant mindre relativt avvik enn metode 1.



Figur 15: Resultat for parametrene α_x og α_y fra eksperiment 2.



Figur 16: Resultat for parametrene u_0, v_0 og s fra eksperiment 2



Figur 17: Spredningen i relativt avvik for α_x over 250 simuleringer på hvert støynivå, representert ved hjelp av median og kvartiler for normaliseringsmetodene benyttet i eksperiment 2.

	α_x	α_y	s	u_0	v_0
1-dimensjonal	1159.6	1104.6	0.0072 (90.00°)	592.5	543.1
med metode 2	1340.1	1290.5	0.0262 (90.00°)	748.6	485.8
med metode 3	1180.6	1134.6	6.3628 (89.68°)	620.6	539.8
2-dimensjonal	1125.5	1125.5	0.0000(90.00°)	799.5	599.5
Relativ differanse	4.67%	0.80%	0.04%	28.83%	11.05%

Tabell 1: Viser resultatet av sammenligning av kalibrering med 1-dimensjonalt og 2-dimensjonalt kalibreringsobjekt, relativ differanse gjelder mellom 2-dimensjonal og 1-dimensjonal normalisert med metode 3.

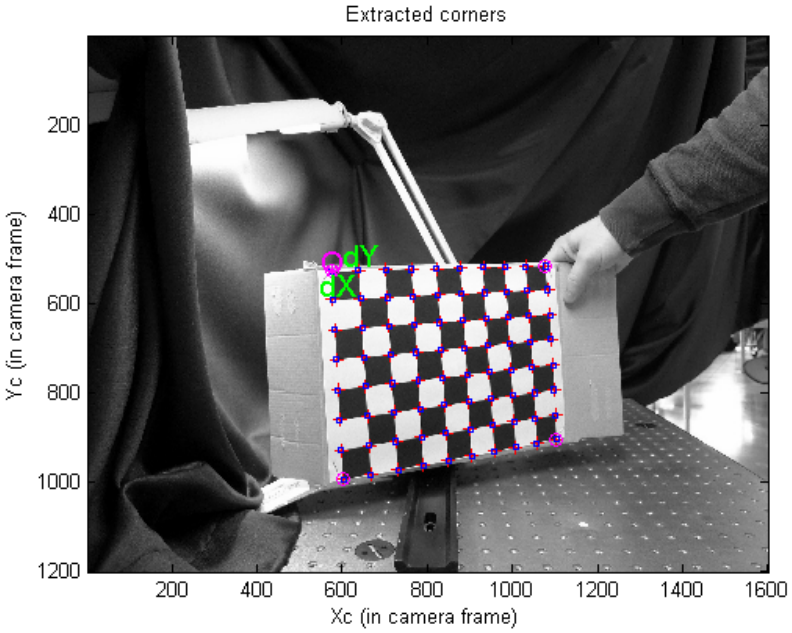
3.3.3 Eksperiment 3: Sammenligning av resultater ved bruk av Zhangs 1-dimensjonale og 2-dimensjonale metode

Forsøket er utført med reelle bilder. Siden en ikke vet kameraparameterne eksakt benyttes Zhangs 2-dimensjonale metode som referanse. For å beregne kameraparametrene med 2-dimensjonalt kaibreringsobjekt benyttes implementeringen i [3]. Kameraet er fokusert manuelt på ca. $0.6m$ (konstant over alle bildene), og bildene er tatt i JPEG-format med oppløsning på 1600×1200 piksler. Kameraet er fastmontert under fotograferingen og det er benyttet selvutløser for å ikke påvirke kameraets posisjon. Det er tatt 6 bilder av et 2-dimensjonalt rutenett med 70 kvadratiske ruter, sidenes lengde er $29mm$. Det er også tatt 15 bilder av et 1-dimensjonalt kalibreringsobjekt bestående av 5 plastperler tredd på en blomsterpinne av tre. Kulene har radius på ca. $5.5mm$ og er plassert med lik avstand mellom hverandre slik at lengden mellom senterpunktene til de to ytterste kulene er $290mm$. Alle avstandene er målt med metermestokk [3] med markering per mm. 15 bilder er bare 10% av det antallet Zhang bruker, men en ønsker å teste algoritmen i en realistisk situasjon. Senterpunktene til kulene på kalibreringsobjektet finnes ved hjelp av den implementerte sirkeldeteksjonsrutinen som er basert på Hough-transform. Denne rutinen krever 5 museklikk per bilde, og har en prosesseringstid på ca. 18 sekunder per bilde, (når en bruker automatisk deteksjon på hele bildet). Innhentingen av bildepunkter tar da ca. 5 minutter for de 15 bildene. 150 bilder hadde tatt opp mot en time, som blir i overkant lenge i en reell situasjon. Resultatene er vist i tabell 1.

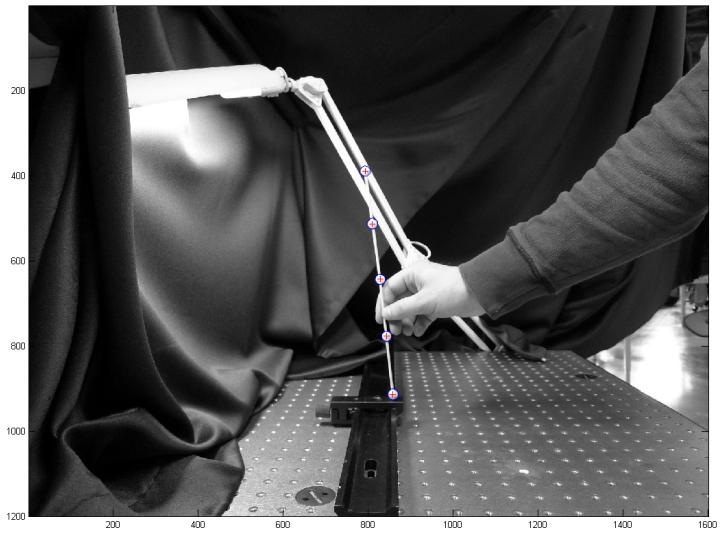
3.3.4 Eksperiment 4: Hvordan påvirker flere referansepunkter på kalibreringsobjektet kalibreringsresultatet?

Det simuleres 25 observasjoner av kalibreringsobjektet med et antall punkter som varierer mellom 3 og 20. Kalibreringsobjektet simuleres tilfeldig for hvert

[3] Meterstokker godkjennes vanligvis fra fabrikk med avvik på opptil $\frac{1cm}{m}$, for at resultatene skal være sammenlignbare er det derfor viktig å benytte samme meterstokk til alle målinger.



Figur 18: Fotografi av 2-dimensjonalt kalibreringsobjekt brukt i eksperiment 3.



Figur 19: Fotografi av 1-dimensjonalt kalibreringsobjekt brukt i eksperiment 3. Her med sentrum av kulene detektert.

antall punkter, slik at orienteringen ikke er lik over eksperimentet. Det legges til støy med $\sigma = 7.5$ (piksler) og en beregner parametrene 250 ganger for hvert antall punkter. De 250 resultatene sorteres etter parameteren α_x, og resultatet er medianen blant de 250 beregningene. Kameraet simuleres som tidligere med følgende parametre

$$K = \begin{bmatrix} 1100 & 0 & 800 \\ 0 & 1100 & 600 \\ 0 & 0 & 1 \end{bmatrix}.$$

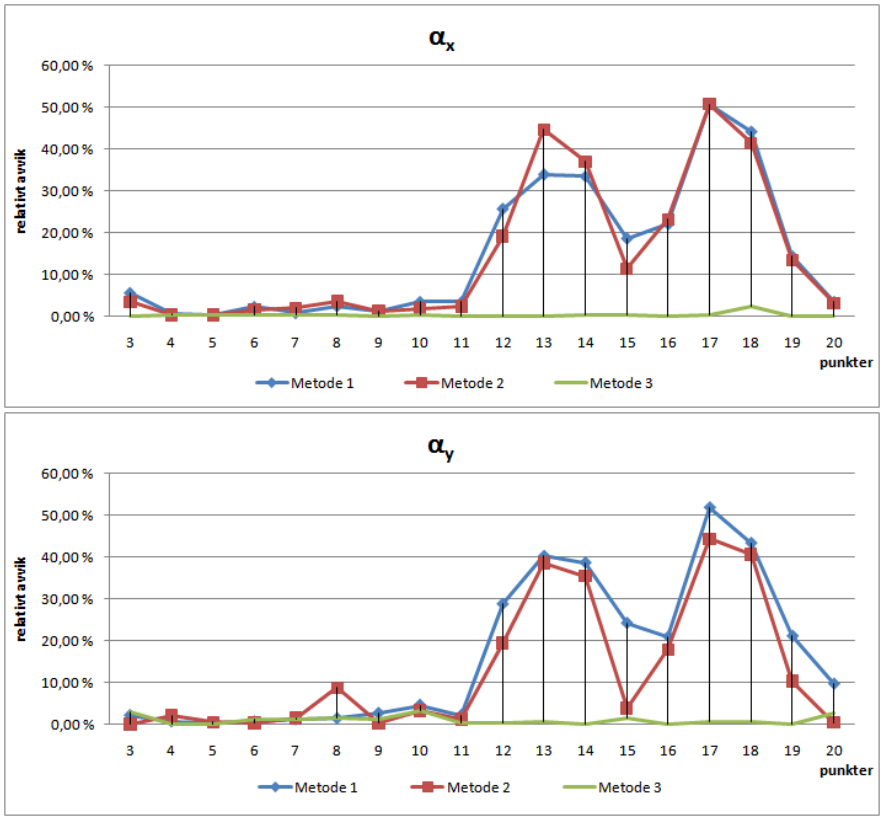
Lengden på kalibreringsobjektet er 290 mm og det faste punktet $A = [20, 220, 660]^T$. Kalibreringsobjektet genereres med tilfeldig orientering der vinklene $\theta \in [\pi/6, 5\pi/6]$ og $\phi \in [\pi/6, 5\pi/6]$ varierer tilfeldig, men uniformt distribuert. Resultatet er vist grafisk i figur 20 og figur 21. En kan se at metode 1 og 2 følger hverandre over alle målingene, mens metode 3 ikke påvirkes av et økende antall referansepunkter. Resultatet for metode 3 sammenfaller med resultatene til Franca et. al. [8]. For metode 2 og 3 er det vist i [8] at feilen stiger tilnærmet lineært når antallet referansepunkter øker, mens en her opplever et sprang i det relative avviket på ca. 15% og 10% henholdsvis for metode 1 og 2 ved en økning fra 11 til 12 referansepunkter. Dette kommer sannsynligvis av at en her genererer nye tilfeldige orienteringer av kalibrerinsobjektet for hvert antall punkter. Ved repetisjon av eksperimentet er det store forskjeller i verdier for metode 1 og 2, og både fallet i relativt avvik for metode 1 og 2 ved 15 punkter og ved 18 og 19 punkter oppfattes som avhengig av orienteringen til det avbildede kalibreringsobjektet.

3.3.5 Eksperiment 5: Hvordan påvirker antall bilder brukt i kalibreringen kalibreringsresultatet?

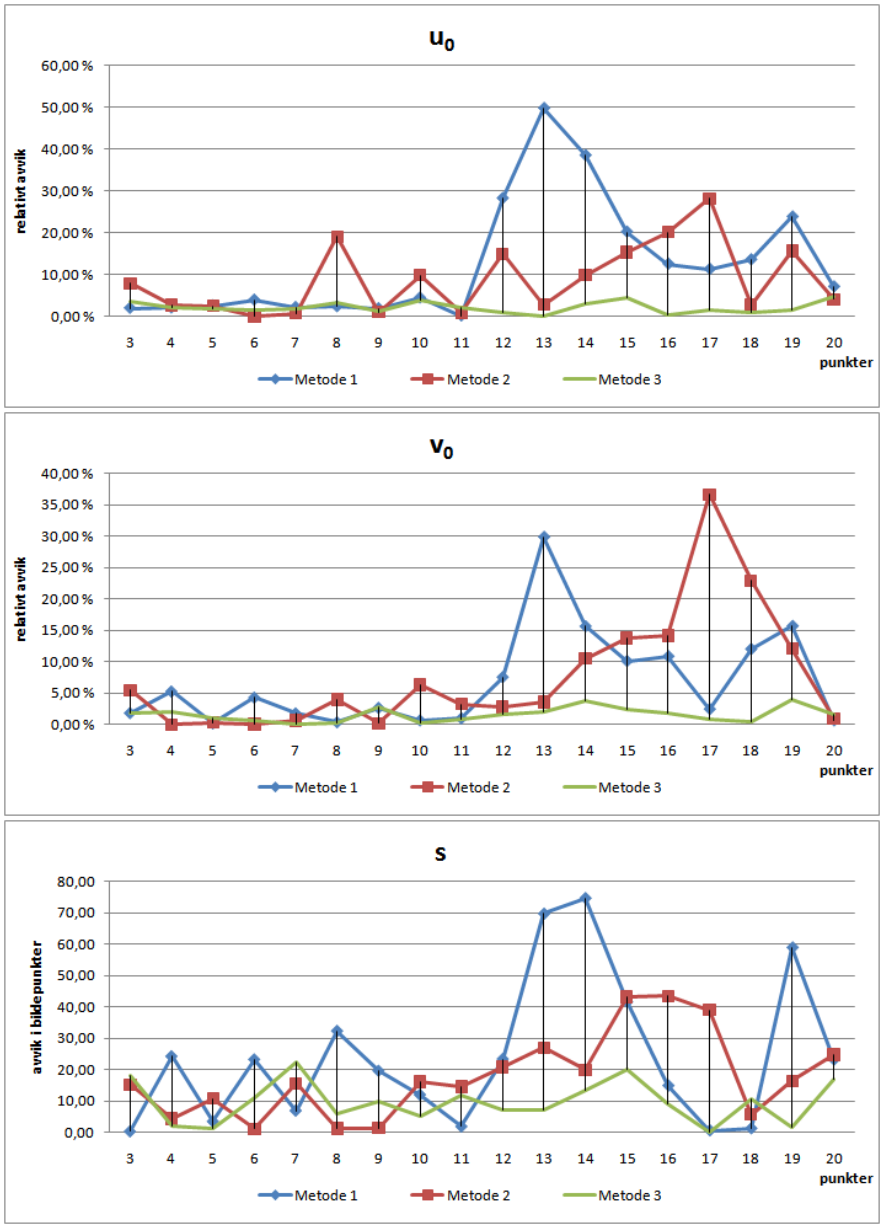
Det genereres et sett med 75 bilder, av et simulert kalibreringsobjekt med 5 referansepunkter. Av utvalget på 75 bilder tas det, for hvert antall bilder n som testes, 250 ganger ut et tilfeldig utvalg på n bilder. Som tidligere legges det til Gaussisk støy med middelverdi 0 og σ standardavvik hvor $\sigma = 7.5$ (piksler).

$$K = \begin{bmatrix} 1100 & 0 & 800 \\ 0 & 1100 & 600 \\ 0 & 0 & 1 \end{bmatrix}.$$

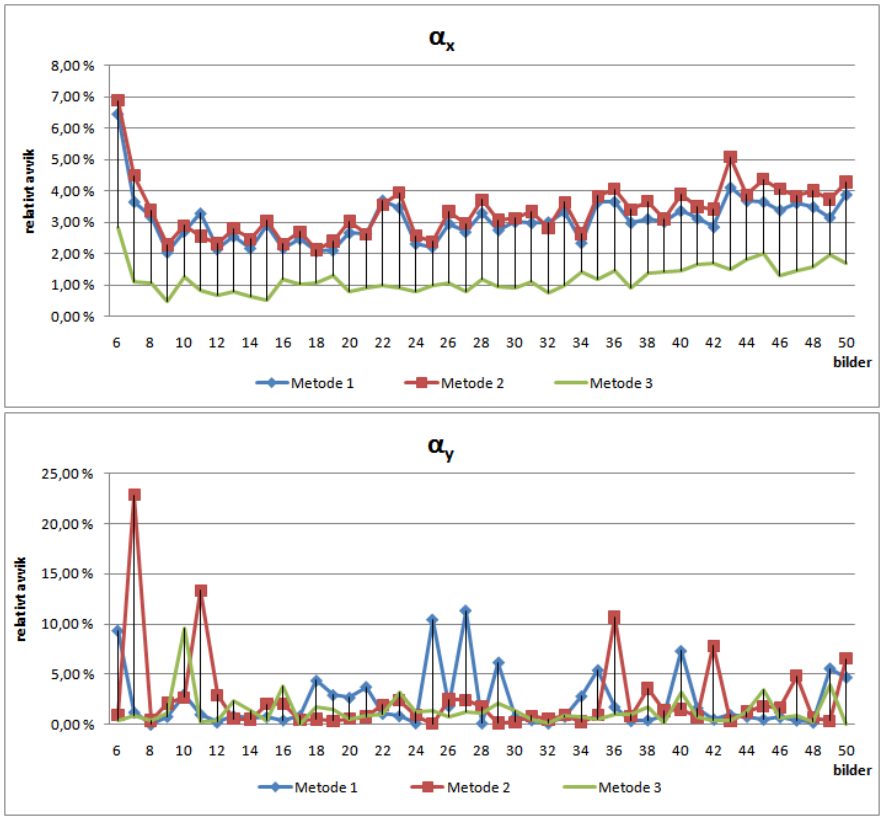
Lengden på kalibreringsobjektet er 290 mm og det faste punktet $A = [20, 220, 660]^T$. Kalibreringsobjektet genereres med tilfeldig orientering der vinklene $\theta \in [\pi/6, 5\pi/6]$ og $\phi \in [\pi/6, 5\pi/6]$ varierer tilfeldig, men uniformt distribuert. Kalibrering foretas for hvert av de 250 utvalgene med metode 1, 2 og 3. Resultatene sorteres etter parameteren α_x og medianen benyttes til sammenligning. Resultatet for de intrinsikke parametrene er plottet i figur 22 og figur 23 for intervallet $6-50$ bilder.



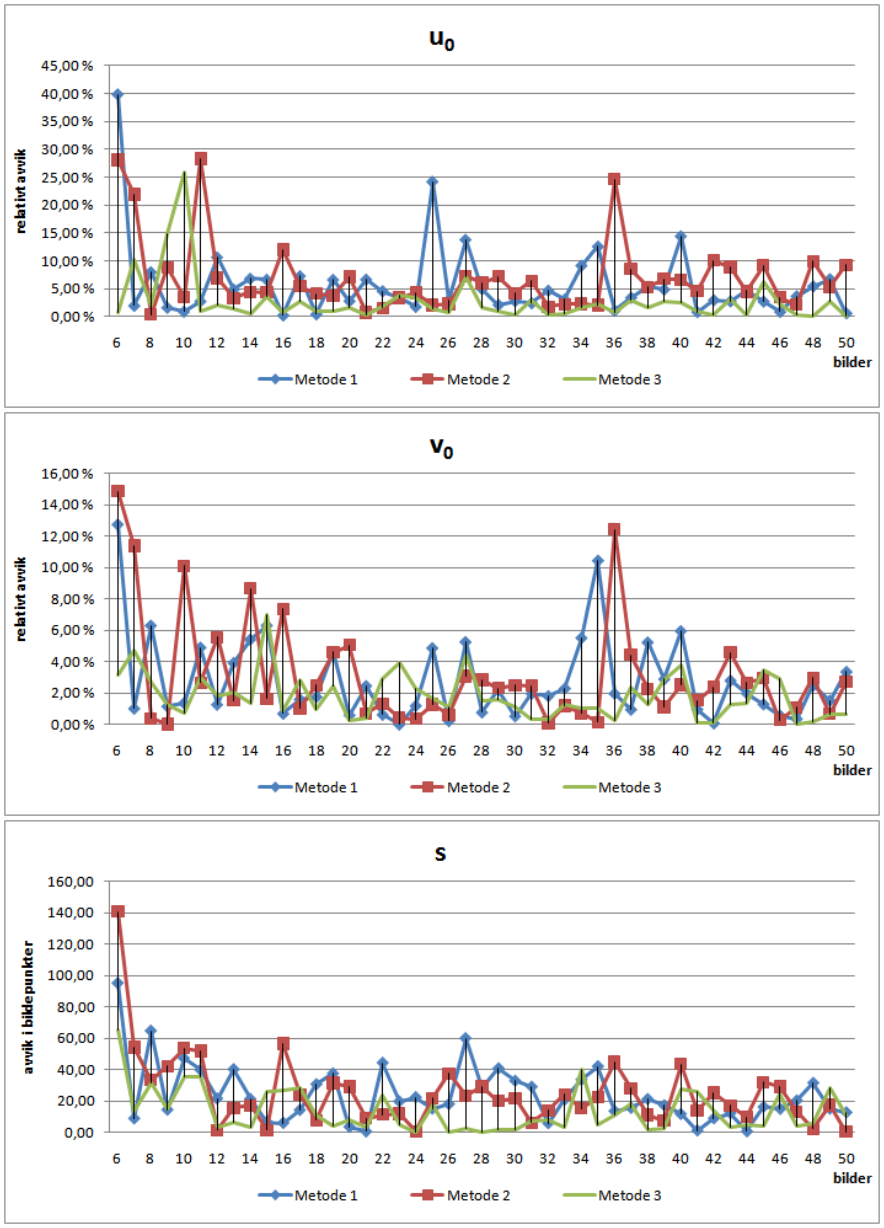
Figur 20: Resultat for parametrene α_x og α_y fra eksperiment 4.



Figur 21: Resultat for parametrene u_0, v_0 og s fra eksperiment 4



Figur 22: Resultat for parametrene α_x og α_y fra eksperiment 5.



Figur 23: Resultat for parametrene u_0, v_0 og s fra eksperiment 5

Som en kan se, synker det relative avviket for parameteren α_x under økningen fra 6 til 10 bilder før det stabiliserer seg omkring 3% opp til 35 bilder. Fra 35 bilder stiger det relative avviket igjen. Dette gjelder alle kalibreringsmetodene. Dette ligner på resultatene til Franca et. al. i [8], men økningen i relativt avvik er her mindre per ekstra bilde. For de andre parametrene er ikke resultatene like klare, dette er ikke uventet da resultatene jo er sortert etter α_x. Metode 3 utmerker seg likevel også her som den beste. Selv om det relative avviket ikke er konsekvent lavere enn for de andre metodene, leverer metode 3 jevnt over mer stabile resultater.

4 Diskusjon

Eksperiment 1 viser at sirkeldeteksjonen basert på Hough-transform fungerer etter intensjonen. Resultatet for bilde 1 i eksperimentet indikerer avvik $< 0.7\%$ som for disse bildene tilsvarer et støybidrag på < 0.8 bildepunkter. Om man overfører resultatet til oppløsningen som er brukt i eksperimentene med syntetiske bildene tilsvarer dette et avvik på < 11.2 bildepunkter. Hvis en sammenligner dette med resultatene fra eksperiment 2, ser en at en kan oppnå brukbare kalibreringsresultater, særlig med metode 3, for dette støynivået. Ved bruk av sirkeldeteksjonsalgoritmen på reelle bilder av 1-dimensjonale kalibreringsobjekt må en anta at støynivået er noe høyere. Likevel viser erfaringene at algoritmen fungerer godt med reelle bilder. Særlig imponerende er kanskje resultatet for den øverste kulen i figur 19, der sentrum og radius blir detektert selv om nesten hele kulen har lampearmen i bildet, med nesten lik gråtoneintensitet, som bakgrunn. Eksperiment 1 viser også at en ved deteksjon av større sirkler får et høyere avvik, sammenlignet med deteksjon av mindre sirkler, når støyen øker. Dette går mot antakelsen om at større sirkel burde gi bedre signal-/støyforhold og dermed mindre avvik. Forklaringen er trolig at når støyen øker blir kanten rundt sirkelen, som blir detektert med gradient (figur 4), ujevn. Dette medfører at toppen i akkumulatortabellen, se figur 12, blir lavere og bredere, noe som gjør deteksjon av sentrum mindre nøyaktig. Det antas at effekten av dette blir større for den større sirkelen fordi omkretsen er større. Dette kan sannsynligvis forbedres ved å tilpasse filterparametrene som er innebygd i Hough-transformen som sirkeldeteksjonsalgoritmen er basert på. Disse har i eksperimentet standardverdier.

Formålet med eksperiment 2 er å vise forskjellen med og uten normalisering. På bakgrunn av resultatene til Franca et. al. [8], antas det på forhånd at normalisering med metode 3 vil gi vesentlig bedre resultater enn metode 1. Metode 2 som ikke tidligere er dokumentert brukt på denne algoritmen antas å skulle ligge mellom disse målt i relativt avvik. Årsaken til at metode 2 er tatt med er for å se i hvilken grad den forbedrer resultatet, da implementeringen av denne metoden er enklere enn for metode 3. Resultatet av eksperimentet viser at forskjellen mellom metodene er mindre enn antatt. For parameteren α_x er det lite forskjell

for $\sigma < 4.5$. For $4.5 < \sigma < 10.5$ øker det relative avviket med 0.5% for metode 1, mens metode 2 og 3 forblir uforandret med relativt avvik $< 0.1\%$. Metode 1 har et vesentlig større relativt avvik fram til $\sigma = 16.5$. For $16.5 < \sigma$ øker det relative avviket raskt for alle metodene og det er vanskelig å konkludere med at noen skiller seg vesentlig ut. For de andre parametrene er det omtrent ikke forskjell mellom metodene. Resultatene svinger også mye for de andre parametrene. Dette kommer av at resultatene er sortert på parameteren α_x og at det tellende resultatet er medianen av 250 gjennomkjøringer for hvert støynivå. De store forskjellene på de andre parametrene forteller om stor variasjon i de forskjellige gjennomkjøringene. For å gi et bilde på denne variasjonen er det for α_x i tillegg til medianen også registrert øvre og nedre kvartil, dette er vist i figur 17. Figuren viser altså spredningen til de 50% "beste" resultatene. Det har ikke blitt beregnet kvartiler for de øvrige parametrene, men hvis en ser på avvikene av de andre parametrene sortert på α_x, er det rimelig å anta at spredningen er tilsvarende også for disse.

I eksperiment 3 er det benyttet reelle bilder av selvproduserte fysiske kalibreringsobjekter. Dette medfører en del feilkilder som kan ha betydning for resultatet. De viktigste er:

- Blomsterpinnen som er brukt er av tre og dermed ikke 100% rett.
- Hullene i plastperlene måtte forstørres for å passe på blomsterpinnene, dette er gjort med stativmontert boremaskin og perlene fastmontert i skrustikke, det må likevel antas at hullet avviker noe fra sentrum.
- Plasseringen til perlene er oppmålt med meterstokk.
- Det 2-dimensjonale mønsteret er skrevet ut på laserprinter og stiftet fast i en papplate. Det er derfor ikke 100% plant.
- Rutenes sidelengder er målt med meterstokk.

Dette eksperimentet er nok det viktigste for å kunne besvare problemstillingen, formålet er å avdekke om bruk av 1-dimensjonalt kalibreringsobjekt gir tilstrekkelig nøyaktighet i forhold til kalibrering med 2-dimensjonalt kalibreringsobjekt, som benyttes som referanse, til at det kan vurderes som et alternativ dersom en ønsker å utføre kamerakalibrering. Som en ser av den relative differansen mellom metode 3 og den 2-dimensjonale metoden er dette ikke tilfelle. En har riktignok et støybidrag fra sirkeldeteksjonen, samt et ukjent støybidrag fra de nevnte feilkildene, men dette er forsøkt minimert ved å være nøyaktig i tillagingen av kalibreringsobjektene. Resultatene anses her derfor som representative for kvalitetsforskjellene mellom metodene.

Eksperiment 4 har som formål å vise effekten av å utvide det 1-dimensjonale kalibreringsobjektet med flere punkter/kuler. Zhang utforsker ikke dette i sin

presentasjon av algoritmen, men antar at en økning i antall punkter vil forbedre nøyaktigheten til kalibreringsresultatet fordi flere punkter vil gi mer informasjon om kalibreringsobjektets orientering, og dermed et bedre signal-/støyforhold. Eksperimentet ville gitt et riktigere resultat dersom en hadde brukt samme orientering på kalibreringsobjektet over hele eksperimentet, fordi en da kun hadde vurdert antallet referansepunkter mot hverandre. Det ble her isteden valgt å generere nye orienteringer, da det allerede var implementert en metode for dette. Alternativet, som vil bestå av å generere kalibreringsobjektet som en rett linje, og "montere" et ulikt antall referansepunkter i etterkant ble ikke prioritert. Resultatet av eksperiment 4 er derfor påvirket av kalibreringsobjektets orientering over bildene i tillegg til antall kuler. Det kan derfor kun benyttes som en indikasjon på hvordan flere kuler påvirker kalibreringsresultatet. Det er likevel nyttig, da det viser at kalibreringsobjektets orientering, eller variasjon i orientering har betydning for kalibreringsresultatet. Resultatet er ut fra Zhangs antakelser noe overraskende og viser at en ved å øke antallet punkter på kalibreringsobjektet får en kraftig økning i det relative avviket for metode 1 og 2 for $11 <$ punkter. Dette, og det faktum at metode 3 påvirkes i liten grad, sammenfaller imidlertid med resultatene til Franca et. al.[8]. Resultatene antyder også at 5 punkter på kalibreringsobjekt gir det kollektivt beste resultatet for alle parametrene. Dette bør imidlertid undersøkes nærmere, og eksperimentet utføres med fast orientering over alle bildene.

Formålet med eksperiment 5 er å vise hvordan antallet bilder som benyttes i kalibreringen påvirker resultatet. Minstekravet for algoritmen er 6 bilder, og en ser at en ved å øke dette antallet moderat, får merkbart bedre resultater. Også her utmerker metode 3 seg som den mest stabile og med minst relativt avvik. I området fra 10 til 30 bilder er resultatene forholdsvis jevne for parameteren α_x over alle de tre metodene, mens en ved å fortsette økningen i antall bilder får en svak stigning i det relative avviket. Også dette strider mot intuisjonen om at mer data skulle gi bedre resultater, men resultatet er også her sammenfallende med resultatene til Franca et. al. For de andre parametrene ser en at resultatene er like ujevne som i tidligere eksperimenter.

4.1 Oppsummering

Det har blitt gjort eksperimenter for å fastslå om Zhangs 1-dimensjonale metode kan være et alternativ til 2-dimensjonale metoder. Metoden har blitt implementert i Matlab (metode 1). Metoden har også blitt implementert i en forbedret form som foreslått i Franca et. al. [8] (metode 3), som tar utgangspunkt i å normalisere de ekstraherte bildepunktene til kulene på kalibreringsobjektet og bruke disse normaliserte bildepunktene som inndata i Zhangs algoritme. I tillegg er det implementert en metode som baserer seg på å normalisere verdiene i matrisen $\mathbf{V}$ før denne inverteres, og resulterer i vektoren x som brukes til å finne kameraparametrene (metode 2). Metode 3 gir som forventet bedre resultater enn både

metode 1 og metode 2 over alle eksperimentene. Metode 2 gir en forbedring over metode 1 i eksperiment 2, og påvirkes i noe mindre grad av moderat støy. For eksperiment 4 og 5 gjør den resultatet verken bedre eller dårligere og påvirkes på samme måte som metode 1 av å øke antallet referansepunkter og antall bilder (hvis en ser på parameteren α_x som resultatene er sortert etter i isolasjon). I eksperiment 3, er resultatet fra metode 2 vesentlig dårligere enn for metode 1 og metode 3. Det må derfor konkluderes med at metode 3 gir de beste resultatene totalt sett, men det kunne vært interessant å gjøre forsøk med en kombinasjon av metode 2 og metode 3.

Hvis en ser på resultatene for de andre parametrene (α_y, u_0, v_0 og s) i eksperiment 2, ser en at disse varierer mye. Dette skyldes at det er store avvik i resultatene over de 250 gjennomkjøringene. Dette indikerer at algoritmen er mer støyfølsom enn resultatet tilsier når en benytter median som mål. Dette er ikke problematisert av Franca et. al. [8], som benytter samme måte å beregne resultat på. Dette er heller ikke problematisert av Zhang i [28], som benytter gjennomsnittet av 120 målinger som resultat. Disse måtene å vise resultatet på gir små relative avvik, selv om spredningen kan være stor. Dette bør derfor undersøkes nærmere.

Det er her ikke implementert forbedring vha. ulineær optimering. Dette er vist i [8] og [28] å skulle gi betydelig reduksjon i det relative avviket, samt gi bedre støytoleranse. Det er ikke rapportert om at denne metoden gir mindre spredning i kalibreringsresultatene, men siden dette heller ikke er rapportert som et problem for de andre metodene bør også dette undersøkes nærmere.

4.2 Forslag til videre arbeid

- Undersøke om Zhangs metode kan utvides til å inkludere parametre for distorsjon.

- Implementere ulineær optimering og undersøke om en får mindre spredning i kalibreringsresultatene.

- Undersøke om en ved å kombinere metode 2 og 3 kan oppnå bedre resultater enn med metode 3 alene.

5 Konklusjon

Det er bekreftet eksperimentelt at kalibrering med 1-dimensjonalt kalibreringsobjekt er mulig. Nøyaktigheten er derimot dårligere sett i forhold til populære eksisterende metoder. Tidligere publiserte arbeider tilsier at en kan få nøyaktighet tilsvarende de med 2-dimensjonale kalibreringsobjekter. Erfaringen her er at spredningen i kalibreringsresultatene er for store til at gode resultater kan oppnås

effektivt. Det konkluderes derfor, med forbeholdene som er gitt i oppsummeringskapittelet, med at 1-dimensjonalt kalibreringsobjekt ikke kan konkurrere med de allerede brukte 2-dimensjonale metodene. Unntaket er for spesialtilfellet der en ønsker å simultankalibrere et nettverk av kameraer hvor alle må se kalibreringsobjektet samtidig.

Referanser

[1] K.H. Tan A.A. Kassim, T. Tan. A comparative study of efficient generalised hough transform techniques. Image and Vision Computing, 17:737–748, 1999.

[2] D.H. Ballard. Generalizing the hough transform to detect arbitrary shapes. Pattern Recognition, 13(2):111–122, 1981.

[3] Jean-Yves Bouguet. Camera calibration toolbox for matlab. http://www.vision.caltech.edu/bouguetj/calib_doc/.

[4] D.C. Brown. Decentering distortion of lenses. Photogrammetric Engineering, 7:444–462, 1966.

[5] D.C. Brown. Close-range camera calibration. Photogrammetric Engineering, 37(8):855–866, 1971.

[6] B.Triggs. Autocalibration from planar scenes. In Proceedings of the V European Conference on Computer Vision, pages 89–105.

[7] Xiaochun Cao and Hassan Foroosh. Camera calibration without metric information using 1d objects. In IEEE International Conference on Image Processing, Singapore, October.

[8] José A. de França, Marcelo R. Stemmer, Maria B. de M. França, and Elaine G. Alves. Revisiting zhang's 1d calibration algorithm. Pattern Recognition, 43(3):1180–1187, March 2010.

[9] Fabio Remondino & Clive Fraser. Digital camera calibration methods: Considerations and comparisons. In ISPRS Commission V Symposium 'Image Engineering and Vision Metrology', pages 266–272, Dresden, Germany, September 2006.

[10] T.A. Clarke & J.G. Fryer. The development of camera calibration method and models. Photogrammetric Record, pages 51–66, 1998.

[11] C. Harris and M. Stephens. A combined corner and edge detector. In Proceedings of the 4th Alvey Vision Conference.

[12] R. Hartley. In defence of the eight point algorithm. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(6):580–593, 1997.

[13] Richard Hartley and Andrew Zisserman. Multiple View Geometry in computer vision. Cambridge university press, 2nd edition, 2003.

[14] Janne Heikkila and Olli Silven. A four-step camera calibration procedure with implicit image correction. In CVPR '97 Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition. IEEE Computer Society Washington, DC, USA ©1997.

[15] Paul Hough. Machine analysis of bubble chamber pictures. In High Energy Accelerators and Instrumentation, 1959.

[16] Michael Isard Jon Deutscher and John MacCormick. Automatic camera calibration from a single manhattan image. In Proc European Conf. Computer Vision, volume 2352 of LNCS.

[17] Quang-Tuan Luong and Olivier Faugeras. Self-calibration of a stereo rig from unknown camera motions and point correspondences. Technical Report RR-2014, INRIA, 1993.

[18] Arthur A. Magill. Variation in distortion with magnification. Journal of the Optical Society of America, pages 148–149, 1955.

[19] Tao Peng. Implementering av sirkulær hough-transform. http://www.mathworks.com/matlabcentral/fileexchange/authors/21266.

[20] Gergerly Vass & Tamás Perlaki. Applying and removing lens distortion in post production. In The Second Hungarian Conference on Computer Graphics and Geometry, Budapest, 2003. 2nd best speech.

[21] G.U. Matus J.H. Nilsen R.H. Bakken, B.G. Eilertsen. Semi-automatic camera calibration using coplanar control points. In NIK-2009.

[22] P.E. Hart R.O. Duda. Use of hough transformation to detect lines and curves in pictures. Technical report, Artificial Intelligence Center, 1971. 36.

[23] Linda G. Shapiro & George C. Stockman. Computer Vision. Prentice Hall, 2001.

[24] Roger Y. Tsai, editor. An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision, 1986.

[25] Roger Y. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. IEEE journal of robotics and automation, RA-3(4):323–344, August 1987.

[26] Reg G. Wilson. Modeling and calibration of automated zoom lenses. Technical report, 3M Engineering Systems and Technology, http://vasc.ri.cmu.edu/IUS/usrp2/rgw/www/spie94.pdf, 1999.

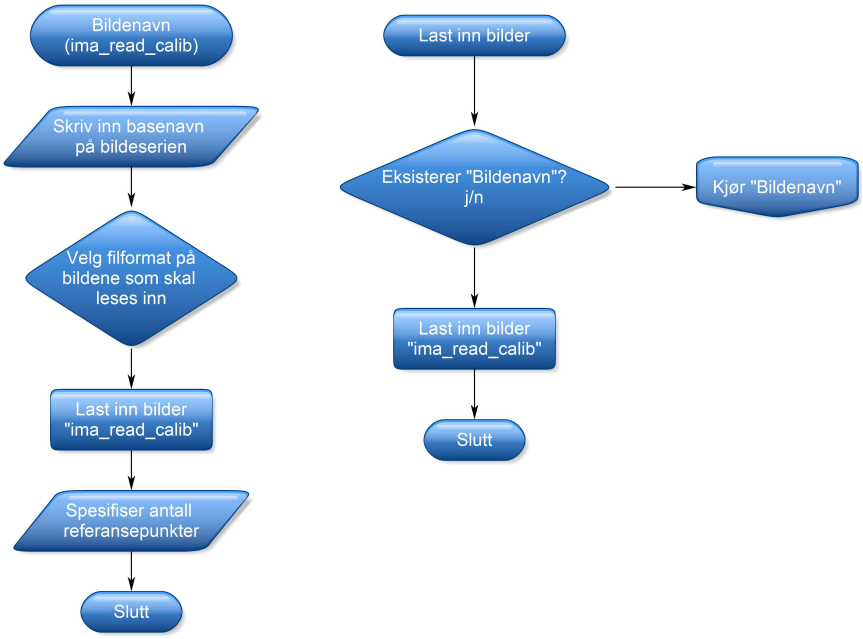
[27] Zhengyou Zhang. A flexible new technique for camera calibration. Technical report, Microsoft Research, 1998.

[28] Zhengyou Zhang. Camera calibration with one-dimensional objects. IEEE transactions on pattern analysis and machine intelligence, 26(7):892–899, 2004.

[29] Zi-jian Zhao and Yun-cai Liu. New multi-camera calibration algorithm based on 1d objects. Journal of Zhejiang University - Science A, 9:799–806, 2008.

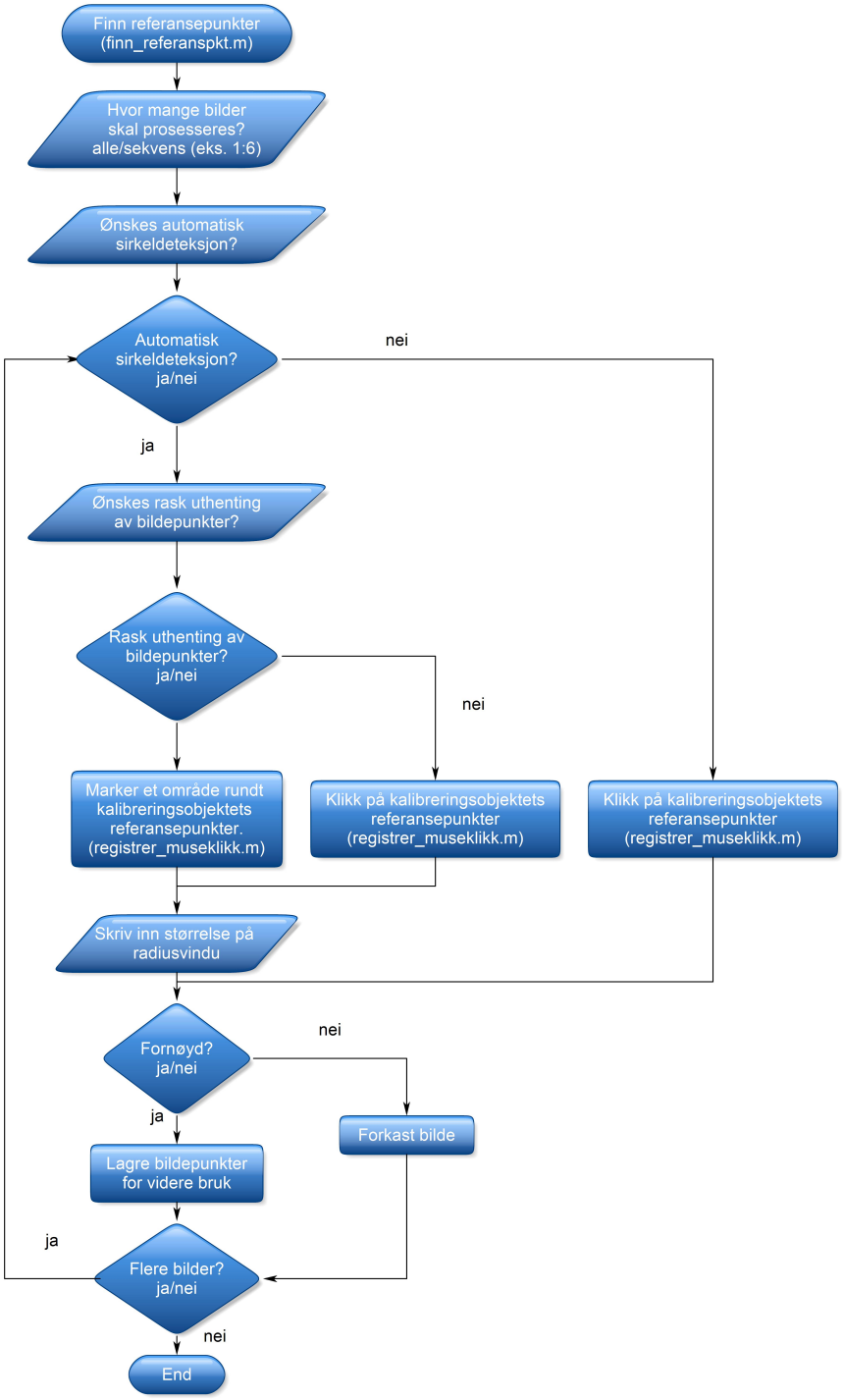
A Implementering

Programflyt vist med flytskjema.



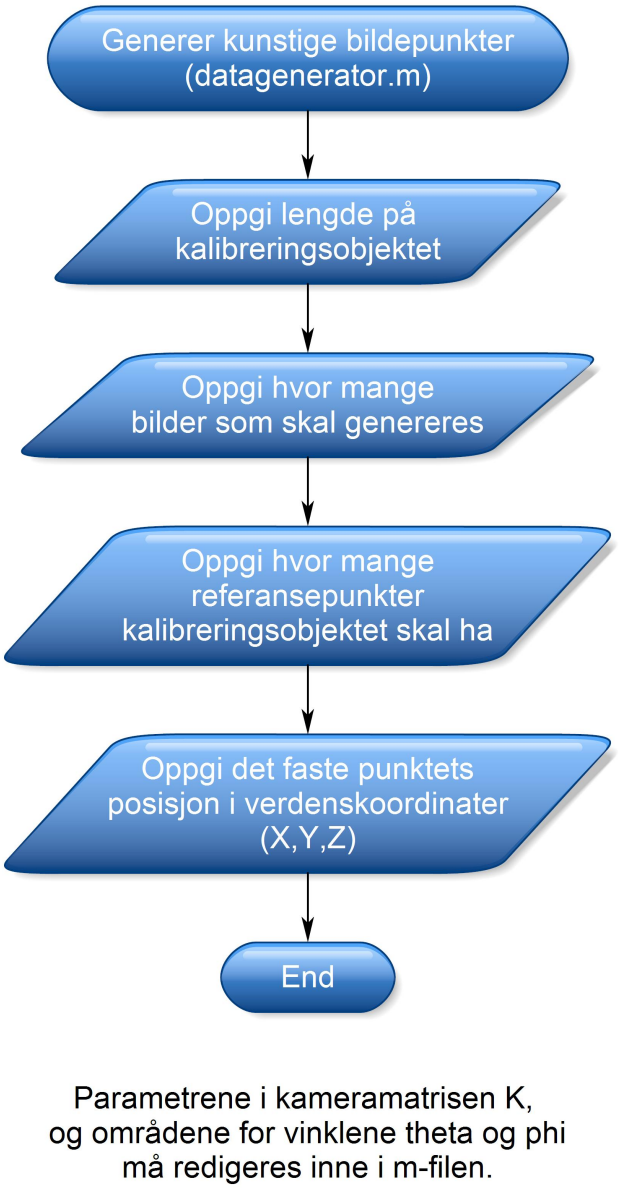
Figur 24: Flytskjema for knappene “Bildenavn” og “les inn bilder” på kontrollpanelet

-



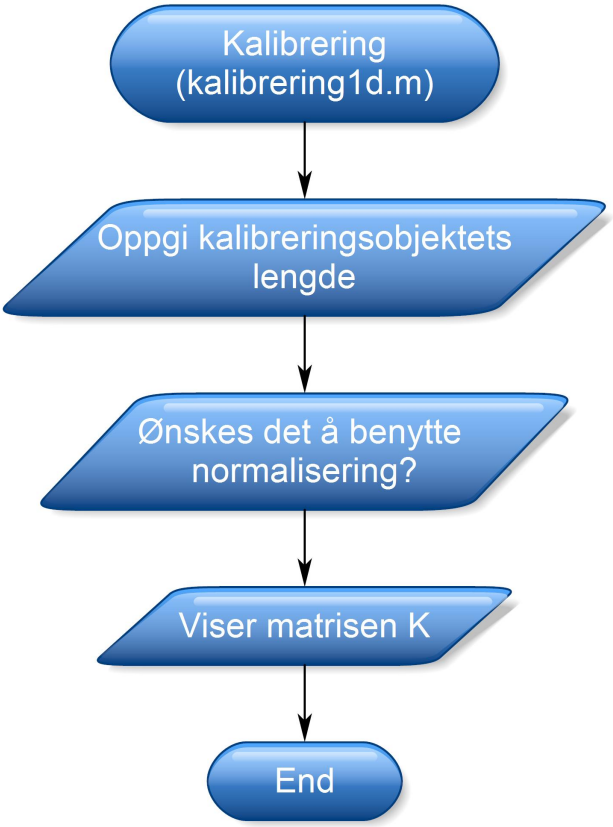
Figur 25: Flytskjema for knappen "Finn referansepunkter" på kontrollpanelet

-



Figur 26: Flytskjema for knappen "Generer kunstige bildepunkter" på kontrollpanelet

-



Figur 27: Flytskjema for knappen "Kalibrering" på kontrollpanelet

-

B Kode

Dette kapittelet inneholder koden som er skrevet i forbindelse med oppgaven. Koden er bygget på "Camera calibreation toolbox for Matlab", denne er skrevet på engelsk. Funksjoner og filer derfra har engelske betegnelser og er ikke tatt med her, men kall til disse forekommer, dette gjelder stort sett basisfunksjonalitet som innlesing av og behandling av bildefiler. Det er i tillegg skrevet en del rutiner for gjennomføring av eksperimentene, disse er lagt ved på CD.

B.1 Kontrollpanelet

```
%function calib_gui_1d
```

```
cell_list = {};


fig_number = 1;

title_figure = 'Kamerakalibrering med Zhangs metode for 1-D kalibreringsobjekt';

clear fc cc kc KK
kc = zeros(5,1);
clearwin;

cell_list{1,1} = {'Bildenavn','image_names;'};
cell_list{1,2} = {'Les inn bilder','ima_read_calib;'};
cell_list{2,1} = {'Finn referansepunkter','finn_referanspkt;'};
cell_list{2,2} = {'Kalibrering','kalibrering1d;'};
cell_list{3,1} = {'Generer kunstige bildepunkter', 'datagenerator;'};
cell_list{3,2} = {'Legg til stoey paa referansepunktene', 'legg_til_stoy;'};
cell_list{4,1} = {'Legg til stoey paa de innleste bildene', 'legg_til_bildestoy;'};
cell_list{4,2} = {'Slett alle data', 'clear;'};

show_window(cell_list,fig_number,title_figure,200,18,0,'clean',12);
```

B.2 Finn referansepunkter

```
%function finn_referanspkt.m
%
% Metode for aa finne ekstrahere referansepunktene
%
% Brukeren klikker saa naer sentrum som mulig, de markerte koordinatene
% sammenlignes med en resultatet fra en automatisk sirkelsentrumdeteksjon,
% omraadet rundt "klikket" sammenlignes med denne, og en benytter naermeste
% detekterte sentrum.

%if exist('images_read');
%   active_images = active_images & images_read;
%end;


var2fix = 'dX_default';

fixvariable;

var2fix = 'dY_default';

fixvariable;

var2fix = 'map';

fixvariable;
```

```
if ~exist('antrefpkt')
    antrefpkt = input('antall referansepunkter paa kalibreringsobjektet?')
end

if ~exist('radmin')
    radmin = 0;
end

if ~exist('radmax')
    radmax = 0;
end

if ~exist('n_ima'),
    image_names;
end;

check_active_images;

if ~exist(['I_' num2str(ind_active(1))]),
    ima_read_calib;
    if isempty(ind_read),
        disp('Kan ikke finne referansepunkter uten bilder');
        return;
    end;
end;

% spoer hvor mange av de innleste bildene som skal brukes

ima_numbers = input('Hvor mange bilder skal prosesseres? ([] = alle bilder) = ');


if isempty(ima_numbers),
    ima_proc = 1:n_ima;
else
    ima_proc = ima_numbers;
end;
offset = 0;
kk=ima_proc;
auto = input('oensker du aa benytte automatisk sirkeldetektor? j/n: ', 's')
roi.j = input('oensker du rask uthenting av bildepunkter?'...
'(Anbefales for store bilder) j/n: ', 's')
%aapner ett og ett bilde for aa velge referansepunktene med musepekeren
for kk = ima_proc,
    if exist(['I_' num2str(kk)]),

        registrer_museklikk;
        %sirkeldeteksjon
        %lagre senterpunkter
        active_images(kk) = 1;

    else
        eval(['dX_' num2str(kk) ' = NaN;']);
```

```
        eval(['dY_' num2str(kk) ' = NaN;']);

        %eval(['wintx_' num2str(kk) ' = NaN;']);
        %eval(['winty_' num2str(kk) ' = NaN;']);

        eval(['x_' num2str(kk) ' = NaN*ones(2,1);']);
        eval(['X_' num2str(kk) ' = NaN*ones(3,1);']);

        %eval(['n_sq_x_' num2str(kk) ' = NaN;']);
        %eval(['n_sq_y_' num2str(kk) ' = NaN;']);
    end;
end;
clear offset
%check_active_images;
```

B.3 Registrer museklikk

```
%function registrer_museklikk.m
%
%Lar brukeren trykke paa
%referansepunkter med musen i bildet. Bildekoordinatene lagres for videre
%behandling
fprintf(1,'\nProsesserer bilde %d...\n',kk);

eval(['I = I_' num2str(kk) ';']);


if ~exist('offset')
    offset = 0;
end

figure(2);
image(I);
colormap(map);
set(2,'color',[1 1 1]);



if roi.j == 'j'
    title(['Marker kalibreringsobjektets punkter med musen (Viktig:'...
        'Marker det faste punktet foerst og fortsett utover)... Bilde '...
        num2str(kk)]);
    disp('Marker punktene paa kalibreringsobjektet med musen, trykk'...
        'for aa starte merking, dra firkanten over punktet og slipp.'...
        'Proev aa sentrer punktet i firkanten for best mulig resultat');
    for k = 1:antrefpkt
        rect{k} = getrect
    end
else
```

```
x= [];y = [];
figure(2); hold on;
%for count = 1:antrefpkt,
    [xi,yi] = ginput(antrefpkt);
    %[xxi] = cornerfinder([xi;yi],I,winty,wintx);
    %xi = xxi(1);
    %yi = xxi(2);
    figure(2);
    plot(xi,yi,'+','color',[ 1.000 0.314 0.510 ],'linewidth',2);
    x = [x;xi];
    y = [y;yi];
    plot(x,y,'-','color',[ 1.000 0.314 0.510 ],'linewidth',2);
    drawnow;
%end;
plot([x;x(1)],[y;y(1)],'-','color',[ 1.000 0.314 0.510 ],'linewidth',2);
drawnow;
hold off;
Klikk =[xi, yi];
end

if auto == 'j' && roi.j == 'n'
title(['Klikk paa kalibreringsobjektets punkter(Viktig: klikk paa det'...
    'faste punktet foerst)... Bilde ' num2str(kk)]);
disp('Klikk paa punktene paa kalibreringsobjektet');
%finner sirkler i bildet automatisk vha hough


radmin_sjekk = input(['Nedre grense radius (piksler) ([] = '...
    num2str(radmin) ') = ']);
if isempty(radmin_sjekk)
    clear radmin_sjekk;
else
    radmin = radmin_sjekk;
end

radmax_sjekk = input(['Ovre grense radius (piksler) ([] = '...
    num2str(radmax) ') = ']);
if isempty(radmax_sjekk)
    clear radmax_sjekk;
else
    radmax=radmax_sjekk;
end
tic
[accum, circen, cirrad] = CircularHough_Grd(I, [radmin radmax]);
disp(size(circen,1))
toc
%sammenligner de automatisk detekterte sirklene og finner den som er naermest hvert
tic
dpos = 100000*ones( size(Klikk,1), 3 );
for i = 1:size(Klikk,1)
    for k = 1 : size(circen, 1)
        d=abs(Klikk(i, 1)- circen(k,1))+abs(Klikk(i,2)-circen(k,2));
```

```
        if d < dpos(i,1)
            dpos(i,1) = d;
            dpos(i,2) = i;
            dpos(i,3) = k;
        end
    end
end
toc
% dpos => [avstand, pos i klikk, pos i circen]
%disp(dpos);

%Plotter sirklene som samsvarer med de markert av brukeren
figure(2); clf; imagesc(I); colormap('gray'); axis image;
 hold on;


 for k = 1 : size(Klikk, 1),
     plot(circen(dpos(k,3),1), circen(dpos(k,3),2), 'r+');
     DrawCircle(circen(dpos(k,3),1), circen(dpos(k,3),2), cirrad(k), 32, 'b-');


      happy = input('Lagre punkter for videre bruk? (j/n) []=j :', 's')

    if happy == 'n';
        offset = offset + 1;
    else

     %lagrer bildepunktene for videre bruk, hvis resultatet godkjennes
     BP{kk-offset}(k,1)=[circen(dpos(k,3),1)];
     BP{kk-offset}(k,2)=[circen(dpos(k,3),2)];
    end




 end
 pause = input('Trykk en tast for aa gaa videre...', 's')
 hold off;

elseif auto == 'j' && roi.j == 'j'
    radmin_sjekk = input(['Nedre grense radius (piksler) ([] = '...
        num2str(radmin) ') = ']);
if isempty(radmin_sjekk)
    clear radmin_sjekk;
else
    radmin = radmin_sjekk;
end

radmax_sjekk = input(['Ovre grense radius (piksler) ([] = '...
    num2str(radmax) ') = ']);
if isempty(radmax_sjekk)
    clear radmax_sjekk;
```

```
else
    radmax=radmax_sjekk;
end
 tic
for k = 1:antrefpkt
        rect{k}=round(rect{k});

% Tar ut delbildet for behandling
I_temp = [I(rect{k}(1,2):(rect{k}(1,2)+(rect{k}(1,4))), rect{k}(1,1):...
    (rect{k}(1,1)+rect{k}(1,3)))];


%Finner sirklene i delbildet

[accum, circen, cirrad] = CircularHough_Grd(I_temp, [radmin radmax]);


midten.x = size(I_temp, 2)/2;
midten.y = size(I_temp, 1)/2;

%Bruker sirkelsentrum som er naermest  senter av det merkede rektangelet
dpos = 100000*ones( 1, 2 );
    for i = 1:size(circen, 1)
        d=abs(midten.y- circen(i,2))+abs(midten.x-circen(i,1));
        if d < dpos(1,1)
            dpos(1,1) = d;
            dpos(1,2) = i;
        end

    end

    if size(circen,2) >= 1
    %naermeste sentrum
    ns = [circen(dpos(1,2),1),  circen(dpos(1,2),2)];

    % Setter senterpunktene funnet fra delpunktene tilbake i hovedbildet
    ns(1) = ns(1) +  rect{k}(1,1);
    ns(2) = ns(2) +  rect{k}(1,2);

    % plotter resultatet
    hold on
     plot(ns(1), ns(2), 'r+');
     DrawCircle(ns(1), ns(2), cirrad(1), 32, 'b-');
    hold off

  BP_temp(k,1)=ns(1);
  BP_temp(k,2)=ns(2);

    else
        disp('Sirkeldeteksjon feilet')
    end
end
```

```
    happy = input('Lagre punkter for videre bruk? (j/n) []=j :', 's')

    if happy == 'n';
        offset = offset + 1;
    else
    %lagrer bildepunktene for videre bruk hvis resultatet godkjennes
    for k = 1:antrefpkt
     BP{kk-offset}(k,1)=BP_temp(k,1);
     BP{kk-offset}(k,2)=BP_temp(k,2);
    end
    end
    clear BP_temp
    toc

else

    BP{kk}=Klikk;

end

 disp(BP(kk-offset));
return
```

B.4 Kalibreringsrutinen

```
%function kalibrering1d.m
%
%Andreas Kristiansen 2011
%
%
%Programmet benytter Zhangs metode for kamerakalibrering vha. 1
%dimensjonalt kalibreringsobjekt.

if ~exist('antrefpkt')
    antrefpkt = ...
        input('hvor mange referansepunkter paa kalibreringsobjektet?')
end
%Lengde paa kalibreringsobjektet
L = input('Kalibreringsobjektets lengde [mm]:');


%beregner saa avtandsforholdet fra endepunkt for hvert punkt mellom
%endepunktene, gamma_A og gamma_B.
for k = 2:(antrefpkt-1)
gamma_A{k-1}=(antrefpkt-k)/(antrefpkt-1);
gamma_B{k-1}=(1-gamma_A{k-1});
end

%estimerer det faste punktets posisjon i bildepunkter som gjennomsnittet av
```

```
%alle observasjoner
a_2=[0; 0; 0];
for k = 1:size(BP,2)
    a_2 = a_2 + [BP{k}(1,1); BP{k}(1,2); 1];
end

a_2= a_2/size(BP,2);

norm = input('oensker du aa benytte normalisering? j/n', 's');
if norm == 'j'
BP2 = BP; %lagrer de originale bildepunktene i tilfelle...
clear BP;
[BP T2] = normalisering1(BP2, antrefpkt);
%normalisering;

end


%estimerer det faste punktets posisjon i bildepunkter som gjennomsnittet av
%alle observasjoner
a_=[0; 0; 0];
for k = 1:size(BP,2)
    a_ = a_ + [BP{k}(1,1); BP{k}(1,2); 1];
end

a_= a_/size(BP,2);

%beregner h-vektoren for hvert bilde



for k = 1:size(BP,2) %antall registrerte bilder
    b_ = [BP{k}(size(BP{k},1)); BP{k}(size(BP{k},1),2); 1];
    clear m_;
    %genererer de midterste punktene
    for i = 2:(size(BP{k},1)-1)
    m_{i-1} = [BP{k}(i,1); BP{k}(i,2); 1]    ;
    end


    %gjoer klar for aa beregne gj.snitt
    kryss_ = 0;
        for i = 1:size(m_,2)
        kryss_ = ((gamma_A{i}*(cross(a_,m_{i}))'*(cross(b_, m_{i})))/...
            (gamma_B{i}*(cross(b_, m_{i})'*(cross(b_,m_{i})))));
        h{k,i} = a_ + (kryss_*b_) ;
        %kryss_ = ((gamma_A{i}*(cross(a_,m_{i}))'*(cross(b_, m_{i})))/...
        %(gamma_B{i}*(cross(b_, m_{i})'*(cross(b_,m_{i})))));
        %disp(kryss_);
        end

   %Mellomregning for aa beregne z_B senere
```

```
   midtpunkt=round(size(m_,2));
   z_Btemp{k}=((gamma_A{midtpunkt}*(cross(a_,m_{midtpunkt}))'*...
       (cross(b_, m_{midtpunkt})))/(gamma_B{midtpunkt}*...
       (cross(b_, m_{midtpunkt})'*(cross(b_,m_{midtpunkt}))));
   endepunkt{k} = b_;
  end



    %lager v_
for k = 1:size(BP,2)
    for i = 1:size(h,2)
    v_{k, i} = [(h{k,i}(1,1)*h{k,i}(1,1));
              2*(h{k,i}(1,1)*h{k,i}(2,1));
                  h{k,i}(2,1)*h{k,i}(2,1);
              2*(h{k,i}(1,1)*h{k,i}(3,1));
              2*(h{k,i}(2,1)*h{k,i}(3,1));
                 h{k,i}(3,1)*h{k,i}(3,1)];
    end
end

%v_ dersom en forutsetter 0 skjevhet

%       v_{k, 1} = [(h{i}(1,1)*h{i}(1,1));
%                                        0;
%                    (h{i}(2,1)*h{i}(2,1));
%                   2*h{i}(1,1)*h{i}(3,1);
%                 2*(h{i}(2,1)*h{i}(3,1));
%                    h{i}(3,1)*h{i}(3,1)];


%Lager V_
V_ = [];
for i = 1:size(v_,2)
    for k = 1:size(v_,1)

    V_(k+(size(BP,2)*(i-1)), :)= [v_{k,i}];

    end
end

% normaliserer V_, dersom valgt
% if norm == 'j'
% if any(V_(:))
%
% minimum = min(min(V_))
% maximum = max(max(V_))
% M = 1 % Tall aa nomalisere til.
% slope = (M-1) / (maximum - minimum);
% NV_ = slope * (V_ - minimum)
% minimum_new = min(min(NV_)) %  0
% maximum_new = max(max(NV_)) % 1
```

```
%Alternativ metode for aa normalisere V_
%     if any(V_(:))
%
%       NV_ = ((V_ - min(V_(:))) ./ (max(V_(:))-min(V_(:))) + eps) ./...
%           (1 + 2*eps);
%
%     else
%      NV_ = repmat(1/2, size(V_));
%     end
% end

%Lager l_
for k = 1:size(v_,2)*size(v_,1)
l_(k, 1)=L^(2);
end

%beregner x_
x_= [];
for k = 1:size(v_,2)

   % if norm == 'j'

 %  x_(:, k)=pseudoinverse(NV_)*l_;

 %  else

     x_(:, k)=pseudoinverse(V_)*l_;

  %  end
end

%Loeser for parametrene i kameramatrisen

 v_0 = ((x_(2)*x_(4))-(x_(1)*x_(5)))/(x_(1)*x_(3)-(x_(2)*x_(2)));
 z_A = sqrt(x_(6)-(x_(4)^(2)+v_0*(x_(2)*x_(4)-x_(1)*x_(5)))/x_(1));
 alpha_x = sqrt((z_A^(2)/x_(1)));
 alpha_y = sqrt(z_A^(2)*x_(1)/((x_(1)*x_(3))-(x_(2)^(2))));
 s= -(x_(2))*alpha_x^(2)*alpha_y/z_A^(2);
 u_0 = s*v_0/alpha_x-x_(4)*alpha_x^(2)/z_A^(2);

%Lager kameramatrisen
K_hatt=[alpha_x, s, u_0; 0, alpha_y, v_0; 0, 0, 1]
if norm == 'j'
K = inv(T2)*K_hatt
%BP = BP2;
else
    K=K_hatt
end
%Finner scenepunktene for de registrerte bildepunktene
%Disse kan brukes som inngangsestimat i optimeringsrutine
%Det faste punktet A
```

```
 A_ = (z_A)*inv(K)*a_2;
 A_ = [A_(1); A_(2); A_(3)]

%Alle endepunktene B
for k = 1:size(BP,2)
        z_B{k} = -z_A*z_Btemp{k};
        B_{k} = z_B{k}*inv(K)*endepunkt{k};

%Alle punktene mellom A og B
    for i=1:antrefpkt-2
        M_{i, k} = gamma_A{i}*A_+gamma_B{i}*B_{k};
    end
end
BP=BP2;
```

B.5 Rutine for å generere syntetiske bildepunkter

```
%function datagenerator.m

%Generer et datasett for kalibreringsrutinen
clear BP
clear SP
L_spor = [];
bilder_spor = [];
antrefpkt_spor = [];
opplosning_x=1600;
opplosning_y=1200;
%Lengden
while isempty(L_spor);
L_spor = input('kalibreringsobjektets lengde :');
   L=L_spor;
end

while isempty(bilder_spor)
bilder_spor = input('antall bilder som skal genereres :');
bilder=bilder_spor;
end
clear bilder_spor,
clear L_spor;

%Antall referansepunkter paa kalibreringsobjektet
while isempty(antrefpkt_spor);
antrefpkt_spor = input('Hvor mange punkter paa kalibreringsobjektet?: ')
   antrefpkt = antrefpkt_spor;
end
clear antrefpkt_spor;

%Koordinater til det faste punktet
x_pos_default = 20;
```

```
y_pos_default = 220;
z_pos_default = 660;

%Lar brukeren bestemme koordinatene til det faste punktet A
x_pos = input(['X-koordinater for det faste punktet A ([] = '...
    num2str(x_pos_default) ') = ']);
    if isempty(x_pos)
        x_pos = x_pos_default;
    end


y_pos = input(['Y-koordinater for det faste punktet A ([] = '...
    num2str(y_pos_default) ') = ']);
    if isempty(y_pos)
        y_pos = y_pos_default;
    end


z_pos = input(['Z-koordinater for det faste punktet A ([] = '...
    num2str(z_pos_default) ') = ']);
    if isempty(z_pos)
        z_pos = z_pos_default;
    end



figure(2), clf
hold off
clear PLOTT
clear SP
clear M
clear BP
for k = 1:bilder

%Vinkelen til z-aksen
phi =  random('unif', pi/6, 5*pi/6);

%Vinkelen til x-aksen
theta =  random('unif', (pi/6)-pi, (5*pi/6)-pi);

%sammenhengen mellom vinkler og verdenskoordinater

x_v2= (L/(antrefpkt-1)) * sin(theta)*cos(phi);
y_v2= (L/(antrefpkt-1)) * sin(theta)*sin(phi);
z_v2= (L/(antrefpkt-1)) * cos(theta);


%posisjonen til det faste punktet
M{1} = [x_pos, y_pos, z_pos];
%De andre punktene beregnes ut fra lengde og vinkel
for i = 1:antrefpkt-1
M{i+1} = [M{0+i}(1)+(x_v2), M{0+i}(2)+(y_v2), M{0+i}(3)+(z_v2)];
```

```
%M3 = [M1(1)+x_v, M1(2)+y_v, M1(3)+z_v];
%M2 = [(M3(1)+M1(1))/2, (M3(2)+M1(2))/2, (M3(3)+M1(3))/2]
end

for i = 1:size(M,2)
PLOTT{k}(i,1:3) = M{i};
end


SP{k} = PLOTT{k};


plot3(PLOTT{k}(:,1), PLOTT{k}(:,2), PLOTT{k}(:, 3), 'ro-')
%plot3(PLOTT{k}(:,3), PLOTT{k}(:, 2), PLOTT{k}(:,1), 'ro-')
hold on
for k=-15:0
plot3([0; 0; 0],[0; 0; 0],[0; 0; k], 'bo-')
%plot3([0; 0; 0],[0; 0; 0],[0; 0; 0], 'bo-')
end
plot3([0; 0; 0],[0; 0; 0],[0; 0; z_pos], 'bx-')
grid on
xlabel('x-akse')
ylabel('y-akse')
zlabel('z-akse')
end



%Definerer et imaginaert kamera med intrinsikke parametre
alpha_x_gen = 1100;
alpha_y_gen = 1100;
s_gen = 0;
u_0_gen = 800;
v_0_gen = 600;

K_gen = [alpha_x_gen, s_gen, u_0_gen; 0, alpha_y_gen, v_0_gen; 0, 0, 1]

%Definerer de ekstrinsikke parametrene, som Zhang
R = [1 0 0; 0 1 0; 0 0 1];
t = [0; 0; 0];

%Tar bilder av det genererte datasettet. Kameraet har intrinsikke parametre
%gitt ved matrisen K_gen.
figure(3), clf
for k = 1:size(SP,2)
   for i = 1:antrefpkt
m{i} = K_gen*[R t]*[SP{k}(i,1);SP{k}(i,2);SP{k}(i,3);1];

%skalerer bildepunktene
m{i}=m{i}/(m{i}(3));
```

```
%Kopierer bildepunktene inn i bildepunktsmatrisen for kalibreringsrutinen
BP{k}(i, 1:2) = [m{i}(1), m{i}(2)];
plot(BP{k}(:,1), BP{k}(:,2), 'mo-')
hold on
axis([0 opplosning_x 0 opplosning_y])
axis ij
   end
end
```