UNIVERSITETET I AGDER

Fartøy til fartøy lastoverføring

Øyvind Bergby Handeland

Veileder
Morten Ottestad

Masteroppgaven er gjennomført som ledd i utdanningen ved Universitetet i Agder og er godkjent som del av denne utdanningen. Denne godkjenningen innebærer ikke at universitetet innestår for de metoder som er anvendt og de konklusjoner som er trukket.

Universitetet i Agder, 2015
Ingeniøravdelingen
Fakultetet for teknologi og realfag

Summary

In the offshore industry there is a need to compensate motions caused by waves and wind. It is important if loading is to be transferred from one vessel to another in a safe and efficient manner. To make this happend, the relative movement between the two vessels must be measured. A common way for doing this is to use a MRU (Motion Reference unit). This measures the orientation of boat related to world coordinates.

This assignement is based on a scenario where two vessels are in motion related to the world coordinate system asynchronously relative to each other. A MRU on one of the vessels will not be able to measure this movement on it's own, but must be connected to a common control system along with a MRU on the second vessel. Linking two vessels together with a physical wire is neither safe or convenient.

As a substitute for the MRU system it was made a passive vision system for measuring the relative movement. The system operates by having a camera attached to one of the vessels, and a pattern in the form of a chessboard to a plan surface on the other vessel. The camera acquires images of the familiar pattern. The image processing and homography is used in LabVIEW to calculate the relative pose between the camera and the plan surface.

It is shown that this system is operating correctly, but the measurements are vulnerable to noise and small changes in the ambient light. For this assignement the problem statement has been to create a vision system that is more robust for changes in light conditions and noise.

Two new vission systems were made for this master thesis. One of them is a passive vision system that incorporates much of the same theory that the previously developed system. The pattern which is analyzed in the new system is concentric contrasting Circles(CCC). The second is an active vision system where four laser pointers projecting a pattern is analyzed. The latter system is using trigonometry to calculate the heave, roll and pitch.

For testing the systems, it was built a platform with the 4 lasers and a camera centered in the middle. Static tests were conducted for the three vision systems using this platform.

The noise were defined using standard deviation. A circle with a diameter of 40mm had a noise level about 24% relative to a corner. The noise level of a laser dot was measured to 140% relative to a corner. The results indicated that the circle is most robust with respect to noise.

The 3 systems were tested with 5 different light intensities. The corner system managed to calculate the pose for 1 of the 5 lighting conditions. When the lightning conditions became too dark or too bright, this system failed to calculate the pose. The system analyzing concentric circles calculated pose for the 4 brightest lightning conditions, and the laser system for the 3 darkest lighting conditions.

Together, the two new systems will cover all of the 5 lightning conditions that have been tested, and they will also overlap in two of them. It will then be possible to get a complementary system by merging these two systems. The complementary system will be able to measure results for all of the 5 lighting conditions, and be redundant for the conditions overlapping. For this system the most convenient pattern to be analyzed must be desided at any given time. This can be determined by measuring the average grayscale value in the image up against two thresholds. In case the average grayscale value is smaller than the smallest threshold, the lazer dots will be analyzed. In case it is larger, both the patterns are analyzed. For an average grayscale value larger then the largest threshold, only the circles are analyzed.

Sammendrag

I offshoreindustrien er det behov for å kompensere bevegelser forårsaket av sjø og vind. Det er viktig dersom last skal overføres fra et fartøy til et annet på en trygg og effektiv måte. Den relative bevegelsen mellom de to fartøyene må da måles. En MRU(motion reference unit) benyttes vanligvis for å måle orienteringen av båten relatert til verdenskoordinater.

Det er tatt utgangspunkt i et scenario der 2 fartøyer er i bevegelse relatert til verdens koordinatsystem, asynkront i forhold til hverandre. En MRU på ett av fartøyene vil ikke kunne måle denne bevegelsen alene, men må kobles til et felles kontrollsystem sammen med en MRU på det andre fartøyet. En fysisk kobling mellom to fartøyer er hverken praktisk eller trygt.

Som en erstatning for MRU-systemet ble det laget et passivt vision system for å måle den relative bevegelsen. Systemet fungerer ved at et kamera festes på det ene fartøyet, og et mønster i form av et sjakkbrett til en plan flate på det andre fartøyet. Kameraet henter inn bilder av det kjente mønsteret. Bildebehandling og homografi benyttes i LabVIEW for å regne ut den relative posituren mellom kamera og den plane flaten.

Det er vist at dette systemet fungerer, men at målingene er utsatt for støy, og lysforholdene kan ikke variere nevneverdig. Problemstillingen i denne oppgaven har vært å lage et vision system som er mer robust for varierende lysforhold og støy.

Det ble laget 2 nye vision systemer. Det ene er et passivt vision system som bygger på mye av den samme teorien som det tidligere utviklet systemet. Mønsteret som analyseres i det nye er konsentriske kontrasterende sirkler(CCC), mot tidligere et sjakkbrett. Det andre er et aktivt vision system der fire laserpekere projiserer et mønster som analyseres. Det sistnevnte systemet benytter trigonometri til å beregne heave, roll og pitch.

Det ble bygget en plattform med de 4 laserne og et kamera sentrert i midten. Ved å sette 4 bein på plattformen ble denne benyttet som testbenk, og statiske tester ble utført på de 3 vision systemene.

Standardavvik ble benyttet for å angi støy. En sirkel med diameter 40mm hadde et støynivå på rundt 24% i forhold til et hjørne. Støynivået på en laserprikk ble målt til 140% i forhold til et hjørnet. Resultatene viste følgelig at sirkelen er mest robust med tanke på støy.

De 3 systemene ble testet under 5 forskjellige lysintensiteter. Hjørne-systemet klarte å beregne positur for 1 av de 5 lysforholdene. Det taklet ikke at det ble for mørkt eller for lyst. Systemet som analyserer konsentriske sirkler beregnet positur for de 4 lyseste, og laser-systemet for de 3 mørkeste lysforholdene.

De to nye systemene ville altså tilsammen dekke alle de 5 lysforholdene. De ville også til en viss grad overlappe hverandre ved at begge fungerte for lysforhold 2 og 3. Systemene vil derfor kunne slås sammen til et komplementært system som da vil fungere for alle de testede lysnivåene og være redundant for de overlappende. I det komplementære systemet må det velges hvilket mønster som er mest fornuftig å analysere til enhver tid. Dette kan gjøres ved å sammenligne 2 terskelverdier med den gjennomsnittlige gråtoneverdien i bildet. Terskelverdiene kan settes til en høy og en lav gråtoneverdi. Dersom den gjennomsnittlige gråtoneverdien er lavere enn den lave terskelverdien vil laserprikkene analyseres. Dersom den er høyere vil begge mønstrene analyseres, og dersom den er høyere enn den høyeste terskelverdien vil kun sirklene analyseres.

Forord

Denne oppgaven er skrevet som en del av den toårige masterutdanningen innen mekatronikk på universitetet i Grimstad. Skribenten er takknemlig for å ha fått lov til å jobbe med en interessant og relevant problemstilling.

Oppgaven er en videreutvikling av masteroppgaven *Development of vision-based measurement system for relative motion compensation* av Johan Lindal Haug.

En stor takk går til veileder under gjennomføringen av oppgaven, Morten Ottestad. En ryddig oppfølging med faste møter hver uke er blitt utført. Det er også vært gode muligheter for veiledning dersom det trengtes på tidspunkter utenfor møtene også.

En takk går også til Johan Lindal Haug og Rune Husveg. Haug har vært hjelpsom og tilgjengelig i perioder. Husveg skal takkes for utlån av LabVIEW-blokker.

Grimstad, 20.mai, 2015

Øyvind Bergby Handeland

Innholdsfortegnelse

Figurliste

Tabelliste

Kapittel 1

Introduksjon

1.1 Bakgrunn

Det er en utfordring å overføre last eller personale mellom flytende installasjoner som beveger seg asynkront i forhold til hverandre, eksempelvis når personale skal overføres fra et skip til en flytende vindmølle som vist i figur 1.1. For at landgangen skal ligge stabilt i dårlig vær/høye bølger, kan en løsning være å måle den relative bevegelsen mellom landgangen og vindmøllen, og så kompensere bort den relative bevegelsen mellom de to ved å benytte seg av hydrauliske eller elektriske systemer til å styre landgangen.

Figur 1.1: Personale skal fraktes fra et skip og over til en vindmølle

I offshoreindustrien er det vanlig å benytte seg av MRU(motion reference unit) til å måle orienteringen til båten i forhold til et fast punkt. Dersom en MRU blir plassert på båten og en på vindmøllen, kan disse kobles sammen og den relative bevegelsen vil kunne måles. Det må være en fysisk kobling med kabel da en trådløs overføring ikke er ønskelig for kritisk informasjon. Kinematikken og MRU-signal fra de 2 fartøyene må samordnes og sendes inn i kontrollsystemet for landgangen.

Å koble fartøyene sammen med en fysisk kabel har både sine praktiske og sikkerhetsmessige utfordringer. Det ble derfor laget et alternativ til dette systemet i masteroppgaven; *Development of a vison-based measurement system for relative motion compesation* [3]. I oppgaven ble det laget et *vision system* der et kamera benyttes som visuell sensor. Kameraets positur relatert til en plan flate ble beregnet ved at flaten inneholder et kjent mønster som vil være innenfor kameraets synsfelt. Et sjakkbrett ble den gangen benyttet som mønster. Ved å benytte seg av det indre koordinatsystemet til kameraet, pikselkoordinatene og verdenskoordinatene var det mulig å beregne seg frem til den relative posituren mellom kamera og den plane flaten. Påliteligheten til dette systemet viste seg derimot og være for dårlig.

1.2 Problembeskrivelse

Det tidligere utviklede vision systemet [3] fungerer, men er ikke robust nok til å fungere i praksis. Detektering av hjørnene på sjakkbrettet krever gode lysforhold, og systemet er forholdsvis lite robust mot støy. Dette gjør at systemet ikke kan benyttes utendørs. Formålet med denne masteroppgaven vil derfor være å videreutvikle dette vision systemet slik at det blir mer robust for støy og varierende lysforhold.

1.3 Problemløsning

For å løse problemet med varierende lysforhold og støy blir systemet som detekterer et sjakkbrett byttet ut med et system som detekterer konsentriske kontrasterende sirkler. Dette er et mere robust mønster enn et sjakkbrett og teorien bak dette mønsteret blir forklart i kapittel 2.3.1.

Det bygges også et uavhengig aktivt systemet. Det samme kameraet blir benyttet som visuell sensor men vil nå detektere en aktiv lyskilde i form av strukturert lys(laser) istedenfor et passivt mønster. Dette systemet bygges ved å sette 4 punktlasere inn i en plate sammen med kameraet. Laserne projisere prikker med en viss intensitet på målskiven. Det vil da være mulig å gjenkjenne disse prikkene i bildeplanet. Dersom de fire laserne og kameraet er festet normalt i platen, vil det være mulig å beregne *heave, roll og pitch*. Teorien bak dette systemet blir forklart nærmere i kapittel 2.3.2.

Laserprikkene gjør det unødvendig å finne noe karakteristisk mønster på målskiven. Dette vil fungere best når det er overskyet og mørkt og kompenserer egenskapene til det passive vision systemet som vil fungere best når det er lyst.

1.4 Oversikt over rapporten

Kapitlene i denne rapporten er satt opp som følger:

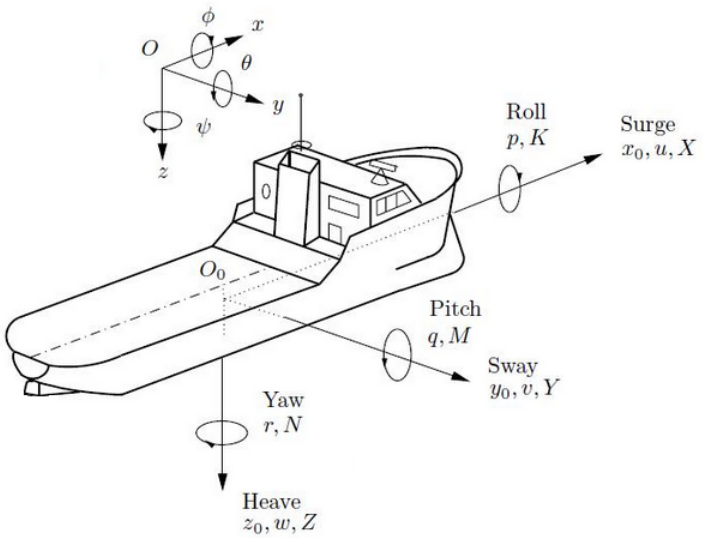
- Kapittel 2: Definerer teorien bak de forskjellige systemene som skal bygges, og valg av ord som benyttes videre i rapporten. Teorien bak kameramodellen og de to vision systemene blir gjennomgått.
- Kapittel 3: En kort gjennomgang av hvordan plattformen som inneholder lasere og kamera blir fysisk bygget, samt justert til bruk.
- Kapittel 4: En gjennomgang av hvordan teorien bak det aktive vision systemet blir benyttet i praksis for å lage koden. De viktigste LabVIEW-blokkene blir forklart.
- Kapittel 5: Oppbyggingen er lik som for kapittel 4 men nå blir den praktiske oppbyggingen av det passive vision systemet forklart.
- Kapittel 6: Utstyr som blir benyttet for testing av vision systemene.
- Kapittel 7: De 2 nye vision systemene blir testet opp mot det tidligere utviklede vision systemet.
- Kapittel 8: En konklusjon basert på prosjektet som helhet
- Vedlegg A og B er oversiktsbilder av hele koden for de 2 nye vision systemene som er laget i denne oppgaven. Blokkene som er benyttet i disse oversiktsbildenene er ikke lagt ved som vedlegg da de er tatt med og forklart i selve rapporten. Blokkene som er benyttet i *Observ_punkt_ordnet.vi* illustrert i figur 5.10 er tatt med i vedlegg C. Disse blokkene er ikke tatt med i selve rapporten da de er hentet fra et annet prosjekt [7].

Kapittel 2

Definisjoner

2.1 Frihetsgrader

I figur 2.1 under er de 6 frihetsgradene til et skip definert.



Figur 2.1: Alle 6 frihetsgradene på et skip [3].

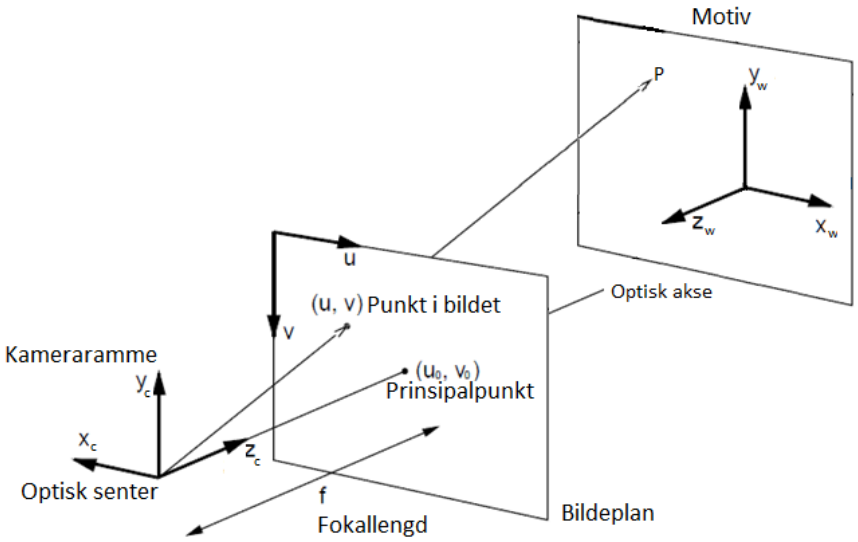
De definerte frihetsgradene som er tatt i bruk i figuren ovenfor benyttes videre i rapporten som følger:

- Heave: Translasjon i z-aksen
- Yaw: Rotasjon i z-aksen
- Surge: Translasjon i x-aksen
- Roll: Rotasjon i x-aksen
- Sway: Translasjon i y-aksen
- Pitch: Rotasjon i y-aksen

Skipets roll og pitch er begrenset til $\pm 15°$ [3].

2.2 Modell av kamera

[3] Kameramodellen som er benyttet i til å beskrive kameraparameterne kalles på engelsk *frontal pinhole camera*. Dette er en *nålehullsmodell* der bildeplanet er plassert foran nålehullet, altså det optiske senter som vist i figur 2.2 under.



Figur 2.2: Nålehullsmodell med bildeplanet plassert foran optisk senter

Forklaring av figur 2.2:

- Bildeplanet, altså (u,v) planet, er sensoren i kameraet. Det er denne platen som tar opp lys(fotoner) og omdanner det til et elektrisk signal. Signalet digitaliseres og leses av som gråtoneverdier.
- X_c,Y_c og Z_c definerer bilderommet og er kameraets koordinatsystem. Origo for dette koordinatsystemet kalles optisk senter.
- u og v er piksel-koordinatene i bildeplanet og har origo oppe i venstre hjørnet av pikselplaten.
- Fokallengden er lengden mellom optisk senter og bildeplanet.
- Optisk senter er blenderåpningen i kameraet. Denne åpningen antas uendelig liten. Det er her lyset fra omverden slepper inn og blir projisert på bildeplanet. Modellen kan bli tolket misvisende fordi den plasserer bildeplanet foran blenderåpningen. Dette er en forenkling for at motivet ikke skal bli snudd på hodet i bildeplanet. Dette er løst ved at aksene i bildeplanet(u,v) er motsatt rettet av aksene (X_c, Y_c) i kameraets koordinatsystem. Siden bildeplanet er plassert foran optisk senter vil $z = f$, mens i virkeligheten vil $z = -f$.
- Prinsipalpunktet er der hvor den optiske aksen treffer bildeplanet, altså der hvor kameraets(X_c,Y_c) koordinat er (0,0). Dette punktet er ofte ikke helt i senter av bildeplanet grunnet imperfeksjoner i optikken. Dette avviket bestemmes gjennom kaibreringsprosedyrer og vil bli videre diskutert i avsnitt 2.2.3

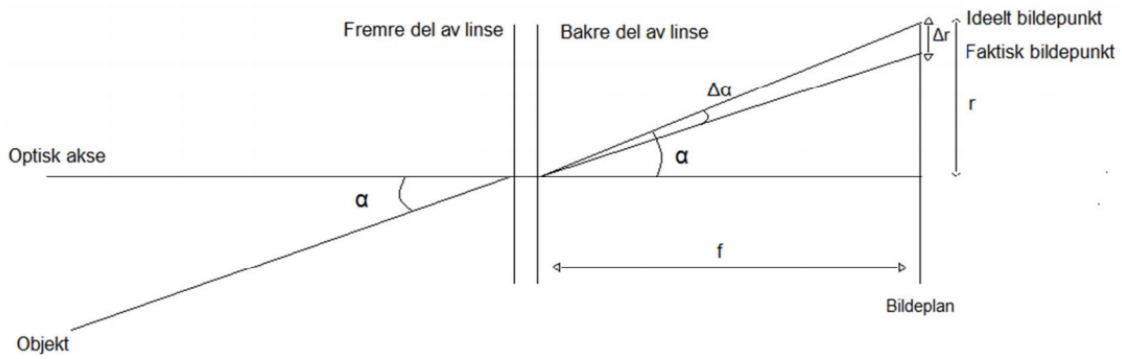
2.2.1 Interne paramtre

De interne parametrene beskriver kameraets optikk. Disse parametrene er listet opp i tabellen under:

Tabell 2.1: Innvendige kameraparametre [3].

Fokallengde, u-retning	f_u
Fokallengde, v-retning	f_v
Prinsipielt punkt, u-retning	u_0
Prinsipielt punkt, v-retning	v_0
Størrelse på piksel i u-retning	S_u
Størrelse på piksel i v-retning	S_v
Forvrengning	k_1, k_2, k_3

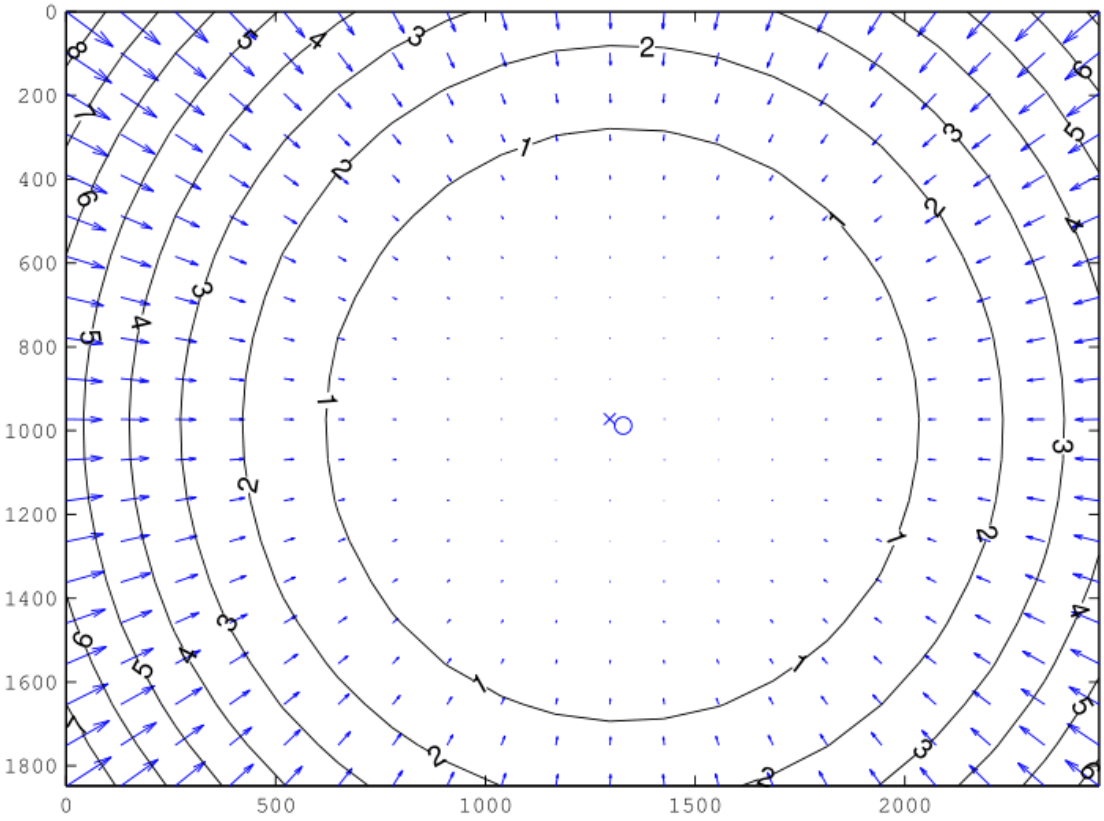
Siden kameraet har en linse som er sfærisk formet er ikke nålehullsmodellen, som er uten linse, helt nøyaktig. Dette betyr at et bilde som blir tatt med en sfærisk linse vil ha aberrasjoner i mer eller mindre grad. En aberrasjon gjør at bildet ikke blir representert helt perfekt i samtlige punkter. Dette kan føre til feil dersom det skal beregnes positur av et objekt ut ifra piksel-lengder i bildet. Det finnes fem ulike aberrasjoner: Sfærisk aberrasjon, koma, astigmatisme, bildekrumming og linseforvrengning. Den sistnevnte påvirker måleresultatene og må derfor korrigeres gjennom geometrisk kalibrering. Det er kun bildekvaliteten som påvirkes av de andre aberrasjonene [2] og vil derfor ikke bli videre diskutert. Under illustreres effekten av linseforvrengning som stort sett beveger seg radielt utover i bildet med utspring i prinsipalpunktet. Den radielle forvrengningen er en radiell forskyvning av de korrekte posisjonene til pikslene i bildet.



Figur 2.3: Effekten av linseforvrengning [2].

Som figuren viser øker Δr med fokallengden.

Dersom effekten av linseforvrengning sees forfra, altså med den optiske aksen innover i bildet, vil det se ut som på figur 2.4 under. I illustrasjonen vil retningen på pilene og nummereringen av sirklene indikere hvor mange piksler bildet er forvrengt.



Figur 2.4: Effekten av linseforvrengning. [3]

Radiell forvrenging kan rettes opp ved å benytte forvrengningskoeffisientene k_1, k_2 og k_3. Hvordan disse koeffisientene hentes ut beskrives i avsnitt 2.2.3. Dersom koordinatene er normalisert rundt prinsipalpunktet, vil forvrengningen kunne bli korrigert som vist i formel 2.1. Indeksering med d indikerer at koordinatet er utsatt for forvrengning(distortion) og indeksering med n indikerer at koordinatene er i det normaliserte bildelanet. Ligning 2.2 gir avstanden fra prinsipalpunktet og punktet som skal korrigeres [8].

$$\begin{aligned} x_n &= \frac{x_{nd}}{1 + k_1 r^2 + k_2 r^4 + k_3 r^6} \\ y_n &= \frac{y_{nd}}{1 + k_1 r^2 + k_2 r^4 + k_3 r^6} \end{aligned} \tag{2.1}$$

$$r^2 = x_{nd}^2 + y_{nd}^2 \tag{2.2}$$

Fokallengden som er oppgitt i mm blir dimensjonsløs ved at den deles på piksel-størrelsen også målt i mm for u og v retning som vist i matrisen 2.3.

Siden fokallengden og prinsipalpunktet er dimensjonsløse i denne rapporten vil de være oppgitt i piksler. Fokallengden som er oppgitt i mm må derfor deles på størrelsen på en piksel (s_u *og* s_v). Parametrene er satt opp som vist i K-matrisen 2.3 under. Dette er formen på kameraets kalibreringsmatrise. $o(\alpha)$ er en justeringsparameter dersom pikslene ikke er kvadratiske. Siden kameraet benyttet i denne rapporten har kvadratiske piksler, blir denne parameteren satt til null.

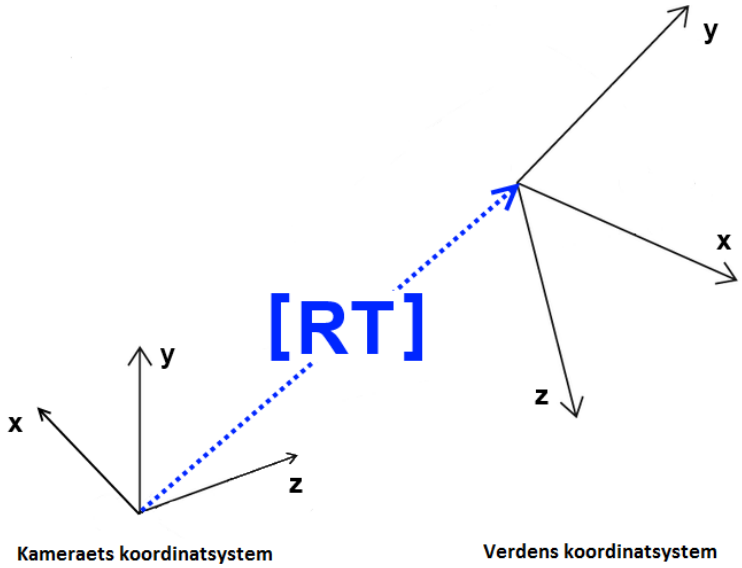
$$K = \begin{bmatrix} f/s_u & o(\alpha) & u_0 \\ 0 & f/s_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} f_u & o(\alpha) & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

2.2.2 Eksterne paramtere

De eksterne eller utvendige parameterne beskriver forholdet mellom kamerakoordinatene og verdenskoordinatene, med respekt til verdenskoordinatene. Disse to koordinatsystemene beskrives av en rotasjon (R) og en translasjon(T). Rotasjonsmatrisen er en 3x3 matrise, men translasjonen er en 3x1 vektor. Disse to parameterne gir den samlede matrisen referert til som RT-matrisen vist i ligning 2.4 under.

$$RT = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

En illustrasjon av de to koordinatsystemene er vist i figur 2.5 under.



Figur 2.5: Eksterne paramtre

I figuren vil translasjonen T være avstanden fra origo i kameraets koordinatsystem til origo i verdens koordinatsystem. Verdens koordinatsystem vil ha origo på en hensiktsmessig utvalgt plass. Rotasjonen vil være hvordan de to koordinatsystemene er rotert i forhold til hverandre.

2.2.3 Kalibrering av kamera

I motsetning til de ytre parameterne, vil de indre paramtrene i kameraet forbli uendret dersom linsen ikke byttes ut, uavhengig av posisjon og omgivelser. For å finne disse konstante parameterne benyttes et kalibreringsverktøy i MATLAB, *Camera Calibration Toolbox for MATLAB* [9]. Siden kameraet og optikken er lik i denne rapporten som i rapporen til Haug [3], benyttes den samme k-matrisen som i rapporten til Lindal Haug. Den ble funnet ved å ta 20 bilder av et sjakkbrettmønster med kjente størrelser på rutene. Bildene ble benyttet i den nevnte verktøykassen i MATLAB som kalkulerte de invendige parameterne ved å analysere disse bildene.

Linsen som ble benyttet har en fokallengde på 12mm. Parametrene fra kalibreringsverktøyet hentes ut og settes inn i matrisen 2.5 som vist under.

$$K = \begin{bmatrix} f/s_u & o(\alpha) & u_0 \\ 0 & f/s_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 5562 & 0 & 1327 \\ 0 & 5548 & 987 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

Forvrengingskoeffisientene er vist i tabellen under

Tabell 2.2: Linseforvrengning [3]

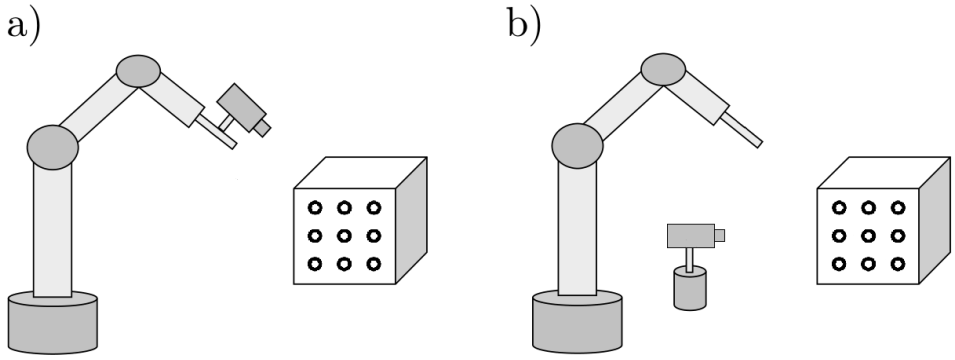
Koeffisienter	Verdi
k_1	-0.09195
k_2	0.36222
k_3	-0.00302

2.3 Vision system

De to mest fundamentale systemkonfigurasjonene innen vision systemer i industrien er [10]:

- Øye-i-hånd. Dette betyr at kameraet er festet til den bevegelige robotarmen
- Hånd-til-øye. Kameraet er fiksert med verdenskoordinater og observerer målobjektet og den bevegelige robotarmen.

De 2 systemkonfigurasjonene er illustrert i figur 2.6 under.



Figur 2.6: a) øye-i-hånd konfigurasjon. b) Hånd-til-øye konfigurasjon

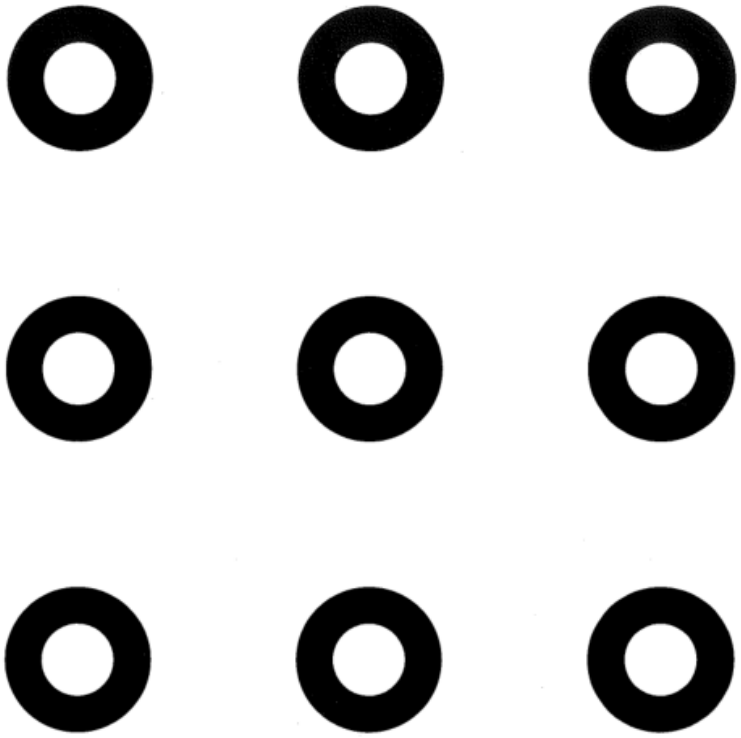
øye-i-hånd er den mest vanlige systemkonfigurasjonen [1] og benyttes videre i rapporten. I denne rapporten blir det bygget 2 øye-i-hånd systemer som blir kalt:

- Passivt vision system
- Aktivt vision system

Videre i kapittelet vil disse 2 systemene bli forklart.

2.3.1 Passivt vision system

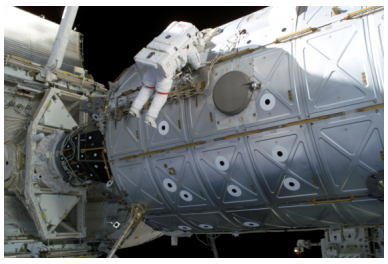
For dette systemet blir det benyttet et mønster som tar i bruk sirklenes egenskaper fremfor et sjakkbrett. Sirkler er lette og finne i et bilde og LabVIEW har egne blokker som detekterer sirkler. Sirkelteknikken som her benyttes kalles *Konsentriske kontrasterende sirkler* eller på engelsk *Concentric Contrasting Circles*(CCC) [4]. I figuren nedenfor 2.7 er sirkelmønsteret avbildet. Diameteren på den svarte sirkelen er 40mm og på den lille hvite 20mm.



Figur 2.7: Konsentriske kontrasterende sirkler

Teknikken går ut på at du detekterer sirkler med sentrum i samme punkt, altså er de konsentriske. De konsentriske sirklene har kontrasterende farge slik at det skal være mulig å detektere svarte og hvite sirkler hver for seg. Dersom alle sirklene detekteres kan sentrum av sirklene sammenlignes. Ved å benytte en lav terskling på denne sammenligningen kan støypartikler filtreres vekk på en effektiv måte. Dersom to sentrumspunkter sammenfaller benyttes sentrumspunktet fra den største sirkelen videre i koden. En stor sirkel vil gi et mer nøyaktig sentrumspunkt enn en liten. Dette er bevist i kapittel 7.1.1. Sentrum av sirkelen måles ved å beregne tyngdepunktet på den.

Deteksjon av romfartøy er et eksempel på hvor denne teknikken benyttes som vist i bildet 2.8 under.



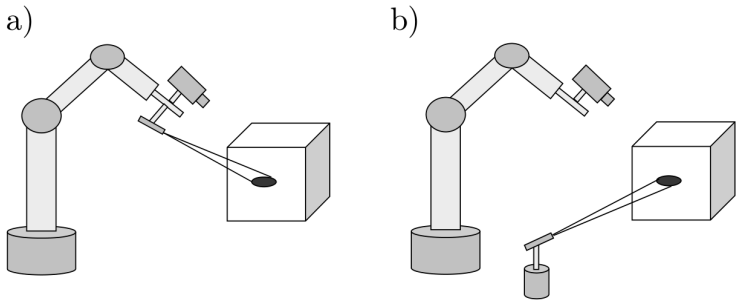
Figur 2.8: CCC-teknikken blir benyttet for å måle avstander i rommet [4]

2.3.2 Aktivt vision system [1]

Et problem med et passivt vision systemet vil være når en robot skal være parallell med et plan uten landemerker. Et annet problem er dersom det ikke er tilstrekkelig lys for å detektere landemerkene.

Det er derfor laget et aktivt system som benytter strukturert lys/laser til å projisere visuelle landemerker. Eksempler på applikasjoner med denne type behov kan være maleindustrien, kutting av plater og sveising.

Dersom laser benyttes finnes det to konfigurasjoner for den nevnte teknikken, *øye-i-hånd*. Den ene gjelder dersom laserkilden separeres fra kamera og robotarmen. Da vil det projiserte landemerket fra laserkilden være statisk på objektet-overflaten. Denne teknikken fungerer ikke dersom objektet flytter på seg, eller at landemerket kommer utenfor kameraets synsfelt. For disse problemstillingene kan laserkilden festes på robotarmen slik at den beveger seg konstant sammen med kameraet. Disse to konfigurasjonene er avbildet i figuren under 2.9.



Figur 2.9: 2 aktive vision systemer med *øye-i-hånd* konfigurasjon [1].

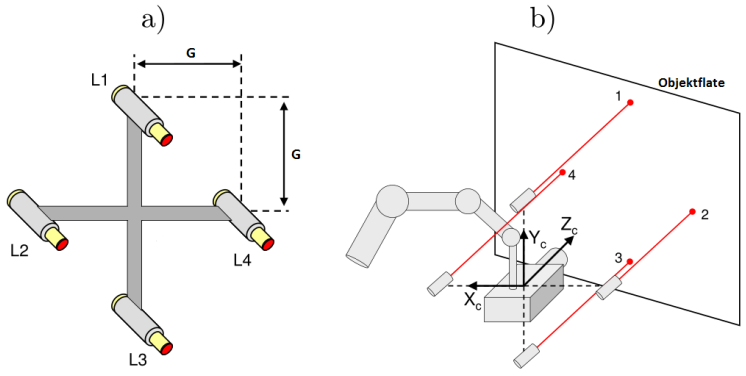
- *a*) Aktivt vision system med *øye-i-hånd* og *laserkilde-i-hånd* konfigurasjon
- *b*) Aktivt vision system med *øye-i-hånd* konfigurasjon og separat laserkilde

Konfigurasjonen gitt i punkt *a* blir valgt videre i denne rapporten. Dette for å sleppe at laserprikkene vil komme utenfor kameraets synsvinkel.

Trigonometri

Dersom laserne er perfekt justert i konfigurasjon *b* i figur 2.10, vil det være mulig å benytte trigonometri til å gjøre beregninger for avstand og orientering av objektflaten relatert til kamera.

Arkitekturen i figuren 2.10 under benytter fire lasere og et kamera for å kunne utføre disse beregningene. **G** i figuren står for *grunnlinje*.

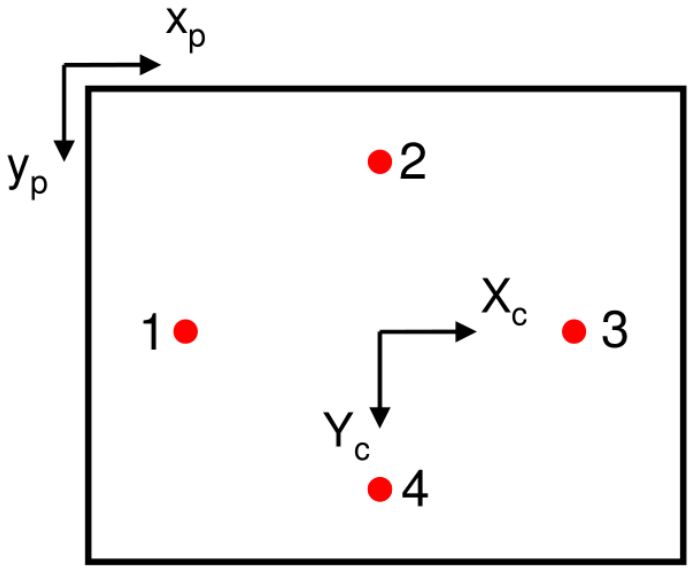


Figur 2.10: Arkitekturen bak lasersystemet. *a*) Rammen til laserkildene. *b*) Kameraet festet til sentrum av rammen som danner plattformen i enden av en maskinarm. [1]

For å utføre trigonometriske beregninger fra kamera til objektflaten, må følgende finjustering forekomme på figur 2.10 *b*:

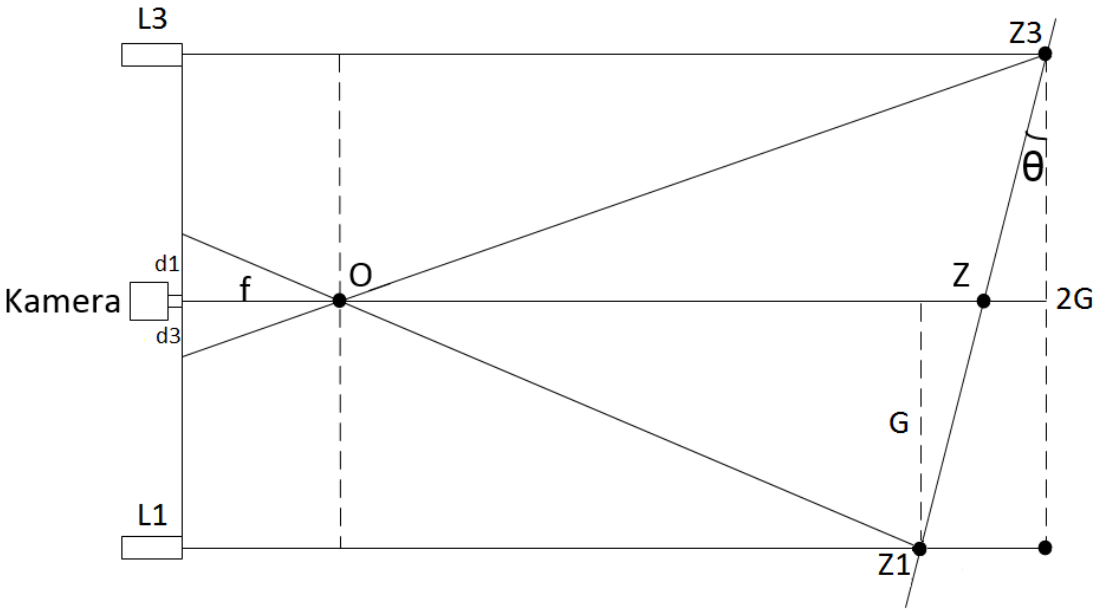
- kameraet festes i senter av rammen
- Laserne og kamera står normalt på rammen
- Grunnlinjen er lik for de 4 laserne

Figuren under illustrer et kamerabilde der objektflaten og plattformen står parallelle og punktene ovenfor er i overensstemmelse.



Figur 2.11: Laserprikkene vist i bildet dersom plattformen er korrekt justert og er parallell med objektflaten. [1]

Figuren under illustrerer hvordan kameraet detekterer strålene fra laser 1 og 3. Dette tilsvarer rotasjon om Yc i figur 2.11 og vil derfor være rotasjon av typen pitch.



Figur 2.12: Refleksjon av laser 1 og 3 på en objektflate som har en relativ vinkel til kamera

Det kan trekkes en sammenligning mellom trekantene i figur 2.12 slik at:

$$\frac{d_1}{f} = \frac{G}{z_1} \tag{2.6}$$

Avstand i Z-retning er målt fra $'O'$(Optisk senter) og kan utledes til:

$$z_1 = \frac{G \cdot f}{d_1} \tag{2.7}$$

$$z_3 = \frac{G \cdot f}{d_3} \tag{2.8}$$

$$z = \frac{z_1 + z_3}{2} \tag{2.9}$$

Vinkelrotasjonen om y-aksen blir:

$$\theta_{pitch} = \arctan(\frac{z_1 - z_3}{2 \cdot G}) \tag{2.10}$$

Den samme trigonometrien kan benyttes på rotasjon om x-aksen(roll). Da benyttes laser 2 og 4 isteden.

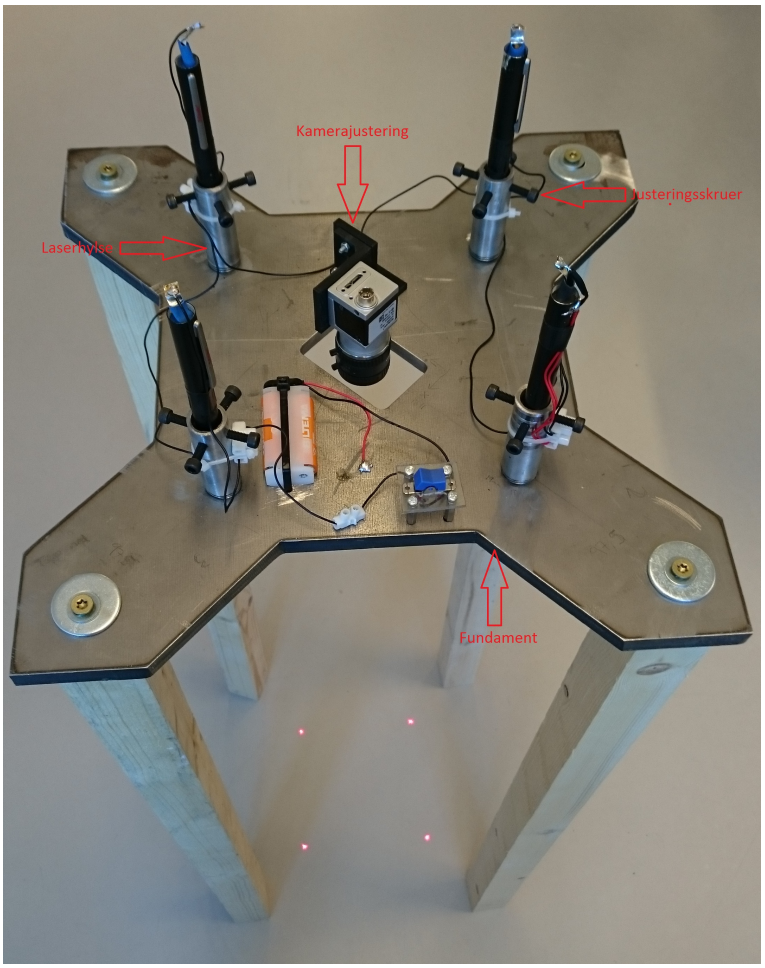
Kapittel 3

Fysisk justering av kamera og lasere

For å få testet vision systemene ble det bygget en plattform basert på figur 2.10 *b*. Det ble viktig med en god rutine på hvordan lasere og kamera i plattformen skal justeres inn slik at det aktive systemet blir så presist som mulig. For å få til en god rutine på dette må plattformen bygges slik at lasere og kamera har gode justeringsmuligheter. Dette kapittelet beskriver litt om prosessen bak bygging av en testbenken og rutinen på hvordan plattformen skal justeres.

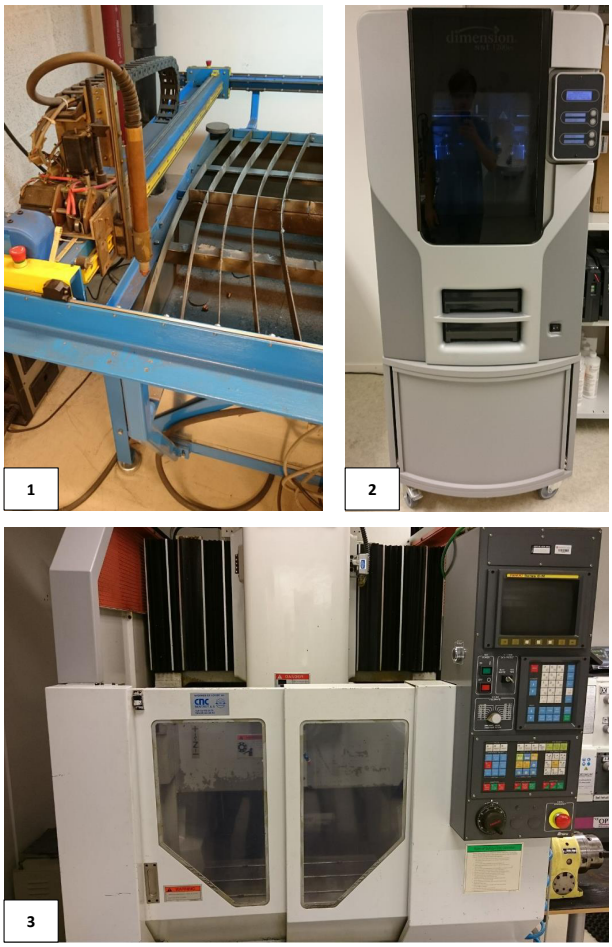
3.0.3 Fysisk bygging av testbenk

Testbenken ble bygget ved å benytte presise maskiner og verktøy som er tilgjengelige på universitet. Under er et bilde av den ferdige testbenken som vil være plattformen med de 4 beina montert på.



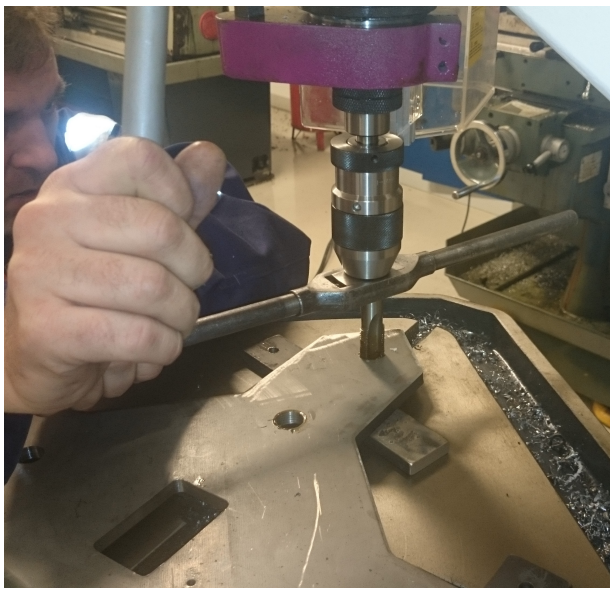
Figur 3.1: Testbenk

Fundamentet og kamerafestet ble tegnet opp i SolidWorks 2014 for å få alle mål så presise som mulig. Fundamentet ble deretter brent ut i en plasmabrenner bygd på universitetet. Presisjonen på fundamentet er mindre kritisk. Plasmabrenneren er vist i figur 3.2 1. Hullene til laserhylsene og kamera i fundamentet ble frest ut i en CNC maskin som vist i figur 3.2 3. For at laserne skal stå normalt på plattformen og at grunnlinjen skal bli korrekt ble det viktig å få disse hullene så presise som mulig. Det er også viktig at hullet hvor kamera står er så sentrert som mulig i firkanten som dannes av de fire hullene til laserhylsene. CNC-maskinen har en oppløsning på 0,003mm. Kamerajusteringen ble bygd i 3D printeren som vist i figur 3.2 . 3D-printeren har en oppløsning på 0,2mm.



Figur 3.2: 1: Plasmabrenner, 2: 3D-printer, 3: CNC-maskin

For å få gjenget opp hullene til laserhylsene så normalt på fundamentet som mulig ble en stasjonær boremaskin benyttet som referanse som vist illustrert i 3.3 under .



Figur 3.3: Gjenging av fundament

For å få gjenget opp laserhylsene utvendig, ble det benyttet en dreiebenk for å få gjengebakken så sentrert som mulig rundt hylsen. Dette er viktig for at laserne skal stå normalt på fundamentet. Dreiebenken er illustrert i figur 3.4 under.

Figur 3.4: Dreiebenk benyttet for å lage utvendige gjenger på laserhylsene

Da laserstrålen ut av laserpekerne kan være noe skjev og laserhylsene ikke står nøyaktig normalt på fundamentet må det lages en justeringsmulighet for laserpekerne. En o-ring er derfor festet rundt tuppen av laserpekerne. Denne gjør at pekerne holder seg i ro i laserhylsen og at de har muligheten til å dreie rundt i o-ringen nede inne i hylsene. Ved hjelp av justeringsskruene merket i figur 3.1 vil laserpekerne derfor kunne justere siktet i x- og y-retning.

Batteriene i laserpekerne er fjernet og koblet sammen i parallell med en felles bryter.

For å kunne stille inn lasere og kamera korrekt må fundamentet være parallelt med en plan flate. Gulvet antas å være plant og benyttes derfor som den plane flaten. For å få fundamentet og gulvet i parallell ble det laget fire like lange trebein som festes på undersiden av plattformen. Dersom beina har identisk lengde vil plattformen og gulvet være parallelle. Dette ble sjekket med å benytte et *vater* på gulvet og plattformen. Dette lar seg også sjekke dersom alle fire beina har kontakt med gulvet på en gang.

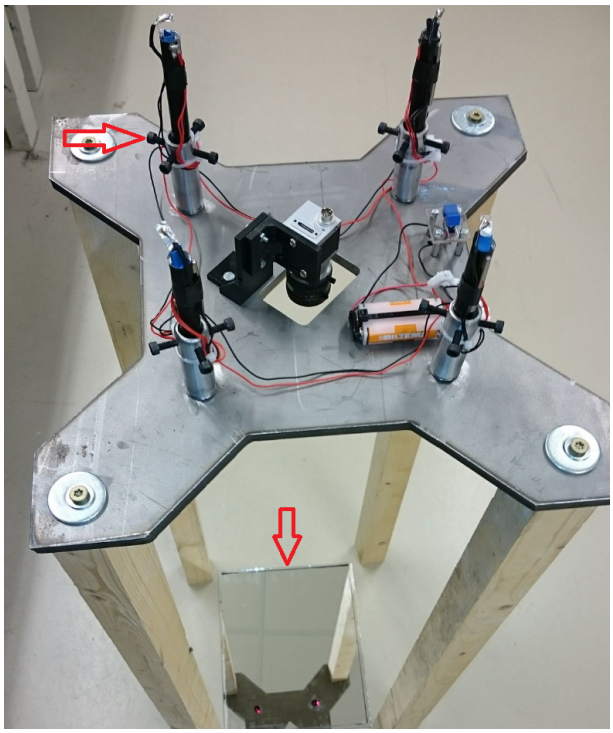
For å få så lik lengde som mulig på beina, ble en stasjonær sirkelsag benyttet sammen med en *skrutvinger* som illustrert i bildet 3.5 under.

Figur 3.5: Sirkelsag og skrutvinger benyttet for å lage lengden på beina til plattformen så like som mulig

3.0.4 Justering av lasere og kamera

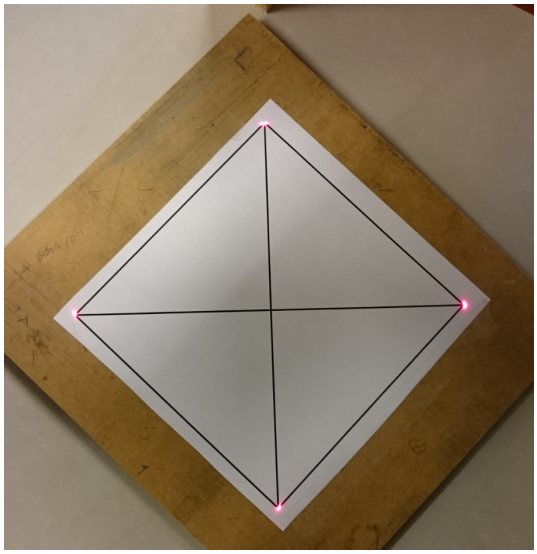
For å gjøre testbenken klar til å utføre positurmålinger av en plan flate er det laget en rutine for å justere kamera og lasere. Den første rutinen går ut på justering av lasere og den andre går ut på justeringen av kamera.

Skrittvis justering av lasere:
1. Et stivt speil legges på gulvet under plattformen. De fire justeringsskruene til hver laserhylse benyttes for å justere laserpekerne slik at laserstrålen reflekteres via speilet og direkte tilbake i laserpekeren som vist i bildet 3.6 under. Fordi Plattformen er parallell med gulvet vil denne teknikken sørge for at laserne står normalt på plattformen. Tester viser at justeringsvinkelen for hver enkelt laser er omtrent 3°.



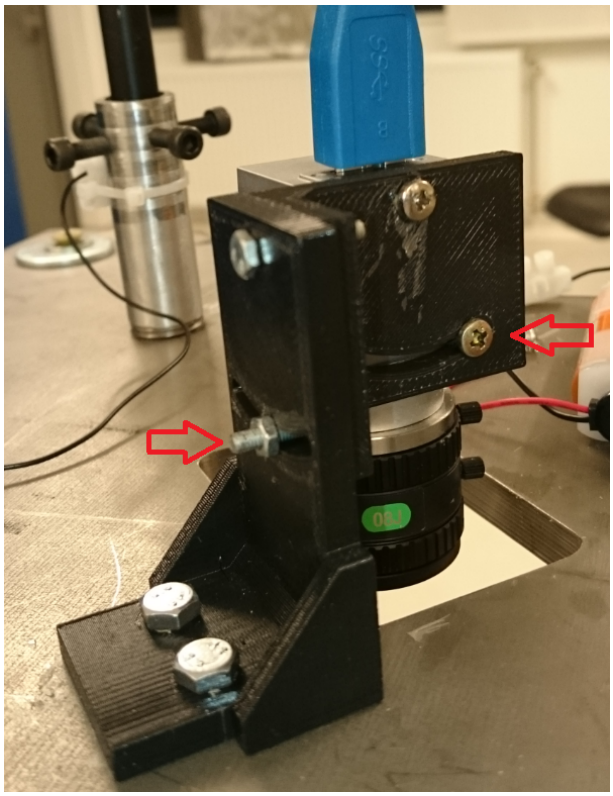
Figur 3.6: Testbenk med avmerket skruer til justering av lasere og speil

2. Fjerner speilet og legger inn en skisse av grunnlinjene(avstanden mellom hver laser og kamera) i x-retning og y-retning som vist i figur 3.7. Dersom laserprikkene treffer hjørnene av kvadratet betyr det at laserpekerne står med korrekt avstand fra hverandre. Dette vil også være en dobbel sikring på at justeringen i punkt 1 er korrekt utført. Laserpekerne er nå ferdig justert.



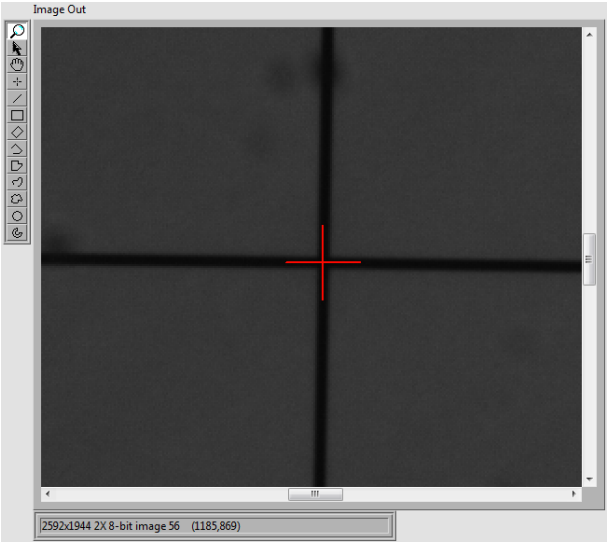
Figur 3.7: Skisse av plattformens grunnlinjer

Justering av kamera:
3. Grovjustering. For å justere kameraet vil det være muligheter for å justere dreining i x-aksen og y-aksen som vist i figur 3.8 under.



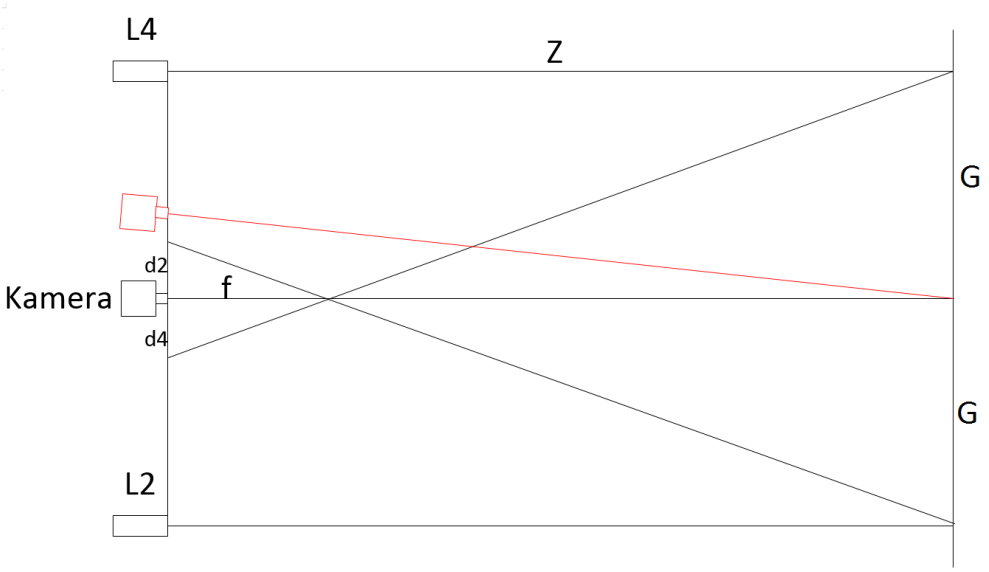
Figur 3.8: Kameraets justeringsmekanisme for dreining om x- og y-aksen

Kamera vil være justert riktig inn dersom den optiske aksen treffer det diagonale krysset i figur 3.7. En del av LabVIEW-programmet detektere prinsipalpunktet med et rødt kors. Kameraet justeres fysisk slik at dette røde korset sammenfaller det diagonale krysset som illustrert i figur 3.9 under. LabVIEW-programmet illustreres i figur 4.2.



Figur 3.9: Program som merker prinsipalpunktet med et kors.

4. Finjustering. Selv om kameraet er justert inn etter prinsipalpunktet i punkt 3, er det ikke sikkert at kameraet står sentrert i plattformen. En slik situasjon er illustrert i figur 3.10 under. I figuren er det røde kamera feiljustert, men den optiske aksen vil likevel treffe det diagonale krysset.

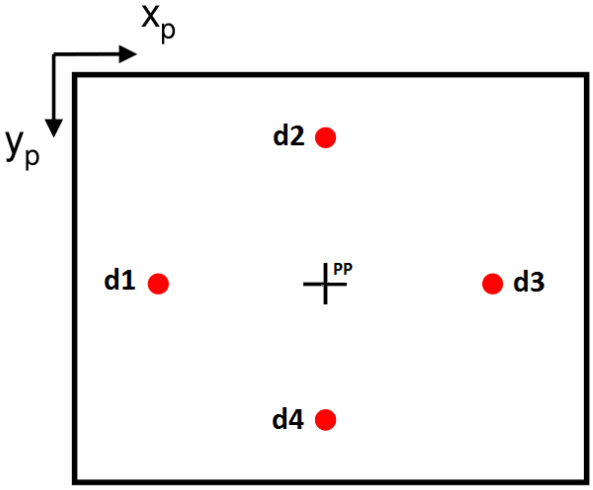


Figur 3.10: En optimalt justert modell av kamera og lasere, kontra et kamera som ikke er sentrert i plattformen markert med rødt.

Beskrivelse av parametrene i figuren over:

- f: Fokallengde
- d2, d4: Lengde fra reflektert laser 2 til prinsipalpunktet, og lengde fra reflektert laser 4 til prinsipalpunktet i bildeplanet.
- Z: Lengde i z-retning
- G: Grunnlinjen, altså avstand fra hver laser og til sentrum av kamera

For å sjekke dette må avstanden fra prinsipalpunktet til hver av de detekterte laserpunktene i bildet være lik. I figur 3.11 under er et bilde skissert opp med distansene fra prinsipalpunktet til de fire laserprikkene markert som d1 til d4.



Figur 3.11: Skisse av et bilde med avstand fra prinsipalpunktet til laserprikkene.

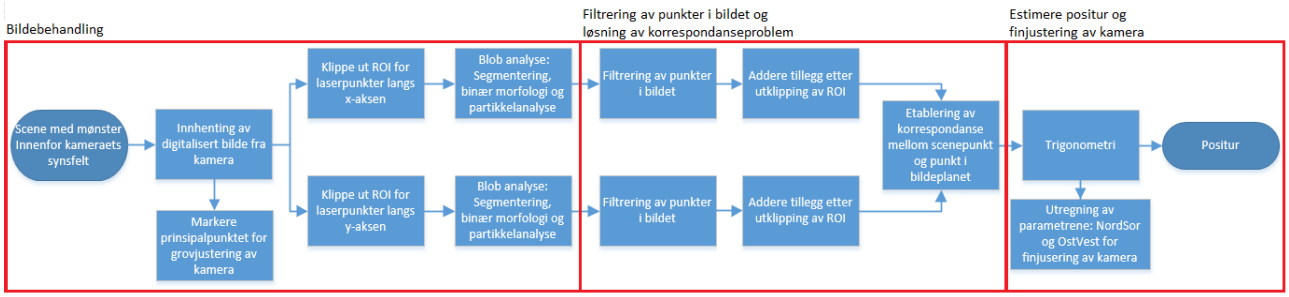
d1 blir sammenlignet med d3 og d2 blir sammenlignet med d4. Dersom d1 er lik d3 og d2 er lik d4 vil kameraet være sentrert i plattformen. Feilen i plassering av kamera måles i piksel og blir kalt *Nordsor* og *VestOst* i LabVIEW. LabVIEW-koden blir sett på i kapittel 4.3.

- $Nordsor = d2 - d4 > 0$: Juster kamera mot nord
- $VestOst = d1 - d3 > 0$: Juster kamera mot vest

Kapittel 4

Oppbygging av koden til aktivt vision system

Koden til det aktive vision systemet er bygget opp som vist i blokkskjema i figur 4.1 under.



Figur 4.1: Blokkskjema for aktivt vision system. De røde rutene illustrerer seksjonene i dette kapittelet

De røde firkantene i bildet illustrerer hvordan kapittelet er bygget opp. Videre i dette kapittelet vil det gis en kort forklaring på hvordan de forskjellige LabVIEW-blokkene bak operasjonene i figur 4.1 fungerer.

4.1 Bildebehandling

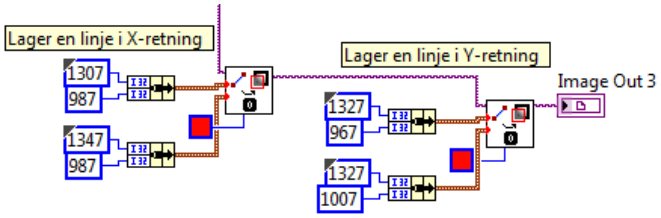
For at bildebehandlingsekvensen skal fungere må laserprikkene være innenfor kameraets synsfelt. Dersom kameraet er for nærme kan det oppstå problemer med at laserprikkene havner utenfor synsfeltet. Utenom dette vil ikke problemet inntreffe da laserkildene følger kameraets bevegelse. Videre i dette underkapittelet metodene bak bearbeidingen av det digitale originalbildet bli forklart. Det er i denne sekvensen informasjonen i bildet hentes ut.

4.1.1 Innhenting av digitalisert bilde fra kamera

Kameraet som er benyttet har en bilderate på 14 fps(frames per second). Hvert 70ms vil altså LabVIEW hente inn et nytt bilde til bufferet sitt. Det betyr at dersom koden for å prosessere bildet har en syklustid lavere enn dette kan det samme bilde bli prosessert flere ganger. For dette kameraet er det derfor ingen grunn til å kjøre en syklustid lavere enn 70ms. Dersom koden skal benyttes på et kamera med høyere fps, vil posituren kunne oppdateres hurtigere ved å ha lavere syklustid.

4.1.2 Markere prinsipalpunktet for grovjustering av kamera

For å justere kamera slik at det har riktig dreining på plattformen blir prinsipalpunktet markert med et kors. Prinsipalpunktet kommer frem av K-matrisen i avsnitt 2.2.3. Koden for dette korset er illustrert under, og forklaring på hvordan dette benyttes til justering kommer frem i avsnitt 3.0.4.



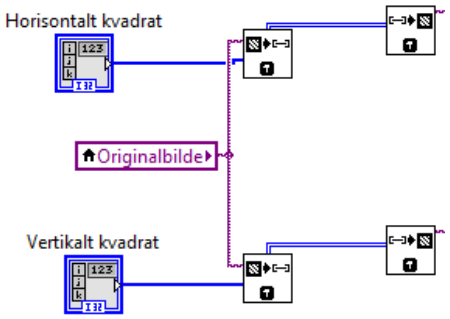
Figur 4.2: Prinsipalpunktet blir markert med et rødt kryss

4.1.3 Finne interesseområdet, ROI

ROI eller *Region of interest* blir på norsk oversatt til *interesseområde*. Siden laserprikkene er små og det er mulig å vite hvilket område i bildet de blir detektert i, kan originalbildet bli kuttet betraktelig ned til en bit av bildet står igjen. Denne biten vil bli behandlet videre som ROI.

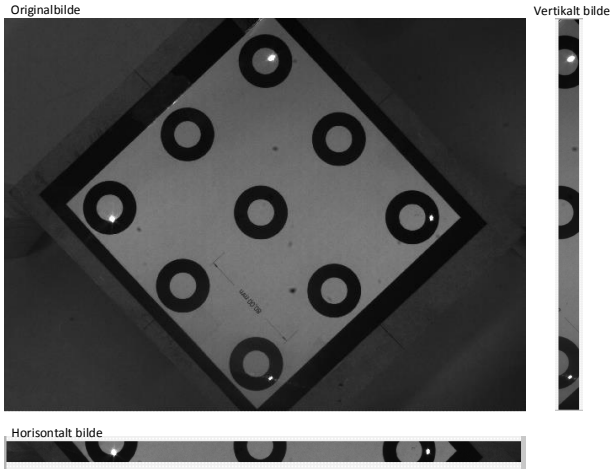
Da det vil være fire laserprikker, to i x-retning og to i y-retning, kan bildet kuttes i to kvadrater. Det ene horisontalt(x-retning) og det andre vertikalt(y-retning). Jo smalere disse stripene er, jo mindre prosesseringstid, og mindre mulighet for å få med uønskede partikler i bildet. Prosesseringstiden går ned jo færre piksler som må behandles videre i programsekvensen. Laserprikkene i horisontal retning vil kun bevege seg langs x-aksen i bildet, og laserprikkene i vertikal retning vil kun bevege seg langs y-aksen. Dette betyr at stripene kan klippes nesten like brede som laserprikkene, med noe margin på hver side. Margin er viktig dersom plattformen ikke er perfekt justert. For å finne denne bredden må man vite når laserprikkene blir representert med flest mulig piksler i bildeplanet. Dette blir målt ved å sette objektplaten så nærme kamera at laserprikkene akkurat vil være innenfor kameraets synsvinkel.

Blokkene som lager ev vertikal og en horisontal ROI er vist i figur 4.3 under.



Figur 4.3: Lager ROI i x- og y-retning for laser-programmet

Det vertikal og horisontale interesseområde er illustrert i figur 4.4 under.



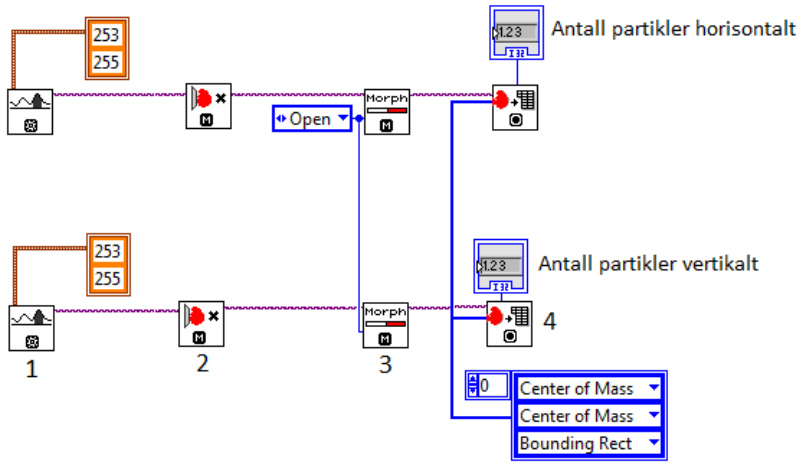
Figur 4.4: Horisontalt og vertikalt ROI

Bredden på det vertikale og høyden på det horisontale ROI på figur 4.4 er satt til 100 piksel, med objektplaten 800mm fra kamera.

4.1.4 Blob analyse

En blob står for *binary large object* og er et areal av piksler med *høy* verdi som grenser til hverandre i et binært bilde. Det finnes 2 ulike gråtoneverdier i et slikt bilde. I resten av rapporten vil *høy* og *lav* verdi i et binært bilde betegnes som gråtoneverdi 255 og 0. Har pikselen verdien 255 er den en del av et binært objekt. Har den verdien 0 vil pikselen ansees som bakgrunn i bildet.

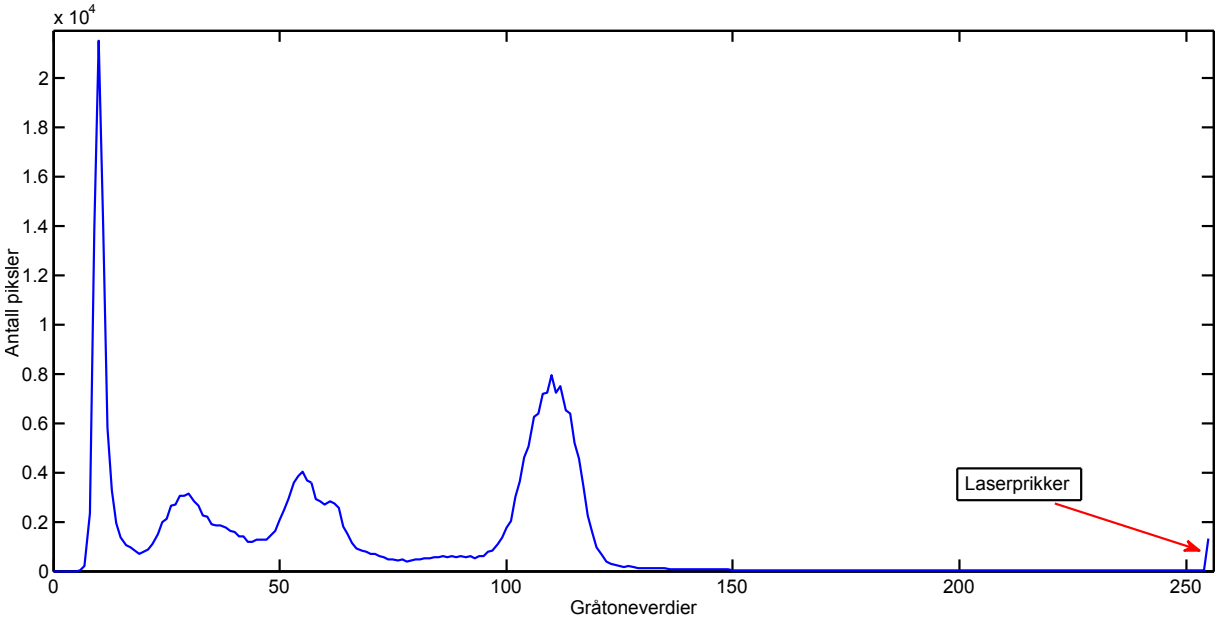
Det horisontale og vertikale bildet blir videre analysert som illustrert i figur 4.5 under.



Figur 4.5: Bildebehandling for vertikalt og horisontalt bilde

Koden gjør følgende.

1. Manuell segmentering. For å forstå dette settes det opp et histogram av gråtoneverdiene i inngangsbildet som vist i figur 4.6 under. Den røde pilen indikerer gråtoneverdien til laserprikkene i bildet.

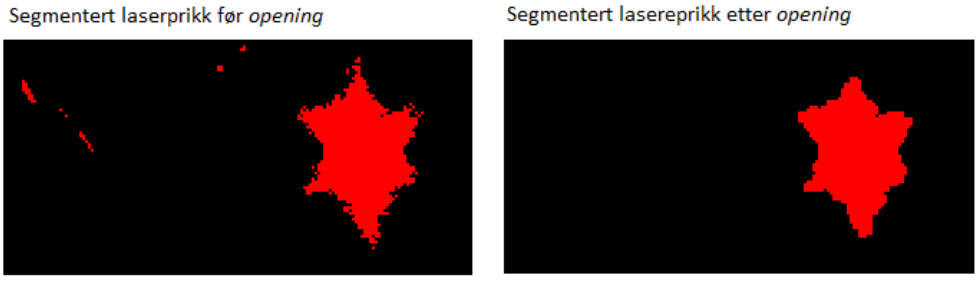


Figur 4.6: Histogram av horisontalt bilde i figur 4.4.

Siden det er kun laserprikkene som er interessante og inteniteten i laserpikkene er upåvirket av lysforhold, benyttes en manuell terskling. Den manuelle tersklingen settes til 253 som vist i figur 4.5. De gråtoneverdiene som er over terskelverdien settes til 255 mens de som er under settes til 0. Resultatet er et binært bilde bestående av verdier på 0 og 255. Siden denne blokken er satt til å se på lyse piksler vil de lyse punktene få verdien 255 og de mørke verdien 0.

2. Fjerner kantobjekter fra det binære bildet. Dersom objekter er i kontakt med kanten av bildet vil objektet fjernes. Et objekt vil være en samling av piksler med gråtoneverdi 255 fra segmenteringen.

3. Åpning [13]. Dette er en sammensatt morfologisk operator som består av de to grunnleggende operatorene *erosjon* og *dilasjon*. $aapning = erosjon + dilasjon$. Dilasjon vil gjøre at alle pikslene med gråtoneverdi 0 som grenser til et objekt vil få gråtoneverdi 255. Objektene vil altså få større areal. Erosjon vil være at alle pikslene i et objekter som grenser til en piksel med piskelverdi 0 vil få pikselverdi 0. Objektet vil altså få mindre areal. I en *åpning* vil det altså forekomme en erosjon først og deretter en dilasjon. Fordi laserprikkene har skiftende form og ujevne kanter vil det være av hensikt å utføre en *åpning*. Dette vil også fjerne de minste partiklene som vil forstyrre analysen. Effekten av *åpning* er vist i figuren under.



Figur 4.7: *Åpning* utført på laserprikk

4. Partikkelanalyse. Her blir det gjort en analyse av hver enkelt partikkel i det binære bildet i den forstand at spesifiserte målinger blir gjort. Det finnes 50 forskjellige partikkelmålinger for denne blokken. I dette tilfellet beregnes det 3. Koordinatet til en partikkel regnes ut ved å finne tyngdepunktet i x- og y-retning til hver partikkel. Dersom dilasjonen i forrige punkt ikke hadde blitt utført ville den oppstykkede laserprikken gjort at denne analysen ville gitt ut mål for flere enn fire partikler. Målingene blir representert med subpiksel nøyaktighet. Dette vil si at parametrene som er regnet ut med bakgrunn i pikselverdier(heltall) er oppgitt i flyttall.

Under er en illustrasjon av hvordan bildet blir segmentert.

Figur 4.8: Horisontalt bilde segmentert

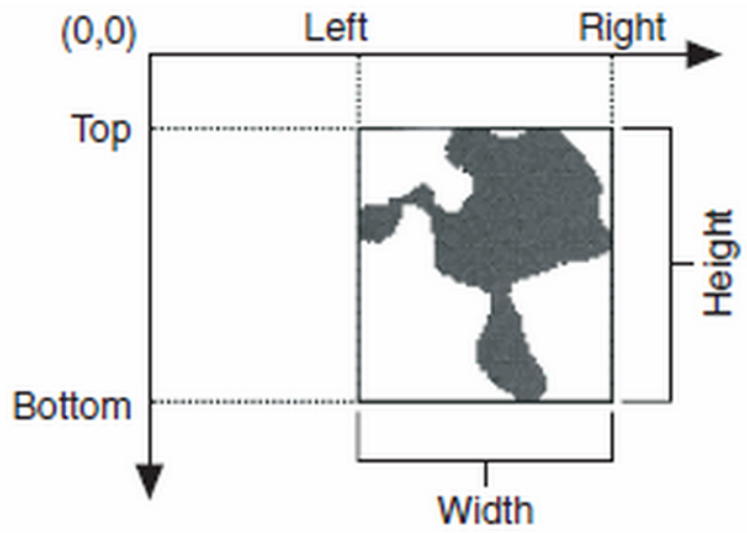
4.2 Filtrering av punkter i bildet og løsning av korrespondanse-problem

4.2.1 Filtrering av punkter i bildet

Det er partikkelanalysen som sender ut data som senere kan benyttes til å filtrere ut de objektene som ikke inneholder den ønskede data og som derfor ansees som støy. Dette kan være mål i henhold til form, størrelse, tyngdepunkt osv.

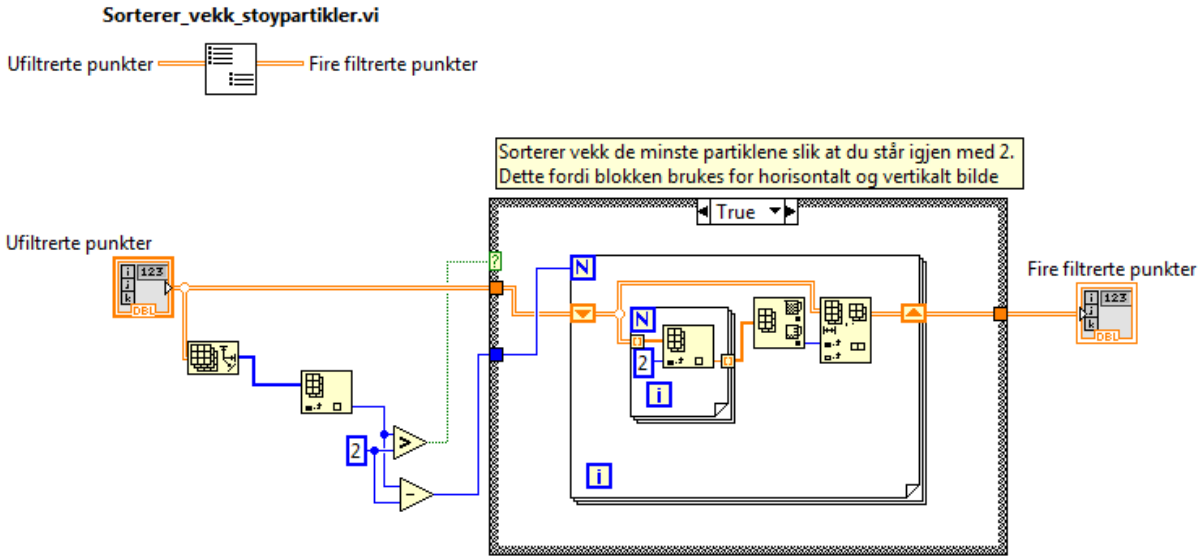
Problemet med de benyttede laserne er at de ikke har muligheten til å fokusere lyset, og de er ikke sirkulære i form. Fordi det er tenkt at deteksjon av laserpunkter og konsentriske kontrasterende sirkler skal foregå på en gang, vil objektplaten som benyttes inneholder hvite og svarte områder. Det er observert forskjeller i hvordan det projiserte lyset reflekteres i henhold til farge på underlaget. Dette vises tydelig i figur 4.8 der den ene laseren reflekteres på det svarte feltet og den andre på det hvite. Dette gjør at de ikke har konstant størrelse.

Selv om laserprikken ikke har konstant størrelse vil de likevel være større en de aller minste støypartiklene som viser seg å være et problem. Det er derfor valgt å filtrere bort de aller minste partiklene ved å sammenligne *Bounding rect diagonal*. Dette er lengden på diagonalen av firkanten som akkurat dekker hele den gjeldende partikkelen, altså $\sqrt{W^2 + H^2}$ [6] i bildet under.



Figur 4.9: *Bounding rectangle*. Firkanten som legges rundt en partikkel i LabVIEW. [6]

Blokken som utfører filteringen er illustert i figuren under.



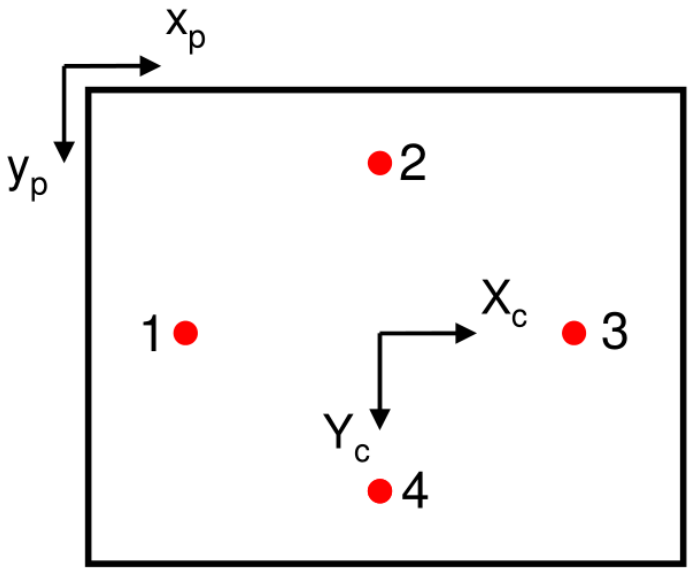
Figur 4.10: Sorterer vekk de minste støypartiklene og står igjen med 2.

Blokken gjør følgende:

- Leser hvor mange partikler som leses av fra partikkelanalysen.
- Dersom det er mer enn 2 partikler fjerner den de partiklene som er minst ved å lese av *bounding rect* for hver partikkel. Grunnen til at det ikke er 4 vil være fordi blokken kjøres på det horisontale og vertikale bildet separat.
- Sender ut de 2 største punktene.

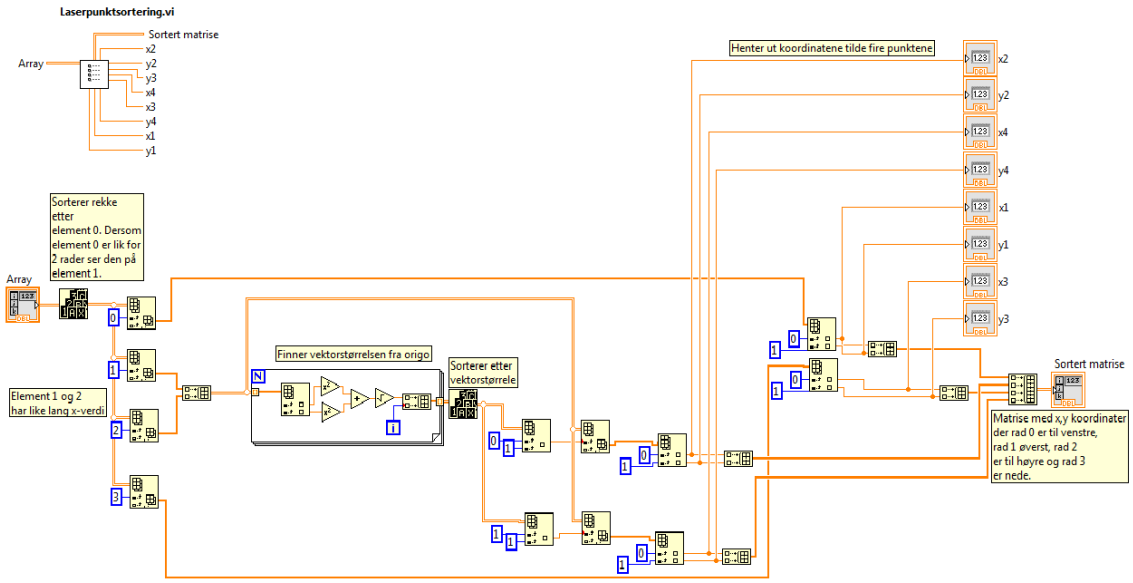
4.2.2 Løsning av korrespondanseproblem

For å kunne utføre trigonometriske beregninger, er det viktig å ha riktig korrespondanse mellom laserprikkene i verdenskoordinater og laserprikkene i bildet. Dette løses enkelt ved å nummerere de fire punktene i bildet som følger:



Figur 4.11: Nummerering av laserprikker i bildet

For å få til dette genereres blokka *Laserpunktsortering* illustrert i figuren under:



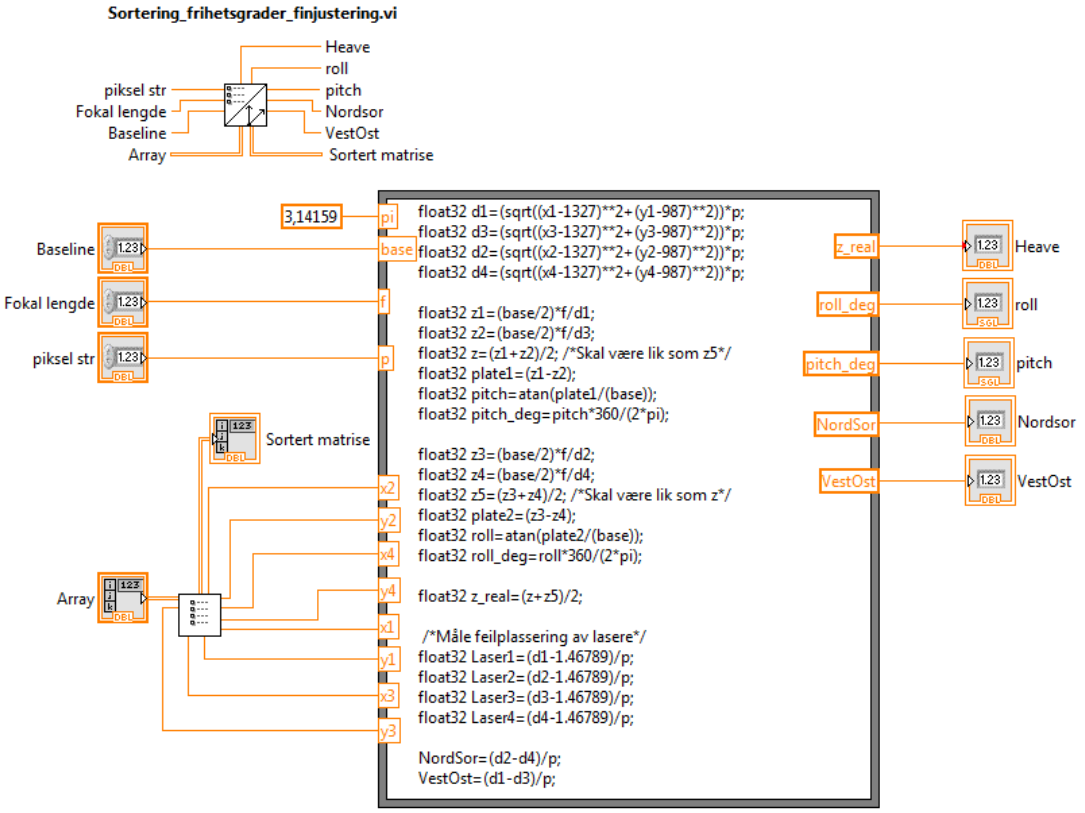
Figur 4.12: Sortering av laserpunkter

Blokken gjør følgende:

- Sorterer de fire laserpunktene (x,y) etter størst x-verdi. Da finner man punkt 1 og 3 i figur 4.11
- Regner ut vektorlengden av punkt 2 og 4
- Sorterer punkt 2 og 4 etter størst vektorlengde
- Sender ut de sorterte koordinatene

4.3 Estimere positur og finjustering av kamera

Roll, *pitch* og *heave* blir beregnet for dette systemet. Formlene som blir benyttet ble gjennomgått i avsnitt 2.3.2. For å regne ut disse parametrene genereres blokka *Sortering_frihetsgrader_finjustering* under.



Figur 4.13: Sortering av laserpunktene og utregning av heave, roll og pitch

Blokken gjør følgende:

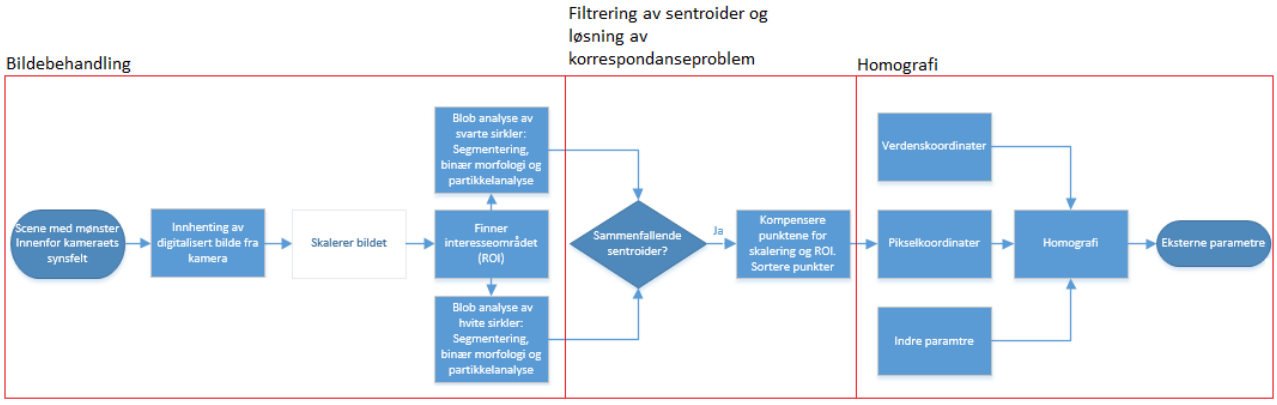
- Henter inn *baseline*, *fokallengde* og *pikselstørrelse*
- Blokken i avsnitt 4.2.2 er en del av denne blokken og sortering av de fire punktene vil derfor inngå her.
- Beregner parametrene *roll*, *pitch* og *heave*
- Beregner parametrene *Nordsor* og *VestOst*

Justeringsparametrene *Nordsor* og *VestOst* blir kun benyttet for finjustering av kamera i plattformen som diskutert i kapittel 3.0.4.

Kapittel 5

Oppbygging av koden til passivt vision system

Koden til det passive vision systemet er bygget opp som vist i blokkskjema 5.1 under.



Figur 5.1: Blokkskjema for det passive vision systemet. De røde rutene illustrer seksjonene i dett kapittelet

De røde firkantene i bildet illustrere hvordan dette kapittelet er bygget opp. Det vil videre i dette kapittelet gis en forklaring på hvordan programkoden i LabVIEW bak de forskjellige blokkene i figur 5.1 er bygget opp.

5.1 Bildebehandling

For at bildebehandlingsekvensen skal fungere må sirklene være innenfor kameraets synsfelt. Videre i underkapittelet vil bearbeidingen av originalbildet beskrives.

5.1.1 Innhenting av digitalisert bilde fra kamera

Kameraet benyttet i dette systemet er likt som for det aktive systemet.

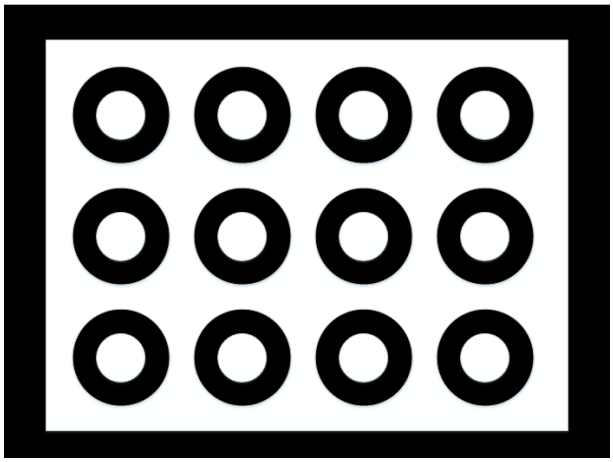
5.1.2 Skalere bildet

For at syklustiden på prosesseringen av hvert enkelt bilde skal være lavere enn 70ms er det kanskje nødvendig å skalere ned bildet. Hvor fort den relative posisjonen skal beregnes kommer selvfølgelig an på applikasjonen som skal kjøres. Kameraet har en oppløsning på 2592x1944 piksler(5MP). Dersom antall piksler i x og y-retning blir halvert, 1296x972, vil iterasjonstiden gå 4 ganger så fort. Om skalering er nødvendig eller ikke kommer an på hvor stor del av bildet som er av interesse(ROI). Ved å skalere ned viktige detaljer i bildet vil detaljene beskrives av færre piksler enn i originalbildet. Det blir en kilde til mer støy i måleresultatene.

5.1.3 Finne interesseområdet, ROI

For å sleppe og skalere ned viktige detaljer i bildet for å holde iterasjonstiden nede, er det gunstig å finne ROI. Da vil bildebehandlingen utføres kun på denne delen av bildet istedenfor hele originalbildet. I dette tilfelle vil ROI bestå av de ni sirklene.

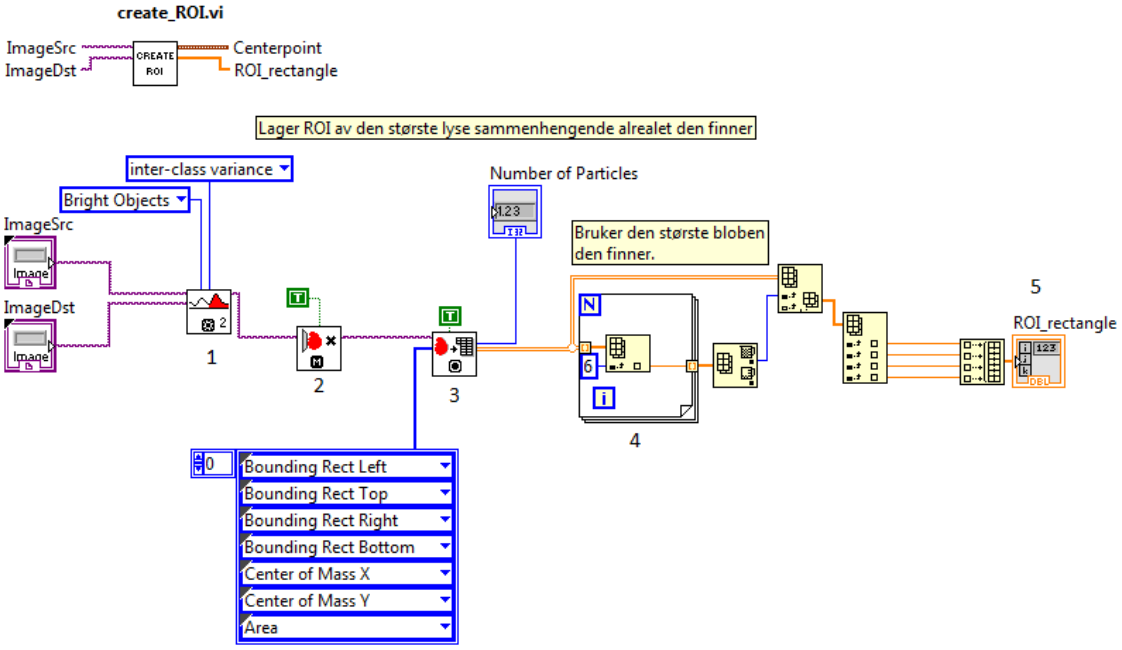
En svart ramme blir lagt rundt de ni sirklene som vist i figur 5.2 under.



Figur 5.2: Svart ramme rundt de 9 konsentriske kotrasterende sirklene for å forenkle oppgaven med å finne ROI

Den svarte firkanten rundt sirklene blir satt på for å sette en kontrasterende grense til den hvite bakgrunnen i bildet. Den hvite flaten mellom den svarte firkanten og de svarte sirklene vil bli behandlet som en sammenhengende blob og vil inneholde ROI. Det faktum at denne hvite flaten vil være den størst hvite flaten i bildet kan benyttes videre.

Blokken som er benyttet for å finne ROI er illustrert i figur 5.3 under.

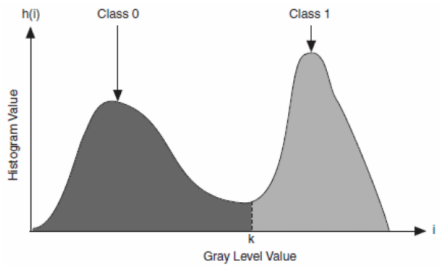


Figur 5.3: Finner ROI

Blokken gjør følgende:

1. Automatisk segmentering. Metoden går ut på å finne en adaptiv terskelverdi. Siden blokken ser på lyse objekter blir de verdiene som er over terskelverdien satt til 255 og de som er under blir satt til 0.

For å automatisk utføre denne tersklingen benyttes det en blokk i LabVIEW som bruker en *Inter-class variance* teknikk. Metoden for å finne den optimale tersklingen blir kalt *Otsus metode* [14] [15]. Metoden går ut på å finne den beste terskelverdien for å skille mellom 2 normalfordelte klasser som vist i figur 5.4.



Figur 5.4: Segmentering av et bilde bestående av 2 normalfordelte klasser med terskelverdi k.

For segmenteringen av CCC-systemet, hvor ROI er et kontrasterende bilde, vil det typisk oppstå et slikt histogram med 2 normalfordelte klasser der klasse 0 er fargen svart og klasse 1 er fargen hvit.

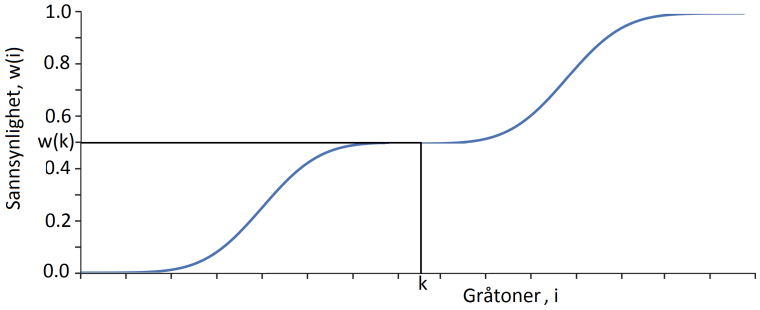
Metoden går ut på å finne *between-class variance*, altså variansen mellom to klasser for alle gråtonevverdier og deretter finne ut hvilke gråtoneverdi som vil være best egnet. i figur 5.4 vil dette være verdien k. Formelen for hvordan Otsus metode er beregnet i LabVIEW blir presentert videre, før en grundigere forståelse av formelen blir beskrevet etter dette.

Between-class variance med Otsus metode i LabVIEW beregnes med formelen [14]:

$$\sigma_B^2(k) = \frac{(\mu_T w(k) - \mu(k))^2}{w(k)(1 - w(k))} \tag{5.1}$$

Bokstav- og funksjonsbeskrivelse:

- k: Gråtoneverdi for terskelverdi
- i: Gråtoneverdi
- N: Antall gråtoneverdier i bildet
- n: Antall piksler i bildet.
- h(i): Antall piksler for hver gråtoneverdi



Figur 5.5: Kummulativ fordelig av normalfordelingen i figur 5.4

- Sannsynligheten for en gråtoneverdi, i, i bildet

$$p(i) = \frac{h(i)}{\sum_{i=0}^{N-1} h(i)} = \frac{h(i)}{n} \tag{5.2}$$

- 1.ordens kummulative moment. Dette vil være gjennomsnittlig antall piksler per gråtoneverdi for klasse 0 i forhold til totalt antall piksler i bildet:

$$\mu(k) = \sum_{i=0}^{k} ip(i) = \sum_{i=0}^{k} \frac{i \cdot h(i)}{n} \tag{5.3}$$

- Gjennomnittlig gråtoneverdi for hele bildet:

$$\mu_T = \sum_{i=0}^{N-1} ip(i) = \sum_{i=0}^{N-1} \frac{i \cdot h(i)}{n} \tag{5.4}$$

- 0.ordens kummulative moment(Sannsynligheten for å treffe klasse 0):

$$w(k) = \sum_{i=0}^{k} p(i) \tag{5.5}$$

Formel 5.1 er den praktiske formelen for hvordan Otsus metode blir benyttet, men er vanskelig å skjønne slik den er oppført.

Formelen er derfor utledet som følger:

$$\begin{aligned}
\sigma_B^2(k) &= \frac{(\mu_T w(k) - \mu(k))^2}{w(k)(1-w(k))} \\
&= w(k)(1-w(k)) \cdot \left(\frac{\mu_T w(k) - \mu(k)}{w(k)(1-w(k))} \right)^2 \\
&= w(k)(1-w(k)) \cdot \left(\frac{\mu_T w(k) - \mu(k) - w(k)\mu(k) + w(k)\mu(k)}{w(k)(1-w(k))} \right)^2 \\
&= w(k)(1-w(k)) \cdot \left(\frac{(\mu_T - \mu(k))w(k) - (1-w(k))\mu(k)}{w(k)(1-w(k))} \right)^2 \\
&= w(k)(1-w(k)) \cdot \left(\frac{(\mu_T - \mu(k))}{1-w(k)} - \frac{\mu(k)}{w(k)} \right)^2
\end{aligned}$$

$$\sigma_B^2(k) = w_0 w_1 (\mu_1 - \mu_0)^2 \tag{5.6}$$

Beskrivelse av 5.6:

- Sannsynligheten for å treffe klasse 0:

$$w_0 = w(k) \tag{5.7}$$

- Sannsynligheten for å treffe klasse 1:

$$w_1 = 1 - w(k) \tag{5.8}$$

- Gjennomsnittlig gråtoneverdi i klasse 0:

$$\mu_0 = \frac{\mu(k)}{w(k)} \tag{5.9}$$

- Gjennomsnittlig gråtoneverdi i klasse 1:

$$\mu_1 = \frac{(\mu_T - \mu(k))}{1 - w(k)} \tag{5.10}$$

Uttrykket 5.6 gir et mer forståelig inntrykk av hvordan variansen mellom klasser regnes ut. Metoden går ut på å finne maksverdien i dette uttrykket. Dette gjøres ved å sjekke denne variansen for alle gråtoneverdier i bildet. Dersom det er 8 bit oppløsning vil dette si å teste gråtoneverdier fra 0 til 255. Denne metoden er god dersom målet er å finne en terskelverdi slik at hver av de 2 klassene som oppstår blir mest mulig homogene samtidig som de 2 blir mest mulig forskjellig. At klassene er mest mulig homogene vil si at variansen i hver av de er minst mulig. At de er mest mulig forskjellig vil si at avstanden mellom middelverdien for de 2 er størst mulig [16].

Den optimale terskelverdien kan settes opp som følger [16]:

$$\sigma_B^2(k_{opt}) = \max_{0 \leqslant k < (N-1)} \sigma_B^2(k) \tag{5.11}$$

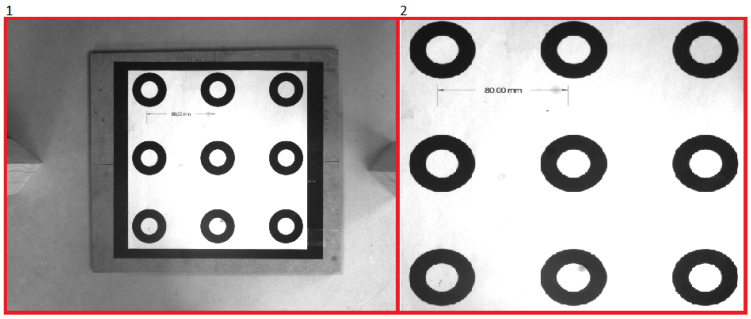
2. Fjerner kantobjekter fra det binære bildet. Dersom ROI med den svarte rammen rundt ligger på en hvit bakgrunn, ville bakgrunnen i værste tilfelle ansees som en større blob enn den hvite bakgrunnen som inneholder ROI. Dersom kantobjekter fjernes vil den hvite bakgrunnen som ble ansett som blob satt til verdien 0 dersom den har kontakt med kantene for synsfeltet til kamera.

3. Partikkelanalysen analyserer objektene og beregner de spesifiserte parametrene for hvert objekt. Det er her spesifisert 7 parametre.

4. Når bildebehandlingen i blokken er over benyttes areal-parameteren, *bounding Rectangle*, i partikkelanalysen for å filtrere ut det største sammenhengende objektet.

5. De 4 første parametrene spesifisert i partikkelanalysen til det største objektet blir sendt videre som parametre i et rektangel som dekker helere ROI. Disse 4 paramtrene vil være de 4 hjørnene i dette rektangelet.

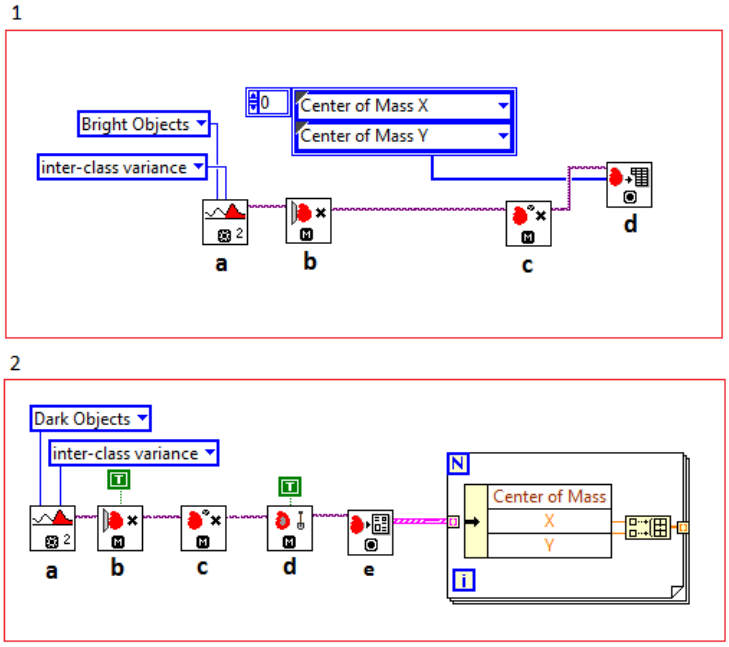
I figur 5.6 under er det illustrert hva blokken gjør med bildet.



Figur 5.6: Finner ROI. Bilde 1 er originalbilde. Bilde 2 er ROI.

5.1.4 Blob analyse

ROI blir videre klippet ut av originalbildet og resten av bildebehandlingen foregår nå i dette vinduet. De ni konsentriske sirkel-parene skal nå detekteres. Når et bilde skal segmenteres kan enten lyse eller mørke objekter bli valgt til å få gråtoneverdien 255. Siden både de svarte og de hvite sirklene nå skal detekteres må det derfor foregå 2 separate blob analyser. Dette er illustrert i figur 5.7 under.



Figur 5.7: 1.ROI hentes inn og blob analyse utføres på små hvite sirkler. 2. ROI hentes inn og blob analyse utføres på store svarte sirkler

1. (a) Automatisk segmentering. Benytter Otsus metode beskrevet i 5.1.3. Denne metoden benyttes fordi det gjerne vil oppstå 2 klasser i bildet, hvit og svart. Siden de hvite innerste sirklene nå skal detekteres vil hvite blobs få verdien 255 og resten 0 i det binære utgangsbildet.

   (b) Kantobjekter fjernes.

   (c) Objekter eroderes med et gitt antall iterasjoner. Dette gjøres for å fjerne de minste støypartiklene.

   (d) Partikkelanalyse. Analyserer de partiklene som er definert som lyse og sender ut sentroidene i x- og y-retning.

2. (a) Automatisk segmentering. Benytter Otsus metode beskrevet i 5.1.3. Siden de svarte ytterste sirklene nå skal detekteres vil svarte blobs få verdien 255 og resten 0 i det binære utgangsbildet.

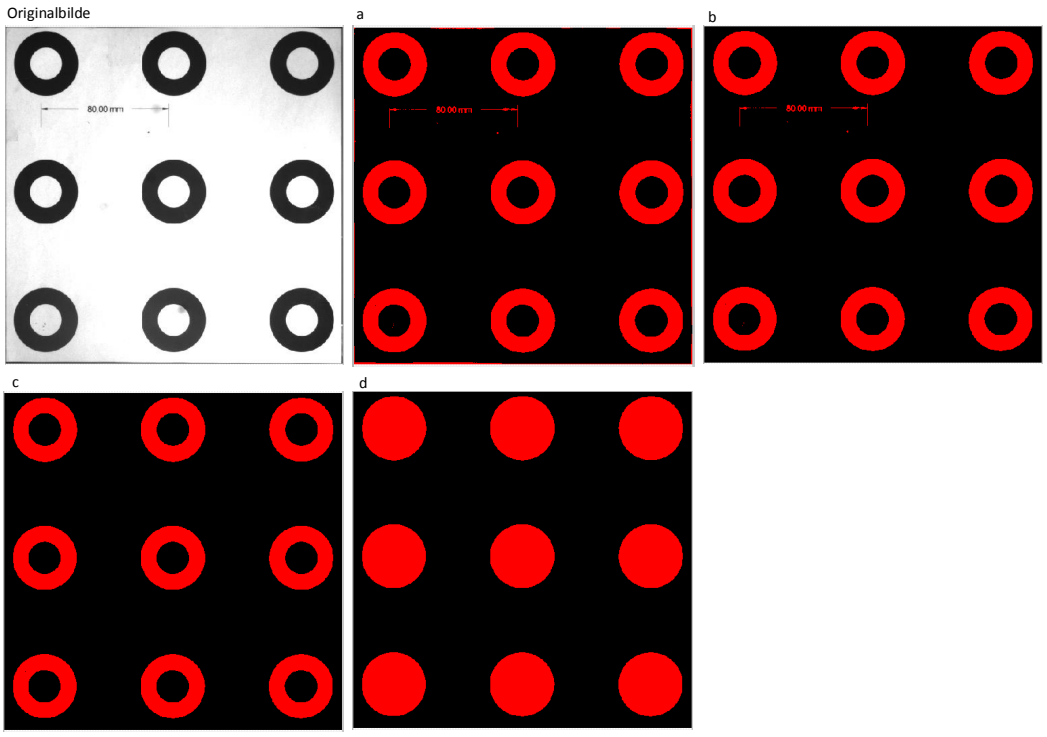
   (b) Kantobjekter fjernes.

   (c) Objekter i bildet eroderes. Dette gjøres for å fjerne de minste støypartiklene.

   (d) Denne blokken fyller hull i et objekt. Et hull vil være piksler med gråtoneverdi 0 inne i et objekt. Dette vil føre til at de lyse sirklene inne i de svarte vil få gråtoneverdi 255 som illustrert i figur 5.8 *d* nedenfor. Grunnen til at den svarte rammen som definerer ROI ikke blir fylt, er at denne rammen alltid vil ligge inntil kanten av bildet og bli fjernet av den forrige blokken i punkt *b*.

   (e) Partikkelanalyse rapport. Denne analysen sender ut en tabell med nyttige paramtre for hver partikkel i bildet. Det er derfor vist hvordan sentroiden plukker ut fra tabellen. Tabellen oppgir målingene med sub-piksel nøyaktighet.

Under er det bilde av hva blokken i den røde firkanten merket *2* i figur 5.7 gjør med bildet.

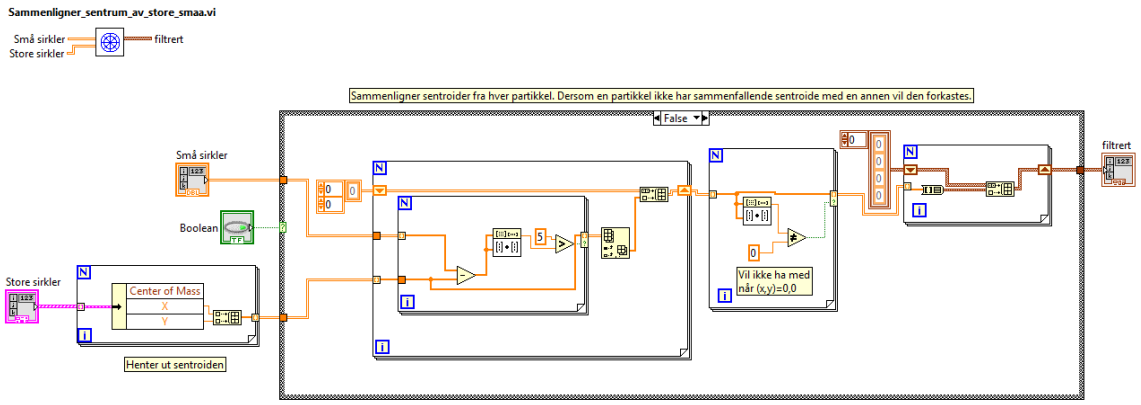


Figur 5.8: Blob analyse utført på de mørke objektene i bildet

5.2 Filtering av sentroider og løsning av korrespondanse-problem

5.2.1 Filtrering av sentroider

Det er kun de 9 sammenfallende sentroidene som er av interesse som punkter for videre utregninger. For å filtrere vekk alle de sentroidene som ikke skal benyttes videre, blir følgende blokk benyttet:



Figur 5.9: Sammenligner sentroidene fra analysen gjort for mørke partikler med sentroidene fra analysen gjort for lyse partikler.

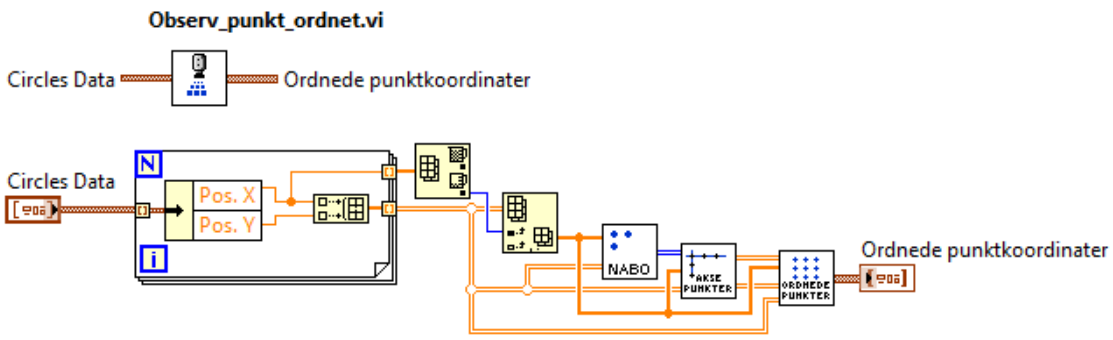
Blokka gjør følgende:

- Henter inn sentroidene fra analysen gjort for små sirkler, altså lyse objekter. Henter inn partikkelrapporten fra analysen gjort for store sirkler, altså mørke objekter. Benytter kun sentroiden for hver sirkel videre
- Finner vektorlengden fra sentroide til sentroide. Merk at det ikke er tatt kvadratrot av lengden $Lengden^2 = x^2 + y^2$. Dette er for å spare prosessorkraft
- Setter toleranseverdi til 5. Dersom $lengden^2 > 5$ blir sentroiden forkastet

De sammenfallende sentroidene går videre som de gyldige pikselpunktene for videre utregning.

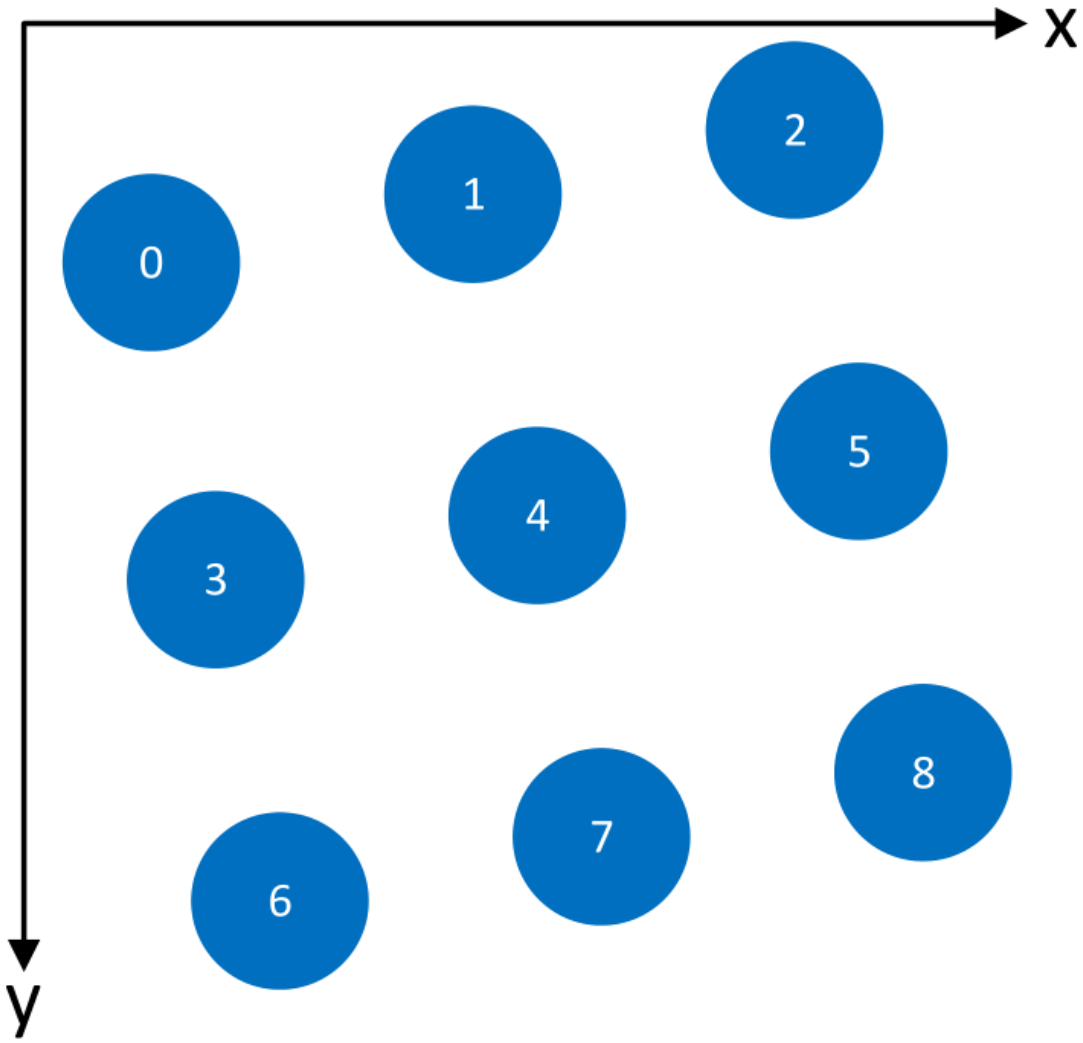
5.2.2 Løsning av korrespondanse-problem

Blokka som tar seg av sortering av sirkelsenterkoordinatene er illustrert i figur 5.10 under. Denne blokka er hentet fra et prosjekt som ble skrevet i faget MAS502 høstsemesteret 2014 [7]. Blokka er nøye inspisert og kan benyttes slik som den står, bortsett fra at alle blokkene har blir forandret til å regne flyttall istedenfor heltall slik at sub-piksel nøyaktigheten opprettholdes. NABO-, AKSE PUNKTER- og ORDNEDE PUNKTER-blokka er også hentet fra dette prosjektet. Programkoden for disse blokkene er tatt med i vedlegg C. Det er kun forklart kort hva sorteringsblokken gjør da det er arbeid gjort av en annen elev.



Figur 5.10: Sorteringsblokka [7]

Blokka henter inn alle de ni sentrumspunktene og sorterer dem slik at punktene får indeksnummerering som vist i figuren under 5.11

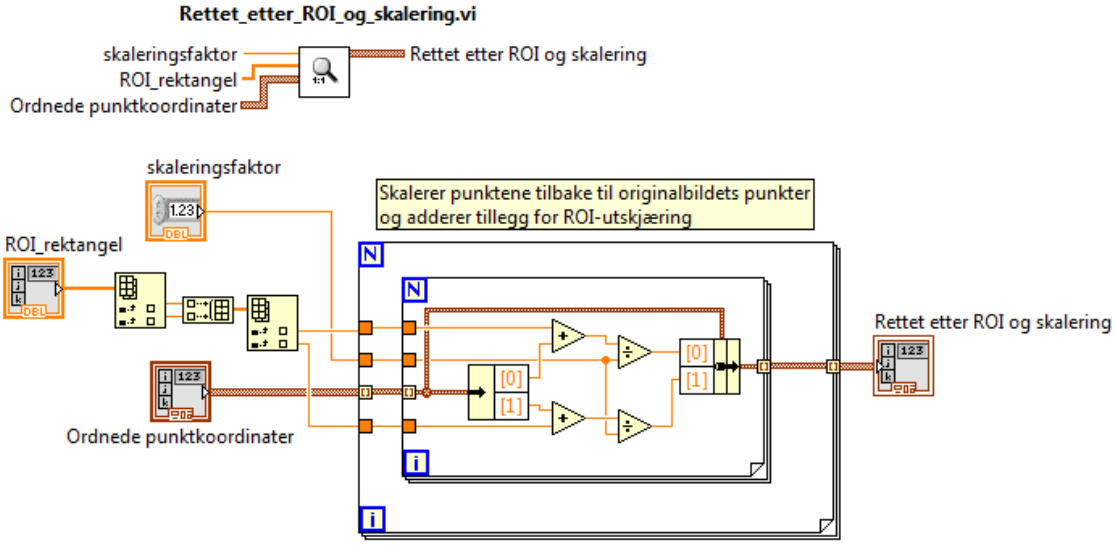


Figur 5.11: Nummerering av de ni sentroidene [7]

Når piskel-koordinatene nå er sortert, er det mulig å opprette en korrespondanse med verdenskoordinatene da de er indeksert på samme måte.

5.2.3 Kompensere punktene for skalering og ROI

Dersom bildet er blitt skalert og ROI er funnet på bakgrunn av det skalerte bildet, må punktene justeres slik at de får den pikselverdien de hadde hatt i originalbildet. Det er da viktig å legge til de pikslene som er klippet ut av ROI først, og deretter skalere punktene tilbake etterpå. Blokken som gjør dette er vist i figur 5.12 under.



Figur 5.12: Blokken som justerer punktene slik at de får pikselverdien de ville hatt før skaleringen av bildet og utklipping av ROI.

Blokken gjør følgende:

- Henter inn parametrene til ROI-rektangelet, skaleringsfaktoren bildet er skaleret med og de ordnede punktene.
- Henter punktet i det øverste venstre hjørnet av ROI-rektangelet. x- og y-verdien i dette punktet blir lagt til x- og y-verdien i hvert ordnede punkt.
- Hvert ordnede punkt blir dividert på skaleringsfaktoren for å få pikselverdien det ville hatt i originalbildet.

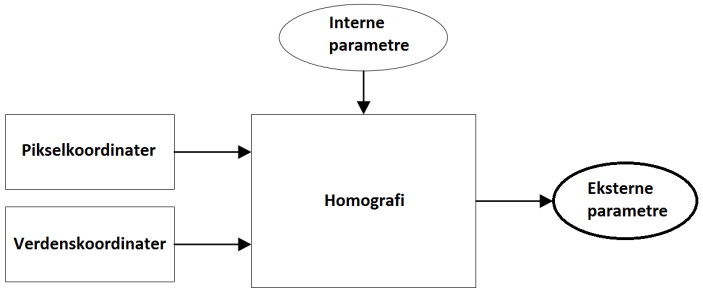
5.3 Estimere positur for CCC-systemet

5.3.1 Homografi

Homografi vil si relasjonen mellom et punkt i verdenskoordinater og det samme punktet i piksel-koordinater. I dette avsnittet vil en stegvis forklaring på hvordan denne transformasjonen tar form bli vist.

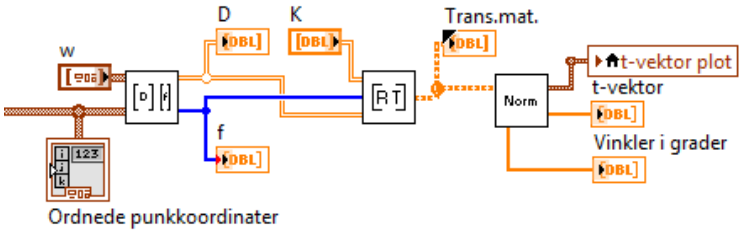
Parametrene som beskriver den relative posituren mellom pikselkoordinatene og verdenskoordinatene vil være de eksterne parametrene. Disse parametrene vil altså si noe om hvor de lokale verdenkoordinatene befinner seg i forhold til kameraets koordinatsystem som diskutert i figur 2.5. De lokale verdenskoordinatene vil bli diskutert videre i dette kapittelet, men illustreres i figur 5.15 sammen med matrisen 5.30.

Et oversiktlig blokkskjema illustrerer hva homografien gjør i figur 5.13. Homografien er avhengig av piksel-koordinatene, de lokale verdenskoordinatene og de interne parametrene til kamera for å generere de eksterne parametrene.



Figur 5.13: Generere ekstrene parametre ved hjelp av homografi [3]

Programkoden for å gjøre disse regneoperasjonene i LabVIEW er illustrert i figur 5.14 under 5.14



Figur 5.14: Oversiktsbilde av blokkene som genererer homografien i LabVIEW [7]

For å finne pikselkoordinatene (u,v) av et punkt i scenen(verdenskoordinater) gjøres følgende:

- Konvertere punktet P_w i verdenskoordinater til et kamerapunkt P_c
- Kamerapunktet, P_c, må deretter transformeres til pikselkoordinater (u,v)

For å transformere kamerapunktet P_c til verdenskoordinater P_w benyttes det homogene koordinater. Dette gjøres for å kunne gjøre en translasjon og en rotasjon i en og samme regneoperasjon. Homogene koordinater skapes ved å legge til en dimensjon i vektoren som vist under:

$$P = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \\ w \end{bmatrix} \; der \; x = \frac{x'}{w}, \; y = \frac{y'}{w} \; pg \; z = \frac{z'}{w} \tag{5.12}$$

RT-matrisen, vist i matrise (2.4), er en homogen matrise. Dersom P_w og P_c gjøres homogene kan RT-matrisen benyttes for gjøre en rotasjon og en translasjon fra P_w tilP_c som vist under:

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \begin{bmatrix} R & T \\ 0^T & 1 \end{bmatrix} \cdot \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = RT \cdot P_w \tag{5.13}$$

For å transformere P_c til pikselkoordinater (u,v) benyttes også homogene koordinater sammen med en skalering, *lambda*. De innvendige paramtrene gitt av K-matrisen vil uttrykke transformasjonsmatrisen mellom pikselkoordinatene og kamerakoordinatene som vist under:

$$\lambda \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & u_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \tag{5.14}$$

Settes uttrykket for P_c i ligning 5.13, inn i uttrykk 5.14 vil transformasjonen mellom et piksel i bildeplanet og det sammenfallende punktet i verdenskoordinater bli som vist under:

$$\lambda * \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & u_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{5.15}$$

Som nevnt i kapittel 2.2.2 vil de utvendige parametrene beskrives med respekt til verdenskoordinatene. Dette betyr at $Z_w = 0$, noe som vil føre til at kolonne 3 i rotasjonsmatrisen blir 0 som visst under.

$$\lambda \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & u_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} r_{11} & r_{12} & 0 & t_x \\ r_{21} & r_{22} & 0 & t_y \\ r_{31} & r_{32} & 0 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_w \\ y_w \\ 0 \\ 1 \end{bmatrix} \tag{5.16}$$

Den komprimerte ligningen når Z_w er fjernet er vist under:

$$\lambda \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & u_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{bmatrix} \cdot \begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} \tag{5.17}$$

Utifra disse ligningene kommer det frem hvordan homografimatrisen, altså transformasjonsmatrisen for å omdanne P_w til en pikselverdi i bildeplanet, (u,v), vil bil:

$$H = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & u_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{5.18}$$

$$\lambda \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \cdot \begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} \tag{5.19}$$

Dersom ligning 5.19 multipliseres ut får vi et sett med tre ligninger:

$$\lambda \cdot u = h_{11} \cdot x_w + h_{12} \cdot y_w + h_{13} \tag{5.20}$$

$$\lambda \cdot v = h_{21} \cdot x_w + h_{22} \cdot y_w + h_{23} \tag{5.21}$$

$$\lambda = h_{31} \cdot x_w + h_{32} \cdot y_w + h_{33} \tag{5.22}$$

Deretter reorganiseres disse ligningene ved å substituere λ i ligning 5.20 og 5.21 med λ i ligning 5.22. Dette gir ligningene som beskriver det samme punktet i pikselkoordinater (u,v) og kamerakoordinater (X_w,Y_w) som vist i ligningene under

$$h_{11} \cdot x_w + h_{12} \cdot y_w + h_{13} - u \cdot h_{31} \cdot x_w - u \cdot h_{32} \cdot y_w - u \cdot h_{33} = 0 \tag{5.23}$$

$$h_{21} \cdot x_w + h_{22} \cdot y_w + h_{23} - v \cdot h_{31} \cdot x_w - v \cdot h_{32} \cdot y_w - v \cdot h_{33} = 0 \tag{5.24}$$

Grunnet proporsjonalitet kan element h_{33} settes lik 1. Ligningene vil da se ut som følger:

$$h_{11} \cdot x_w + h_{12} \cdot y_w + h_{13} - u \cdot h_{31} \cdot x_w - u \cdot h_{32} \cdot y_w = u \tag{5.25}$$

$$h_{21} \cdot x_w + h_{22} \cdot y_w + h_{23} - v \cdot h_{31} \cdot x_w - v \cdot h_{32} \cdot y_w = v \tag{5.26}$$

Introduserer så vektoren h:

$$\begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{21} & h_{22} & h_{23} & h_{31} & h_{32} \end{bmatrix}^T \tag{5.27}$$

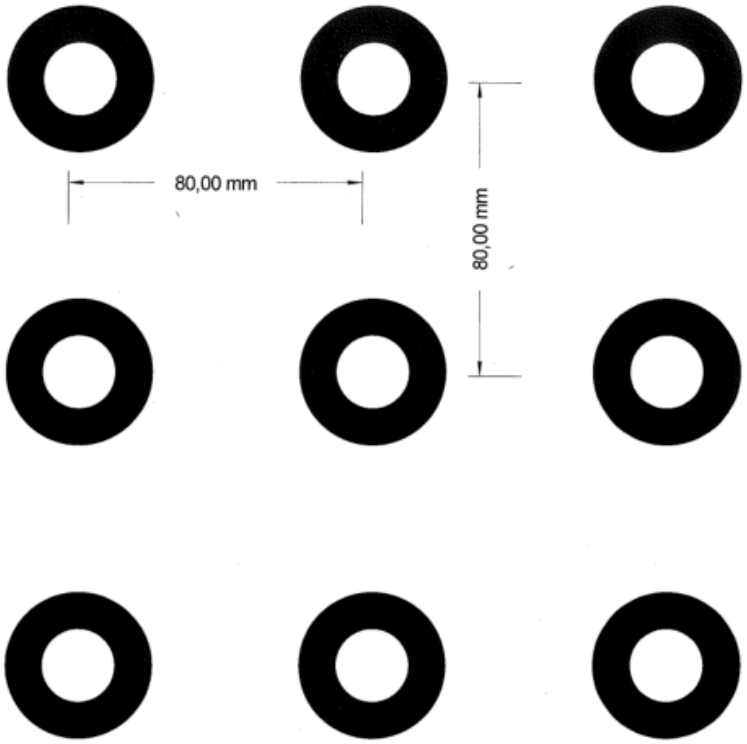
Dersom disse ligningene nå blir organisert tilbake i et uttrykk der vektoren h står alene, vil uttrykket kunne skrives som følger:

$$\begin{bmatrix} x_{w1} & y_{w1} & 1 & 0 & 0 & 0 & -x_{w1} * u_1 & -y_{w1} * u_1 \\ 0 & 0 & 0 & x_{w1} & y_{w1} & 1 & -x_{w1} * v_1 & -y_{w1} * v_1 \\ x_{w2} & y_{w2} & 1 & 0 & 0 & 0 & -x_{w2} * u_2 & -y_{w2} * u_2 \\ 0 & 0 & 0 & x_{w2} & y_{w2} & 1 & -x_{w2} * v_2 & -y_{w2} * v_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{wn} & y_{wn} & 1 & 0 & 0 & 0 & -x_{wn} * u_n & -y_{wn} * u_n \\ 0 & 0 & 0 & x_{wn} & y_{wn} & 1 & -x_{wn} * v_n & -y_{wn} * v_n \end{bmatrix} \cdot \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ u_1 \\ v_1 \\ \vdots \\ \vdots \\ u_n \\ v_n \end{bmatrix} \tag{5.28}$$

$$D \cdot h = f \tag{5.29}$$

I uttrykk 5.28 vil (x_{w1}, y_{w1}) være (x,y) koordinatene til det første punktet i verdenskoordinater lokalt. I dette systemet tilsier altså dette da sentrumskoordinatet til den første sirkelen på objektet som skal observeres. Dette punktet tilsvarer pikselkoordinatet (u_1, v_1) i bildeplanet. De lokale verdenskoordinatene er vist i matrisen under, sammen med en illustrasjon av sirklene som skal detekteres.

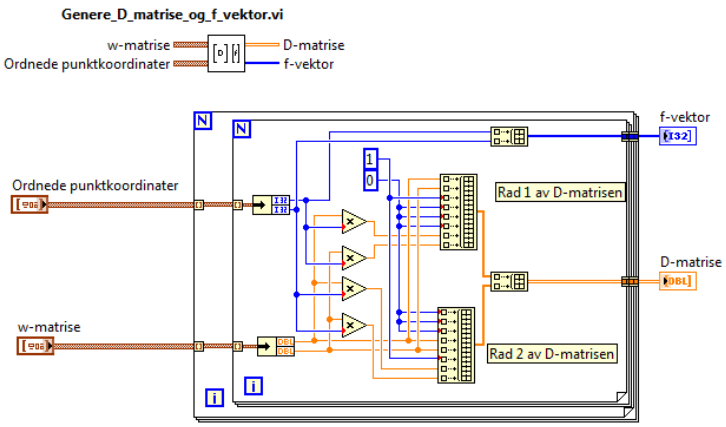
$$w = \begin{bmatrix} x_{w1} & x_{w2} & x_{w3} \\ y_{w1} & y_{w2} & y_{w3} \\ x_{w4} & x_{w5} & x_{w6} \\ y_{w4} & y_{w5} & y_{w6} \\ x_{w7} & x_{w8} & x_{w9} \\ y_{w7} & y_{w8} & y_{w9} \end{bmatrix} = \begin{bmatrix} 0 & 80 & 160 \\ 0 & 0 & 0 \\ 0 & 80 & 160 \\ 80 & 80 & 80 \\ 0 & 80 & 160 \\ 160 & 160 & 160 \end{bmatrix} \tag{5.30}$$



Figur 5.15: w-matrisen er basert på avstanden mellom sirkelsentrene. 80mm mellom sirkelsentrene vertikalt og horisontalt.

Origo vil være plassert i sentrum av det kontraherende sirkel-paret øverst til venstre i figur 5.15.

Blokka som henter ut D-matrisen og f-vektoren er vist i figur 5.16 under.



Figur 5.16: Koden til blokka som beregner D-matrisen og f-vektoren [7]

Videre må uttrykket for h-vektoren generes med hensyn på D-matrisen. Det vil si at D-matrisen må flyttes over på andre siden av likhetstegnet. Siden matrisen ikke er kvadratisk, vil det ikke være mulig å invertere den. Derfor må den multipliseres med den transponerte først så den blir kvadratisk, for så å invertere. Dette regneuttrykket kalles *pseudoinvers* av D-matrisen, altså D^+ og er illustrert under:

$$h = (D^T D)^{-1} \cdot D^T \cdot f = D^+ \cdot f = \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} \tag{5.31}$$

Vektoren h i uttrykk 5.31 kan skrives på matriseform H:

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{5.32}$$

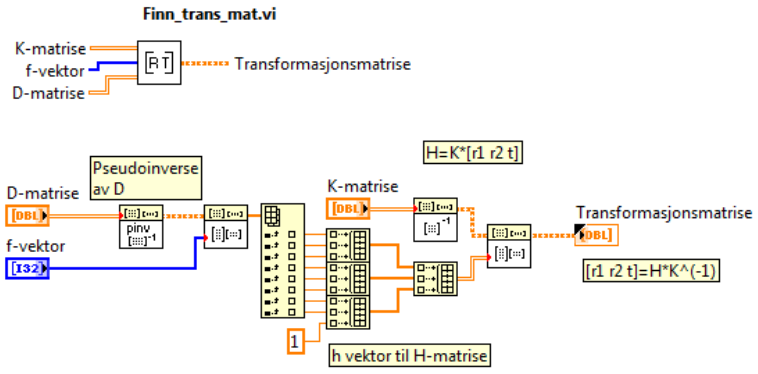
Det er transformasjonsmatrisen(RT-matrisen) som inneholder den relative rotasjon og translasjon som er de interessante måleverdiene for programkoden. Fra formel 5.18 kommer det frem at:

$$H = K * \begin{bmatrix} r_1 & r_2 & t \end{bmatrix} \tag{5.33}$$

Siden K-matrisen er kvadratisk vil inversen av K kunne regnes ut og utrykket i 5.33 kan settes opp som følger:

$$K^{-1} \cdot H = \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{bmatrix} \tag{5.34}$$

Blokka som tar inn *D-matrisen*, *f-vektoren* og *K-matrisen* og genererer *transformasjonsmatrisen* er illustrert under:



Figur 5.17: Koden til blokka som beregner transformasjonsmatrisa(RT), modifisert utgave [7]

Normalisering av rotasjonsmatrisen

Rotasjonsmatrisen
Rotasjonsmatrisen er en ortonormal matrise. Dette betyr at kolonnevektorene i matrisen er enhetsvektorer(lengde på en), og de må stå normalt på hverandre [17].

For å finne ut om R_1 og R_2 er ortogonale(står vinkelrett på hverandre) tar man prikkproduktet mellom dem som vist under.

$$Feilmargin = R_1^T \cdot R_2 = \begin{bmatrix} r_{11} & r_{21} & r_{31} \end{bmatrix} \cdot \begin{bmatrix} r_{12} \\ r_{22} \\ r_{32} \end{bmatrix} \tag{5.35}$$

Dersom prikkproduktet ikke blir null er det vanskelig å si om det er R_1 eller R_2 som er skeiv. Derfor fordeles feilen likt mellom de to som vist under.

$$R_{1orto} = R_1 - (\frac{Feilmargin}{2})R_2 \tag{5.36}$$

$$R_{2orto} = R_2 - (\frac{Feilmargin}{2})R_1 \tag{5.37}$$

For å finne R_3 slik at den er ortogonal med R_{1orto} og R_{2orto} tar man kryssproduktet mellom de to som vist under.

$$R_{3orto} = R_{1orto} \times R_{2orto} \tag{5.38}$$

For å være sikker på at vektorene er enhetsvektorer finnes det to metoder. Den ene er å dele på lengden av vektoren som vist under.

$$\begin{bmatrix} \frac{R_{1orto}}{\|R_{1orto}\|} & \frac{R_{2orto}}{\|R_{2orto}\|} & \frac{R_{3orto}}{\|R_{3orto}\|} \end{bmatrix} = \begin{bmatrix} R_{1norm} & R_{2norm} & R_{3norm} \end{bmatrix} \tag{5.39}$$

Absoluttverdi krever både kvadratrotoperasjon og potensoperasjon. For å spare prosessorkraft kan følgende metode benyttes isteden:

$$R_{1norm} = \frac{1}{2}(3 - R_{1orto} \cdot R_{1orto})R_{1orto} \tag{5.40}$$

$$R_{2norm} = \frac{1}{2}(3 - R_{2orto} \cdot R_{2orto})R_{2orto} \tag{5.41}$$

$$R_{3norm} = \frac{1}{2}(3 - R_{3orto} \cdot R_{3orto})R_{3orto} \tag{5.42}$$

Rotasjonsmatrisen er nå ortonormal.

Rotasjonene er definert i figur 2.1 og kan settes opp i komprimert form som vist under:

$$R = R_x(\theta_{roll}) \cdot R_x(\theta_{pitch}) \cdot R_x(\theta_{yaw}) \tag{5.43}$$

Dersom 5.43 utvides:

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_{roll}) & -\sin(\theta_{roll}) \\ 0 & \sin(\theta_{roll}) & \cos(\theta_{roll}) \end{bmatrix} \cdot \begin{bmatrix} \cos(\theta_{pitch}) & 0 & \sin(\theta_{pitch}) \\ 0 & 1 & 0 \\ -\sin(\theta_{pitch}) & 0 & \cos(\theta_{pitch}) \end{bmatrix} \cdot \begin{bmatrix} \cos(\theta_{yaw}) & -\sin(\theta_{yaw}) & 0 \\ \sin(\theta_{yaw}) & \cos(\theta_{yaw}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.44}$$

Ved å multiplisere ut 5.44:

$$R = \begin{bmatrix} C_yC_p & C_yS_pS_r - S_yC_r & C_yS_pC_r + C_yS_r \\ S_yC_p & S_yS_pS_r - C_yC_r & S_yS_pC_r - C_yC_r \\ -S_p & C_pS_r & C_pC_r \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{5.45}$$

Beskrivelse av elementene i 5.45 listet opp under:

- $C_r = \cos(\theta_{roll})$
- $C_p = \cos(\theta_{pitch})$
- $C_y = \cos(\theta_{yaw})$
- $S_r = \sin(\theta_{roll})$
- $S_p = \sin(\theta_{pitch})$
- $S_y = \sin(\theta_{yaw})$

For å finne vinklene i radianer benyttes formlene [3]:

$$\theta_{roll} = \arctan 2(\frac{-r_{23}}{r_{33}}) \tag{5.46}$$

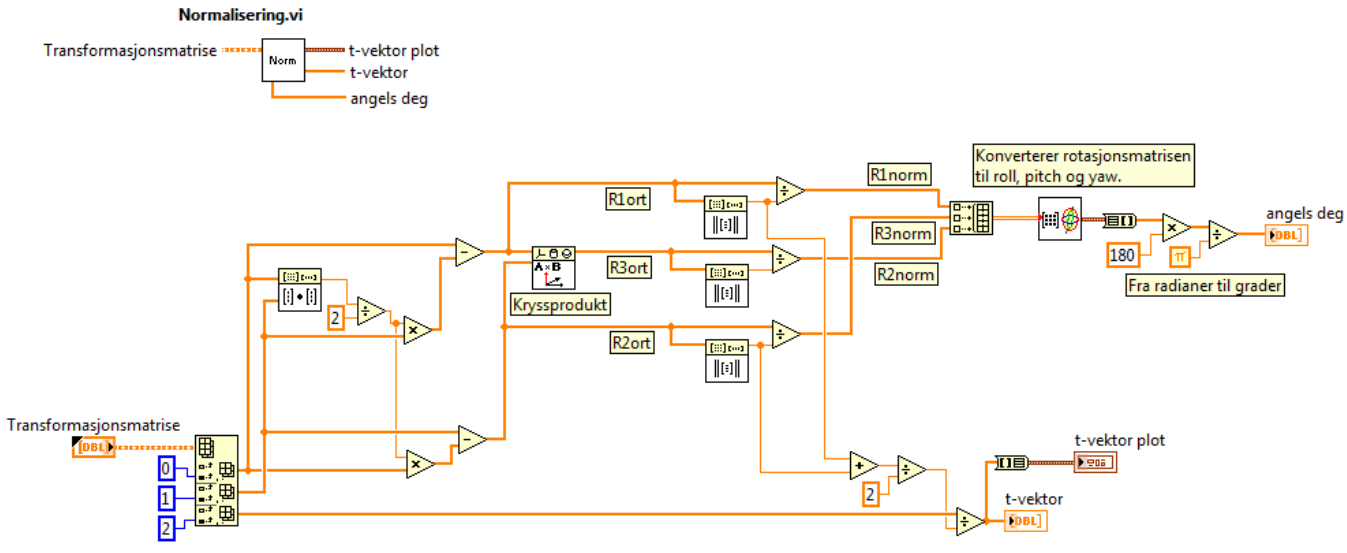
$$\theta_{pitch} = \arctan 2(\frac{r_{13}}{\cos(\theta_r) \cdot r_{33} - \sin(\theta_r) \cdot r_{23}}) \tag{5.47}$$

$$\theta_{yaw} = \arctan 2(\frac{-r_{12}}{r_{11}}) \tag{5.48}$$

Translasjonsvektoren
Fordi r_1 og r_2 er normalisert må translasjonsvektoren også deles på en faktor for at den skal ha samme skaleringen som rotasjonen. Siden skaleringen kan være noe forskjellig i r_1 og r_2, beregnes gjennomsnittslengden for de to, som benyttes som skaleringsfaktor for translasjonen som vist under.

$$T = \frac{t}{\frac{\|r_1\|+\|r_2\|}{2}} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{5.49}$$

Blokka som utfører normaliseringen er illustrert under.



Figur 5.18: Blokka som normaliserer transformasjonsmatrisen(RT-matrisen) og konverterer rotasjonsmatrisen til roll, pitch og yaw.

Normalisering av pikselkoordinatene og kompensering for radiell forvrenging

For å kompensere for radiell forvrengning ble formlene 2.1 og 2.2 generert i LabVIEW. For å sleppe å bla frem og tilbake i rapporten er formlene gjengitt som følger:

$$\begin{aligned} x_n &= \frac{x_{nd}}{1 + k_1 r^2 + k_2 r^4 + k_3 r^6} \\ y_n &= \frac{y_{nd}}{1 + k_1 r^2 + k_2 r^4 + k_3 r^6} \end{aligned} \tag{2.1}$$

$$r^2 = x_{nd}^2 + y_{nd}^2 \tag{2.2}$$

For å kunne generere denne koden må x_{nd} og y_{nd} defineres. Dette vil være det normaliserte pikselkoordinatet med prinsipalpunktet som origo, og utsatt for forvrenging(distortion). Ligning 5.17 repeteres under:

$$\lambda \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & u_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{bmatrix} \cdot \begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} \tag{5.17}$$

Dersom K-matrisen flyttes over på andre siden av likhetstegnet og glemmer λ litt, vil venstresiden se ut som følger:

$$K^{-1} \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{f_u} & 0 & -\frac{u_0}{f_u} \\ 0 & \frac{1}{f_v} & -\frac{v_0}{f_v} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{f_u - u_0}{f_u} \\ \frac{f_v - v_0}{f_v} \\ 1 \end{bmatrix} = \begin{bmatrix} x_{nd} \\ y_{nd} \\ 1 \end{bmatrix} \tag{5.50}$$

Uttrykket 5.17 vil derfor se ut som følger:

$$\lambda \cdot \begin{bmatrix} x_{nd} \\ y_{nd} \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{bmatrix} \cdot \begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} \tag{5.51}$$

Dette betyr at dersom pikselkoordinatene blir normalisert og sentrert rundt prinsipalpunktet vil H-matrisen være lik *RT-matrisen* som følger:

$$H = \begin{bmatrix} r_1 & r_2 & t \end{bmatrix} \tag{5.52}$$

En grunn til å normalisere pikselkoordinatene rundt prinsipalpunktet vil være for å gjøre matrisen med pikselkoordnater bedre tilpasset endringer. Dette er sterkt anbefalt av Hartley og Zissermann [3]. Ved å normalisere matrisene som har med utregningen av D-matrisen å gjøre, vil D-matrisen bli mer tilpasset endringer. Hvor godt en matrise er tilpasset endringer kan relateres til *kondisjonstall* eller på engelsk *condition number*. Et høyt kondisjonstall vil øke effekten av støy i måledataen. Dersom dette tallet er lavt vil effekten av støy forminskes. Hartley og Zissermann anbefaler å beregne koordinatene slik at sentroiden(tyngdepunktet) til matrisen med alle koordinatene vil være origo og gjennomsnittlig distanse til origo lik $\sqrt{2}$ [3].

Det er dette som blir gjort med de sorterte pikselkoordinatene illustrert i blokken i figur 5.19 under. Verdenskoordinatenes nullpunkt blir også flyttet til matrisens sentroide. Dette gjør kondisjonstallet på matrisen lavere og vil også sørge for at beregningen av translasjon i x- og y vil være 0 dersom kameraet står direkte ovenfor den midterste av de konsentriske kontrasterende sirklene. Origo ble i utgangspunktet plassert øverst til venstre, slik som vist i figur 5.15.

Tilpasningsnummeret er definert som [18]:

$$Kondisjonstall = \frac{|relativ\ utgangsfeil|}{|relativ\ ingangsfeil|} \tag{5.53}$$

D-matrisen er et resultat av pikselkoordinater og verdenskoordinater som repetert i 5.28 under. Verdenskoordinatene er konstante mens pikselkoordinatene blir målt til en ny verdi hver gang et nytt bilde blir prosessert. Kondisjonstallet til D bestemmer derfor hvor mye støy måling av pikselkoordnatene har å si for hvor mye feil det blir i utregningen av D-matrisen.

$$D = \begin{bmatrix} x_{w1} & y_{w1} & 1 & 0 & 0 & 0 & -x_{w1} * u_1 & -y_{w1} * u_1 \\ 0 & 0 & 0 & x_{w1} & y_{w1} & 1 & -x_{w1} * v_1 & -y_{w1} * v_1 \\ x_{w2} & y_{w2} & 1 & 0 & 0 & 0 & -x_{w2} * u_2 & -y_{w2} * u_2 \\ 0 & 0 & 0 & x_{w2} & y_{w2} & 1 & -x_{w2} * v_2 & -y_{w2} * v_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{wn} & y_{wn} & 1 & 0 & 0 & 0 & -x_{wn} * u_n & -y_{wn} * u_n \\ 0 & 0 & 0 & x_{wn} & y_{wn} & 1 & -x_{wn} * v_n & -y_{wn} * v_n \end{bmatrix} \tag{5.28}$$

Dette sees i sammenheng med utregningen av h-vektoren(som gir posituren av platen) repetert i formelen 5.31 under:

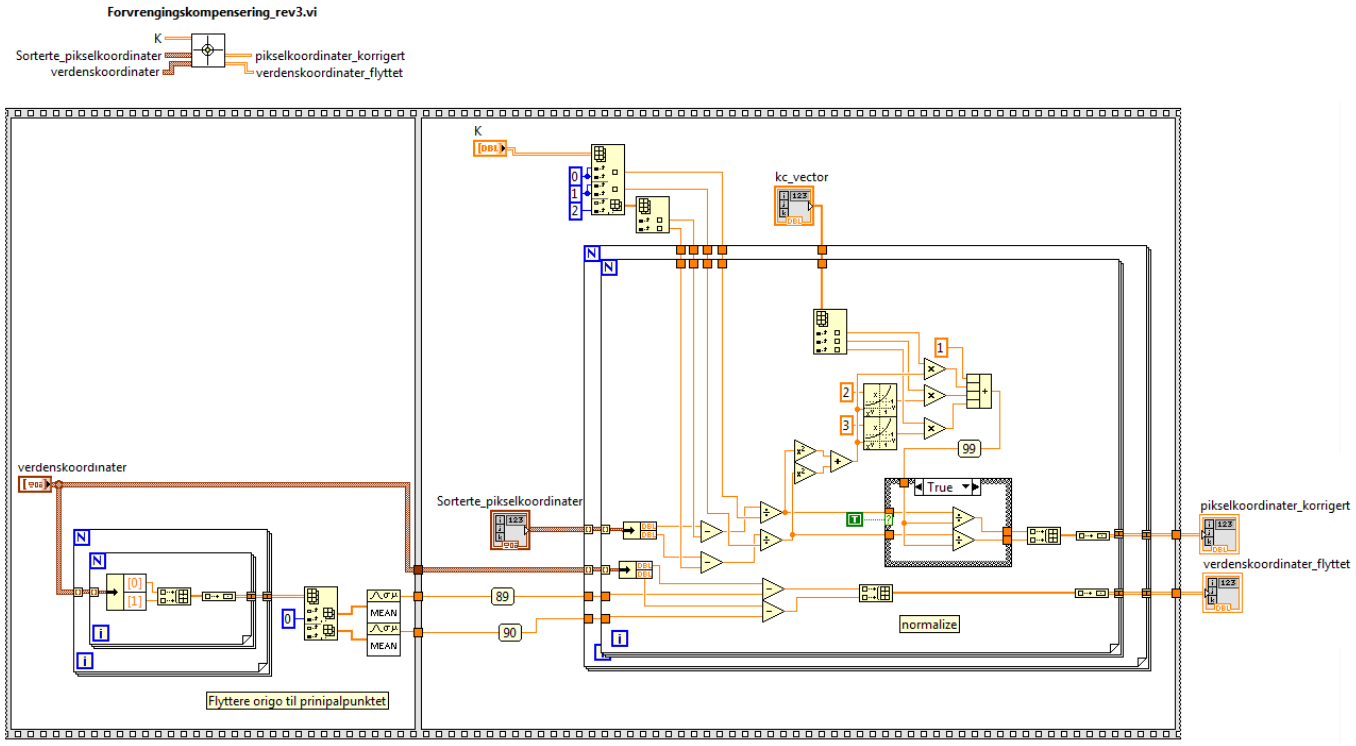
$$h = D^{+} \cdot f \tag{5.31}$$

I uttrykket over vil h være relatert til utgangsfeil og f til inngangsfeil, og effekten av kondisjontallet til D kan tolkes som følger:

$$Kondisjonstall(D) = \frac{|relativ\ feil\ i\ h|}{|relativ\ feil\ i\ f|} \tag{5.54}$$

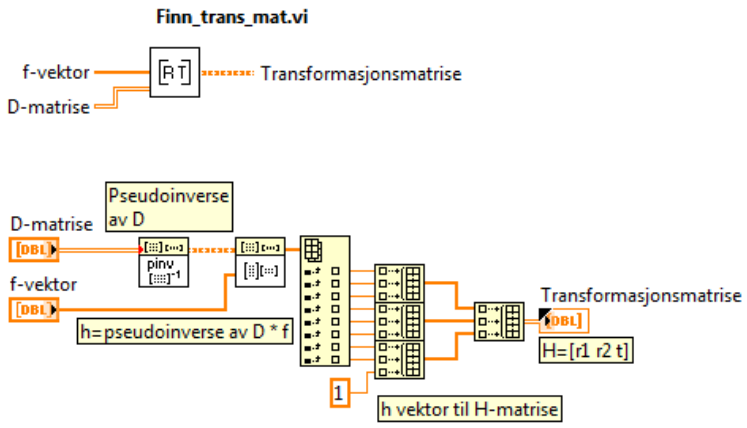
Det tolkes derfor at et høyt kondisjonstall på D-matrisen vil gi en relativt stor feil i h for en liten feil i f.

Blokken som tar seg av normaliseringen av pikselkoordinatene og kompensering for radiell forvrengning er vist i figuren under:



Figur 5.19: Blokka som utfører normalisering av pikselkoordinatene rundt prisipalpunktet og kompeneserer for radiell forvrengning.

Ved å benytte seg av denne metoden må blokken som beregner transformasjonsmatrisen (figur 5.17) endres. Siden K-matrisen er tatt med i utregningen av de normaliserte pikselkoordinatene, skal ikke denne være med videre. Derfor er den nye blokken helt lik bortsett fra at K-matrisen ikke er med, og ser ut som følger:



Figur 5.20: Blokka som utfører overgangen fra f-vektor og D-matrise til H-matrisen. Lik som figur 5.17, men nå er H-matrisen=RT-matrisen

Det er gjort forøk for å se på effekten av å normalisere pikselkoordinatene og kompensere for radiell forvrenging. Dette blir behandlet i kapittel 7.1.2

Kapittel 6

Testoppsett

6.1 Utstyr

6.1.1 Kamerasystemet

Det er kun benyttet et kamera til testing og utviklingen av programvarene. Navnet på kamera er *Basler ace acA2500-14um* [19]. Det er et USB3.0 monokromt(svart/hvitt) kamera med CMOS bildesensor og noe som på engelsk kalles *rolling shutter*, som betyr at kamera leser av pikslene rad for rad. Det kan hente inn 14fps, altså 14 bilder per sekund. Pikseldybden er på 12 bit. Bildet kan representeres med minimum 8 bit, altså representeres hver piksel med $2^8 = 256$ gråtoneverdier.

Linsen er av typen $computar M1214 - MP22/3 Fixed Lens(12mm)$. Den har fokallengde på 12mm. Blenderåpningen kan justeres fra f-nummer=1.4 til fullstendig lukket.

6.1.2 Datamaskin og programvare

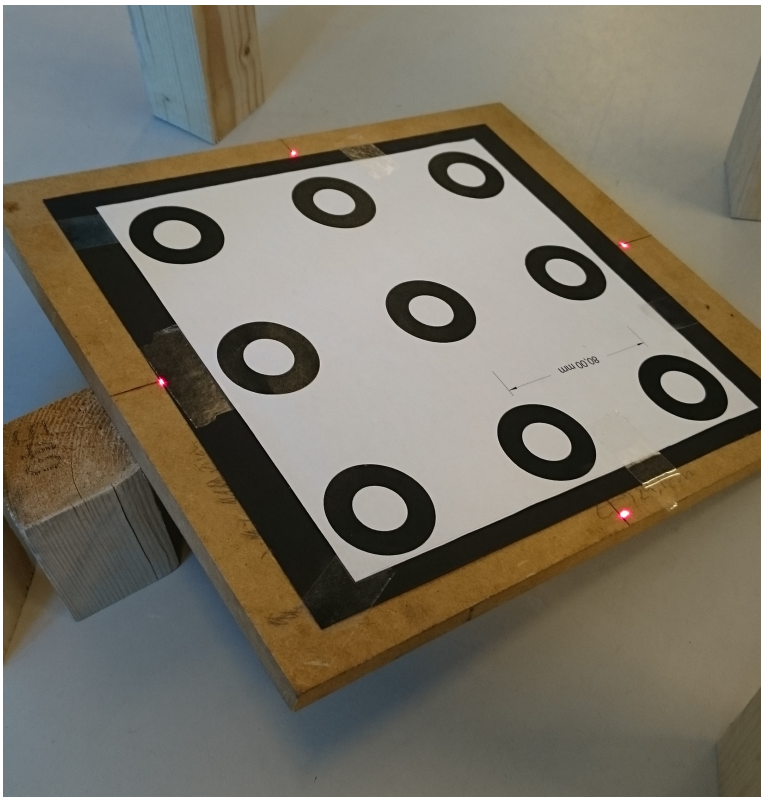
Programvaren er generert i LabVIEW av National Instruments. Datamaskinen som er benyttet for å kjøre koden har følgende spesifikasjoner:

- Intel(R) Core(TM) i7-4510U CPU @ 2.00GHz 2.60GHz
- 6.00 GB RAM
- Windows 7 Professional

6.2 Testbenk

Det er laget to systemer som skal testes opp mot det som ble som ble utviklet under masteroppgaven i fjor [3]. Hovedhensikten med rapporten er å lage et system som er mer robuste med tanke på varierende lysforhold og støy. Testbenken, se kapittel 3.0.3, benyttes videre for å teste vision systemene.

For å teste vinkelpresisjon er det benyttet klosser som settes under en plate, slik at platen blir liggende i en vinkel til gulvet. Med kjente størrelser på plate og kloss kan vinkel regnes ut ved å benytte trigonometri. Laserstrålene benyttes som peilere for å se at platen er rotert om riktig akse som vist i figuren under.



Figur 6.1: Plate med 15° rotasjon om x-aksen. Dette er justert inn ved å benytte laserprikkene som peilere på aksene

Kapittel 7

Eksperimenter og resultater

For å teste om de to nye systemene er bedre utviklet mot støy og varierende lysforhold gjøres det forskjellige eksperimenter. De tre systemene blir videre i dette kapittelet testet opp mot hverandre og måledataen blir diskutert. Siden det er 2 passive og et aktivt vision system, vil de 2 passiv bli referert til som CCC-systemet og hjørnesystemet. Det aktive blir refert til som lasersystemet.

7.1 Tilfeldig feil i målesignalet

Det finnes 3 forskjellige typer feil som kan inntreffe i signalet. Tilfeldig, dynamisk og statisk/systematisk feil. I dette forsøket testes systemet for tilfeldig feil, altså støy i signalet. Støy er en form for feil i signalet som har en gjennomsnittsverdi på 0. Dette betyr at støy ikke kan kompenseres vekk, slik statisk og dynamisk feil kan.

Standardavvik altså kvadratroten av variansen, er et mål på variasjon, altså hvor mye et målesignal varierer i størrelse. Dersom standardavviket blir målt på positurmålinger av et objekt som ligger helt i ro med konstante lysforhold vil dette være et godt mål for støyen i bildet. For å kunne benytte resultatene fra testing for å sammenligne de 3 systemene må lysforholdene være like. Testene ble derfor gjort i et rom uten vinduer og uten bevegelser som kan skape skygger. Blenderåpningen var uendret for testing på de tre ulike systemene. For å måle standardavviket er det benyttet en blokk i LabVIEW. Denne blokken måler standardavviket med følgende formel [20]:

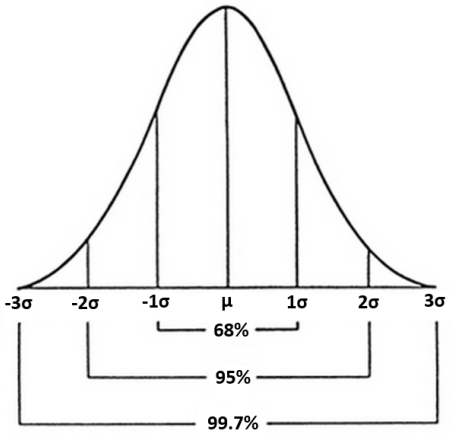
$$\sigma = \sqrt{\sum_{i=0}^{n-1} \frac{(x_i - \mu)^2}{n-1}} \tag{7.1}$$

I formelen gjelder:

- σ: standardavvik
- x_i: målepunkt
- n: Antall målinger
- μ: Gjennomsnittlig måleverdi

$$\mu = \sum_{i=0}^{n-1} \frac{x_i}{n} \tag{7.2}$$

Dersom måleresultatene er normalfordelt vil standardavviket kunne si noe om spredningen av måleresultatene i prosent, som illustrert i figur 7.1 under. Videre i kapittelet vil standardavvikets benevning være sub-piksel.

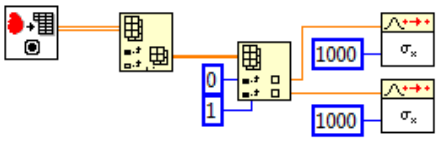


Figur 7.1: Standardavvik for en normalfordeling

7.1.1 Standardavvik etter bildeanalyse

Dersom man forstørrer opp et piksel i bildet som oppdaterer seg 14 ganger i sekundet, vil man se at gråtoneverdien forandrer seg, uten at det er endringer i lysforholdene. Dette kan være grunnet elektrisk støy inn på CMOS bildesensoren. En CMOS tar imot et lyssignal og gjør det om til en spenningsverdi lokalt. Denne spenningsverdien vil videre bli omgjort til en gråtoneverdi. Elektrisk støy på bildesensoren vil derfor skape variasjon i gråtoneverdier.

Under en bildebehandling vil støyen fra gråtoneverdiene gi utslag under segmenteringen av bildet. Gråtoneverdier som ligger med en verdi rundt terskelverdien vil variere mellom å få verdien 255 og 0. Det vil derfor være viktig å fokusere linsa slik at det blir et skarpt skille mellom hvit og svart bakgrunn på mønsteret. Fokuset blir derfor justert inn først. For å finne ut hvilke system som baserer seg på det mønsteret med minst støy i, måles standardavviket på en blob i hvert system. Dette gjøres ved å måle standardavviket på sentroiden gjort av partikkelanalysen som vist under.

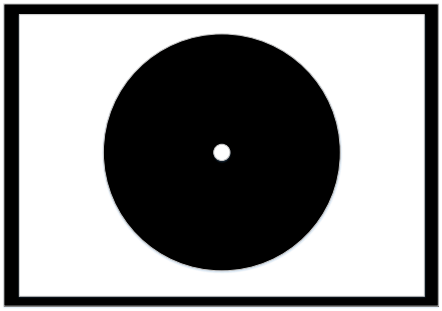


Figur 7.2: Finner standardavviket for blob med 1000 stikkprøver

Senterpunktet for både store og små sirkler blir målt ved å regne ut tyngdepunktet i x-og y-aksen for hver blob. Disse målene blir regnet ut med subpiksel nøykaktighet, altså høyere nøykatighet enn hele piksel. Under er formelen for hvordan tyngdepunkt regnes ut [6]:

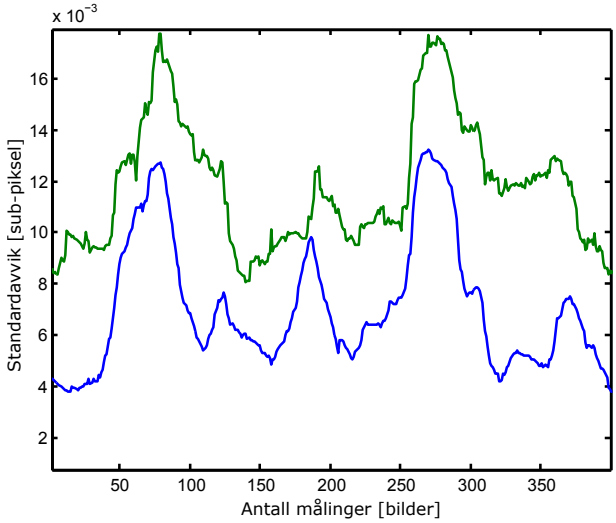
$$(x_c, y_c) = \frac{1}{M} \sum_{i=1}^{n} (x_i, y_i) \tag{7.3}$$

I formelen ovenfor vil M være antall piksler i en blob. Det antas derfor at dersom en sirkulær blob er stor(består av mange piksler) vil tyngdepunktet få et lavere standardavvik enn for en mindre blob. Dette grunnet at støyen blir midlet ut dersom flaten er stor. Dette er påvis ved å gjøre et forsøk med en liten og en stor sirkel som vist i figuren under.



Figur 7.3: Bilder som benyttes for testing av forskjellen i standardavvik for stor og liten sirkel

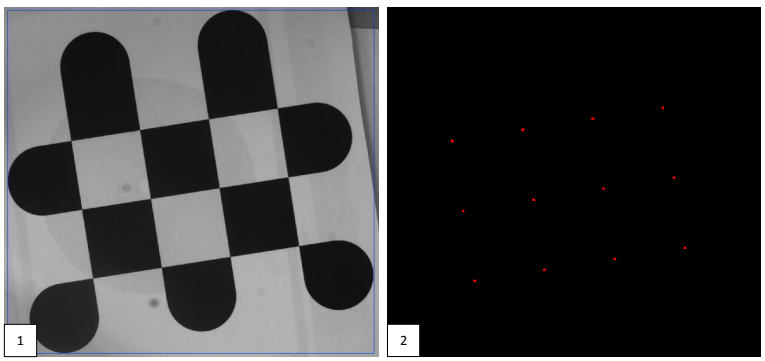
Forskjellen i standardavvik illustreres i figur 7.4 under. Antall målinger vil være antall bilder som hentes inn og prosesseres. Standardavviket måles med sett på 40 bilder. Det vil altså være et vindu som måler standardavviket for de 40 nyste bildene. Målingene gjøres på sentroiden i x-retning for den svarte store sirkelen og den lille hvite sirkelen.



Figur 7.4: Standardavvik for sentroide i stor og liten sirkel. Stor er blå og liten er grønn

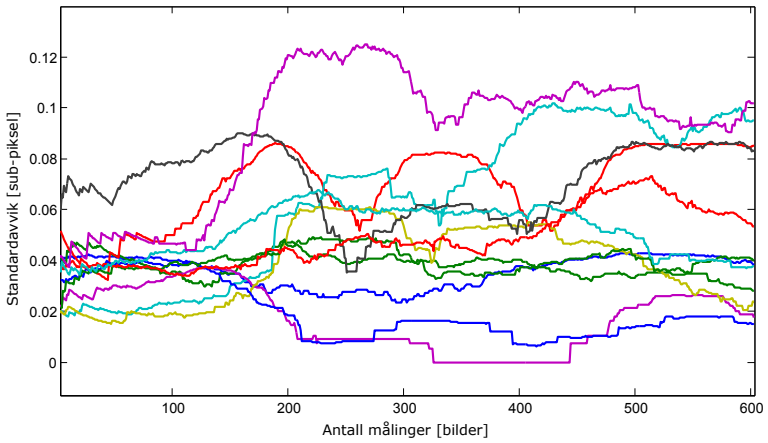
Det er ikke gjort noe mer analyse rundt resultatet annet enn at det registreres at standardavviket blir noe lavere ved større sirkel, men overraskende lite.

For programmet som benytter sjakkbrettet som ROI vil hvert hjørne detekteres som vist under ved å benytte *Harris corner detection*.



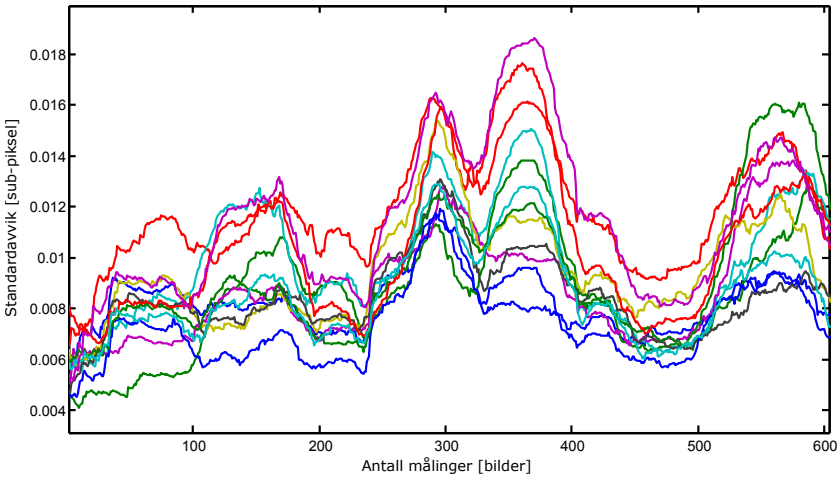
Figur 7.5: Benytter *Harris corner detection* for å detektere hjørner. Under prosesseringen fra bilde 1 til bilde 2 er det også gjort en segmentering som vist i figur 7.8

Sentroidene for de detekterte hjørnene blir benyttet videre for utregning av posituren. Partiklene vil være små i forhold til sirklene. Det vil videre gjøres en analyse på standardavviket for sentroiden av hver hjørneblob etter segmenteringen av bildet. Under er grafene for alle 12 sentroidene. Det er 100 målinger per sett.



Figur 7.6: Standardavviket for 12 hjørnesentroider etter blob analyse. Det er 100 målinger per sett. Sammenligner kun sentroider i x-retning.

Det blir benyttet 12 konsentriske kontrasterende sirkler istedenfor de 9 i figur 5.15 for å sammenligne standardavviket for disse sentroidene kontra sentroiden for hjørnene. Grafene for sirkelsentroidene er avbildet i figur 7.7 under. Grafene indikerer at det ikke er noen store sprang i standardavvik.



Figur 7.7: Standardavvik for 12 sirkelsentroider etter blob analyse. Det er 100 målinger per sett. Sammenligner kun sentroider i x-retning.

For å få et mest mulig representativt standardavvik for 1 blob, estimeres et gjennomsnittlig standardavvik for disse sentroidene i x-pos som følger:

$$\sigma = \frac{\sigma_1 + \sigma_2 + \sigma_3 + \sigma_4 ... + \sigma_{12}}{12} \tag{7.4}$$

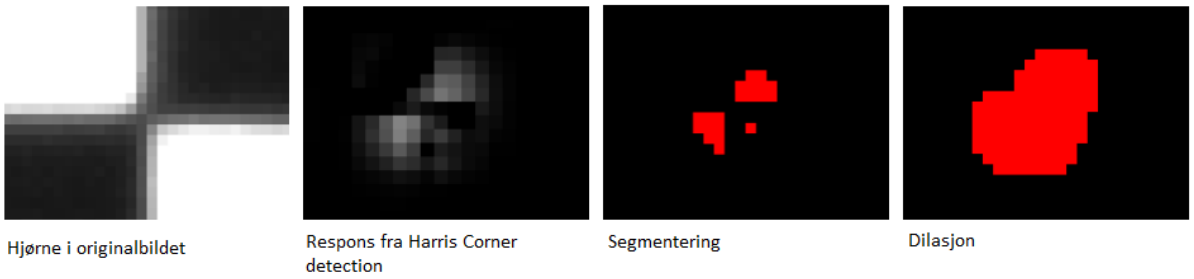
Standardavviket for 1 hjørnesentroid vil da ligge på rundt:

$$\sigma_{hjorne} = 0.045 \tag{7.5}$$

For 1 sirkelsentroid:

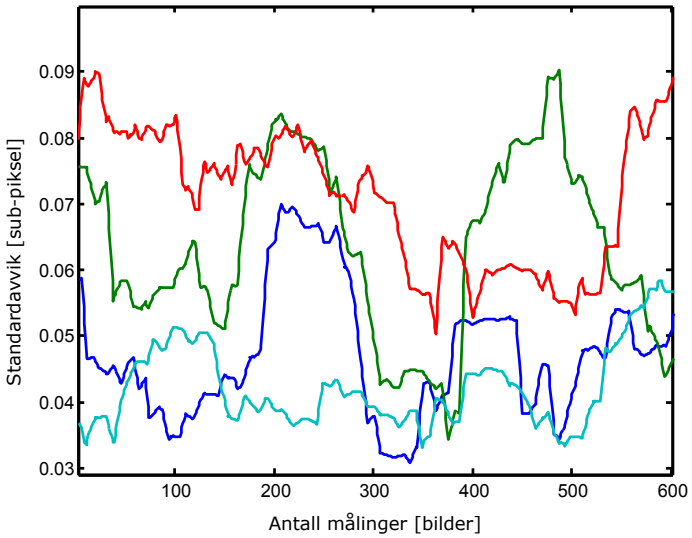
$$\sigma_{sirkel} = 0,011 \tag{7.6}$$

Standardavviket for et hjørne og en sirkel sammenlignes: $\frac{0.011}{0.045} = 0.244$. Standardavviket for 1 sirkelsentroid er altå rundt 24% av standardavviket for 1 hjørnesentroid. En sirkel-blob representeres av mange piksler i forhold til en hjørne-blob og vil være noe av årsaken. En hjørne-blob vil også være mindre fokusert enn en sirkel-blob. Dette vil gjøre at flere pikselverdier vil ligge rundt terskelverdien under segmenteringen. Disse pikslene vil hoppe fra å ha verdien *høy* til verdien *lav* i det binære bildet, og vil oppleves som støy. Dette vil være under transformasjonen fra bilde 2 til bilde 3 i figur 7.8 under. Symmetrien på en blob vil også spille inn på standardavviket. En usymmetrisk form vil gjøre at støy har en varierende påvirkning for et standardavvik målt i en sentroide utifra hvor støyen registeres. Dersom det observeres støy i en smal del av en blob vil dette påvirke standardavviket mer enn støy i en bredere del av samme blob.



Figur 7.8: Bildeprosessering, segmentering og binær morfologi utført i programmet som detekterer hjørner [3]

Lasersystemets stadardvviket for de fire punktene er illustrert under:



Figur 7.9: Standardavvik for de 4 lasersentroidene etter blob analyse. Det er 100 målinger per sett. Sammenligner kun sentroider i x-retning.

Standardavvik for 1 laser-sentroide i x-retning ligger på rundt:

$$\sigma = 0,063 \tag{7.7}$$

Dersom dette resultatet sammenlignes med resultatet i 7.5: $\frac{0.063}{0.045} = 1.4$. Standardavviket for 1 lasersentroide er altså rundt 40% høyere enn standardavviket for 1 hjørnesentroid. Siden lysnivået skal være konstant for de 3 testene er blenderåpningen noe stor og slepper inn mye lys. Dette kan føre til at støy i pikselverdier på hvit bakgrunn kan få gråtoneverdier som varierer mellom å være under og over terskelen for segmentering. Dersom disse pikslene er i kontakt med en laser-blob vil dette påvirke standardavviket. Det er store individuelle forskjeller på lasere. Disse er av dårlig kvalitet. De har forskjellig og usymmetrisk form og de har ikke et godt fokus.

Det viser seg at en sentroide for en sirkel-blob er mye mer robust med tanke på støy enn for de 2 andre. Dette er ikke uventet da den skiller seg fra de 2 andre ved at den er symmetrisk, godt fokusert og er mye større enn de 2 andre.

7.1.2 Standardavvik og presisjon av positurmålinger

I tabell 7.1 under blir presisjonen for de tre systemene målt for pitch, roll og heave. Posisjonen, altså avstanden fra kamera til måleobjektet er konstant for måling av pitch og roll. Det er i denne testen benyttet 48 sirkler og 48 hjørnepunkter. Dette fordi filtreringsrutinen til hjørnesystemet ikke fungert for færre enn 48 punkter.

Tabell 7.1: Presisjonsmåling for de 3 vision systemene

		Sjakkbrett		CCC		Laser	
Positur		Gjennomsnitt	standardavvik	Gjennomsnitt	standardavvik	Gjennomsnitt	standardavvik
Roll	15°	15,68	0,038	14,93	0,002	14,47	0,039
	0°	-0,03	0,025	0,17	0,003	-0,62	0,060
	-15°	-15,50	0,034	-16,64	0,002	-15,95	0,020
Pitch	15°	15,55	0,033	17,79	0,004	14,91	0,024
	0°	0,00	0,020	0,03	0,005	0,10	0,053
	-15°	-15,63	0,018	-13,49	0,002	-15,04	0,0177
Heave	994mm	990,03	0,024	991,18	0,022	975,76	0,025
	893mm	888,1	0,018	886,78	0,013	874,67	0,042
	792mm	785,49	0,013	784,16	0,011	773,39	0,022

Standardavviket går ned jo nærmere platen kommer kamera, altså heave i tabell 7.1 blir mindre. Dette er naturlig da ROI vil få en høyere oppløsning og hver blob da blir presentert med flere piksler. Effekten av støy vil derfor gå ned.

Standardavviket går ned for måling av roll og pitch i forhold til standardavviket målt for hver enkelt blob. Grunnen til dette vil være fordi måling av frihetsgrader baserer seg på en blob-populasjonen på 48 istedenfor en enkelt blob. Det nye standardavviket blir da:

$$\sigma_\mu = \frac{\sigma}{\sqrt{48}} \tag{7.8}$$

Derfor vil det nye standardavviket for CCC ligge på rundt:

$$\sigma_{\mu,CCC} = \frac{0,011}{\sqrt{48}} = 0,0016 \tag{7.9}$$

Standardavvikene målt i tabell 7.1 ligger ikke så langt fra denne verdien.

Hjørnesystemet

Presisjonen på hjørnesystemet er forholdsvis god. Den statiske feilmarginen er omtrent like stor rundt 0° for 15° og −15°. Dette kan kompenseres vekk på følgende måte:

$$k_{roll} = \frac{\frac{15}{15,68} + \frac{-15}{-15,50}}{2} = 0,962 \tag{7.10}$$

$$k_{pitch} = \frac{\frac{15}{15,55} + \frac{-15}{-15,63}}{2} = 0,962 \tag{7.11}$$

$$k_{heave} = \frac{\frac{994}{990,03} + \frac{893}{888,1} + \frac{792}{785,49}}{2} = 1,0059 \tag{7.12}$$

Tabell 7.2: Presisjonsmåling før og etter kompensering for statisk feilmargin av hjørnesystemet

		Hjørnesystem	
Positur		Gjennomsnitt før	Gjennomsnitt etter
Roll	15°	15,68	15,08
	6,9°	7,4	7,11
	0°	-,0,3	0,29
	−6,9°	-7,15	-6,88
	-15°	-15,50	14,91
Pitch	15°	15,55	14,95
	6,9°	7,16	6,89
	0°	0,00	0,00
	−6,9°	-7,2	-6,92
	-15°	-15,63	15,04
Heave	994mm	990,03	995,83
	893mm	888,10	893,33
	792mm	785,49	790,12

Tabell 7.3: Presisjon av endring i heave for hjørnesystemet

Endring i heave	Målt endring i heave	Feilmåling i prosent
994mm-792mm=202mm	995,83-790,12=205,71	1,84%

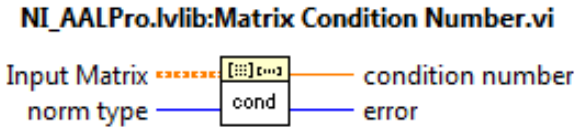
CCC-systemet

Presisjonen er ikke helt som ønsket på CCC-systemet. Dette kan skyldes at pikselkoordinatene ikke er normalisert rundt prinsipalpunktet, og den radielle forvrengningen ikke er kompensert vekk. Dersom dette gjøres, som vist i kapittel 5.3.1, vil blokken i figur 5.19 bli tatt med, og blokken i figur 5.17 blir erstattet med blokken i figur 5.20. Resultatene før og etter denne endringene er vist i tabell 7.4 under.

Tabell 7.4: Presisjonsmåling før og etter normalisering rundt prinsipalpunktet av pikselkoordinatene og kompensering for radiell forvrenging

		CCC	
Positur		Gjennomsnitt før	Gjennomsnitt etter
Roll	15°	14,93	15,79
	6,9°	6,77	7,31
	0°	0,17	0,32
	−6,9°	-7,5	-6,80
	-15°	-16,64	-15,17
Pitch	15°	17,79	15,36
	6,9°	8,27	7,03
	0°	0,03	-0,02
	−6,9°	-6,17	-7,32
	-15°	-13,49	-15,47
Heave	994mm	991,18	990,43
	893mm	886,78	888,07
	792mm	784,16	785,75

Det er tatt med flere målinger i tabellen ovenfor for å se at den konstante feilmarginene er lineær. Den nye presisjonen er bedre enn den som var tidligere. Ved å se på kondisjonstallet til D-matrisen benyttes følgende blokk i LabVIEW:



Figur 7.10: Blokken som beregner kondisjonstallet til en matrise.

Kondisjonstallet for D-matrisen før og etter normalisering av pikselkoordinatene og verdenskoordinatene er vist i tabell 7.5 under.

Tabell 7.5: Kondisjonstallet for D-matrisen før og etter normalisering av pikselkoordinatene og verdenskoordinatene

Før	Etter
975059	91

Tabell 7.5 gir en klar indikasjon på at normaliseringen rundt prinsipalpunktet gir store utslag for kondisjonstallet til D-matrisen. Dette utgjorde en presisjonsendring på rundt 2.5° for +15° pitch. Kompenseringen for radiell forvrenging utgjorde rundt 0.3° for denne endringen, mens normaliseringen rundt prinsipalpunktet utgjorde resten av endringen, altså omkring 2.2°.

For å kompensere vekk den statiske feilmarginen blir følgende parametre regnet ut:

$$k_{roll} = \frac{\frac{15}{15,79} + \frac{-15}{-15,17}}{2} = 0,9694 \tag{7.13}$$

$$k_{pitch} = \frac{\frac{15}{15,36} + \frac{-15}{-15,47}}{2} = 0,9731 \tag{7.14}$$

$$k_{heave} = \frac{\frac{994}{990,43} + \frac{893}{888,07} + \frac{792}{785,75}}{3} = 1,0057 \tag{7.15}$$

Dette medførte følgende endringer

Tabell 7.6: Presisjonsmåling før og etter kompensering for statisk feilmargin av CCC

		CCC	
Positur		Gjennomsnitt før	Gjennomsnitt etter
Roll	15°	15,79	15,30
	6,9°	7,31	7,17
	0°	0,32	0,24
	−6,9°	-6,80	-6,66
	-15°	-15,17	-14,65
Pitch	15°	15,36	15,01
	6,9°	7,03	6,77
	0°	-0,02	-0,05
	−6,9°	-7,32	-7,01
	-15°	-15,47	-15,10
Heave	994mm	990,43	996,14
	893mm	888,07	893,68
	792mm	785,75	790,82

Tabell 7.7: Presisjon av endring i heave for CCC

Endring i heave	Målt endring i heave	Feilmåling i prosent
994mm-792mm=202mm	996,14-790,82=205,32	1,64%

Lasersystem

Lasersystemet beregner resultatene sine ut ifra trigonometri og presisjonen vil være avhengig av hvordan plattformen er fysisk justert. Det er derfor vanskelig å sammenligne dette systemets presisjon med presisjonen av de to andre systemene. En svakhet med oppbyggingen av plattformen er at den fysiske justeringsevnen av kameraet på riggen, illustrert i figur 3.9, er litt vanskelig når justeringen skal ned på pikselnivå. Måling av pitch gir høy presisjon i forhold til måling av roll. Dette har med at kamera er justert mer presist inn i x-retning enn i y-retning i forhold til laserpekerne. Istedenfor å kompensere vekk statisk feil, vil det være en fordel å bruke mere tid på å justere inn systemet.

Dersom en ser på presisjonen av *Heave* i tabell 7.1, er resultatene forholdsvis presise i forhold til mål av endringer i heave:

Tabell 7.8: Presisjon av endring i heave for lasersystemet

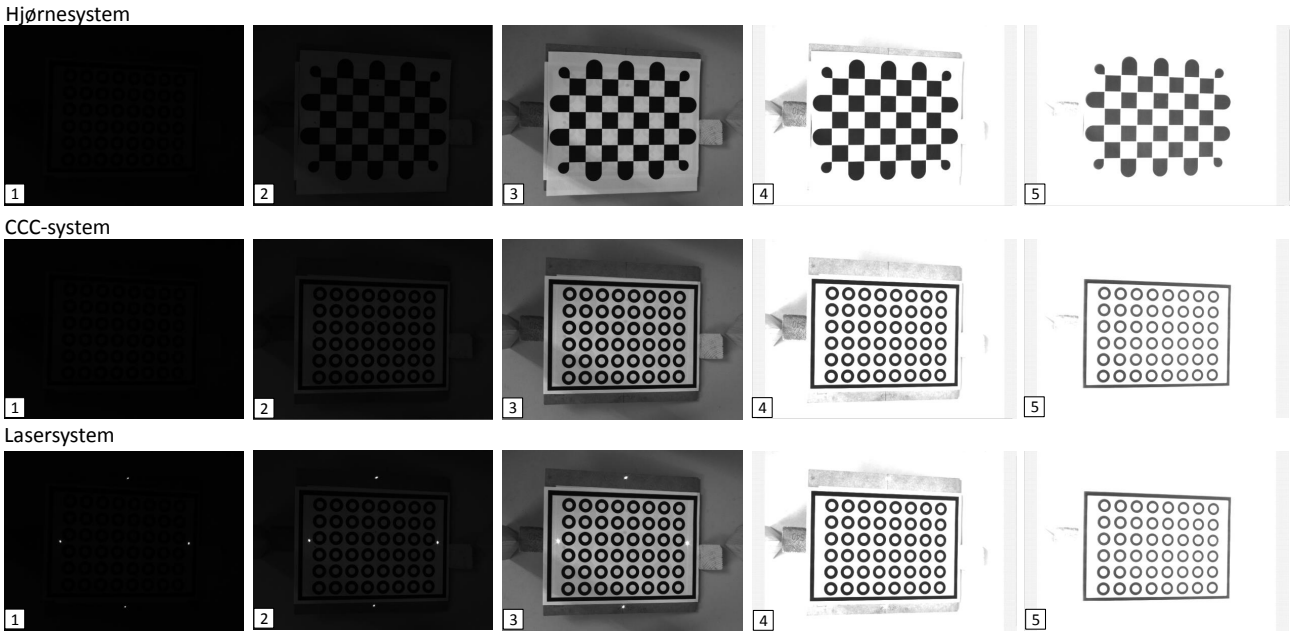
Endring i heave	Målt endring i heave	Feilmåling i prosent
994mm-792mm=202mm	975,76-773,39=202,37	0,18%

Standardavviket for positurmålinger er ganske likt som ved måling av standardavvik for en laser-blob i det binære bildet. Dette skyldes at det benyttes trigonometri og ikke homografi for disse utregningene. 2 lasere blir benyttet for å finne roll og z1, og de 2 andre laserne blir benyttet for å finne pitch og z2. Heave blir deretter et gjennomsnitt av z1 og z2. Formel 7.8 vil derfor ikke gjelde i dette tilfellet.

7.2 Varierende lysforhold, justere blenderåpning

Denne testen skal måle hvor stor forskjell det er på de 3 forskjellige systemene under varierende lysforhold.

For å få de samme lysforholdene for hvert enkelt system justeres blenderåpningen til en bestem verdi før de 3 systemene testes på denne verdien. Deretter justeres blenderåpningen og de 3 systemene testes på nytt. Under er en illustrasjon av hvilke lysforhold som er blitt testet.



Figur 7.11: De tre systemene testet under 5 forskjellige lysforhold ved å justere blenderåpningen

Platen i figuren ovenfor er satt til å stå i $-15°$ pitch. Målinger ble gjort for de 3 systemene under de 5 forskjellige lysforholdene for å se om de klarte å gi korrekte måleresultater. Resultatet for testene er vist i tabell 7.9 under.

Tabell 7.9: De 3 systemene testes under 5 forskjellige lysforhold

System	Pitch	Lysforhold	Gjennomsnittlig målt pitch	Standardavvik
Sjakkbrett-system	-15°	1	Leverer ikke måling	-
		2	Leverer ikke måling	-
		3	-15,30	0,0155
		4	Leverer ikke måling	-
		5	Leverer ikke måling	-
CCC-system	-15°	1	Leverer ikke måling	-
		2	-15,31	0,0043
		3	-15,30	0,0031
		4	-15,30	0,00205
		5	-15,35	0,00318
Laser-system	-15°	1	-15,73	0,0385
		2	-16,26	0,022
		3	-15,89	0,0536
		4	Leverer ikke måling	-
		5	Leverer ikke måling	-

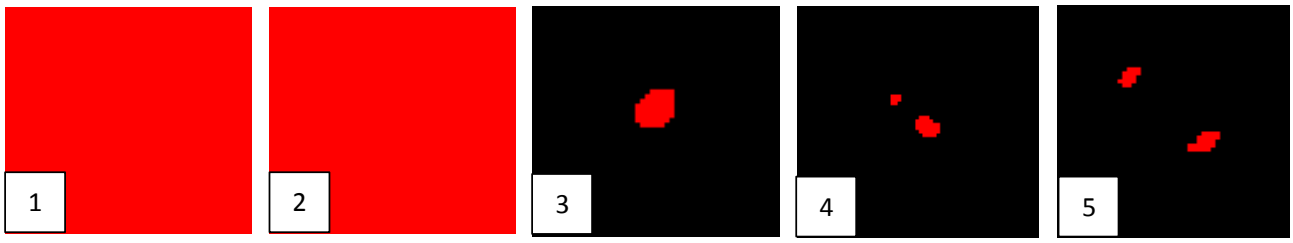
7.2.1 Hjørnesystemet

Hjørnesystemet viser seg å være veldig sensitivt for varierende lysforhold. Dersom det ble for mørkt klarte det ikke å finne hørner, og dersom det ble for lyst klarte det heller ikke å finne hjørner. Kun *Lysforhold 3* ga riktig måleresultatet for dette sytemet. En av grunnene til at det ikke fungerer for de lyse bildene er at ROI ikke detekteres. Det er fordi det blir dårligere kontakt mellom de svarte rutene når lyset øker i intensitet som vist i figur 7.12 under.

Figur 7.12: Hjørnene etter at Otsus metode er benyttet under segmentering ved de 5 forskjellige lysforholdene

Systemet er avhenging av at alle de svarte rutene henger sammen for å finne ROI. Ved å øke antall iterasjoner for dilasjon vil de svarte rutene henge sammen og det er mulig å finne ROI. Dette setter krav til en dynamisk dilasjon som altså må forandres utifra lysintensiteten i bildet. Dette problemet oppstår ikke når ROI i CCC-systemet skal detekteres. CCC-systemet finner den største hvite bloben i bildet istedenfor den svarte.

Dersom ROI detekteres er det likevel et problem til som gjør at hjørnesystemet ikke klarer å beregne positur for varierende lysforhold. Segmenteringen etter *Harris corner detection* blokka deler hver hjørne-blob mer og mer jo lysere det blir som vist i figur 7.13. Ved å øke antall iterasjoner på dilasjonsblokka etter denne rutinen også vil hver hjørne-blob henge sammen igjen, og du posituren kan måles.

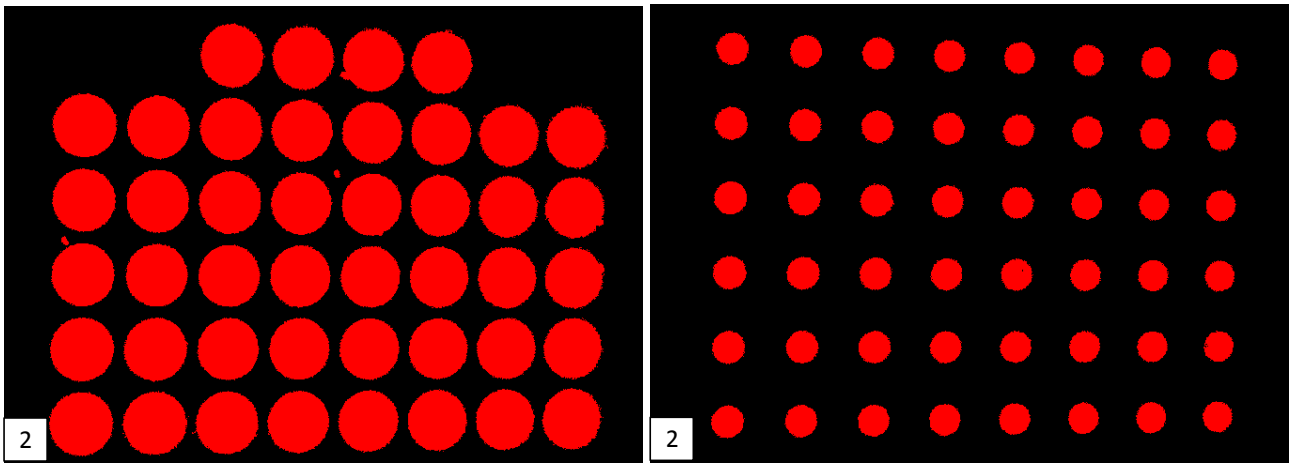


Figur 7.13: Segmenteringen ved bruk av Otsus metode etter *Harris corner detection* med de 5 forskjellige lysforholdene

Figur 7.13 viser også at *Harris corner* blokken ikke klarer å finne hjørner for de 2 mørkeste lysforholdene. Dette betyr at selv om det forekommer dilasjon i flere iterasjoner for både ROI og segmenteringen etter *Harris corner* blokken vil systemet ikke levere positurmålinger for de 2 mørkeste bildene.

7.2.2 CCC-systemet

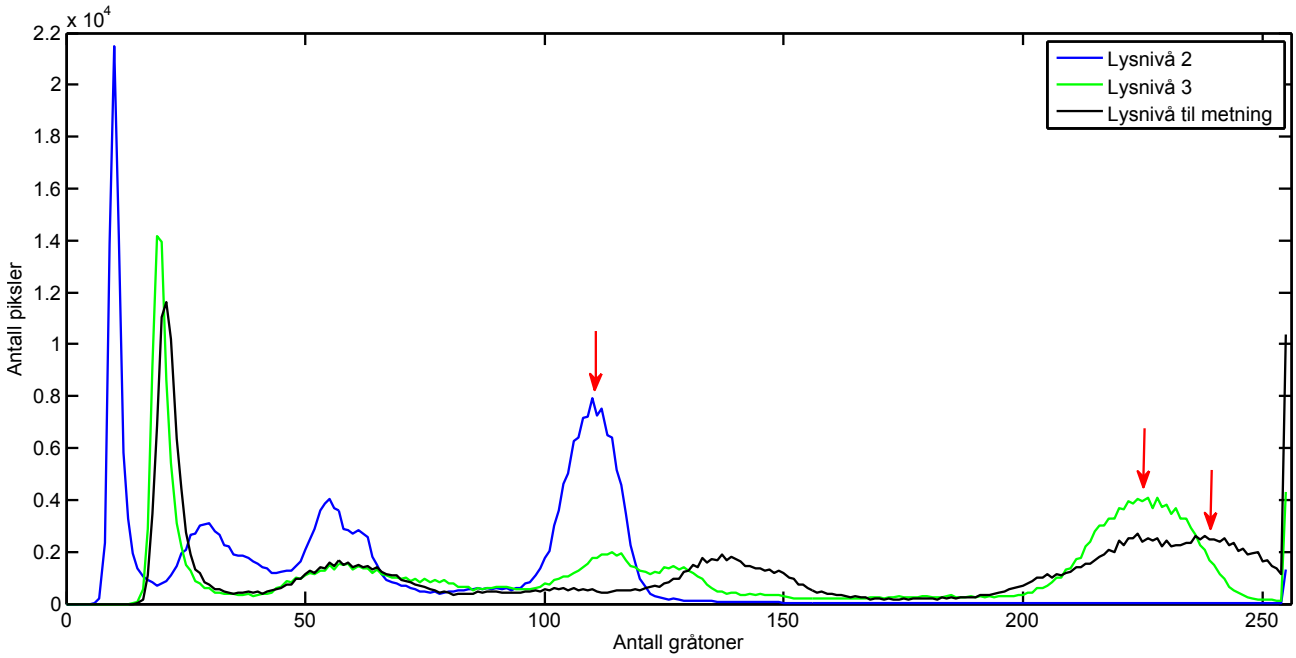
CCC-systemet målte korrekt positur for alle lysforholdene bortsett fra det mørkeste. Til og med på det mørkeste ble de hvite sirklene detektert men alle de mørke delene av de konsentrisk kontrasterende sirklene ble ikke detektert som vist i figur 7.14. Sentroidene godkjennes ikke dersom det ikke er en sentroide fra en hvit og en svart sirkel som har samme posisjon. Systemet leverte derfor ikke resultater for det mørkeste lysforholdet.



Figur 7.14: Segmentering ved bruk av Otsus metode ved lysforhold 1 for CCC-systemets svarte sirkler til venstre og hvite sirkler til høyre.

7.2.3 Lasersystemet

Lasersystemet leser riktig positur for de 3 mørkeste lysforholdene. Når det slepper for mye lys gjennom blenderåpningen går den hvite bakgrunnen i metning sammen med laserpunktene. Det vil si at den hvite bakgrunnen får gråtoneverdi 255 og laserunktene kan ikke få høyere verdi enn 255. Det vil derfor ikke være mulig å skille disse pikslene fra hverandre. Dette synes spesielt godt under lysforhold nr.5 i figur 7.11. I figur 7.15 under er dette illustrert ved å skissere opp histogrammet for lysnivå 2,3 og lysnivå til metning. Det sistnevnte lysnivået er justert litt høyere enn nivå 3 for å illustre metning. Pilene i figuren illustrer hvilke gråtoneverdier den hvite fargen ligger på for de forskjellige lysnivåene og indikerer hvordan den hvite bakgrunnsfargen flytter seg mot høyre ettersom lysnivået øker i intensitet. For lysnivå 3 indikerer pilen at den hvite bakgrunnen akkurat ikke går i metning. For lysnivå til metning, indikerer pilen at den hvite bakgrunnen akkurat går i metning og det blir derfor en umulig oppgave å skille laserprikkene fra den hvite bakgrunnen. Siden lasersystemet skal fungere sammen med CCC-systemet må den hvite bakgrunnen kunne atskilles fra laserprikkene, og lysforholdene setter derfor en naturlig stopper for hvor lyse forhold det kan være før dette systemet ikke vil fungere mer.



Figur 7.15: Histogrammet for 3 lysnivåer ved analyse av laserprikker. Pilene indikerer hvordan gråtoneverdien til den hvite bakgrunnsfargen forandrer seg med for de 3 lysnivåene.

7.2.4 Resultater og analyse

For tilfeldige feil i målingene, viser det seg at den målte sentroiden i et sirkelformet stort objekt gir lavere stadardavvik enn for et sirkelformet lite objekt. Dette er ikke uventet da en blob representert av mange piksler vil midle ut støy i kantene grunnet dens store overflate.

Standardavviket for måling av positur går ned dersom flere målepunkter i mønsteret benyttes videre i homografien. Dette vil kun gjelde for de 2 passive systemene, da lasersystemet benytter trigonometri for å regne ut roll, pitch og heave.

Fordi en blob i CCC-systemet er sirkelformet og større enn en blob i hjørnesystemet, vil standardavviket for en sentroide gitt av CCC være omtrent 24% relatert til hjørnesystemet. Standardavviket for en laser-blob vil være omtrent 40% høyere enn standardavviket for en hjørne-blob. Siden en laser-blob har større areal enn en hjørne-blob er det naturlig å tenke at støynivået målt på lasersentroiden vile være lavere enn for en hjørnesentroid. Lysforholdene under støymålingene tilsvarte lysforhold 3 i figur 7.11. Da den hvite bakgrunnsfargen blir representert med en gråtoneverdi som begynner å nærme seg terskelverdien i lasersystemet, kan dette ha bidratt med at støynivået målt i en lasersentroid ble større enn for en hjørnesentroid. Dette indikeres også i tabell 7.9 der stadardavviket i lasersystemet er lavere i lysforhold 2 enn 3. Andre grunner vil være at laserne er av dårlig kvalitet. I denne sammenheng vil det si at de har dårlig fokus, intensiteten er lav, formen på laserprikken er usymmetrisk og forskjellig fra laser til laser. Dårlig fokus gjør at det ikke blir et skarpt skille mellom hva som er laser og hva som er bakgrunn i bildet. I det binære bildet vil dette føre til støy i form av at pikslene i kantene på en laser-blob vil variere mellom å være høy og lav i større grad enn dersom laserne hadde vært fokusert. Dette gir utslag i målingene av sentroiden. At de er usymmetriske vil gjøre at pikslene som varierer mellom å ha høy og lav verdi i en laser-blob vil gi større utslag i en smal del av en blob i forhold til en bred del.

For presisjonsmålinger av heave viser det seg at lasersystemet er det beste med en feilmargin på 0,18% mens CCC-systemet måler 1,64% og hjørnesystemet 1,84%. For roll er hjørnesystemet best med $< 0,3°$ feilmargin, CCC$< 0,35°$ og laser$< 0,95°$. Hjørnesystemet har også best presisjon for pitch med $< 0,08°$, laser$< 0,10°$ og CCC$< 0,13°$. Statisk feil er ikke kompensert vekk i lasersystemet for disse sammenligningene slik som CCC og hjørnesystemet og vil derfor være en grunn til at det ikke er like presist. Dette fordi kameraet ikke var fysisk perfekt justert inn etter prinsipalpunktet. Det ble tidkrevende å justere kamera helt perfekt inn med den justeringsmekanismen som er bygd på plattformen.

Det viser seg at kondisjontallet på D-matrisen og kompensering for radiell forvrenging er av forholdsvis stor betydning for presisjonen av positurmålinger for CCC. På det meste ble presisjonen forbedret med 16,2%. Dette ble målt for 15° pitch med og uten normalisering og kompensering av D-matrisen. Det ble også gjort forsøk med en ny k-matrise. Dette ga ikke nevneverdig stor forskjell i resultatene og er derfor ikke kommentert andre steder i rapporten.

De 3 systemene ble testet under 5 forskjellige lysforhold. Hjørnesystemet klarte i utgangspunktet bare å måle riktig resultatet for 1 av de 5 lysforholdene. Etter at justeringer ble gjort i forhold til dilasjon klarte systemet å gi resultater på lysnivå 3,4 og 5. CCC-systemet klarte å registrere riktig positur for lysnivå 2,3,4 og 5. Lasersysmet klarte lysnivå 1,2 og 3.

De 2 nye systemene utfyller hverandre altså bra i forhold til varierende lysforhold. Det er en god glidende overgang fra lasersystemet som gir gode målinger under lysnivå 1,2 og 3 og CCC-systemet som gir gode målinger under lysnivå 2,3,4 og 5.

Kapittel 8

Videre arbeid

Siden CCC-systemet og lasersystemet er komplementære med tanke på forskjellige lysforhold, vil en naturlig videreføring være å slå sammen programkoden for de to. For å vite hvilket mønster som skal detekteres i det nye komplementære systemet, vil en løsning være å måle den gjennomsnittlige lysintensiteten i bildet opp mot 2 terskelverdier, en med lav og en med høy gråtoneverdi. For gjennomsnittlig gråtoneverdier under den laveste terskelverdien vil lasersystemet gjelde. Fra den laveste til den høyeste terskelverdien vil begge systemene benyttes og redundans oppnås. Fra den høyest terskelen og opp til 255 vil kun CCC-systemet operere.

Det går med en del prosesseringstid på å finne ROI for CCC-systemet og løsningen er heller ikke optimalt robust. Dette kunne vært løst annerledes. En mulig løsning på dette kan være:

1. Sirklene blir detektert og sentroidene filtrert.
2. ROI blir bestemt utifra sirklenes totale tyngdepunkt i bildet med litt margin.
3. Et nytt bilde hentes inn. Istedenfor å finne ROI i originalbildet, detekterer man nå sirklene kun inne i ROI fra punkt 2, med den antagelsen at sirklene ikke har beveget seg så mye på 70ms at de har flyttet seg vekk fra ROI.
4. ROI oppdaterer sin posisjon med litt margin utifra sirklenes nye totale tyngdepunkt.
5. Begynner på punkt 3 igjen.

Denne løsningen for å finne ROI ble det benyttet noe tid på, men systemet ble ikke stabilt. Ved å få dette til å fungere, vil systemet kun være avhengig av å detektere konsetriske sirkler og ikke den hvite bakgrunnen for å finne ROI. Dette kan gjøre systemet mer robust dersom det er stor avstand mellom kamera og den plane flaten og mye forstyrrende mønstre innenfor kameraets synsvinkel.

Plattformen som ble bygget til dette prosjektet fungerer bra, men når kamera skal justeres etter prinsipalpunktet må presisjonen ned på pikselnivå. Da blir justeringsmulighetene litt for grove og det blir tidkrevende å få det helt nøyaktig. Den fysiske løsningen på denne mekanismen kan gjøre det mulig å justere inn kameraet mer presist. Nå som det er bevist at et slikt system fungerer, kan det være en mulighet å gå til innkjøp av industrilasere, og justeringsmekanismer for å justere kameraet med høyere presisjon. Z-LASER [21] har mye slikt utstyr.

Dersom en plane flaten eksponeres for sollys oppstår problemer med å skille prjojisert strukturert lys med det skarpe sollyset. Den frekvensen som er mest synlig i sollys vil være 532nm(nanometer) [11]. Dette tilsvarer fargen grønn. Effekten av laseren vil også være av betydning for synligheten. Det ble benyttet en del tid på å få kjøpt inn slike lasere. Dette arbeidet ble stoppet opp fordi det tok for mye tid. Dersom en får godkjente papirer fra Statens Strålevern, ville det være interessant å se hvor mye lys man kan ha og likevel detektere en grønn laserprikk for å dekke enda flere lysforhold. Dersom laseren er synlig ved fult sollys vil et RGB-kamera klare å detektere den.

Kapittel 9

Konklusjon

De to nye vision systemene som ble bygget fungerer under ulike lysforhold og utfyller hverandre. De kan derfor dekke et stort antall lysnivåer. Det ene systemet baserer seg på at et kamera detekterer 4 laserprikker og vil derfor fungere best under mørke forhold og måling i forbindelse med homogene overflater. Det andre systemet baserer seg på å detektere konsentriske kontrasterende sirkler(CCC). Dette skal fungere godt under de fleste lysnivåer bortsett fra de aller mørkeste. Ideen bak CCC-systemet er å sammenligne sentroider for konsentriske kontrasterende sirkler i bilde. De konsentriske sirklene vil ha sammefallende sentroider. Dette gjør at filtreringen av punkter i bildet blir enkel, effektiv og sikker.

Det ble bygget en testbenk med 4 lasere og et kamera for å teste de to nye vision systemene, samt vision systemet tidligere beskrevet. Statiske tester ble utført på de 3 systemene og resultatene ble sammenlignet. Robustheten med hensyn på støy ble testet ved å analysere tilfeldig feil i målingene. Testing av varierende lysforhold ble gjort ved å justere blenderåpningen i kamera med 5 forskjellige lysnivåer i et rom uten vinduer.

Blob analysen for de 3 systemene ble sammenlignet. Det viste seg at et hjørne i det tidligere beskrevne systemet, gir en liten ujevn blob som er svært utsatt for varierende lysforhold, og måling av sentroiden gir forholdsvis stort stadardavvik. En sirkel i det nye systemet ga en mer robust blob som tåler varierende lysforhold. I en sirkel vil alle kanter ha samme avstand til sentrum, samt at den gir en blob som er forholdsvis stor i areal. Dette viste seg å ha positive virkninger med tanke på støy. Målinger viser at standardavviket for en sirkel med diameter 40mm er omtrent 24% av stadardavviket for en hjørne-blob. Støynivået er altså 24% av støynivået for hjørnesystemet.

Det viste seg at en laser-blob har omtrent 40% høyere støynivå enn en hjørne-blob. Dette kan skyldes kvaliteten på laserne som er benyttet. De benyttede laserne gir en blob som ikke er symmetrisk og ikke spesielt godt fokusert. De pikslene som ikke er godt fokusert i en blob vil typisk ha gråtoneverdier rundt terskelverdien, og variere mellom å være i klasse 0 og klasse 1 i et binært bilde. Dette vil skape støy. Dersom det benyttes bedre laserutstyr vil dette systemet mest sannsynlig bli betraktelig bedre rustet mot støy.

Hjørnesystemet klarte kun å måle positur ved det midterste, altså lysforhold nr.3 av de 5 lysforholdene som ble testet. En hjørne-blob er sensitiv ovenfor varierende lys fordi et hjørne er avhengig av at 2 svarte ruter er i kontakt, og kontaktpunktet representeres av svært få piksler i et binært bilde. Når det blir lysere vil de svarte firkantene i det binære bildet miste denne kontakten og avstanden mellom rutene vokser når lysintensiteten øker. Dilasjon må derfor benyttes med stigende antall iterasjoner for at hjørnene i det binære bildet skal holde på denne bindingen dersom lysintensiteten øker. Det viser seg også at *Harris corner detection* ikke klarer å finne hjørner for lave lysintensiteter og den gir ufokusert gjengivelse av hjørner.

CCC-systemet målte positur på de 4 lyseste lysforholdene. Det var kun i det mørkeste bildet at det ble vanskelig å detektere den svarte delen av sirklene. Lasersystemet målte positur for de 3 mørkeste lysforholdene. Når den hvite bakgrunnen i et bilde representeres av gråtoneverdier som har gått i metning er det ikke lenger mulig å skille laserprikkene fra den hvite bakgrunnen i et monokromt kamera.

De 2 nye systemene vil tilsammen dekke alle de 5 lysforholdene som er testet, og de vil også overlappe hverandre for 2 av dem. Det vil da være mulig å få et komplementært system ved å slå sammen disse 2 systemene til en felles kode. Det komplementære systemet vil kunne måle resultater for alle de 5 lysforholdene og være redundant for de lysforholdene som overlapper. Om dette systemet skal detektere laserprikker, sirkler eller begge på en gang kan bestemmes ved å måle den gjennomsnittlige lysintensiteten i bildet opp mot 2 terskelverdier, en med lav gråtoneverdi og en med høy. Er den gjennomsnittlige gråtoneverdien lavere enn den lave terskelen vil laserprikkene analyseres, er den høyere vil både sirklene og laserprikkene analyseres og dersom den er høyere enn den høyeste terskelen vil bare sirklene analyseres.

Referanser

[1] F. c. C. J. S. J. Pag‘ es, Christophe Collewet, “Visual servoing by means of structured light for plane-to-plane positioning,” *Univeritetet i Agder*, p. 103, 2006, [Accessed feb-2015]. [Online]. Available: https://hal.inria.fr/inria-00070427/document

[2] T. T. Røren, “3d-overflatemodellering basert på stereoskopiske bilder,” *NTNU*. [Online]. Available: http://www.diva-portal.org/smash/get/diva2:566544/FULLTEXT01.pdf

[3] J. L. Haug, “Development of a vison-based measurement for relative motion compensation,” *Univeritetet i Agder*, p. 67, 2014, [Accessed jan-2015]. [Online]. Available: http://brage.bibsys.no/xmlui/handle/11250/221816

[4] B. Tweddle, “Relative computer vision based navigation for small inspection spacecraft,” *Massachusetts Institute of Technology*, p. 23, 2011, [Accessed 02-2015]. [Online]. Available: http://ssl.mit.edu/spheres/library/tweddle_AIAA_GNC_2011_presentation.pdf

[5] D. Fofi, T. Sliwa, and Y. Voisin, “A comparative survey on invisible structured light,” in *Electronic Imaging 2004*. International Society for Optics and Photonics, 2004, pp. 90–98.

[6] N. Instruments, “Paarticle measurements.” [Online]. Available: http://zone.ni.com/reference/en-XX/help/372916L-01/nivisionconcepts/particle_measurements/

[7] S. N. Rune Husveg, Tore Medberg, “Mas502 instrumentation,” *Univeritetet i Agder*, p. 19, 2014, [Accessed jan-2015].

[8] G. G. K. A. W. B. Barbara Frank, Cyrill Stachniss, “Robotics 2 camera calibration.” [Online]. Available: http://ais.informatik.uni-freiburg.de/teaching/ws09/robotics2/pdfs/rob2-08-camera-calibration.pdf

[9] J.-Y. Bouguet, “Camera calibration toolbox for matlab,” 02.02.2013. [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc/

[10] L. V. Vincenzo Lippiello, Bruno Siciliano, “Eye-in-hand/eye-to-hand multi-camera visual servoing,” *44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, 2005, [Accessed feb-2015]. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1583013

[11] T. P. Classroom, “Visible light and the eye’s response,” *Light Waves and Color - Lesson 2 - Color and Visionv*. [Online]. Available: http://www.physicsclassroom.com/class/light/Lesson-2/Visible-Light-and-the-Eye-s-Response

[12] S. strålevern, “Laserklasser.” [Online]. Available: http://www.nrpa.no/fakta/90813/laserklasser

[13] L. Aurdal, “Matematisk morfologi.” [Online]. Available: http://www.aurdalweb.com/morpho.pdf

[14] N. instruments, “Thresholding.” [Online]. Available: http://zone.ni.com/reference/en-XX/help/372916L-01/nivisionconcepts/thresholding/

[15] N. Otsu, “A tlreshold selection method from gray-level histograms,” *IEEE Transactions on systems, man, and cybernetics, vol 9 no 1, pp 62-66, 1979*. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4310076

[16] L. Aurdal, “A tlreshold selection method from gray-level histograms,” *Norsk regnesentral*. [Online]. Available: http://www.uio.no/studier/emner/matnat/ifi/INF3300/h06/undervisningsmateriale/intro_2006.pdf

[17] B. H. Shahid Ayub, Alireza Bahraminisaab, “A sensor fusion method for smart phone orientation estimation.” [Online]. Available: http://www.cms.livjm.ac.uk/pgnet2012/Proceedings/Papers/1569603133.pdf

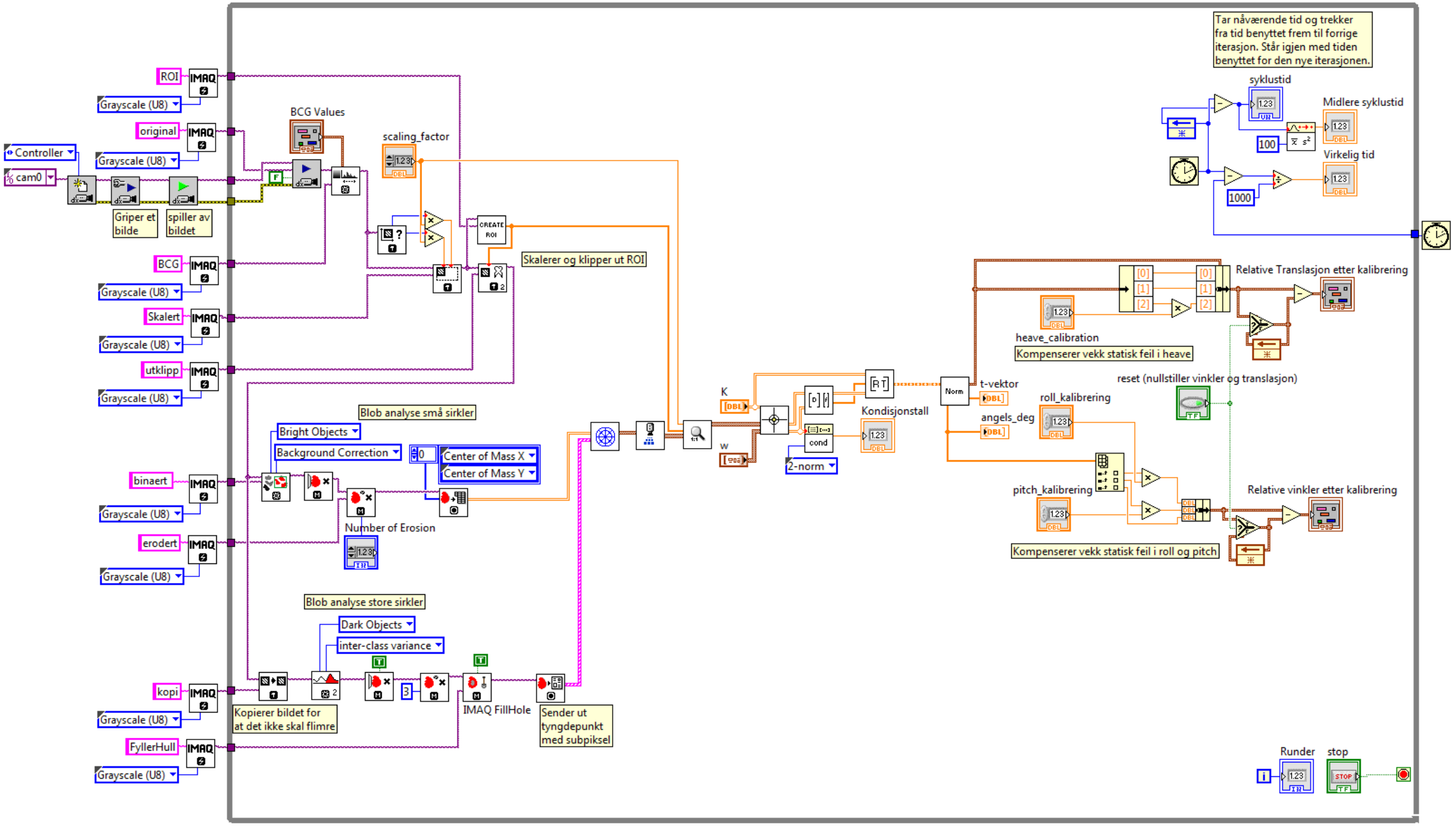
[18] UiB, "Sensitivitet og kondisjonering." [Online]. Available: http://www.ii.uib.no/~anto/i162/Slides/Forel5.pdf

[19] Basler, "Product information for aca2500-14um camera." [Online]. Available: http://www.baslerweb.com/en/products/area-scan-cameras/ace/aca2500-14um

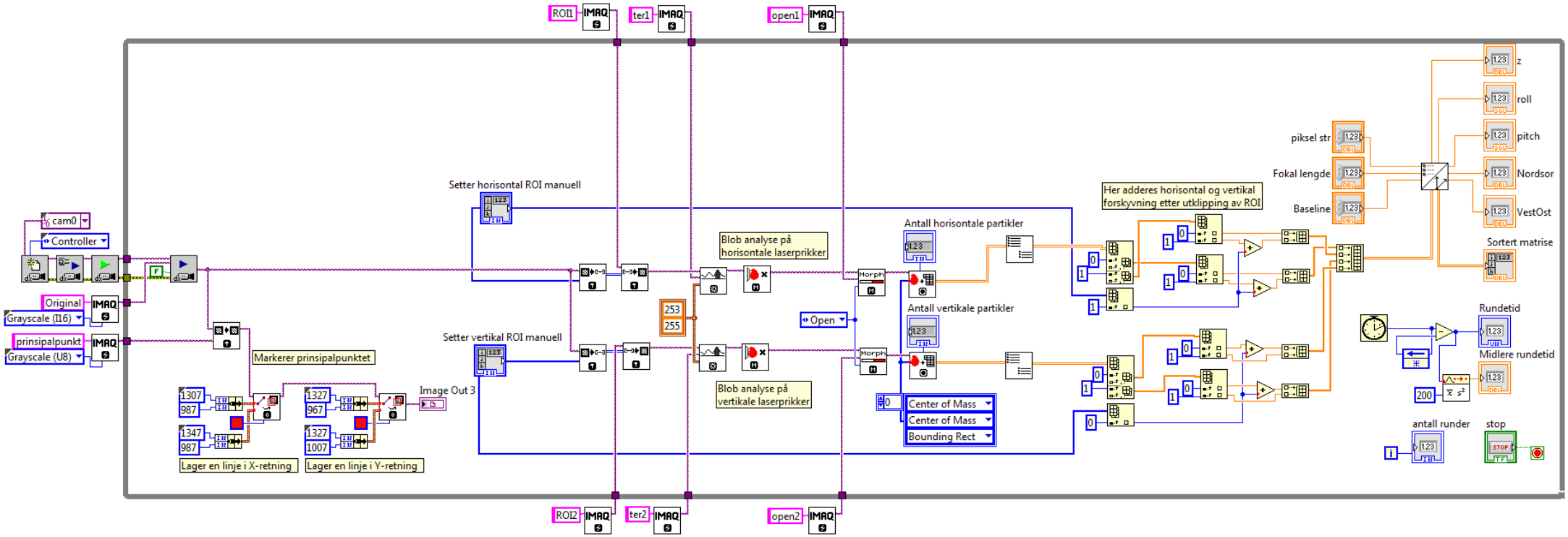
[20] N. Instruments, "Standard deviation and variance vi." [Online]. Available: http://zone.ni.com/reference/en-XX/help/371361J-01/gmath/stand_deviation_variance/

[21] Z-LASER. [Online]. Available: http://www.z-laser.com/index.php?&L=1

Vedlegg A: LabVIEW-kode for passivt vision system

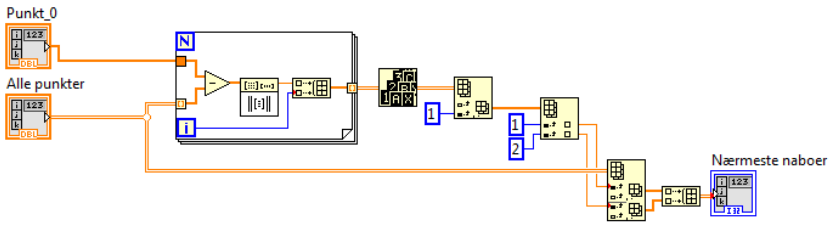

Tar nåværende tid og trekker fra tid benyttet frem til forrige iterasjon. Står igjen med tiden benyttet for den nye iterasjonen.
syklustid
Midlere syklustid
100
Virkelig tid
1000
ROI
Grayscale (U8)
original
Grayscale (U8)
Controller
cam0
Griper et bilde
spiller av bildet
BCG Values
scaling_factor
CREATE ROI
Skalerer og klipper ut ROI
BCG
Grayscale (U8)
Skalert
Grayscale (U8)
utklipp
Grayscale (U8)
Blob analyse små sirkler
Bright Objects
Background Correction
Center of Mass X
Center of Mass Y
binaert
Grayscale (U8)
erodert
Grayscale (U8)
Number of Erosion
Blob analyse store sirkler
Dark Objects
inter-class variance
kopi
Grayscale (U8)
Kopierer bildet for at det ikke skal flimre
FyllerHull
Grayscale (U8)
IMAQ FillHole
Sender ut tyngdepunkt med subpiksel
K
w
cond
2-norm
Kondisjonstall
Norm
t-vektor
angels_deg
roll_kalibrering
pitch_kalibrering
heave_calibration
Kompenserer vekk statisk feil i heave
Kompenserer vekk statisk feil i roll og pitch
Relative Translasjon etter kalibrering
reset (nullstiller vinkler og translasjon)
Relative vinkler etter kalibrering
Runder
stop

Vedlegg B: LabVIEW-kode for aktivt vision system

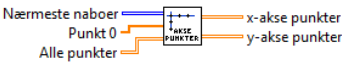

ROI1
ter1
open1
cam0
Controller
Original
Grayscale (I16)
prinsipalpunkt
Grayscale (U8)
Markerer prinsipalpunktet
Lager en linje i X-retning
Lager en linje i Y-retning
Image Out 3
Setter horisontal ROI manuell
Setter vertikal ROI manuell
Blob analyse på horisontale laserprikker
Blob analyse på vertikale laserprikker
253
255
Open
Antall horisontale partikler
Antall vertikale partikler
Center of Mass
Center of Mass
Bounding Rect
Her adderes horisontal og vertikal forskyvning etter utklipping av ROI
piksel str
Fokal lengde
Baseline
z
roll
pitch
Nordsor
VestOst
Sortert matrise
Rundetid
Midlere rundetid
200
antall runder
stop
ROI2
ter2
open2

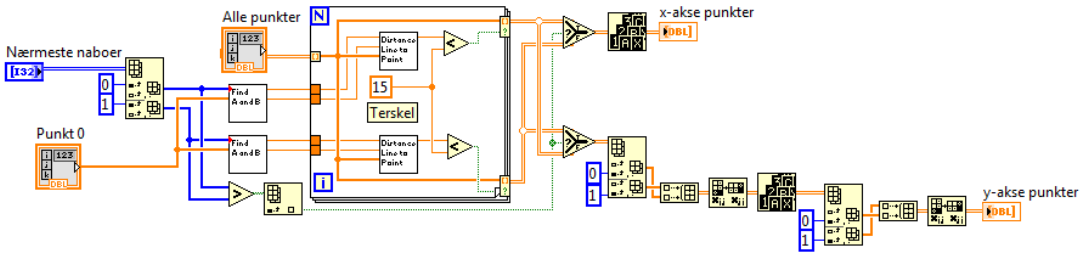
Vedlegg C: LabVIEW-kode for blokkene benyttet i Observ_punkt_ordnet.vi

Observ_punkt_ordnet_SubVI1.vi

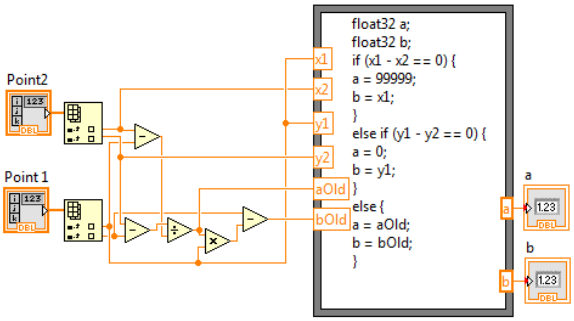


Observ_punkt_ordnet_SubVI2.vi

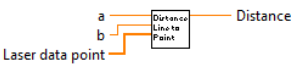


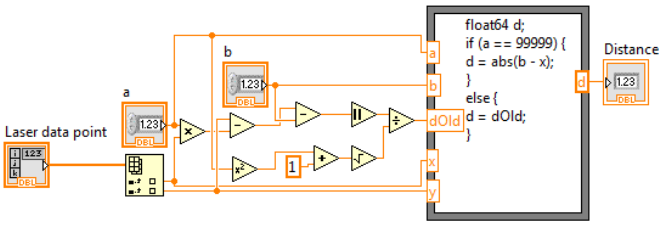


Find_a_and_b.vi



Distance_line_to_point.vi





Observ_punkt_ordnet_SubVI3.vi