NTNU
Det skapende universitet

Sanntids følging av nyrestein i ultralydavbildning

"Jakten på nyresteinen"

Jens Kristian Tøraasen

Master i teknisk kybernetikk
Oppgaven levert: Juni 2010
Hovedveileder: Geir Mathisen, ITK

Norges teknisk-naturvitenskapelige universitet
Institutt for teknisk kybernetikk

Oppgavetekst

5 15 % av verdensbefolking utvikler på et tidspunkt nyrestein, og andel nyresteinpasienter er økende. Etter en fjerning av nyrestein er sannsynligheten for igjen å få nyrestein ca 50%. Behandlingsformen for fjerning av nyrestein er enten ved bruk av ultralyd sjokkbølger, ved åpen (kikkhulls) operasjon eller ved inngrep gjennom urinleder, hvor bruk av ultralyd sjokkbølger er den foretrukne. Ved denne behandlingsformen plasseres pasienten på et bord, og ultralyd sjokkbølge apparatet fokuseres mot nyresteinen ved hjelp av en avbilningsteknikk og sjokkbølgebehandlingen startes. Ut fra pasientbevegelser (forflytting, hosting, pusting) kan nyresteinen komme ut av apparatets fokus permanent eller periodisk. Dette medfører at sjokkbølgen ikke hele tiden treffer nyresteinen, men omliggende anatomi.
Vi ønsker i denne oppgaven å se på metoder for å øke sjokkbølgenes treffsikkerhet

Oppgaven gitt: 15. januar 2010
Hovedveileder: Geir Mathisen, ITK

Sammendrag

Denne masteroppgaven omhandler design og utvikling av et system for tracking av nyrestein i et ultralydbilde. Problemstillingen innebefatter et literturstudie for å først undersøke muligheten for å lage et tracking systemet. Studiet tar for seg det biologiske aspektet som nyrefunksjon og nyresteinens oppbygning samt dagens behandlingsmetodikk for nyrestein. Videre beskrives avbildningsteknikker med spesiell vekt på ultralydavbildning samt et studium av eksisterende trackingalgoritmer fra bildebehandlingslitteratur. På bakgrunn av literaturstudiet, drøftes og designes et trackingsystem. Mean-Shift algoritmen blir valgt for tracking og sammen med forbehandling av bildet for å forbedre resultatet, blir systemet implementert. Bildebehandlingsbiblioteket OpenCV velges for å utføre selve trackingen. Et brukergrensesnitt blir bygget rundt trackingalgoritmen ved hjelp av programmeringsrammeverket Qt. For å teste og validere nøyaktigheten til systemet ble det utviklet en simulator hvor en teller ble inkrementert med samme frekvens som skuddfrekvensen til en ESWL maskin hver gang steinen var i sjokkbølgenes brennpunkt. Testen sammenlignet reel posisjon og tracket posisjon, samt fant forskjellen mellom et behandlingssystem med og et uten tracking av nyresteinens posisjon og bevegelser.

Forord

Arbeidet med masteroppgaven generelt har vært både spennende og utfordrende. Formen skiller seg mye fra alt vi tidligere har gjort ved NTNU, kanskje bortsett fra Prosjektoppgaven. Der vi vanligvis har fulgt en fagplan, lest forhåndsdefinert pensum og jevnt hatt frister i form av øvinger, måtte denne gangen både relevant litteratur og arbeidsfordeling bestemmes helt og holdent av oss selv. Måten å jobbe på har vært med på å gi en stor grad av mestringsfølelse, jeg har blitt sikrere på mine egne ferdigheter og kvalifikasjoner rent faglig og har på mange måter blitt klar for en fremtidig arbeidssituasjon.

Faglig og innholdsmessig har jeg vært veldig heldig med valg av oppgave. Jeg oppdaget en spesiell interesse for medisin og medisinsk kybernetikk under arbeidet og på denne måten kom motivasjonen av seg selv. Etter hvert som oppgaven gikk fremover og en løsning lå foran meg var det særdeles tilfredsstillende å kunne utvikle en fungerende prototyp.

Selv om det er mange som har vært til stor hjelp og støtte under arbeidet med denne masteroppgaven, vil jeg rette en spesielt stor takk til:

- Min veileder gjennom denne oppgaven, Geir Mathisen. Han har vist stor interesse og engasjement for både temaet og oppgaven min. Dette har vært svært motiverende for arbeidet. Geir har også hjulpet meget mye med å komme i kontakt med de riktige personene og ført en dialog med samarbeidspartnerne på St. Olavs Hospital.
- Carl-Jørgen Arum, Unni Bergan, Niels Thomas Finsen og Ingrid Høye ved St. Olavs Hospital for å ha satt i gang prosjektet og hatt troen på et resultat. Hjelp og tilgang til utstyr, samt anskaffelse av opptak fra pasienter til bruk under utvikling og testing. Til sist positiv tilbakemelding på prototypen, noe som gav mye motivasjon.
- Mine medstudenter Stig Hornang og Andreas L. Carlsen som har hjulpet meg svært mye med kildekodeutvikling. De har i tillegg vært gode kollegaer også sosialt i pauser på kontoret, G234.
- Medisinstudent Sindre Woxholt for oppklarende samtaler om biologi og anatomi.
- Familien min hjemme i Arendal og i tillegg Aslak Normann for god tilbakemelding på rapportens innhold like før levering.

Innhold

Figurer

Tabeller

Kapittel 1

Introduksjon

1.1 Motivasjon

Nyrestein rammer 5-15% av verdens befolkning og antallet er på stigende kurve. Kostnaden for samfunnet ved behandling er store, særlig med tanke på at opp mot 50% av alle tidligere nyresteinspasienter utvikler nye tilfeller av nyrestein. Den mest brukte behandlingsteknikken per dags dato er "extracorporeal shockwave lithotripsy" eller ESWL. Metoden bruker sjokkbølger rettet mot steinen for å riste, knuse og fragmentere opp steinen nok til at de gjenværende delene kan passere gjennom urinveiene. Til avbildning for å posisjonere pasienten og nyresteinen brukes røntgen. Ulempen med røntgen er at pasienten utsettes for skadelig stråling og det er derfor begrensninger på hvor lenge og hvor mange ganger en pasient kan avbildes. Dette er grunnen til at den ufarlige ultralydavbildingen er valgt. Selv om ESWL er tryggere enn åpen kirurgi og behandling gjennom urinveiene, er det fortsatt visse komplikasjoner knyttet til denne behandlingen. Pasienten kan oppleve smerter under behandling og blod i urinen og store blåmerker etter behandling. Dette er fordi pasientens bevegelser fører steinen ut av treffpunktet for sjokkbølgene fra behandlingsutstyret, noe som fører til at omliggende anatomi blir truffet. Denne bevegelsen kan skyldes periodiske bevegelser av innvollene pga respirasjon, men også annen pasientbevegelse forårsaket av for eksempel smerter.

På bakgrunn av dette er det derfor svært ønskelig å utvikle et system for tracking av nyresteinen inne i pasienten og å bruke ultralyd som avbildningsteknikk. Et slikt system vil kunne øke treffprosenten for sjokkbølgene og begrense komplikasjoner hos pasienten. I tillegg kan behandlingstiden om mulig begrenses fordi pulsfrekvensen og effekten kan økes da man vet at omliggende vev ikke blir skadet.

1.2 Disposisjon

Oppgaven kan deles opp i tre deler. Først kommer litteraturstudiet, deretter implementasjon og til sist test. Litteraturstudiet går inn på biologi, avbildningsteknikker og teknikker for tracking. I implemenatsjonsdelen blir først fremgangsmåter evaluert før det overordnede systemet designets. Dette designet implementeres for så å testes i kapittelet om test og resultater. Resultatene diskuteres i diskusjonskapittelet.

Biologi: Forklarer urinsystemet, nyrene og nyrestein, samt går inn på behandlingsmetoder for nyrestein.

Avbilding: Tar for seg teknikker som brukes både for å diagnostisere nyrestein og teknikker som brukes under behandling av nyrestein.

Tracking: Her blir flere metoder for bildebehandlingsmetoder for tracking presentert og virkemåte blir forklart.

Design: På bakgrunn av teori fra tidligere kapitler, blir et system foreslått.

Implementasjon: Systemet fra designkapittelet implementeres med eksterne bibliotek og rammeverk. Et brukergrensesnitt for operatørkommunikasjon utvikles og dokumenteres.

Test: Ut fra en definert testplan beskrives resultatene av systemets prestasjon. Flere videofiler av ultralyd med nyrestein brukes for å undersøke kvaliteten på trackingalgoritmen og implementasjonen.

Vedlagt oppgaven ligger en DVD med program og installasjonsfil kompatibel og testet på MS Windows Xp, Vista og 7. Programkoden og Qt prosjektfilen er lagt ved samt testvideo og rådata fra testkjøringene utført i Test og Resultater kapittelet.

Kapittel 2

Biologi

2.1 Urinsystemet

2.1.1 Urinsystemets formål

Urinsystemet består i all hovedsak av nyrer, urinledere og urinblære. I tillegg har vi også urinlukkemuskel, urinrør og urinrørsåpning som har som oppgave å kvitte seg med den ferdig produserte urinen. Urin er et avfallsstoff som skilles ut fra kroppen gjennom urinrørsåpningen etter at den har blitt produsert i nyrene og lagret i urinblæren.

Urinsystemets hovedfunksjon er å kontrollere sammensetningen og volumet av kroppsvæskene. Denne veskekontrollen blir ivaretatt ved hjelp av flere prosesser som alle foregår i nyrene. [22]

1. Utskilling. I urinsystemet skilles det meste av kroppens avfallsstoffer ut. Dette skjer ved en filtrering av blodet i nyrene. Mange av stoffene som skilles ut i nyrene er skadelige for mennesker. Selv om avfallstoffer også skilles ut i hud, lever, lunger og tarmsystemet vil det være umulig for kroppen å kvitte seg med alt avfallet ved svikt av nyrefunksjonen.

2. Regulering av blodvolum og blodtrykk. Ved å produsere en stor mengde tynn urin, eller en liten mengde konsentrert urin regulerer urinsystemet hvor mye væske blodet inneholder til enhver tid. Når det er mye væske i blodet vil blodtrykket øke. I motsatt tilfelle får vi lavere blodtrykk. Det er verdt å merke seg at det er snakk om den ekstracellular væsken, det vil si den væsken som befinner seg fritt utenfor cellene. Væsken inne i cellene kalles intracellular væske og denne væsken er viktig for at cellen skal fungere normalt. Den

ekstracellular væsken er med på å tilføre cellen den væsken de trenger, derfor er det svært viktig at blodtrykk og blodvolum er nøye regulert av blant annet urinsystemet[1].

3. Regulering av konsentrasjonen av stoffer i blodet. En av de viktigste oppgavene til blodet er å frakte forskjellige næringsstoffer rundt i kroppen. Urinsystemet er med på å regulere konsentrasjonen av molekyler og forskjellig ioner som til enhver tid må fraktes med blodet til cellene i kroppen. Stoffer som kan nevnes er blant annet glukose, natrium (Na^+), klor (Cl^-) og kalium (K^+). Natrium, kalium og klor er noen av de viktigste stoffene for å iverksette muskelkontraksjon.

4. Regulering av ekstracellurært pH nivå. Ved å skille ut forskjellige mengder H^+ regulerer nyrene til enhver tid surhetsgraden i blodet

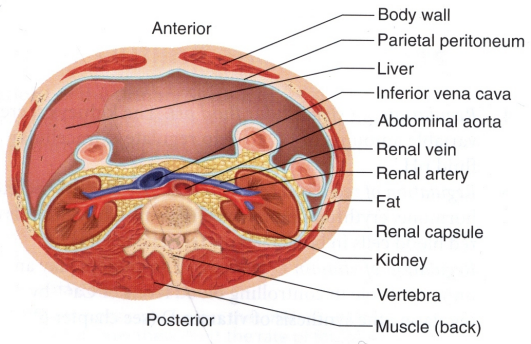
5. Nyrene spiller også en sentral rolle i produksjon av nye røde blodceller samt vitamin D syntesen.

2.1.2 Anatomi

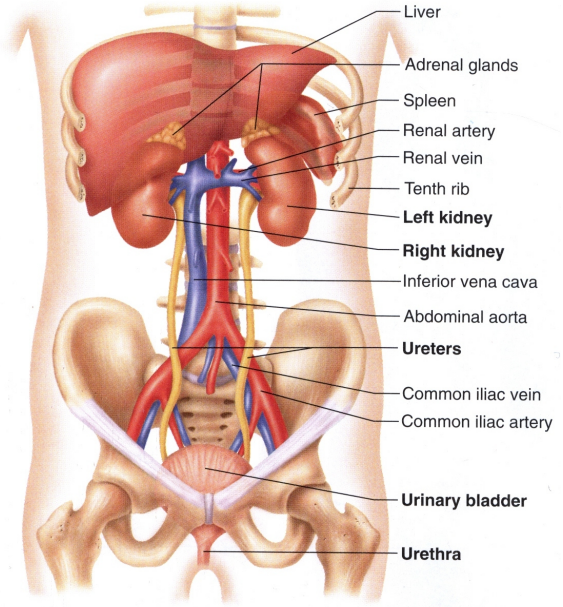
Øverst i urinsystemet ligger nyrene. Hos mennesker er nyrene plassert inntil bakre bukvegg. Den høyre nyren ligger liker under mellomgulvet og like bak leveren mens den venstre nyren ligger under mellomgulvet og like bak milten. Nyrene har form som minner om to bønner. De ligger speilvendt i forhold til hverandre og med arterie og vene samt urinleder inn på medial (inn mot ryggsøylen) side. Denne siden av nyren kalles hilum. Urinlederne heter ureter og det er gjennom disse at urinen blir fraktet fra nyrene og til urinblæren for lagring. Når urinblæren tømmes renner urinen gjennom urinrøret (urethra) og ut urinveisåpningen. Hos begge kjønn går urinrøret fra urinblæren og til kjønnsorganet, men hos menn går urinrøret også gjennom prostata. Dette er fordi urinrøret hos menn også brukes til forplantning.

2.1.3 Urinblære og urinveier

Urinlederne er som nevnt rørene som frakter urinen fra nyrene og til urinblæren. Hver leder går ut fra hilum på de to nyrene og møtes i den bakre underliggende delen av urinblæren. Hver urinleder er vanligvis mellom 25-30 cm og har et tverrsnitt på ca 5mm [15]. Veggene i urinlederen består av bindevev og glatt muskulatur. Glatt muskulatur er muskulatur som vi finner i de hule innvoldsorganene i kroppen. De skiller seg fra for eksempel skjelettmuskulaturen, som vi bruker for å bevege oss, ved at de har en annen form som gjør dem spesielt egnet til de oppgavene de utfører i organene. For at urinen skal fraktes fra nyrene til urinblæren,



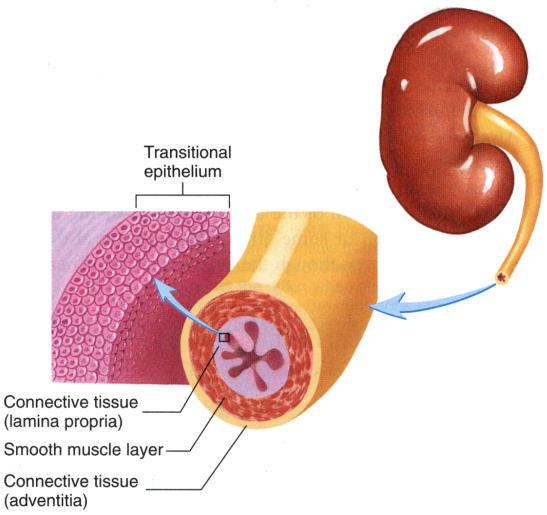
Figur 2.1: Nyrene og omsluttende organer sett ovenfra. © Seeley's Essentials of Anatomy and Physiology



Figur 2.2: Urinsystemet og omsluttende organer sett fra anterior synsvinkel. © Seeley's Essentials of Anatomy and Physiology

foregår det peristaltiske bølgebevegelser i den glatte muskulaturen i urinlederne. Denne er den samme type bevegelse som finner sted i tarmene når mat fraktes gjennom fordøyelsessystemet. Urinen pumpes altså ned i urinblæren [22].

Etter at urinen har blitt fraktet gjennom urinlederne kommer den til urinblæren for lagring. Urinblæren er bygget opp av de samme lagene som urinlederen, et glatt muskellag, og to lag med bindevev. Størrelsen på urinblæren er avhengig av hvor mye urin den inneholder. Den kan romme fra et par milliliter til rundt 1000



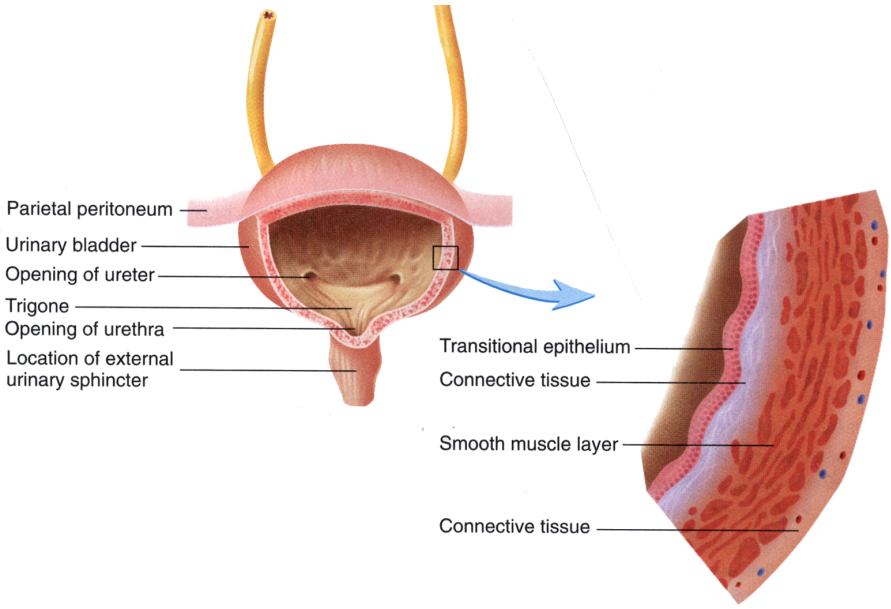
Figur 2.3: Et utsnitt av urinlederen med muskler og bindevev. © Seeley's Essentials of Anatomy and Physiology.

milliliter. På de innerst på veggene i både urinlederne og urinblæren finnes et lag som består av overgangsepitel. I tillegg til å beskytte underliggende bindevev[14], er dette laget spesialisert for å la seg strekke. Når blæren eller urinlederen er liten, for eksempel når det er lite urin i blæren, vil cellene i epitel laget ligge tett og i flere lag utenpå hverandre. Så fort blæren strekkes vil dette laget strekke seg utover slik at cellen vil legge seg ved siden av hverandre i stedet for bak hverandre. Urinblæren munner ut i urinrørt nederst på blæren. Det er gjennom dette røret at urinen til slutt tømmes ut av kroppen. Begge kjønn har en lukkemuskel, også kalt ytre lukkemuskel, for å kontrollere når urinblæren skal tømmes. [22].

Når urinblæren vegger strekkes aktiveres urineringsrefleksen. Urin får urinblæren til å strekke seg. Denne strekken stimulerer nerver i urinblærens vegger som sender et signal til ryggmargen. Her sendes nye signaler til musklene i urinblæren som trekker seg sammen samtidig som den ytre lukkemuskelen avslappes. Denne prosessen er egentlig en automatisk refleks, men det er mulig å overstyre den fra høyere hjernesentre. Når vi overstyrer refleksen vil ikke urinblæren trekke seg så mye sammen og den ytre lukkemuskelen vil holde seg lukket. Evnen til å overstyre urineringsrefleksen utvikler seg normalt i 2-3 års alderen.[22]

2.1.4 Nyrene

Nyrene er den viktigste delen av urinsystemet. Det er her urinen produseres når blodet filtreres. Rundt nyrene er det ytterst et lag med fet. Dette laget er til for

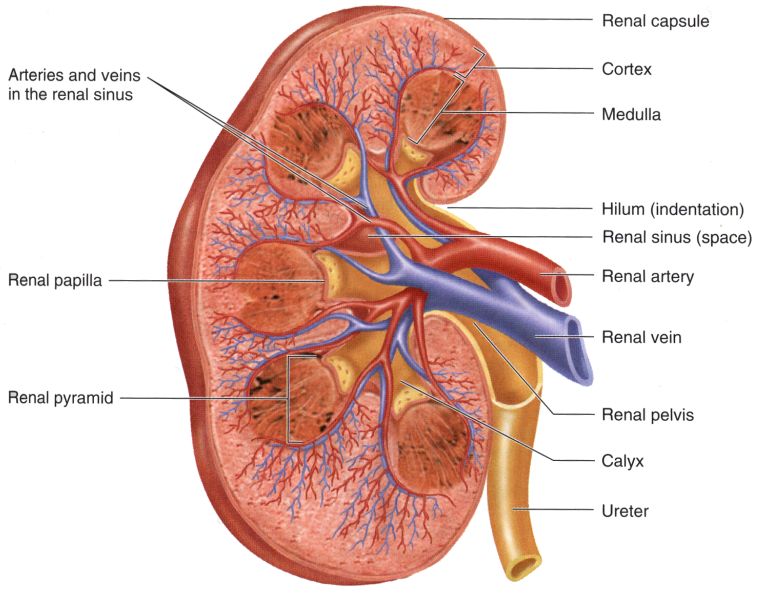


Figur 2.4: Et snitt av urinblæren med muskler og bindevev. © Seeley's Essentials of Anatomy and Physiology.

å absorbere og beskytte nyren mot mekaniske støt. Innenfor fettlaget er det et omsluttende lag med bindevev som kalles renal capsule. Renal kommer fra det latinske ordet for nyre. På den mediale siden av nyren, hilum, er blodårer tilkoblet nyren. Ureter har også utgang fra hilum via renal pelvis, som er navnet på et litt større rom hvor urin samler seg fra forskjellige beger kalt calyx. Calyx igjen, er munningen på en av flere nyrepyramider i nyren. Nyrepyramider består av nepfroner og det er i disse nepfronene at selve filtreringen av blodet og dermed produksjon av urin foregår.

Vi kan dele nyrepyramidene inn i to deler. Den øverste delen, altså den som ligger nærmest renal capsul, kaller vi for cortex. Her foregår selve filtreringen av blodet. I cortex finner vi blant annet Bowman's kapsel. Denne kapselen omslutter arterier og ved hjelp av porer i arteriene og spesialiserte celler i veggene på Bowman's kapsel filtreres væske og oppløste stoffer over i rør som urin. Det neste steget i produksjonen er å gjøre urinen konsentrert og å føre tilbake vann til blodet. Som nevnt i 2.1.1 er en av urinsystemets oppgaver å regulere blodvolum og blodtrykk, dette ble gjort gjennom å regulere mengden væske i blodet. Prosessen med å føre tilbake væske foregår i den nedre delen av nyrepyramiden, også kalt medulla. Her ligger foller av tynne urinførende rør. Medulla er hyperosmolar, det vil si at den har en lav konsentrasjon av væske. Når den vannholdige urinen passerer gjennom medulla i de urinførende rørene vil væsken trekke over i medulla og deretter trekke

over i det filtrerte blodet som også er hyperosmolart etter å ha gått gjennom Bowman's kapsel. På denne måten blir avfalsstoffene værende igjen i urinen mens den rene væsken fortsetter med blodet. Natrium og klor blir også filtrert ut i medulla og tatt opp igjen i blodet i denne prosessen. Den konsentrerte urinen vil samle seg i større rør og til slutt ende opp i en calyx [22].



Figur 2.5: Et snitt av nyren. © Seeley's Essentials of Anatomy and Physiology.

2.2 Nyrestein

2.2.1 Hva er nyrestein

Nyrestein er det vanligste urologiske problemet både hos menn og kvinner. I Norge rammer nyrestein opp til 10 % av den mannlige delen og 5 % av den kvinnelige delen av befolkningen. Nyrestein er en gjentagende tilstand, det vil si at pasienter med nyrestein har anlegg for å få nye nyresteiner og nyresteinsanfall. Innen 5 år vil 50 % av pasientene få nye anfall og innen 20 år vil 70 % av tidligere pasienter være rammet på nytt[17].

Nyrestein kan deles inn i fire grupper etter hva som de består av og hva som er grunnen til at de forekommer: Kalsiumsteiner, Urinsystestein, Cystinstein og Struvitestein. Kalsium steiner er den vanligste formen og kan deles inn i kalsium oxalat og kalsium fosfat. Denne gruppen utgjør 75-85 % av alle steiner og er vanligst

hos menn[12]. Steinene er i all hovedsak bygget opp av krystaller av kalsiumkoksalt eller kalsiumfosfat. Urinsyresteiner er den nest mest vanlige formen for nyrestein med en hyppighet på 5 til 10%. Også denne typen stein er vanligst hos menn. Steinene er krystallisert urinsyre og steinene betegnes som organiske. Fordi steinene er organiske vises de ikke nødvendigvis ved røntgenavbildning, men den kan sees ved bruk av ultralyd. En annen organisk steintype er cystinstein. Steinene dannes av aminorsyren cystein og er relativt sjelden. Den siste steintypen utgjør 5 til 10 % av alle forekomstene av nyrestein. Den er mest vanlig hos kvinner og består av magnesim-ammoniumfosfat som er et salt [21]. Dette saltet er ikke naturlig i urinen, men oppstår ved bakteriell urinveisinfeksjon.

Nyresteinene vil etter hvert passere ut med urinen. Noen ganger er steinene så små, under 5mm, at de forsvinner uten symptomer. Når steinene blir større enn dette vil de gi symptomer som smerte ved utskillelse. Da kan det være nødvendig med behandling for å gjøre det lettere og mindre smertefullt å skille ut steinen. Hvis urinstrømmen fra nyren blir helt avstengt vil det være helt nødvendig å fjerne steinen for å berge nyren.

2.2.2 Grunn til forekomst av nyrestein

I all hovedsak dannes alle typer nyrestein ved at salter i urinen binder seg og blir faste masser. Som nevnt i 2.1.1 er et av urinsystemets hovedoppgaver å regulere konsentrasjoner av stoffer i blodet. Blant disse er saltene som kan danne nyresteiner. Nyresteiner oppstår som regel når balansen mellom væske og salter i nyre og blod bryter sammen. Nyrene må sørge for å bevare væske i kroppen samtidig som stoffer med lav oppløselighet skal skilles ut. Mennesket er i tillegg i aktivitet, temperatur forandrer seg og næringsinntaket varierer. For å forhindre krystallisering av salter inneholder urinen stoffer som motarbeider krystallisering, men disse stoffene fungerer bare til en viss metningsgrad. Når saltene overstiger denne metningsgraden sier vi at urinen er overmettet. Urinen kan være overmettet på grunn av dehydrering eller ved for høyt nivå av kalsium, cystin eller urinsyre. Urinen må gjennomsnittlig være overmettet over en lengre periode for at saltene skal danne krystaller. Urinens pH verdi er også avgjørende for dannelsen av steiner. Urinsyre krystalliseres ved en pH på under 5,5, basisk urin har letter for å få opphopninger av kalsiumfosfat, mens kalsiumoxalatverdien er forholdsvis upåvirket av syrenivået [12].

Krystallisering oppstår så lenge urinen er overmettet. I tillegg kan andre krystaller og avleiringer fungere som grobunn for steinkrystaller slik at saltene krystalliserer seg selv om metningsgraden ikke er nådd. Krystallene gror så lenge urinen fortsetter å være overmettet og små krystaller vil vokse sammen til en nyrestein.

2.2.3 Symptomer

Det vanligste symptomet på nyrestein er smerter. Dette er fordi steinen oppnår en størrelse som er for stor til å kunne passere gjennom urinlederen og kiler seg fast i enten selve ledere eller i overgangen mellom nyre og leder. Når steinen kiler seg fast mellom nyre og leder vil trykket i nyren øke pga fortsatt urinproduksjon. Dette vil igjen føre til store smerter og såkalt nyrekolikk. Hvis steinen setter seg fast i urinlederen, gir det tilsvarende smerte ned mot skrittet og hvis steinen fester seg nede i urinledere kjennes smerten i blæren. Felles er at smerten fører til kvalme, blek hud og rask pust. Nyresteinsmerter er av de sterkeste smertene vi kan oppleve[21].

I tillegg til smerter er blod i urinen og infeksjoner tegn på at man kan ha nyrestein.

2.2.4 Konsekvenser

Nyresteinspasienter får ofte urinveisinfeksjon fordi den lokale irritasjonen fører til nedsatt motstandskraft mot infeksjoner. Dette kan igjen føre til videre steindannelser og mer smerter og plager. Hvis ikke løpet stanses kan det føre til nyresvikt og ødeleggelse av nyren. Når steinen blokkerer urinlederen kan pasienten få akutt nyrebekkenbetennelse. Trykket i nyrene vil øke slik at puss fra betennelsen kan gå tilbake i blodet og pasienten blir blodforgiftet. Dette er dødelig uten behandling. [21]

2.2.5 Behandling

Når det har gått så langt at pasienten trenger behandling for å fjerne steinen finnes det både medisinske og kirurgiske hjelpemidler. Ved hjelp av medisinsk behandling kan passeringen av mellomstore og små steiner forenkles. Hvis steinen derimot er for stor til å i det hele tatt passere gjennom urinlederen kreves det kirurgiske tiltak. Et alternativ er åpen opperasjon hvor kirurgene åpner nyren og tar ut steinen. Denne formen for behandling har blitt mindre og mindre vanlig takket være teknologiske fremskritt innen urologien. Jameson et. al[12]presenterer tre teknikker som brukes i dag. “Extracorporeal shock wave lithotripsy” (ESWL), “Percutaneous nephrolithotomy” og “Ureteroscopy”. De to siste teknikkene baserer seg på å gå inn i kroppen for deretter å ødelegge steinene direkte. “Percutaneous nephrolithotomy” blir utført ved at et instrument føres inn via et snitt i siden på pasienten og inn i renal pelvis. Deretter blir instrumentet rettet mot steinene og enten utsatt for ultralyd sjokkbølger eller holmium laser. Ved “Ureteroscopy” går mann inn via urinrøret og opp i urinlederen. Her bruker man en holmium laser for å knuse steinen.

ESWL er den vanligste formen for behandling og utgjør 80-85% av totalt antall behandlinger[17], det er også denne behandlingsmetoden denne oppgaven skal se på å forbedre ved å tracke nyresteinen hos pasienten. Jeg vil derfor gå litt nærmere inn på denne behandlingsmetoden.

ESWL ble utviklet i 1980 av Tyske Donier Medizintechnik. Før denne teknikken ble introdusert ble alle nyresteinsbehandlinger for store steiner utført ved åpen kirurgi. Åpen kirurgi utsetter pasienten for både stress og fare for betennelser samt tung bedøvelse. ESWL ble raskt den mest brukte behandlingsmetoden. Når en pasient behandles med ESWL vil steinen pulveriseres inne i selve nyren for så å bli skylt ut med urinen. Pulveriseringen blir utført ved at steinen blir plassert i et brennpunkt hvor flere ultralydsjokkbølger treffer samtidig. Energien i dette punktet vil være stor nok til at steinen ristes i stykker og pulveriseres. Sjokkbølgene vil ikke skade vevet det forplanter seg i, siden energien i hver enkelt sjokkbølge ikke er høy nok. Dette er med på å gjøre behandlingsmetoden trygg for omliggende organer. For å fokusere sjokkbølgene i et punkt bruker man de geometriske egenskapene til en ellipse. I ellipsens ene brennpunkt blir sjokkbølgene generert og skutt mot ellipsen. På grunn av ellipsens egenskaper vil sjokkbølgene møtes igjen i det andre av ellipsens brennpunkter. Dette punktet kan beregnes og man kan fokusere det over nyresteinen. Bølger taper energi i overgangen mellom stoffer av forskjellig tetthet. For å ikke tape energi mellom sjokkbølgegeneratoren og nyresteinen bruker man derfor gel eller vann for å få en kobling mot huden [7]. St. Olavs Hospital har en Donier Compact Delta maskin hvor en ball med ultralydgel er kontaktmateriale mellom maskin og pasient.

Selv om behandlingsmetoden er en forbedring i forhold til tidligere åpen kirurgi, vil det også ved bruk av ESWL kunne oppstå komplikasjoner hos pasienten. Det er et mål med oppgaven å forminske eller eliminere noen av disse komplikasjonene og ettervirkningene hos pasientene. Noen pasienter utvikler høy konsentrasjon av bakterier i urinen etter behandling, særlig de pasientene som har infeksjonsrelatert stein som for eksempel struvitestein.Grasso[7] foreslår antibiotikabehandling for å forebygge dette. Når pasienten er under behandling vil steinen bevege seg inn og ut av brennpunktet pga pasientens respirasjon eller bevegelse ved smerte. Selv om pasienten er bedøvet og derfor ikke reagerer sterkt på smerte vil alltid bevegelsen være til stede. Når steinen er utenfor brennpunktet treffer sjokkbølgene direkte i vevet og kan påføre skade. Også når sjokkbølgene treffer steinen kan omgivelse ta skade. Sjokkbølgene vil skape kavitasjon i steinens omliggende urin. Kavitasjons-boblene gir en mer effektiv destruksjon av steinen, men det vil også påvirke det omliggende vevet. Pasienter kan oppleve blod i urinen, indre blødninger, smerter i forbindelse med indre blødninger og store blåmerker. Noen pasienter kan utvikle kronisk høyt blodtrykk etter behandling, men dette er en sjelden ettervirkning[7]. Hvis steinene ikke knuses tilstrekkelig vil noen pasienter få smerter og kolikk når

de skal skille ut nyresteinrestene gjennom urinrøret. Ved tracking av nyresteinens posisjon vil forhåpentligvis indre blødninger minskes fordi treffprosenten vil være høyere enn med dagens system. Hvis man er garantert å treffe nyresteinen kan man også øke effekten slik at steinene knuses mer og smerter ved utskilling av fragmenter vil bli begrenset. Dette gjenstår å testes og valideres i videre arbeide og er kun en motivasjon for denne oppgaven.

Kapittel 3

Avbildning

I dette kapittelet skal jeg ta for meg forskjellige avbildingsteknikker som brukes for å diagnostisere nyrestein og som brukes som støtte under ESWL behandling. Jeg vil trekke frem fordeler og ulemper med teknikkene i forbindelse med avbildning av nyrestein og jeg vil gå mer i dybden på ultralyd for så se på hvilke utfordringer som i forbindelse bildebehandling og tracking av nyrestein. Jameson et. al. trekker frem Røntgen, Ct og Ultralyd for bruk i diagnostikk av nyrestein [12]. Det er de metodene som vil omtales i påfølgende kapitler.

3.1 Røntgen

3.1.1 Virkemåte

Røntgen er en form for elektromagnetisk stråling. I teorien fungerer det på samme måte som vanlig synlig lys, men har en mye høyere energi og kortere bølgelengde. Når et atom kolliderer med en fri partikkel vil et elektron, så lenge det er nok energi i kollisjonen, hoppe ut i en høyere bane rundt kjernen. Denne tilstanden er ikke naturlig for atomet og elektronet vil hoppe tilbake i den originale banen det var i. Når dette skjer, frigjøres energi fra atomet i form av et foton. Hvis energien er lav nok faller bølgelengden inn under vanlig synlig lys, men hvis den er høy vil røntgenstråler frigjøres[8].

I en røntgenmaskin produseres disse strålene ved at en elektronstrøm drives mellom en katode og en tungsten anode. Spenningsforskjellen mellom katoden og anoden er meget høy slik at elektronene treffer tungstenatomene i høy fart og med mye kinetisk energi. Det produseres mye varme i anoden så derfor er den roterende for å forhindre at elektronene treffer samme punkt og anoden smelter. Fotonene

frigjøres på samme måte i tungstenatomet, men utløsermekanismen er forskjellig fra den som inntreffer før eksitasjon av synlig lys. Når er fritt elektron treffer et av elektronene i bane rundt tungstenkjernen, vil de slå elektronet helt ut av banen. Det blir da en ledig plass på en lavere bane slik at et elektron på høyere bane vil flytte seg ned og frigjøre røntgenstråler. Fotoner kan også bli frigitt uten at elektronet kolliderer med atomet. Når et elektron passerer nære tungstenkjernen kan det bli dratt inn av atomkjernen på samme måte som når en komet kommer for nære en planet. Elektronet vil da bremses litt ned i det det bøyer seg rundt kjernen. Energi frigjøres i nedbremsingen i form av røntgenstråler. Elektronet vil slippe løs fra kjernen fordi det ikke har noen bane å gå i [10].

I en røntgenmaskin vil røntgenstrålene hindres i å bevege seg i andre retninger enn den en ønsker. Pasienten plasseres slik at det man ønsker å gjennomlyse er i fotonenes bane. Når fotonene treffer kroppen vil de absorberes i forskjellig grad ettersom hva de treffer. Fett, musker og væske har lavere tetthet enn ben. Det vil derfor ikke være like mange fotoner som blir absorbert i disse stoffene, mens mye av røntgenbølgene vil bli dempet mot bein. På andre siden av pasienten har man et stoff som reagerer med fotonene som kommer seg gjennom pasienten. I dag har man gjerne en digital sensor for å detektere røntgenstråler og generere bilder[8].ja

3.1.2 Ulemper

Når en pasient utsettes for røntgenstråling utsettes vedkommende også for såkalt inonisert stråling. Ionisert stråling er stråling hvor energien er så høy at den kan ødelegge strukturen i stoffene i kroppen. Dette er fordi fotonene for eksempel kan treffe atomer slik at elektroner blir slått ut av bane rund kjernen. Atomet vil da bli positivt ladet siden kjernen nå inneholder flere protoner enn det er elektroner i bane. Vi sier at atomet er ionisert. Dette kan i sin tur utløse kjemiske prosesser i kroppen som kan føre til ødeleggelse av for eksempel DNA og protein molekyler. Skaden som påføres et menneske kan bare begrenses ved å ikke utsette seg for stråling, det vil si at det beste er å ikke utføre røntgenavbildning i det hele tatt. I alle fall hvis det ikke er ytterst nødvendig[8].

3.1.3 Nyrestein i et Røntgenbilde

Røntgen brukes både til diagnose av nyrestein og under ESWL behandling. For å diagnostisere en pasient tar man et ultralydbilde forfra eller bakfra. Det bløte vevet rundt steinen vil ha liten dempningseffekt på røntgenstrålene, mens steiner og ben vil absorbere stråling. Ulempen med denne typen avbildning av nyrestein er at det bare er mulig å oppdage steiner som er laget av kalsium. Dette er allikevel

mesteparten av steinene siden 75-85% av alle nyresteiner er kalsium steiner. I tillegg vil mange av urinsyre- og struvitesteiner føre til at man også danner kalsium steiner, fordi de fungerer som grobunn og endrer den kjemiske sammensetningen i urinen[12]. Da vil kalsiumsteinene gjøre at også denne typen steiner blir synlige.

Under ESWL behandling bruker man røntgen for å fokuserer brennpunktet for sjokkbølgene over nyresteinen. Siden man ikke kan se dybde i et røntgenbilde roterer man vinkelen fra rett ovenfra til ca 30 grader fra siden. På denne måten får man plassert steinen i brennpunktet, men det tillater ikke konstant overvåkning av steinen i og med at røntgenstråler er skadelige for mennesker. Man ønsker å begrense eksponeringstiden og bruker derfor bare røntgenmaskinene til å innstille første gang og til å rette på pasientens posisjon få ganger i løpet av behandlingen. Gjennomlysning brukes også for å se om steinen har begynt å løse seg opp.

Figur 3.1: Et røntgenbilde av en nyresteinspasient tatt forfra. Bildet avslører flere nyresteiner. WIKIMEDIA

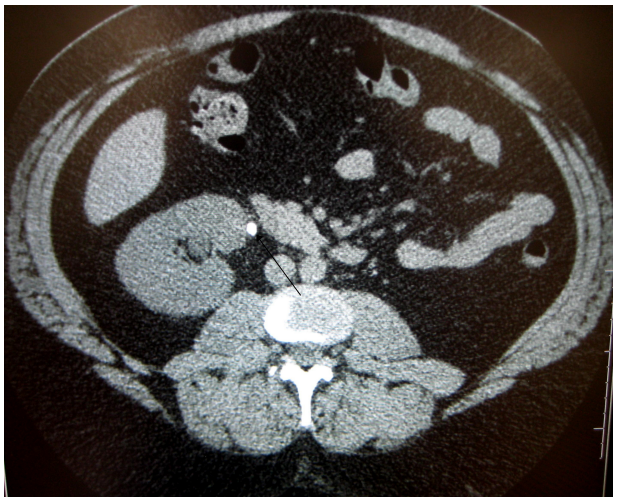
3.2 CT

3.2.1 Virkemåte

CT skanning baserer seg på røntgenstråling, men istedenfor å ta et bilde gjennom pasienten fra ett plan, tar man et tynt snittbilde rundt hele pasienten fra forskjellige vinkler. For å få til dette blir pasienten ført inn i en ring hvor røntgengeneratoren roterer rundt med en deteksjonsflate på motsatt side. Snittene blir prosessert og kombinert i en datamaskin og presentert som et helt snitt av pasienten gjennom hele planet. En av de store fordelene med CT skanning kontra bilder med standard røntgen er at evnen til å skille mellom endringer i vev er mange ganger større. Hvor vanlig røntgen krever en endring på 10 % for å vise forskjell vil CT oppdage forskjeller på 0.5 % [8].

3.2.2 Nyrestein i CT bilde

Siden CT skanning krever mer omfattende utstyr brukes det bare til å diagnostisere nyrestein. Utstyret som brukes krever mye plass og lar seg ikke kombinere med ESWL utstyr. Fordelen med CT er at steintyper som ikke er er kalsium blir avbildet. På denne måten kan alle steiner oppdages og man kan bruke bilder tatt på forhånd som veiledning under ESWL behandling.



Figur 3.2: Et snitt av en pasient med nyrestein tatt med CT. Nyresteinen er tydelig hvit midt i bildet. WIKIMEDIA

3.3 Ultralyd

3.3.1 Virkemåte

Ultralyd skiller seg fra de to forrige avbildningsteknikkene. Hvor røntgen og CT baserer seg på penetrasjon baserer ultralyd seg på refleksjon. Som navnet tilsier er ultralyd høyfrekvent lyd. Frekvensen ligger i området mellom 2 til 15 MHz. Lyden genereres i en transduser ved å utsette piezoelektriske krystaller i transduseren for elektromagnetiske bølger. Krystallene vil vibrere og danne ultralydbølger. Etter at ultralydbølgene er sendt, bytter transduseren funksjon og gjør seg klar til å motta lyd. Den registrer den mottatte lyden ved at de samme krystallene som genererte ultralyd også genererer elektriske signaler når de vibrerer. Svingningene i lyden som treffer transduseren vibrerer altså krystallene og generere et elektrisk signal. Trancduseren, som også kalles probe, plasseres inntil det legemet man ønsker å undersøke. Deretter sørger ultralydmaskinen for å bytte på å sende ut og registrere ultralyd. Lydbølgene trenger inn i legemet og vil reflekteres tilbake ettersom den treffer forskjellige medium som vann, bløtvev og ben. Ekko signalet sier noe om egenskapene til det som blir undersøkt. Tiden det tar før signalet blir returnert forteller hvor langt unna ekkoet kom fra, altså hvor langt inn i kroppen det reflekterende vevet er. Amplituden på det returnerende signalet forteller noe om tettheten til objektet. Forskjell i amplitude på ekko signalet skyldes at lyd beveger seg med forskjellig hastighet i forskjellige typer vev. Dette er fordi hvert type medium har forskjellig impedans for ledning av lydenergi, akustisk impedans. Desto større forskjellen i akustisk impedans mellom nærliggende medier er, desto høyere er amplituden på retursignalet. Ekko inntreffer altså bare når lyden forplanter seg fra et type medium til et annet, for eksempel mellom bløtvev og ben. Det er verdt å merke seg at når ultralydfrekvensen øker, vil også oppløsningen på bildene bli bedre. Samtidig vil evnen til å gå dypt inn i kroppen bli mindre[2].

I følge Bowra og Russell [2] kan vi dele ultralyd inn i 4 grupper. A-mode er når ekko signalet vises som en graf hvor x-aksen er dybden, mens y-aksen er amplitude. M-mode er en sekvens av A-mode avbildninger som har til hensikt å vise bevegelse. B-mode er når bildet representeres med et 2d bilde av ultralydplanet. Her vil amplituden på det returnerte signalet bestemme lysintensiteten objektet får og tiden det tar for å returnere ekkoet vil avgjøre hvor langt ned på bildet objektet er. Det er denne formen som brukes videre i oppgaven. I tillegg har man doppler avbildning som bruker doppler effekten til å analysere strømninger for eksempel blod.

Det finnes flere forskjellige ultralydprober. De vanligste er lineær rekke, kurvelineær rekke og fase rekke. Lineær er typisk for høye frekvenser og for å se detaljert i overflaten. Bildet fra disse probene er rektangulært. Kurvelineær gir et bredere

synsfelt og har lavere frekvens slik at de kan se dypere. De gir et bilde med form som en trapes med buet flate på det grunneste og på det dypeste. Det er denne typen probe som brukes for se på nyre og nyrestein. Fase rekke probe har en samling av krystaller som sveiper over et område i løpet av en tid i motsetning til de to foregående som hadde flere krystaller som dekket hver sitt område av synsfeltet[2].Det finnes også egne ultralydprober for å lage 3D bilder. Disse har en matrise av krystaller som en utvidelse av de vanlige rekketypene for 2D. De tar da flere snitt som kan settes sammen til en 3D form av bildet.

3.3.2 Egenskaper i ultralydbildet

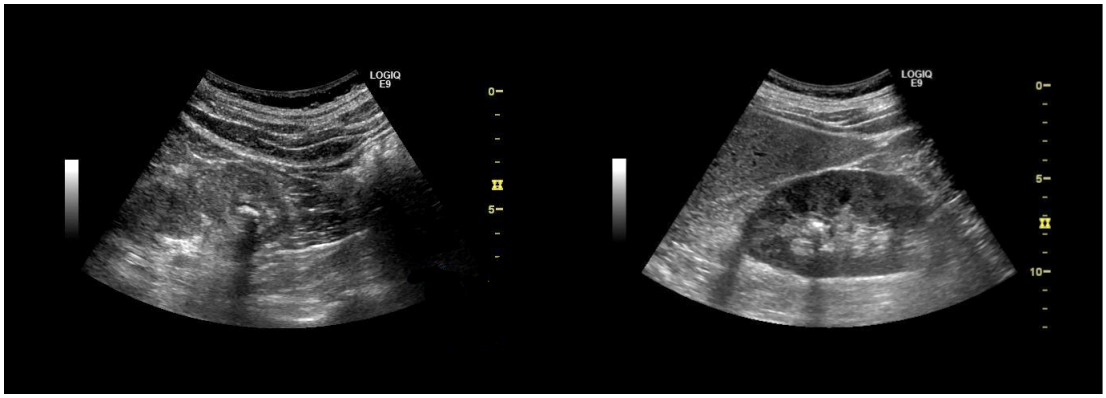
Et ultralydbilde kan inneholde informasjon som ikke er direkte knyttet til pasientens anatomi. Disse egenskapene heter artefakter. De kan være ødeleggende for forståelsen av bildet, men hvis det tolkes riktig kan de gi en bedre forståelse[2].

- Akustisk forsterkning og akustisk vindu: Dette inntreffer nå lyden går gjennom væske. Væsken er uniform slik at mindre lyd returneres som ekko. Signalet blir ikke dempet like mye som når det går gjennom for eksempel omliggende bløtvev og strukturene nedenfor væskeansamlingen vil sees tydeligere enn om den hadde ligget nedenfor andre medium.
- Akustisk skygge: Når lyden treffer en struktur med høy akustisk impedans i forhold til omliggende struktur vil svært mye av lyden reflekteres. Når nok ultralyd reflekteres vil det ikke være nok energi igjen til å nå dypere strukturer og det vil oppstå en skygge.
- Kant skygge: Skygger under buede strukturer som kanten av en blære eller kanten av nyrene. Skyldes høy tetthet og mye refleksjon.
- Tilbakekastning: Skjer når lyden reflekteres flere ganger mellom proben og en struktur eller mellom to strukturer. Signalet vil da returnere til proben mer enn en gang og kan sees som streker nedover i bildet fordi ultralydmaskinen tror at de senere reflekterte kopiene ligger lenger inn i pasienten.

3.3.3 Nyrestein i Ultralydbilde

Alle typer nyrestein har en høy akustisk impedans og vil derfor reflektere mye ultralyd. Bak steinen vil det også være en akustisk skygge noe som gjør steinen svært gjenkjennelig. Formen på steinen vil typisk være som en halvmåne siden ultralyden reflekteres fra toppen av steinen og lite reflekteres under dette. Fordelen

med ultralyd for avbilding av nyrestein er at pasienten ikke utsettes for skadelig stråling. Ultralyd vil også i motsetning til røntgen, vise alle typer nyrestein.



Figur 3.3: Bildene viser nyrestein hos to forskjellige pasienter tatt med ultralyd B-mode og en buet lineær arrayprobe. Begge steinene er tydelig hvite med akustisk skygge. På bildet til høyre kan man også se kantskygge fra nyren. Bildene er tatt i samarbeid med St. Olavs Hospital og med pasientens samtykke

3.3.4 Utfordringer med ultralydavbildning

Når ultralydbildet skal brukes for å følge nyresteinen vil det være flere egenskaper i bildet som vil utgjøre mulig problemer og utfordringer. Disse utfordringene tas høyde for og overkommes i utformingen av algoritmen som skal utvikles og i valg av for eksempel tracking metode. Yeung et al.[23] identifiserer i sin artikkel fra 1998 en rekke utfordringer ved bruk av ultralydavbildning i tracking sammenheng. Mange av disse egenskapene er fortsatt gjeldende i dag selv om utviklingen innen ultralydavbildningen har gjort fremskritt.

- Vev deformasjon: Vev er mykt og kan endre form når de påføres press. Bevegelse hos pasienten kan flytte nyren og dermed steinen ut av brennpunktet til ESWL maskinen.

- Støy: Ultralydbilder i medisin har en lav signal til støy ratio, det vil si forholdet mellom det vi er ute etter og bakgrunnstøy. Ekko fra små endringer i samme type vev kan sees på som støy og i tillegg inneholder signalet gausisk fordelt elektrisk støy.

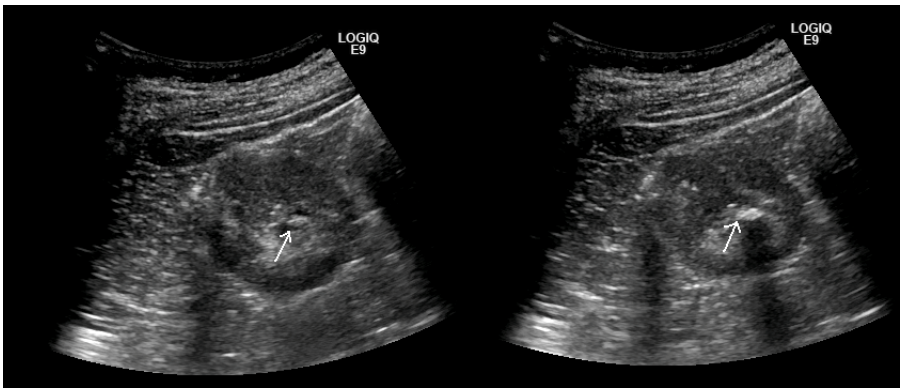
- "Ut-av-planet"bevegelse: Siden 2D ultralyd som vi bruker i denne oppgaven bare er en representasjon i form av et snitt i et tredimensjonalt rom, vil

ikke steinen kunne sees når den beveger seg ut av snittplanet. Steinen kan bevege seg opp eller ned i forhold til snittflaten uten at det er mulig å se hvor den er eller hvilken reting den gikk i, uten å analyse av omgivelsenes kjente anatomiske egenskaper. Dette er en av de største ulempene med å bruke 2D ultralyd og bildebehandling fordi det ikke er mulig å behandle bildet for å få frem informasjonen man er ute etter. En fordel er derimot at når steinen faktisk er i brennpunktet og snittplanet stemmer dette helt overens med virkeligheten. I kapittel 3.1.3 nevnte jeg hvordan røntgenbilder måtte tas fra to vinkler for å vite om steinen var i fokus. Dette trengs ikke ved bruk av 2D ultralyd.

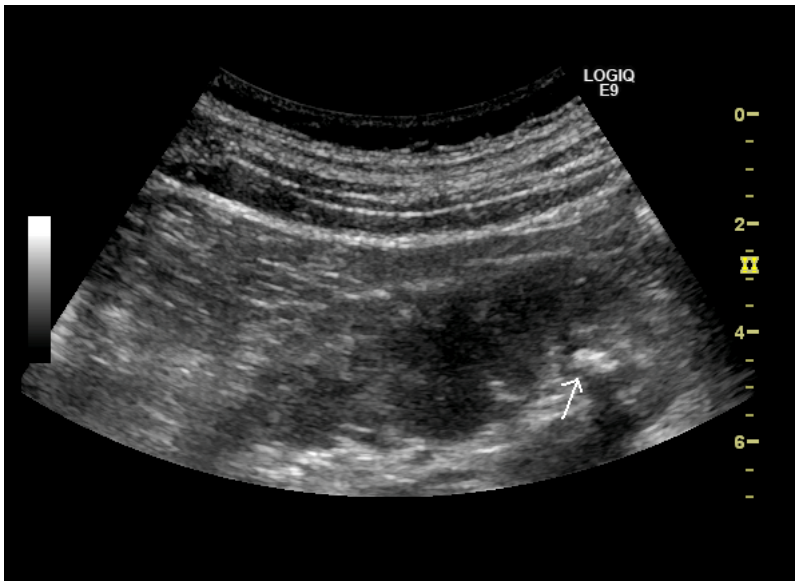
- Artefakter: Artefakter er også omtalt i kapittel 3.3.2. I tillegg til de nevnte egenskapene derfra legger Yeung et al.til artefakter som oppstår pga bevegelse hos pasienten. Disse artefaktene befinner seg i samme kategori som "Tilbakekasting", det vil si at det blir flere refleksjoner fra samme signal som vil avbildes uten at det er en del av anatomien.

- Oppløsning: For lav oppløsning kan gjøre det vanskelig å skille forskjellige medier fra hverandre og overgangene mellom disse mediene kan bli diffuse og vanskelige å tyde. Dette er som sagt avhengig av frekvensen på ultralyden. Høyere frekvens gir bedre oppløsning, men lavere penetrasjon.

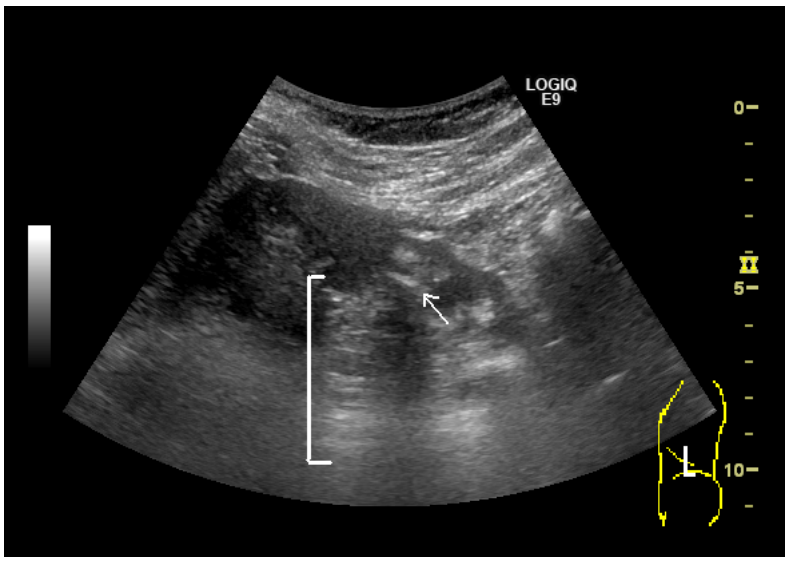
Ut fra egne målinger tatt på nyresteinspasienter i samarbeide med St. Olav har jeg identifisert de punktene som også må tas høyde for videre i denne oppgaven. Jeg fant "Ut-av-planet" bevegelse til å være den desidert største utfordringen med bildet. Videre vil støy og lav oppløsning på bildet utgjøre en utfordring i og med at steinen kan skille seg mer eller mindre fra omgivelsen ettersom disse parameterne er store eller små. Artefakter er også tilstede i bildene, da mest i form av akustisk skygge og kantskygge som vist i figur 3.3. Vevdeformasjon var vanskelig å påvise uten inngående kjennskap til anatomien og fordi den ble maskert av respirasjonene hos pasienten og "Ut-av-planet" bevegelse. Eksempler finnes i figur 3.3, 3.4 og 3.5.



Figur 3.4: Bildet er tatt med et snitt som gir maksimalt utslag for ut av plan bevegelse, altså i bevegelsesretningen for nyrene under respirasjon. Til venstre er steinen på vei nedover eller inn i bildet, mens på høyre side har er steinen i planet med en tydelig skygge. Steinen var bare kort tid i fokus før den forsvant igjen pga respirasjon hos pasienten



Figur 3.5: Her er et eksempel på at oppløsningen er noe lavere enn på de andre bildene. Steinen er synlig, men skiller seg ikke så mye ut. Som alle de andre eksempelbildene inneholder også dette mye støy.



Figur 3.6: Her er et eksempel på artefakten akustisk vindu. Det forsterkede området er markert med hvitt mens steinen i bildet er markert med en pil.

Kapittel 4

Tracking

Denne oppgaven har som hovedmål å undersøke og realisere muligheten for å tracke en nyrestein i ultralydbildet. Til nå har jeg tatt for meg hva nyrestein er og hva egenskapene til steinen er i ultralydbildet. Dette kapittelet vil ta for seg bildebehandlingsteknikker for å tracke objekter i en videofil eller en videostrøm. Egenskapene til disse algoritmene skal senere sammenlignes med egenskapene i ultralydbildet for å kunne ta en korrekt avgjørelse om hvilken teknikk som egner seg best før jeg til sist implementerer antatt beste løsningen.

4.1 Hva er tracking

I sin aller enkleste form går tracking ut på å se på hvordan et objekt forflytter seg fra en frame i en video til en etterfølgende frame. Tracking prosessen består i følge Bradski et al [3] av to hovedmomenter: Identifikasjon av objektet og modellering av bevegelsen. Identifikasjon er å identifisere objektet fra frame til frame. Objektene kan være både definerte og udefinerte på forhånd. Modellering er en viktig komponent fordi målinger gjort i et bilde og fra frame til frame inneholder støy og er upresise. De neste kapitlene vil gå gjennom algoritmene som er presentert av både Bradski et al [3] og Sunka et al[16].

4.2 Optical Flow

Optical flow befinner seg i kategorien identifikasjon av udefinerte objekters bevegelse. Metoden tar for seg avstanden et sett med piksler har beveget seg mellom to frames for på denne måten å danne seg et bilde av om det har vært bevegelse

og om hvilken retning og fart denne bevegelsen har. Vi skiller mellom dense (tett) optical flow og sparse (spredd) optical flow. Dense optical flow tar for seg bevegelsen til hver enkelt piksel i bildet. Horn-Schunck er et eksempel på en algoritme for å utføre dette. En annen tett teknikk er block matching. Her kan bildet deles inn i ruter hvor hver rute inneholder flere pixler. Dette kan få ned beregningstiden noe, men de tette teknikkene krever generelt mye beregningskraft siden hele bildet analyseres. Sparse optical flow algoritmer bruker mindre regnekraft for å tracke bevegelse fordi de først velger seg ut et sett med gode kandidater for tracking. Kandidatene kjennetegnes med at de er unike i forhold til sine omgivelser. Et kryss er et eksempel på en unik kandidat så sant det ikke er omringet av lignende kryss. Beregninger utføres bare på kandidatene for å analysere deres bevegelse. Et eksempel på en spredd teknikk er Lukas-Kanade metoden [3].

4.2.1 Lukas-Kanade

Lukas- Kanade baserer seg på tre antagelser.

1. Konstant lysintensitet: Antar at intensiteten i hver piksel innenfor trackingkandidaten ikke endrer seg fra frame til frame. I realiteten betyr dette at endringen er så liten at den er neglisjerbar.
2. Liten bevegelse: Bevegelsen til objektet er lav slik at objektet ikke forflytter seg mye fra frame til frame
3. Romlig koherens: Nabopunkter i bildet tilhører den samme flaten, har samme bevegelse og projekterer til nærliggende punkter i bildeplanet.

Konstant lysintensitet kan formuleres matematisk som:

$$f(x,t) = I(x(t),t) = I(x(t+dt),t+dt) \tag{4.1}$$

som er det samme som

$$\frac{\partial f(x)}{\partial t} = 0 \tag{4.2}$$

I er intensitetsfunksjonen.

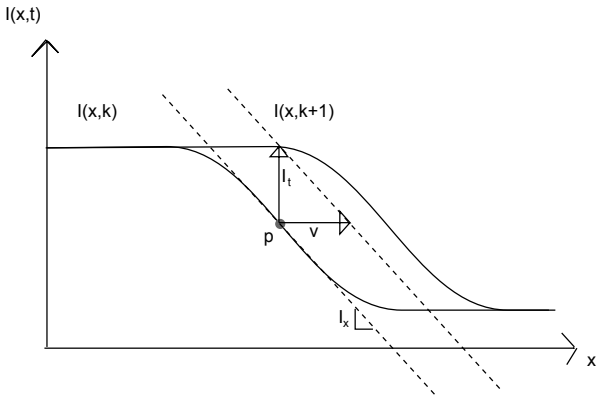
Fra den andre antagelsen kan vi uttrykke bevegelsen ved den deriverte av intensiteten med hensyn på tiden. I en dimensjon kan dette skrives som:

$$\frac{\partial I}{\partial x}|_t \left(\frac{\partial x}{\partial t}\right) + \frac{\partial I}{\partial t}|_{x(t)} = I_x v + I_t = 0 \tag{4.3}$$

hvor $I_x$er den avstandsderiverte i det første bildet og I_t er den deriverte mellom flere bilder over tid. v er her farten til bevegelsen i bildet. Det er denne vi er ute etter å finne. Hvis vi skriver om ligningen kan vi uttrykke farten i en dimensjon som:

$$v = -\frac{I_t}{I_x} \tag{4.4}$$

Figur 4.1 viser hvordan en endimensjonal kant forflytter seg og hvordan de forskjellige variablene korresponderer.



Figur 4.1: Bildet viser en endimensjonal kant og dens forflytning registrert ved to tidspunkt k og k+1. v er fartsvektoren og er bare en approksimasjon. Ved hjelp av flere iterasjoner med Newton's metode vil v til slutt være korrekt. I_t og I_x er tegnet inn.

Ligning 4.4 er basset på antagelser og er derfor bare en tilnærming. Antagelsen som lå til grunne for Lukas-Kanade metoden holder ikke i virkeligheten. Allikevel vil ligningene gi et godt estimat for den korrekte løsningen og vi kan bruke en iterativ teknikk kalt Newton's metode for å komme frem til den korrekte løsningen. Vi beholder beregnet I_x, men oppdaterer I_t for hver iterasjon. v utvides for hver nye beregnede I_t helt til den korrekte verdien er funnet. Newton's metode finner løsningen innen fem iterasjoner.

For å gå fra en til to dimensjoner utvider vi ligningssettet 4.3 med en y koordinat og får et nytt ligningssett:

$$I_x v + I_y u + I_t = 0 \tag{4.5}$$

$$\begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} v \\ u \end{bmatrix} = -I_t \tag{4.6}$$

$$\nabla I^T d = -I_t \tag{4.7}$$

I denne ligningen finnes det to ukjente for hver piksel og vi kan ikke finne en entydig løsning for den todimensjonale bevegelsen i punktet. For å finne en entydig løsning må vi bruke den tredje antagelsen som gjøres for Lukas-Kanade metoden. Den sier at pikslene i et område hører til samme objekt og har samme bevegelse som dette objektet. Tar vi dette i betrakting kan vi sette opp et ligningssett med nxn piksler rundt gjeldende piksel og få n ganger n ligninger for bevegelsen.

$$\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_n) & I_y(p_n) \end{bmatrix} \begin{bmatrix} v \\ u \end{bmatrix} = - \begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_t(p_n) \end{bmatrix} \tag{4.8}$$

$$\mathbf{A}d = -\mathbf{b} \tag{4.9}$$

For å løse dette ligningssettet bruker vi minste kvadraters metode hvor $\min ||\mathbf{A}d - b||^2$ med hensyn på d løses på standard form:

$$(\mathbf{A}^T\mathbf{A})d = \mathbf{A}^T\mathbf{b} \tag{4.10}$$

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} v \\ u \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix} \tag{4.11}$$

$$\begin{bmatrix} v \\ u \end{bmatrix} = (\mathbf{A}^T\mathbf{A})^-1\mathbf{A}^T\mathbf{b} \tag{4.12}$$

Ligning 4.12kan bare løses når $(\mathbf{A^T A})$ er invertibel, altså når den har full rank. Dette inntreffer når den har to store egenverdier, noe som gjelder når de valgte kandidatene for tracking har tekstur som endrer seg i minst to retninger og er unike.

Selv om den tredje antagelsen ble brukt til å finne en entydig løsning på bevegelsesproblemet fører den også med seg en ulempe. Gode tracking kandidater kan ikke være store fordi sannsynligheten for at et sett med piksler tilhører samme flate

minker etter som størrelsen på settet med piksler øker. Dette vil igjen sette en begrensing for hvor store bevegelser som kan detekteres. Problemet kalles blendeåpning problemet (aperture problem). For å løse dette er det utviklet en metode som heter Pyramide Lucas-Kanade. Metoden går ut på å iterere seg gjennom større og større søkeområder mens man bruker estimatet fra foregående søkeområde som startpunkt for hvert nye.

4.2.2 Block matching

Block matching er et fellesbetegnelse på en rekke metoder for å tracke bevegelse i bildet. Metodene er av typen dense eller tett optical flow. Det er fordi de ser på hele bildet uten å velge seg ut et mindre antall punker å følge. Felles for de alle metodene er at de deler inn bildet i mindre, typisk firkantede, biter kalt "blocks". Blokkene kan overlappe hverandre[3]. Vi kan dele inn block matching i to hoveddeler: Sammenligning og søk. Blokkene sammenlignes i et område rundt sine egne koordinater i neste frame helt det finnes et godt treff. Sammenligningen kan utføres ved for eksempel minste kvadraters metode, gjennomsnittlig absoluttverdi, eller krysskorrelasjon [20][9]. Det andre steget i prosessen og den mest vitale for effektiviteten er søk. Med søk menes teknikken som brukes for å lete etter det området man skal utføre sammenligningen i mellom to frames. Turaga et al [20] omtaler 10 metoder for å utføre dette søket hvor et av teknikkene er Spiral Search som også er metoden som er implementert i bildebehandlingsrammeverket OpenCV[3]. Søketeknikken har en innvikning på kjøretiden til metoden, mens sammenligning avgjør trackingegenskapene. Oppløsningen på bevegelsesbildet er avhengig av hvor mye hver blokk overlapper hverandre og størrelsen på blokkene.

Siden metodene ser på en samling piksler og ikke hver enkelt piksel i bildet er de mindre beregningstunge enn andre tette optical flow metoder.

4.3 Kjernebasert tracking

Denne kategorien av tracking brukes for identifikasjon av definerte objekters bevegelse. De er definerte fordi vi på forhånd bestemmer et område eller en kjerne som skal være basis for videre beregninger. I dette kapittelet skal jeg ta for meg Mean-Shift og Camshift som begge er støttet i OpenCV

4.3.1 Mean-Shift

Mean-Shift er en robust metode for å finne massesenteret, eller lokale ytterpunkt, i en intensitetsdistribusjon. Hva denne distribusjonen er basert på defineres ut ifra egenskapene til hva en ønsker å tracke.

For fargebilder kan man bruke distribusjonen av farger i bildet for å beregne bak projeksjonen. Dette er en binær sannsynlighetsdistribusjon som forteller noe om sannsynligheten for at en piksel er en del av objektet når det har en gitt farge. I et ultralydbilde har man allerede en intensitetsdistribusjon siden B-mode ultralydavbildning som nevnt i kapittel 3.3.1, gjengir ultralyd i et sorthvitt bilde hvor høy fasthet på vev eller ben fører til høy intensitet i bildet, og motsatt. Jeg vil derfor ikke gå nærmere inn på behandling av fargebilder

Mean-Shift algoritmen er som følger:

1. Velge et passende søkevindu. Vinduet kan defineres av brukeren over objektet som vil trackes. Dette vil være initiell posisjon for søkevinduet.
2. Beregne søkevinduets massesenter.
3. Flytte søkevinduet slik at det ligger sentrert over massesenteret.
4. Returnere til punkt 2 til vinduet slutter å bevege seg. Mean-Shitft vektoren er lik 0.

Punkt én i algoritmen går ut på å velge et passende område for initiering av søkevinduet. Hvis søkevinduet velges feil i forhold til objektet som skal trackes, vil ikke metoden fungere. Søkevinduet må være på rett sted, altså over objektet. Det må også ha riktig størrelse. Er vinduet for lite kan det feile i å oppdage bevegelse hvis sentrum av objektet har en uniform intensitet. Er vinduet for stort kan det bli forstyrret av endringer rundt objektet som skal trackes. Vi sier allikevel at algoritmen er robust siden det ikke vil bli påvirket av data utenfor søkevinduet. Videre beregnes søkevinduets massesenter, det vil si hvor den største konsentrasjonen av intensitet befinner seg.

$$M_{00} = \sum_x \sum_y I(x, y) \tag{4.13}$$

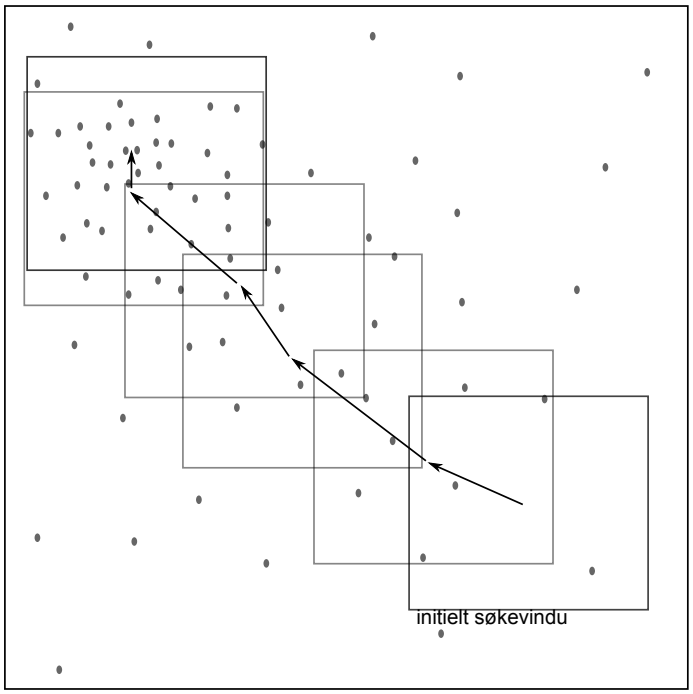
$$M_{10} = \sum_x \sum_y xI(x, y) \tag{4.14}$$

$$M_{01} = \sum_x \sum_y yI(x, y) \tag{4.15}$$

$$x_{cm} = \frac{M_{10}}{M_{00}} y_{cm} = \frac{M_{01}}{M_{00}} \tag{4.16}$$

Søkevinduet flyttes deretter slik at det har sentrum i det nye massesenteret. Deretter utføres en ny beregning av massesenter og en ny forflytning. Hvis vi ser for oss et statisk bilde uten bevegelse, men hvor søkevinduet ikke er plassert direkte over massesenteret til objektet, vil denne iterasjonsprosessen gjenta seg helt til søkevinduet er over det "globale" massesenteret til objektet og dermed stoppe opp. Vi sier at Mean-Shift vektoren er lik 0.

$$d = \frac{\delta x}{\delta y} \tag{4.17}$$



Figur 4.2: Mean-Shift algoritmen utført på et statisk bildet for å finne det globale massesenteret. Fra initielt vindu sentreres søkevinduet over det lokale massesenteret i flere iterasjoner frem til metoden er ferdig. Pilene representerer Mean-Shift vektorene mellom hver forflytning. I den siste ruten er vektoren lik 0.

I en virkelig situasjon vil objektets massesenter også forflytte seg, men algoritmen vil fortsatt prøve å senterer søkevinduet over søkevinduet og dermed tracke objektet.

4.3.2 Camshift

Camshift, eller "continuously adaptive Mean-Shift" er egentlig bare et utvidelse av standard Mean-shift. Forskjellen på de to metodene er at Camshift har et dynamisk søkevindu. Metoden tilpasser seg selv ettersom det trackede objektet endrer størrelse. For at metoden skal fungere er det viktig at objektets identitetsdistribusjon er godt segmentert i forhold til omgivelsene, altså skiller seg ut. Hvis for eksempel et ansikt trackes og beveger seg mot kamerat vil det være behov for å øke søkevinduet i takt med at ansiktet tar opp større plass i bildet. Her vil Camshift algoritmen gjøre nettopp dette. Det er verdt å merke seg at hvis et objekt som trackes forsvinner ut av søkevinduet vil metoden utvide vinduet for å finne en distribusjon som passer med kriteriene for tracking.[3]

4.3.3 Robusthet

Som nevnt er ikke Mean-Shift og camshift avhengige av farger for å tracke et objekt. Metodene tracker en identitetsdistribusjon. Faktisk kan metodene tracke alle typer distribusjoner det måtte være ønskelig å bruke for å beskrive trackingobjektet. På grunn av dette er metodene i følge Bradski et al[3] lite ressurskrevende, robuste og effektive.

4.4 Konturbasert tracking

En siste gruppen algoritmer for tracking av objekter som jeg skal ta med i denne oppgaven og evalueringen av løsning for implementasjon er konturbasert tracking. Metodene ble utviklet for å tracke kurver i et rotete og uoversiktlig område.

Et bilde er en upresis måling pga ting som skygger, okklusjon eller endring av form. Felles kan vi benevne disse usikkerhetene som støy. En måte å eliminere denne støyen på og å maksimere bruken av målinger er ved bruk av estimatorer. For å estimere systemets tilstander brukes en modell av objektets bevegelser.

Prosessen med å estimere tilstandene kan deles inn i to faser: prediksjon og korreksjon. I prediksjonsfasen brukes informasjon fra tidligere tilstander og modellen for å si noe om hva de neste tilstandene vil være. I korreksjonsfasen sammenlignes de predikere tilstandene med målinger og brukes for å opptattere og forbedre modellen. [3]

4.4.1 CONDENSATION

En av ulempene med Kalmanfilter for å modellere objektets bevegelse er at kalmanfilteret bare kan modellere et utfall av gangen. Dette er fordi den underliggende sannsynlighetsdistribusjonen for objektets tilstander er basert på en Gausisk distribusjon, en distribusjon som bare har et toppunkt. Etter en okklusjon kan et objekt bevege seg både i samme retning, forandre bevegelsen til en ny vinkel eller gå tilbake i motsatt retning. Kalmanfilteret kan bare oppdage objektet igjen dersom det beveger seg i samme retning. CONDENSATION eller "Conditional Density Propagation", baserer seg på en type estimatorer kalt partikkelfiltre og er en metode som prøver å håndtere dette. I stedet for å tilpasse en spesifikk ligning til observert data brukes tilfeldig vektet sampling for å modellere og approksimere kurven.

Vi tar utgangspunkt i objektets tilstander ved tiden t, x_t og med historien $X_t = \{x_0 \cdots x_t\}$ samt et sett med målinger z_t og med historien $Z_t = \{z_o \cdots z_t\}$. Algoritmen utfører en rekke vektede sampler iterativt på etterfølgende bilder i en serie. Vi skriver samplesettet fra hver iterasjon som $\left\{s_t^{(n)}, n = 1, \cdots, N\right\}$ med vektene $\pi_t^{(n)}$. Settet er tilnærmet lik den avhengige tilstandstettheten $p(x_t|Z_t)$. For å utlede tilstanden ved tiden t, ser vi først på tilstanden før, altså $p(x_t|Z_{t-1})$. Denne igjen utledes fra det forrige sampelsett $\left\{(s_{t-1}^{(n)}, \pi_{t-1}^{(n)}), n = 1, \cdots, N\right\}$fra $p(x_{t-1}|Z_{t-1})$.

Første steg i algoritmen er å sample N elementer tilfeldig fra settet $\left\{s_{t-1}^{(n)}\right\}$ med sannsynligheten $\pi_{t-1}^{(n)}$. For å beholde N konstant vil noen elementer velges flere ganger (de med stor verdi for $\pi_{t-1}^{(n)}$). Samplene med lav vekt vil ha liten sannsynlighet for å velges og derfor vil de dårligste treffene forsvinne. De valgte elementene er nå en del av settet som skal brukes videre i det prediktive steget. Gitt det valgte settet $\left\{s_{t-1}^{(n)}\right\}$ predikteres $\left\{s_t^{(n)}\right\}$. Dette korresponderer til å sample fra tettheten $p(x_t|x_{t-1} = s_t^{(n)})$. Til slutt vektes tilstandene

$$\pi_t^{(n)} = p(z_t x_t = s_t^{(n)}) \tag{4.18}$$

før $\pi_t^{(n)}$ normaliseres slik at $\sum_n \pi_t^{(n)} = 1$. Settet er nå helt oppdatert og iterasjonen kan gjentas.[11]

Fordelene med CONDENSATION algoritmen er dens evne til å tilpasse seg endringer i form og å takle delvis okklusjon. Algoritmen er også robust i form av at den takler støy på en god måte. Til sist krever algoritmen forholdsvis lite ressurser og har gode sanntidsegenskaper, i alle fall i forhold til en annen konturtrackingalgorime som kalmanfilter. En ulempe er at den ikke kan generaliseres på samme

måte som for eksempel Mean-Shift. Det må lages en modell av objektet og dens bevegelse på forhånd.

Kapittel 5

Design av trackingsystem

5.1 Tidligere arbeid

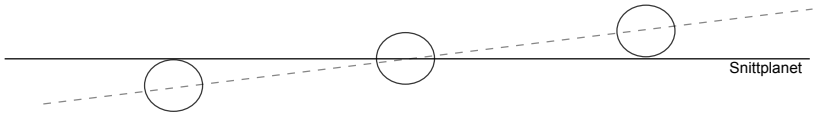
Flere systemer for tracking av nyrestein er tidligere utviklet. Yeung et al utvikler i sin artikkel fra 1998, en metode for å tracke generell bevegelse i et ultralydbilde. Teknikken er bygget på block matching, men er utvidet ved at hver blokk kan endre form og størrelse. Dette er med på å gjengi tettheten i den anatomiske strukturen i form av et nett som fungerer som et høydekart for intensiteten. Metoden er ikke rettet spesifikt mot tracking av nyrestein. Block matching blir utført på hele bildet[23]. Orkiz et al beskriver utviklingen av et system for tracking av nyrestein som de selv kaller Echotrack. Systemet bruker en block matching tilnærming til tracking hvor sammenligningen blir utført ved å beregne korrelasjon. Block matchingen blir dog ikke utført over hele bildet, men i et begrenset søkevindu. Vinduet defineres på forhånd av brukeren, altså den medisinsk kyndige. På denne måten har metoden en del til felles med kjernebaserte trackingalgoritmer. En alarmgrense brukes for å gi beskjed til brukeren om at steinen er ute av planet. "Ut-av-planet" bevegelse er et problemområdet. Men så lenge steinen er i bildet viser de til en reduksjon på 40 % skudd i forhold til konvensjonell behandlingsteknikk. Dette er bevist med test på 65 pasienter [13]. Tsao et al presenterer i likhet med de to foregående en block matching teknikk. Teknikken bruker et nett av blokker over et større område, men har ikke dynamiske blokker som Yeung et al. Motivasjonen til Tsao et al er å forbedre ESWL behandling fordi steinen bever seg ut av brennpunktet på grunn av respirasjon, pasientens bevegelse og trykk fra ESWL bølgene. Metoden analyserer et større område rundt steinen samtidig. Det argumenters for at dette vil takle ut av plan bevegelse fordi området rundt steinen trackes samtidig og vil bevege seg videre i tilnærmet samme retning som steinen selv om den er utenfor planet. Systemet er testet på et videoklipp av nyresteinsbevegelsen hos en pasient[19].

5.2 Utfordringer og kravspesifikasjon

5.2.1 Utfordringer

Nyresteinen skal trackes i et ultralydbilde fordi ultralyd i motsetning til røntgen ikke utsetter pasienten for skadelig stråling og derfor kan brukes til å ha et kontinuerlig innsyn i anatomien. Som nevnt i kapittel 3.3 er det en rekke egenskaper ved ultralydbildet som må tas høyde for i utviklingen av et trackingsystem.

Den mest dominerende egenskapen ved 2D ultralydavbilding er at det bare er et snitt inn i pasienten. Alt over og under dette snittet vil være utenfor synsfeltet i bildet. For en nyrestein vil dette vise seg i "ut-av-planet" bevegelse. Når steinen er ute av planet vil det være vanskelig å si noe om hvor den er. Fordelen er at vi vet med sikkerhet at en stein ute av planet ikke vil befinne seg i ESWL brennpunktet. Det skal altså ikke skytes sjokkbølger. Tsao et al prøver å håndtere oppførselen med et større søkevindu som også tar omgivelsene i betraktning. Testvideo av nyresteinbevegelse som skyldes respirasjon viser at nyresteinen tendenserer til å treffe ultralydsnittplanet i det samme området som den gikk ut av planet. Denne egenskapen kan utnyttes ved å søke i et område rundt steinens forsvinningspunkt.

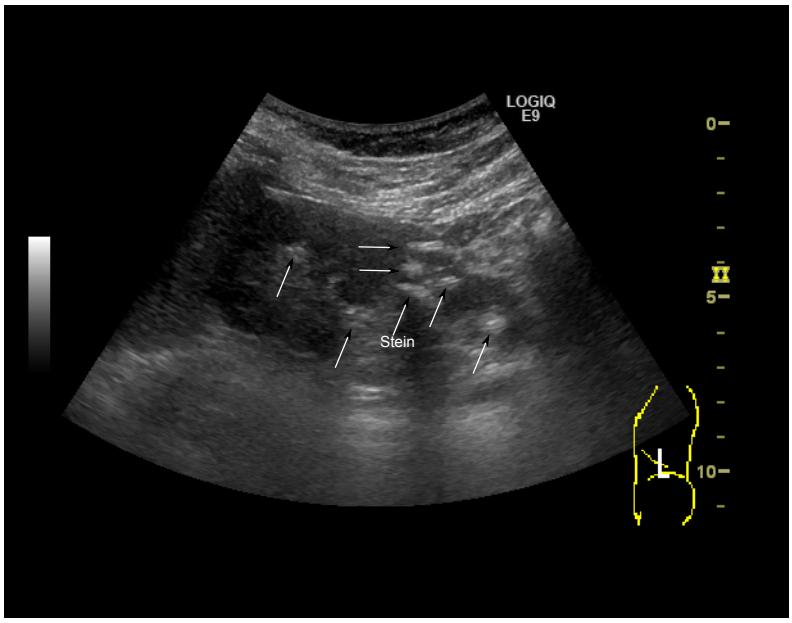


Figur 5.1: En tilnærmet bevegelse av en nyrestein hvor snittplanet ikke ligger parallelt med bevegelsen til nyresteinen. Normal bevegelse er ikke lineær, men figuren beskriver en egenskap som viser seg i flere opptak. Steinen forsvinner og dukker opp igjen i samme område.

Når tracking algoritmer skal brukes på ultralydbilder må en ta hensyn til at flere anatomiske strukturer kan utgjøre en høy intensitet i forhold til omgivelsene sine. Selv om de ikke globalt sett er det lyseste punktet i bildet. Når en stein forsvinner kan det dukke opp andre objekter som trackingalgoritmen reagerer på og begynner å følge. Figur 5.2 viser eksempler på omliggende lyse felter.

Støy og lav oppløsning kan gjøre det vanskelig å skille steinen fra omgivelsene, særlig hvis det er generelt høy intensitet i området rundt steinen. Støyen må håndteres før tracking hvis det ikke brukes en metode som tar høyde for støy i sin funksjon.

Bevegelse som ikke skyldes respirasjon kan i tillegg til å få steinen ut av planet for ultralydsnittet, også føre til at vev i omgivelsen rundt nyre og nyrestein forandrer form. Probens press mot hudflaten kan også gi utslag i forandring av omgivelsene



Figur 5.2: Bildet viser flere objekter med lokalt høy lysintensitet. For et menneske er det mulig å skille ut steinen pga skyggen under, men de lysintensive objektene rundt kan vel så gjerne bli tracket. De to pilene lengst til venstre peker mot sentrum av nyren som ofte vises lysere enn resten av nyrevevet.

rundt nyresteinen. Hvis vevet trackes samtidig med steinen kan det få utslag på den samlede bevegelsesvektoren som beregnes av trackingalgoritmen. Dette må tas høyde for i utvikling av et system.

5.2.2 Krav til systemet

På bakgrunn av de dokumenterte utfordringene ved egenskapene til et ultralydbilde er det utviklet en overordnet kravspesifikasjon til systemet.

Kravspesifikasjon:

Hovedspesifikasjoner:

- Det skal utvikles et system for tracking av en nyrestein i et 2D ultralydbilde.
- Trackingen skal foregå i sanntid fra en videofil.
- Systemet skal ha et brukergrensesnitt for betjening slik at en operatør kan identifisere nyrestein i bildet og initiere en tracking ut fra definert nyrestein.

- Det skal lages en simulering av ESWL for test og rapportering av systemets prestasjon og virkemåte.

Underspesifikasjoner:

- Systemet skal takle støy i bildet.
- Et mål er å være mest mulig robust mot ut av plan bevegelser. "Ut-av-plan" bevegelse skal identifiseres og rapporteres i loggfil. Operatør skal også informeres om ut av plan bevegelse under drift.
- Systemet skal følge steinen uten å la seg påvirke av omgivelsene. Det vil si ikke la vevdeformasjon hindre tracking eller følge andre objekter enn stein etter initiering.

På bakgrunn av disse punktene skal riktig tracking teknikk velges, samt eventuelt forbehandling av bildet utføres.

5.3 Design av algoritme

5.3.1 Tracking teknikk

I dette avsnittet skal jeg ta for meg valg av algoritme for tracking. Algoritmer som skal tas med i betraktning er omtalt i kapittel 4.

Lukas Kanade: Fungerer ved å følge unike hjørner eller kanter i bildet. Mye støy kan føre til mye falske kanter slik at metodene for å velge unike tracking kandidater ender opp med å velge en dårlig kandidat. En annen ulempe som kan vise seg å skape problemer for trackingen er når steinen forsvinner ut av planet, altså opplever full okklusjon, vil dette teoretisk sett være det samme som uendelig forflytning. Noe som bryter sterkt med antagelse nummer to i definisjonen av Lukas-Kanade.

Block-Matching: Teknikken er bevist å fungere godt fra tidligere arbeid. Orkiz et al [13]viser til gode resultater ved å bruke korrelasjon for å sammenligne blokker. Block matching søker på et større område enn bare over selve steinen. Tsao et al[19] argumenterer for at dette kan brukes til å følge steinens bevegelse selv når den beveges ut av planet. Det sies ingenting om virkningen av støy, men det er rimelig å anta at støy vil forholde seg forholdsvis konstant fra frame til frame. Hvis den er uniform vil det ikke virke inn på korrelasjonen. Sanntidsegenskapene til metoden er avhengig av størrelsen på blokkene og hvor stor del av bildet som dekkes.

Mean-Shift: God på støy så lenge den er uniformt fordelt i bildet. Da vil ikke tyngdepunktet bli endret. Metoden er generell i forhold til objektets form så lenge man beregner en sannsynlighetsdistribusjon eller intensitetsdistribusjon. Dette passer meget godt for ultralyd fordi dette som nevnt tidligere allerede er en intensitetsdistribusjon. Hvis steinen kommer tilbake på samme punktet den forsvant, vil den igjen være tyngdepunktet i området. Dette forutsetter at ikke metoden har forvekslet andre lysintensitive objekter i området rundt med steinen. Altså beregnet tyngdepunktet til å ligge et annet sted og forflyttet seg dit.

Camshift: Er lik som Mean-Shift, men utvider søkeområdet. Dette vil skje hvis steinen er ute av planet slik at metoden vil finne andre objekter.

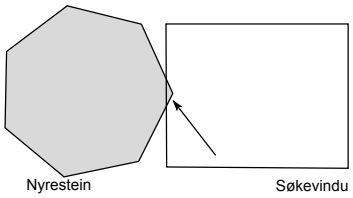
CONDENSATION: Laget spesielt for å takle støy og rotete omgivelser. Metoden har også gode sanntidsegenskaper[11]. Ikke så generell hva gjelder form på objektet og det må utvikles en modell for bevegelsen. Dette kan løses ved kantdeteksjonsalgoritmer på nyresteinen, men beregningstiden vil øke. Isard et al har utført en rekke tester og viser til gode resultater på bruk av algoritmen.

Den eneste algoritmen jeg ser som et dårlig alternativ for bruk under trackig av nyrestein er Lukas-Kanade metoden. Både Block-Matching og CONDENSATION kan vise til gode resultater fra tester og tidligere arbeide. Alikevell skiller Mean-Shift seg ut ved at et ultralydbildet passer direkte som input til metoden fordi det allerede er en binær intensitetsdistribusjon. Mean-Shift er derfor valgt som teknikk for tracking av nyrestein. En ulempe med metoden er muligheten for å finne andre lokale tyngdepunkt når steinen ikke er til stede. Det må derfor utføres forbehandling av bildet.

5.3.2 Forbehandling

For at systemet i minst mulig grad skal la seg påvirke av omliggende strukturer, velger jeg å forbehandle bildet ved å terskelsegmentere (threshold) vekk alt som har lavere lysintensitet enn steinen. På denne måten vil lokale intensive objekter i området med lavere intensitet enn steinen, ikke synes for Mean-Shift algoritmen. Når steinen er borte vil det ikke finnes noe tyngdepunkt i området og søkevinduet vil holde seg stasjonært over forsvinningspunktet for så å gjenoppdage steinen når den kommer tilbake. Området steinen kan komme tilbake innenfor er lik søkevindu størrelsen pluss størrelsen på stein (fig 5.3).

Før terskelsegmenteringen utføres, filtreres bildet for å fjerne små lokale intensitetstopper. Dette gjøres ved hjelp av et midlingsfilter med størrelsen nxn. Filteret vil finne snittet av intensiteten i et området og lagre i hver enkelt piksel. Slik vil små intensitetstopper flates ut, mens store vil bli værende. Størrelsen på området som



Figur 5.3: Med en gang steinen kommer innenfor søkevinduet vil tyngdepunktet beregnes til ytterste kant der steinen er. Deretter vil algoritmen fokusere seg over steinen igjen. Dette skjer så sant det ikke er andre objekter i søkevinduet som forandrer tyngdepunktet.

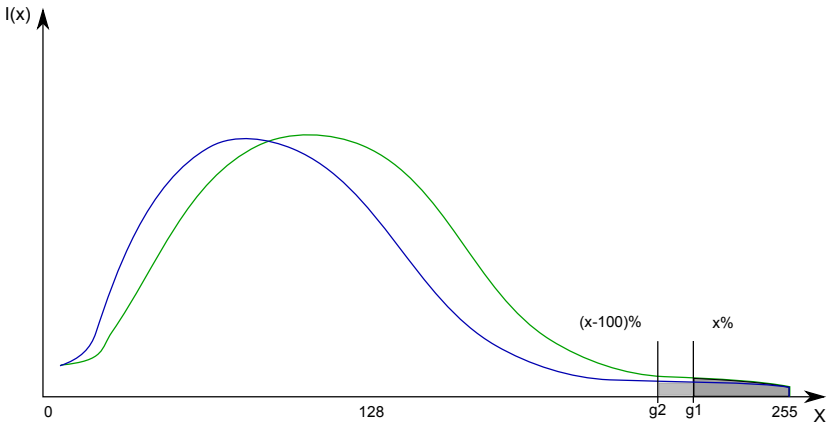
flates ut avgjøres av n. En positiv bieffekt av filtreringen er at støy blir mindre signifikant.[6]

$$R = \frac{1}{n} \sum_{i=1}^{n} z_i \tag{5.1}$$

En ren segmentering med en konstant terskelverdi vil ikke la seg gjennomføre fordi intensiteten vil variere fra pasient til pasient og til og med under en økt. Dette kan løses ved å tillate operatøren å endre grenseverdien i brukergrensesnittet slik at den kan stilles inn til riktig nivå. Dette løser ikke problemet med at steinen kan endre tydelighet og lysintensitet under en økt. Det ble utviklet en egen metode for dynamisk terskelsegmentering.

Metoden tar utgangspunkt i et histogram som beregnes ut fra et område rundt steinen. Dette histogrammet vil vise hvordan intensiteten er fordelt i en graf med intensitet langs x-aksen og mengde av denne graden av intensitet langs y-aksen. De laveste verdiene representerer det sorte, altså der hvor ultralydbølgene ikke har blitt reflektert. De høyeste verdiene vil inneholde for eksempel nyresteinens profil. Et bilde med lite refleksjon vil føre til at størst andel av areal under intensitetskurven vil befinne seg i nedre region. I motsatt tilfelle vil et lyst og grått bildet skylles mye refleksjon og ha en kurve som er forskjøvet mot de høyere intensitetsverdiene. Videre bygger algoritmen på en antagelse som ble gjort ut fra observasjoner av histogrammene fra ett ultralydopptak og senere validert på flere opptak: Nyresteinens bidrag til fordelingen i histogrammet, vil alltid befinne seg innenfor en gitt prosentverdi av arealet under grafen. En stein i et mørkt bilde og en stein i et lyst bilde vil finnes innenfor de samme øvre prosentene av grafarealet. Grenseverdien for lysintensitet vil derimot forandre seg. Dette er illustrert i figur 5.4.

Valg av verdi for x ble gjort eksperimentelt mot opptak av ultralyd og ble valgt til 5 %. Verdien var for lav i enkelte tilfeller og det kan derfor argumenters for å la



Figur 5.4: Dynamisk terskelsegmentering. X beskriver prosent av lysintensiteten. Grenseverdien vil forflytte seg til større eller mindre verdi ettersom det markerte området er større eller mindre enn x. g1 er grenseverdien som blir beregnet når arealet under grafen summeres til x for den grønne grafen. g2 er tilsvarende for den blå. Forskjellen på de to fordelingene er at blå graf vil representere et mørkere bilde enn grønn graf.

operatøren velge mellom flere verdier i området rundt valgt verdi. Testprosedyren for valg av x er dokumentert i kapittel 7.

5.3.3 Algoritme\Programflyt

I dette avsnittet følger en rask gjennomgang av programflyten i hoveddelen av programmet, det vil si det absolutte minimum av funksjonalitet for at trackingen skal fungere. Hoveddelen består av filtrering, grenseverdisegmentering og Mean-Shift. Programflyten er gjengitt i algoritme 5.1

Algorithm 5.1 Programflyt

```
if video tilgjengelig then
    grab frame
frame ⇐ beregne histogram
for i = 0; i < hist_size; i++ do
    summer areal under histogram
beregne 4% av arealet
for i = hist_size; i > 0; i-- do
    summer areal
    if sum areal>4prs then
        sett grenseverdi
frame ⇐ average filter
frame ⇐ grenseverdisegmenter(greneverdi)
if søkevindu definert then
    {Dette er Mean-shift}
    søkevindu ⇐ beregne tyngdepunkt
    søkevindu ⇐ oppdater posisjon
```

Kapittel 6

Implementasjon

6.1 Valg av teknologi

Her følger en gjennomgang av teknologi som er valgt for implementasjon. Fra før er ultralyd valgt som videokilde og algoritmen for tracking er valgt og definert.

6.1.1 OpenCV

OpenCV er et åpen-kildekode bildebehandlingsbibliotek, men ble originalt utviklet av Intel. Biblioteket er skrevet i C og C++ og kjører på både Linux, Windows og Mac OSX. I tillegg finnes det OpenCV grensesnitt mot blant annet Python, Ruby og Matlab. OpenCV er designet for å være beregningsorientert effektivt med fokus på sanntidssystemer og kan utnytte flerkjerne prosessorer. [3]

OpenCV biblioteket kan struktureres inn i fem kategorier:

- CV: Grunnleggende bildebehandling og høyere nivå datasyn algoritmer
- MLL: Maskinlæringsbibliotek.
- HighGUI: I\O rutiner for å hente og lagre video og bilder. Inneholder også kode for enkel prototyping av brukergrensesnitt.
- CXCore: Inneholder alle de grunnleggende datastrukturene og algoritmene samt XML støtte og tegnefunksjoner.
- CvAux: Inneholder både gamle og eksperimentelle algoritmer. For eksempel: Stereosyn, øyne og munn tracking og bakgrunn segmentering.

6.1.2 Qt

Qt er et multiplatform, grafisk, programvareutviklingsverktøy for å kompilere og kjøre programmer for bårde Windows, Linux, Mac OSX og forskjellige Unix versjoner. Qt er utviklet av norske Trolltech Software som i 2008 ble kjøpt opp av Nokia[5]. Tanken bak Qt er å utvikle et plattformnøytralt grensesnitt for det meste av funksjoner fra minnehåndtering til grafikk presentasjon. Rammeverket bruker i utgangspunktet C++, men Trolltech har laget støtte for Java og JavaScript i tillegg er Phyton, Ruby, PHP og .NET støtte utviklet av tredjeparts leverandører.

I tillegg til standard C++ funksjonalitet er Qt utvidet med en ekstra funksjonalitet: *signaler* og *slots*. Dette er en teknikk for å fleksibelt koble objekter sammen. *Signal* er metode som sender signaler når den kalles. Dette implementeres i koden ved å declarere en prototype av signalet i SIGNAL seksjonen av classen. *Slot* er en en funksjon som kan påkalles av et sendt signal. *Slots* kan være både *public*, *protected* og *private*. Denne beskyttelsen gjelder bare når *slot* brukes som en metode slik at en private og en protected slot allikevel kan motta signaler fra andre metoder.[18]

6.2 Programkode

Først ble en trackingmetodikken implementert i C og OpenCV. En prototyp ble laget med OpenCv sine innebygde funksjoner for presentasjon av video og glidebrytere for regulering av parametere. Deretter ble koden konvertert til C++ for å passe inn i Qt rammeverket. Koden som gjengis her vil være den ferdige koden med brukergrensesnitt fra Qt.

Klassen i programmet heter MainWindow og det er her tracking, videovisning og kommunikasjon med brukergrensesnittet blir utført. Det første som skjer i klassens konstruktør er at alle variable blir initiert. Dette innebefatter variable for simulator, definisjon av arbeidsområdet, timere og oppretting av loggfil, i tillegg til åpning av videofil og identifikasjon av video parametere. Sistnevnte utføres ved at brukeren blir presentert med en dialog med mulighet for å bla seg frem til filen i Windows mappestruktur. Til sist opprettes kommunikasjon med brukergrensesnittet. Her er det utført en litt omstendelig prosess for å konvertere OpenCV sitt videoformat til et som kan vises i Qt sitt objekt, QGrapicsView. Mer om dette i kapittel6.3. Videre inneholder klassen slots for kommunikasjon med brukergrensesnittet og verktøy metoder som er støtte for tracking metoden.

OpenCVMeanShift() inneholder hele programløkken. Her utføres henting av frame fra videofil og forbehandling av frame før Mean-Shift metoden kjøres. I tillegg er simulatoren delvis en del av denne metoden og en logging til fil med tidsmål

utføres etter operasjoner i forbindelse med tracking. Metoden er koblet opp mot en timer slik at det kan bestemmes hvor ofte den skal kjøres. Tilsvarende en while-løkke med en pause, men uten at systemet henger. I den nåværende koden er tiden mellom hver programløkke, og altså hver gang en ny frame skal hentes, satt til 25ms. Det tilsvarer en frekvens på 40Hz, noe som ligger over framerate på ultralydopptakene utført ved St. Olavs Hospital og derfor vil gi en sikker tracking. Dette tallet kan settes høyere for å begrense belastningen på systemet hvis det skulle være nødvendig, men bør holde seg innenfor framerate fra video for å takle store bevegelser i bildet.

Listing 6.1: Program timer

```
1 //Oppretter et timer objekt og setter timer intervallet til 25 ↩
     ms
2 timer = new QTimer();
3 timer->setInterval(25);
4 //Kobler timeren med metoden for tracking slik at en ny frame ↩
     hentes inn og behandles hvert 25 ms     connect(timer,SIGNAL(↩
     timeout()),this,SLOT(OpenCVMeanShift()));
5 timer->start();
```

I henhold til algoritme 5.1 er momentene i programmet implementert på følgende måte. Første steg er å hente en frame fra videofilen. Koden rundt gjør at videoen vil loope rundt når den kommer til siste frame.

Listing 6.2: Hente frame fra videofil

```
1 if(cvGetCaptureProperty(capture,CV_CAP_PROP_POS_FRAMES)==frames↩
     -1)
2 {
3     cvSetCaptureProperty(capture,CV_CAP_PROP_POS_FRAMES,1);
4 }
5
6 frame = cvQueryFrame(capture);
7
8 int frameNumber = (int)cvGetCaptureProperty(capture,↩
     CV_CAP_PROP_POS_FRAMES);
```

Deretter beregnes grenseverdien for dynamisk grenseverdisegmentering. Her blir først histogrammet beregnet ut fra et arbeidsområde klippet ut av frame. Dette gjøres for å utføre bildebehandling bare på det nødvendige området og dermed begrense beregningsbelastningen. OpenCV spesifikke metoder for skallering av histogram følger, før histogrammet tegnes opp, en søyle av gangen. Etter dette beregnes totalt areal og arealet av det øvre sjiktet hvor man finner steinens profil og til slutt beregners grenseverdi for dette øvre sjiktet.

Listing 6.3: Dynamisk grenseverdisegmentering

```
1  cvCalcHist( &hist_en_kanal, hist, 0, NULL );
2  cvGetMinMaxHistValue( hist, 0, &max_value, 0, 0 );
3
4  cvScale( hist->bins, hist->bins, ((double)hist_image->height)/↩
       max_value, 0 );
5  cvSet( hist_image, cvScalarAll(255+1), 0 );
6  for(int i = 0; i < hist_size; i++ )
7  {
8      //Tegner opp en søyle for hver grad av intensitet i ↩
           histogrammet
9      cvRectangle(hist_image,cvPoint(i,hist_image->height),cvPoint↩
           (i+2,hist_image->height-cvGetReal1D(hist->bins,i)),↩
           cvScalarAll(0),1,8,0);
10
11     //Sumerer bins, dvs totalt areal under histogram grafen
12     if(i > 0 ){
13         bin_total += cvGetReal1D(hist->bins,i);
14     }
15 }
16
17 bin_vol_limit = bin_total - bin_prosent;
18
19 for(int i = hist_size-1; i>0; i--){
20     bin_top_sum +=cvGetReal1D(hist->bins,i);
21     if(bin_top_sum >= bin_vol_limit){
22         if(dyn_tresh == 1){
23             thresholdValue = i;
24         }
25     break;
26     }
27 }
```

Arbeidsområdet kopieres til et midlertidig minne fordi det skal kopieres tilbake over det gamle ubehandlede bildet. Filtrering og grenseverdisegmentering utføres på dette midlertidige bildet. Det er verdt å legge merke til at det benyttes en binær segmentering. Dette betyr at alt over grensen blir satt til hvitt, mens alt under blir sort. På denne måten vil tyngdepunktet som beregnes av Mean-Shift bli meget dominerende i forhold til omgivelsene.

Listing 6.4: Filtrering og segmentering

```
1 cvSmooth(midl,midl,CV_BLUR,filterSize,filterSize);
2
3 cvThreshold(midl,midl,thresholdValue,256,CV_THRESH_BINARY);
```

Til sist følger beregning av tyngdepunkt og oppdatering av søkevindu før det op-

pdaterte søkevinduet tegnes rundt steinen i videoavspillingsvinduet.

Listing 6.5: Mean-Shift

```
1 cvMeanShift( image, track_window,cvTermCriteria( CV_TERMCRIT_EPS↩
      | CV_TERMCRIT_ITER, 10, 1 ),&track_comp);
2
3 track_window = track_comp.rect;
4             cvRectangle(frame,cvPoint(track_window.x,↩
                 track_window.y),cvPoint(track_window.x+↩
                 track_window.width,track_window.y+track_window.↩
                 height),CV_RGB(150,0,100),2,CV_AA,0);
```

Resten av koden med ESWL simulator og loggkode finnes i filvedlegg medlagt denne oppgaven..

6.3 Brukergrensesnitt

6.3.1 Bakgrunn

For å lage et brukergrensesnitt i Qt kan en bruke den grafiske editoren i Qt Creator. Qt Creator er Trolltech sitt eget utviklingsmiljø for Qt kode. Etter at et Qt prosjekt er opprettet kan en generere og editere *.ui filer. Dette er filer som inneholder det grafiske brukergrensesnittet. I dette grensesnittet er det en samling ferdige objekter som knapper, glidebrytere, grupperingsbokser og grafikkvinduer som kan legges til brukergrensesnittet. Kommunikasjon mellom objektene i *.ui filene og programkoden foregår med signaler og slots. Et eksempel er koblingen mellom et knappetrykk i brukergrensesnittet og metoden, altså slot, som utføres når signalet fra knappen kommer frem.

Listing 6.6: Knapptrykk

```
1 void MainWindow::StartButtonClicked()
2 {
3     pause = 1;
4 }
5 connect(this->ui->startButton, SIGNAL(clicked()), this, SLOT(↩
    StartButtonClicked()));
```

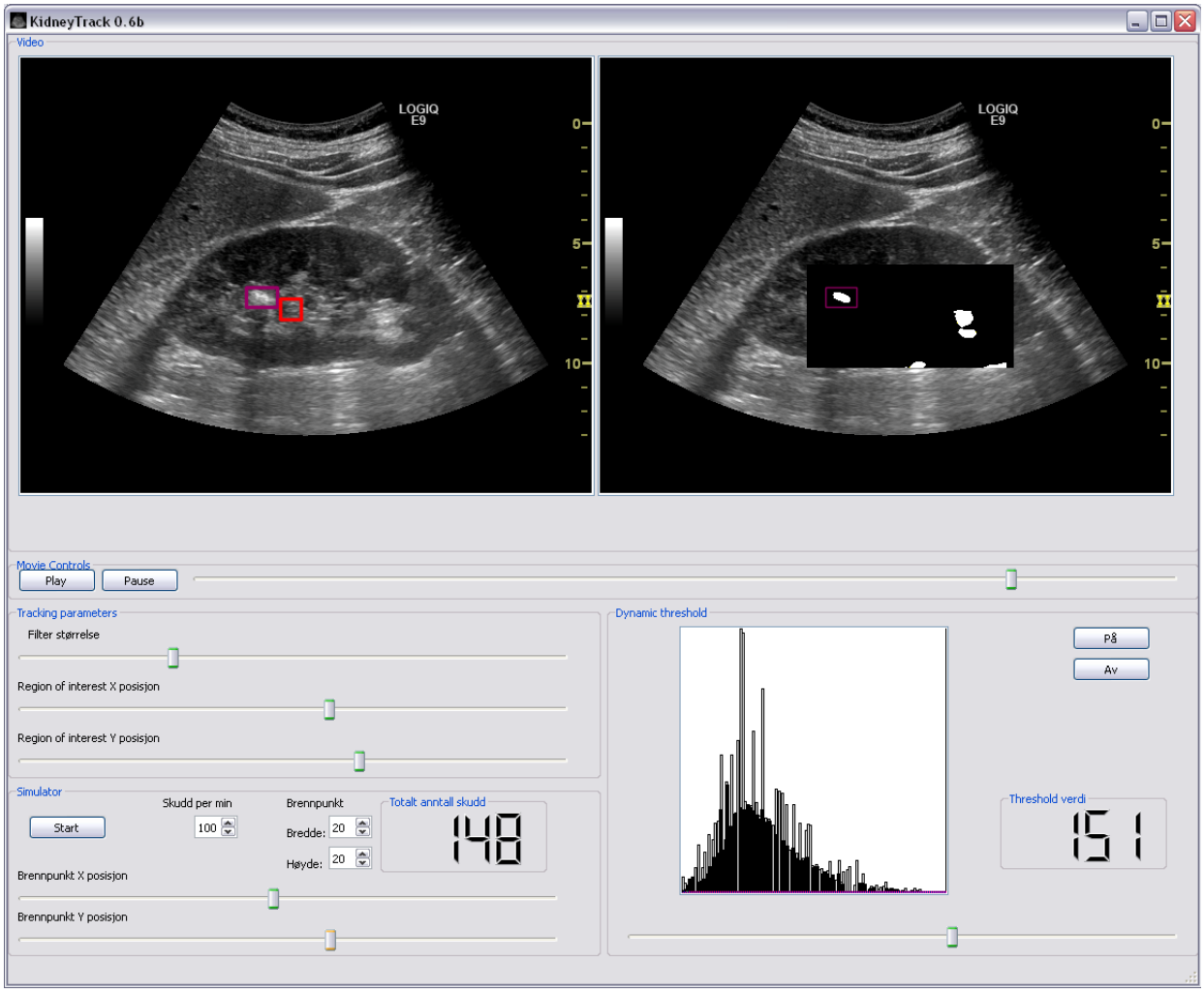
For å vise OpenCV i Qt konverteres bildene til et format som passer til Qt objektet QGrapicsView. Dette er et objekt som er til for å vise grafikk av forskjellige typer. OpenCV sitt bildeformat heter IpImage. Dette kopieres først inn i en char* av

riktig størrelse. Deretter kan bildet bygges opp igjen i riktig Qt format og til sist settes inn i QGrapicsView. Koden er fullstendig dokumentert i appendiks.

6.3.2 Dokumentasjon

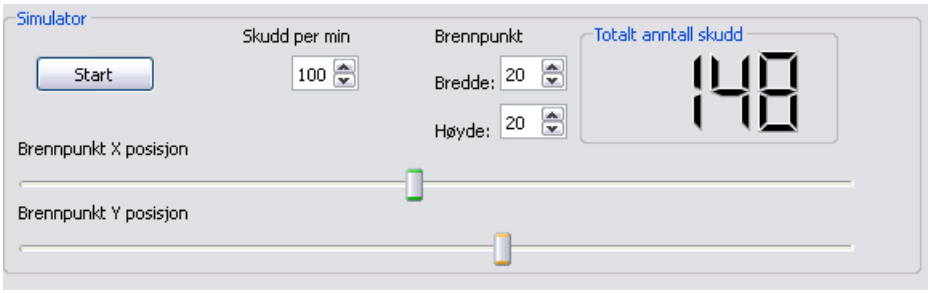
Figur 6.1 viser et bilde av brukergrensesnittet slik det ser ut for operatøren. Øverst i bildet vises to versjoner av videoopptaket som brukes som input til programmet. Til venstre er en ubehandlet utgave hvor brennpunktet er tegnet inn i rødt og søkevinduet er representert ved et lilla rektangel. Brennpunktet er tegnet rødt fordi steinen ikke befinner seg i fokus. Når steinen er i brennpunktet og et skudd vil medføre treff, vil brennpunktet representeres av et grønt rektangel. Bildet på høyre side er hva Mean-Shift algoritmen "ser", altså datasynet. Her er filtrering og grenseverdisegmentering utført og steinen er omringet av et lilla rektangel tilsvarende det i bildet på venstre side. Når steinen er identifisert av operatøren er det i dette vinduet steinen blir markert og søkevinduet initiert. Dette utføres ved at operatøren drar ut et rektangel over steinen med musepekeren på skjermen. Figur 6.2 viser et nærbilde av området hvor generelle parametere kan reguleres. Filter størrelse representerer n for nxn masken i midlingsfilteret, omtalt i kapittel 5.3.2. Region of interest er arbeidsområdet for systemet, altså området hvor algoritme 5.1 utføres innenfor. Ved hjelp av glidebryterne i bildet kan posisjonen til dette området plasseres mer hensiktsmessig i forhold til nyren og nyresteinens bevegelsesområde. Simulatoren styres fra panelet i figur 6.3. Her er det mulighet for operatøren å gjøre en rekke endringer av parametere. Skudd per minutt begrenser hvor ofte simulatoren kan registrere et nytt skudd i likhet med EWSL maskinens regulering av skuddfrekvens. Det er verdt å merke seg at her vil begrensingen være maksimalt antall skudd per min. Hvis steinen er ute av fokus på en periode på flere sekunder vil et skudd bli avfyrt med en gang den kommer i fokus. Hvis steinen fortsatt er i fokus vil det, i tilfelle av 100 skudd per min, gå 0,6 sekunder før neste skudd avfyres. Brennpunktets størrelse og posisjon kan forandres for å gjøre det lettere å teste på forskjellige videofiler. Under en behandling vil pasienten posisjoneres for å få brennpunktet i nyresteinens bane. Det er ikke mulig å flytte pasienten etter at opptaket er utført, men den tilsvarende effekten oppnås ved å forflytte brennpunktet. Totalt antall skudd kan leses av fra displayet. Dette er et mål på hvor mange skudd som faktisk ville ha truffet steinen siden hvert skudd blir avfyrt når steinen befinner seg i brennpunktet. Det siste området av brukergrensesnittet viser status fra den dynamiske grenseverdisegmentering. Se figur 6.4. Her vises et histogram som presenterer den momentane intensitetsfordelingen i den frame systemet behandler. Den beregnede grenseverdien presenteres både som tallverdi og ved posisjonen til glidebryteren nederst i bildet. Operatøren kan velge å skru av den dynamiske grenseverdisegmenteringen for så å bestemme grenseverdi

manuelt ved å regulere glidebryteren.

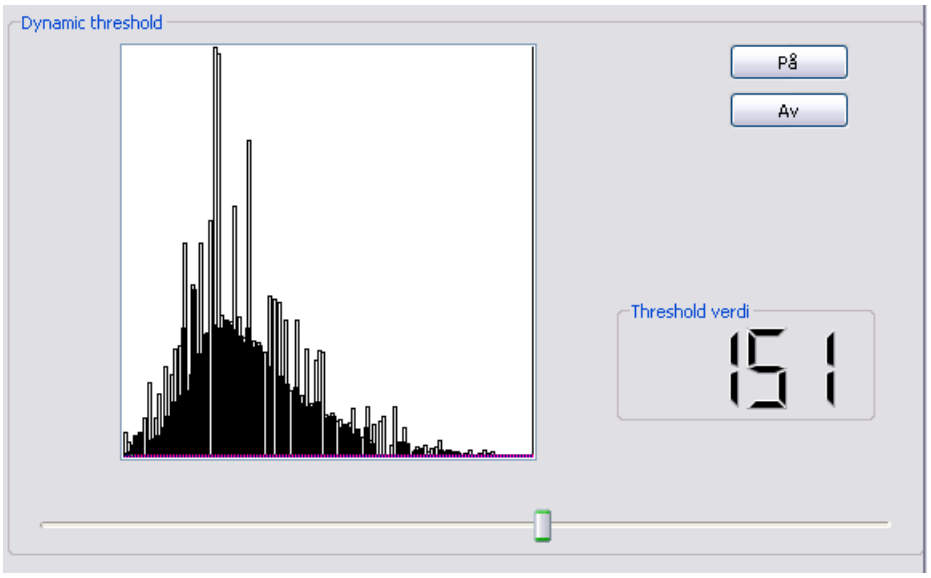


Figur 6.1: Brukergrensesnittet i sin helhet slik det møter operatøren. Øverst til venstre vises en ubehandlet utgave av videoen, men med markert søkevindu (lilla) og brennpunkt simulering (rød). Brennpunktet er rødt fordi steinen ikke er i fokus.

Figur 6.2: Mulighet for regulering av generelle parametere som filterstørrelse og posisjon for arbeidsområdet.



Figur 6.3: Operatørgrensesnittet for kontroll av simulatoren. Operatøren får også tilbakemelding fra simulatoren i form at totalt antall skudd.



Figur 6.4: Histogrammet viser momentan intensitetsfordeling i bildet. Grenseverdien vises både som tall og i form av glidebryterens posisjon. Det er mulig å skru av og på modulen øverst til høyre.

Kapittel 7

Test og Resultater

7.1 Testplan

I test av systemets ytelse er det fire momenter som skal utføres. Det første går ut på å validere kvaliteten av den valgte parameteren for hvor mange prosent av den øvre intensiteten i histogrammet nyresteinen befinner seg innenfor. Som punkt nummer to vil testvideoen beskrives og data om videoene kartlegges for å få et grunnlag for vurdere tracking algoritmens oppførsel. Sanntidsegenskapene vil testes som et tredje moment. Og til sist må systemets overordnede prestasjon undersøkes for å kunne sammenlignes med et system uten tracking av stein.

For å beregne beste verdi for dynamisk grenseverdisegmentering testes flere verdier. En verdi velges og testes. Deretter vil en større og en mindre verdi utprøves hvorpå beste resultat vil føre til et nytt valg i samme retningen. På denne måten vil verdien konvergere mot beste resultat. Et mål for kvalitet vil være tiden steinen er i planet og kan utføres ved å bruke tracking algoritmen til å følge steinen til den forsvinner.

For å kunne sammenligne test data med reelle parametere, kartlegges data fra testvideo. Følgende punkter skal undersøkes:

- Plott av nyresteinens bevegelser.
- Data om hvor lenge steinen er i planet
- Maks tid steinen er ute av planet
- Hvor mange ganger steinen går inn og ut av planet

Et godt mål for sanntidsegenskapene til systemet er tiden det tar for OpenCVMean-Shift metoden å gjennomføre en iterasjon. Denne tiden må være mindre enn den

hyppigste skuddfrekvensen som har blitt oppgitt fra St. Olavs hospital til å være omtrent 120 skudd per minutt, altså 0,5 sekunder. I tillegg må det undersøkes om tiden mellom oppdatert søkevinduposisjon og skudd er så stor at steinen kan ha forflyttet seg ut av fokus.

Systemets overordnede effektivitet kan måles ved å sammenligne antall skudd med og uten tracking. Nøyaktigheten til tracket steins koordinater plottes sammen med reelle koordinater fro å sammenligne. Spørsmål som må dokumenteres vil være: Forsvant steinen? Ble den funnet igjen? Hadde søkevinduets form noe å si for dette? Ble søkevinduet "kapret" av andre, mer lysintensive objekter?

7.2 Utførelse

Alle testene ble utført på NTNU sin pc, KYBPC581 som kjører Windows XP som operativsystem. Før tesptrogrammet ble startet ble det registrert 71 simultane prosesser, omtrent 1928mb av 3930 mb RAM var i bruk og prosessoren vekslet mellom 13 og 25 prosent bruk. Systemspesifikasjoner er gjengitt i tabell 7.1

Tabell 7.1: Testoppsett

Platform	Pentium(R) D 2.80GHz
Opertivsystem	MS Windows Xp
Hovedminne	2Gb RAM
Grafikkort	Intel(R) 82945G Express (innebygget)

7.2.1 Dynamisk grenseverdisegmentering

For å teste og finne beste grenseverdi for segmentering ble det utført test med tracking på tre forskjellige videoer etter at et estimat var gjort visuelt med en fjerde video. Fire prosentverdier ble testet og total trackingtid og tid ute av plan, det vil si når steinen ble segmentert vekk, ble registrert. Deretter ble tiden ute av plan regnet om til prosent av total og denne verdien midlet over resultatet fra de tre testvideoene. I tillegg ble det undersøkt visuelt om andre lysintensive objekter ble tracket når steinen var ute av planet slik at søkevinduet ikke ville sentrere seg over steinen så fort den entret planet.

Dataene ble hentet ut fra en loggfil som automatisk genereres under kjøring. Det er kode i programmet (på vedlagt DVD) som beregner hvor lenge steinen er ute av planet. Disse verdiene ble summert opp til total tid.

7.2.2 Identifikasjon av testvideoparametere

For å identifisere parametere ble filmene spilt av noen frames av gangen mens steinens posisjon ble undersøkt visuelt. Posisjonen ble logget ved å bruke søkevinduinitieringsegenskapene i programmet til å markere steinen så ofte som mulig for å få en nøyaktig måling av steinens midtpunkt. Disse koordinatene ble sammen med framenummer lagret til en separat loggfil. Koordinatene ble importert til Matlab hvor de ble plottet. Nøyaktig frame for "ut-av-planet" bevegelse ble undersøkt visuelt. Prosessen ble gjentatt for hver enkelt videofil.

7.2.3 Sanntidsegenskaper

Sanntidskravene til systemet er at beregningstiden for hver programløkke, med belastning i form av tracking og simulering, må være liten nok til at hver frame rekker å behandles. Altså lavere enn frameraten. Dette kravet er ikke nødvendig for at programflyten skal gå som normalt. Det nødvendige sanntidskravet til systemet er at beregningstiden må være kort nok i gjennomsnitt til at steinen ikke har beveget seg vekk fra den beregnede posisjonen før en ESWL puls avfyres. Ved liten steinbevegelse kan dette kravet fortsatt overholdes selv om ikke hver frame hentes ut fordi forrandringen mellom frames vil være så liten. Testen ble utført ved å logge tiden programmet brukte på en løkkekjøring for noen hundre samples. Makstid og snitt ble beregnet.

7.2.4 Sluttresultat

Med sluttresultatet menes systemets overordnede kvalitet. Hvor godt trackingen fungerer både i forhold til å følge steinens bevegelser og resultatet av å bruke tracking kontra dagens metode. Programkoden inneholder kode for å skrive logg til fil. Det genereres en loggfil som inneholder informasjon om programflyten og en fil som inneholder informasjon om koordinatene til trackingvinduet. Informasjonen om koordinater ble importert i Matlab hvor de ble plottet sammen med informasjonen om steinens faktiske posisjon. Videre informasjon ble lest fra loggfilen om programflyten og presentert i tabellform.

7.3 Resultater

7.3.1 Dynamisk grenseverdisegmentering

Data fra de 12 testkjøringene er gjengitt i tabell 7.2.

Tabell 7.2: Grenseverdi

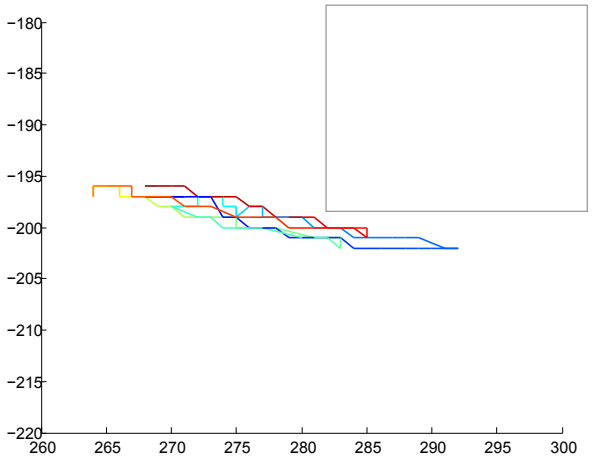
		2 Prosent	4 Prosent	5 Prosent	10 Prosent
Image012.avi	Total	22,191	20,181	21,681	29,728
	Ute av plan	6,233	1,815	2,432	1,969
	Prosent	28	9,29	11,2	6,60
	Obstruhert	Nei	Nei	Nei	Nei
Image02.avi	Total	23,237	23,336	30,282	25,632
	Ute av plan	20,634	16,922	22,30	7,984
	Prosent	88,9	72,8	73,7	31,1
	Obstruhert	Nei	Nei	Nei	Nei
Image13.avi	Total	18,616	50,279	22,424	14,077
	Ute av plan	9,682	12,993	5,901	4,564
	Prosent	52	25,85	26,31	32,42
	Obstruhert	Nei	Nei	Nei	Ja
Alle	Snitt prosent	56,30	35,98	37,07	23,37

7.3.2 Identifikasjon av testvideoparametere

Syv testvideoer ble tatt opp fra tre forskjellig pasienter. En eldre mann, en mellomaldrene kvinne og en mellomaldrene mann. Disse videoene utgjorde grunnlaget for test av systemet. Plott av steinens bevegelse ble generert i Matlab. Koordinatsystemene i OpenCV og Matlab har inverse y akser, derav de negative fortegnene på y-aksen. Bildene er også skalert for å lette kunne se bevegelsen. Tyve piksler tilsvarer en distanse på omtrent en cm.

7.3.2.1 Video1

Video1 har filnavnet "Image01.avi". Steinen i videoen har meget liten bevegelse, faktisk bare marginalt større enn brennpunktet på 20x20 piksler. Mellom ytterpunktene i grafens x-retning er bevegelsen på 29 piksler, mens den er 15 piksler i y-retning. Steinen går ikke ut av planet noen ganger i løpet av den 269 frame lange videoen. Steinen har lav intensitet i forhold til omliggende anatomi. I figur 7.1 er bevegelsen gjengitt. Steinens startpunkt representeres ved mørkeblå farge og ligger i koordinaten (271,-197) og endepunktet er burgunder i koordinaten (266,-195).



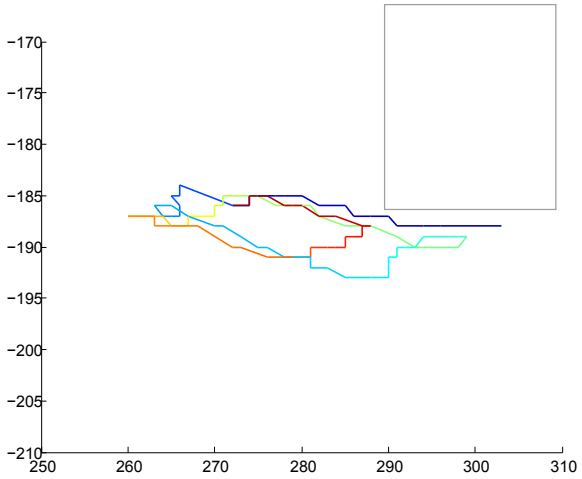
Figur 7.1: Steinens bevegelse i Video1. Totalt er opptaket på 269 frames og det loopes ikke. Figuren er fargekodet med mørk blå i startpunkt. Rektangelet øverst til høyre representerer 20x20 piksler, noe som tilsvarer omtrent 1 cm, altså det samme som brennpunktet.

Tabell 7.3: Faktatabell Video1

Filnavn	Lengde	Ute av plan	Tid ute av plan
“Image01.avi	269 frames	0 ganger	0 frames

7.3.2.2 Video2

Video2 har filnavnet "Image02.avi". Steinen beveger seg innenfor et område på 44 piksler i x-retning altså litt over dobbel brennpunktstørrelse. I y-retning er bevegelsen på bare 9 piksler. Steinen er i planet i alle videoens 269 frames. Filmen kjennetegnes av flere lysintensive objekter i området rundt steinen. Startpunktet til steinen er i koordinaten (303,188) og endepunktet i koordinaten (271,188).



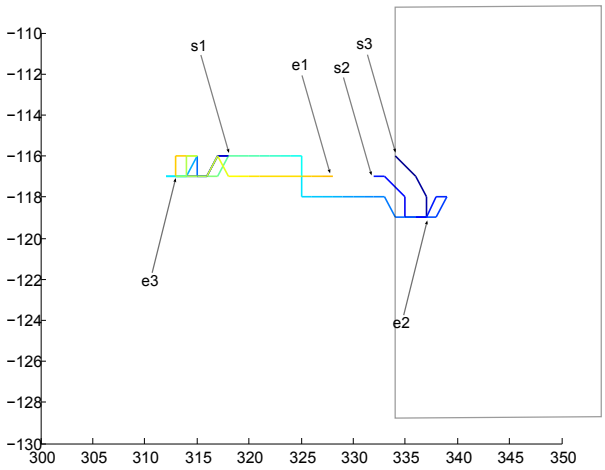
Figur 7.2: Steinens bevegelse i Video2. Totalt er opptaket på 269 frames og det loopes ikke. Figuren er fargekodet med mørk blå i startpunkt. Rektangelet øverst til høyre representerer 20x20 piksler, noe som tilsvarer omtrent 1 cm, altså det samme som brennpunktet.

Tabell 7.4: Faktatabell Video2

Filnavn	Lengde	Ute av plan	Tid ute av plan
"Image02.avi	269 frames	0 ganger	0 frames

7.3.2.3 Video3

Video3 har filnavnet "Image03.avi". Også i denne filmen er det forholdsvis liten bevegelse av steinen. 29 piksler i x-retning og bare 3 piksler i y-retning. Steinen beveger seg inn og ut av planet i to intervaller. Dette foregår i høyre del av bevegelsesområdet. Steinen er ikke lett å identifisere i videoen på grunn av vinkelen på nyren, men den har allikevel en høyere lysintensitet enn omgivelsene som vil gjøre det mulig å tydeliggjøre den ved hjelp av grenseverdisegmentering. Steinens posisjon første gang den er i plan er (314,-117), andre gang (336,-119) og tredje gang er koordinatene (334,-116). Videoen er 269 frames lang.



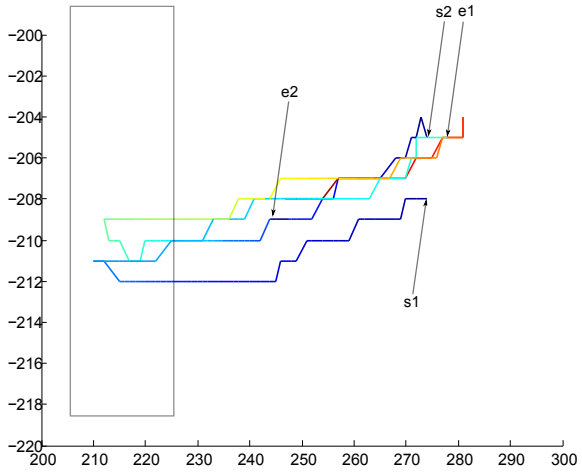
Figur 7.3: Steinens bevegelse i Video3. Totalt er opptaket på 269 frames og det loopes ikke. Figuren er fargekodet med mørk blå i startpunkt. Rektangelet til høyre representerer 20x20 piksler, noe som tilsvarer omtrent 1 cm, altså det samme som brennpunktet. s1-s3 og e1-e3 representerer start og endepunkt for intervallene hvor steinen er i planet.

Tabell 7.5: Faktatabell Video3

Filnavn	Lengde	Ute av plan	Tid ute av plan	Ute av plan intervall
Image03.avi	269	2 ganger	28 frames	108-125, 142-153

7.3.2.4 Video4

Video4 har filnavnet "Image012.avi". I denne videoen er bevegelse noe større enn i de foregående, altså vil tracking av steinen være mer nødvendig fordi steinen vil befinne seg sjeldnere i brennpunktet. Mellom ytterpunktene i x-retning beveges steinen 70 piksler. I y-retning er variasjonen mindre, her beveger steinen seg innenfor 8 piksler. Steinen er ute av planet én gang i løpet av filmens 280 frames. Dette er i det ytre området for bevegelse i x-retning på høyre side i figur 7.4. Startpunktene for steinens to perioder i ultralydplanet er (274,-208) og (274,-205).



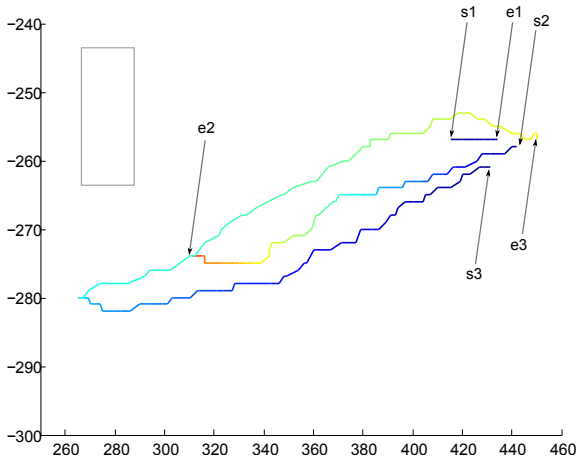
Figur 7.4: Steinens bevegelse i Video4. Totalt er opptaket på 280 frames og det loopes ikke. Figuren er fargekodet med mørk blå i startpunkt. Rektangelet til venstre representerer 20x20 piksler, noe som tilsvarer omtrent 1 cm, altså det samme som brennpunktet. s1 og s2 og e1 og e2 representerer start og endepunkt for intervallene hvor steinen er i planet.

Tabell 7.6: Faktatabell Video4

Filnavn	Lengde	Ute av plan	Tid ute av plan	Ute av plan intervall
Image012.avi	280	1 gang	13 frames	110-123

7.3.2.5 Video5

Video5 har filnavn "Image13.avi". Videoen skiller seg fra de foregående med forholdsvis stor bevegelse. Dette er fordi det er et forstørret utsnitt av steinens bane. Rektangelet i figur 7.5 er fortsatt på 20 piksler, men her vil dette bare være halvparten av brennpunktstørrelsen. Steinen er i planet i tre intervaller. Det første er kort, mens de to andre er lengre. Startpunktene for de tre forløpene er (415,-257), (431, -261) og (442, -258).



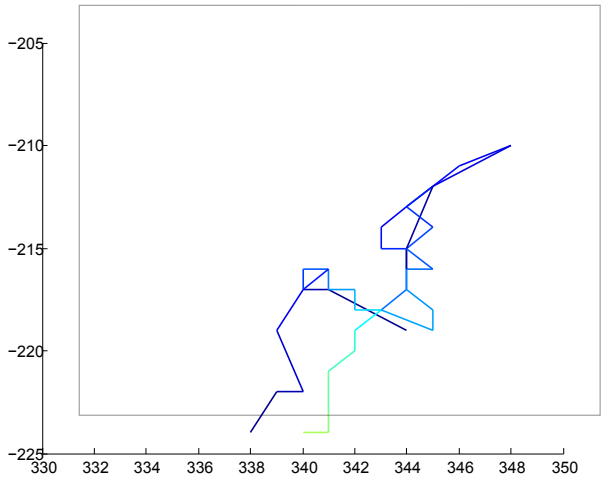
Figur 7.5: Steinens bevegelse i Video5. Totalt er opptaket på 305 frames og det loopes ikke. Figuren er fargekodet med mørk blå i startpunkt. Rektangelet til venstre representerer 20x20 piksler, noe som tilsvarer omtrent 0.5 cm, altså halvparten av brennpunktet. s1-s3 og e1-e3 representerer start og endepunkt for intervallene hvor steinen er i planet.

Tabell 7.7: Faktatabell Video5

Filnavn	Lengde	Ute av plan	Tid ute av plan	Ute av plan intervall
Image13.avi	305 frames	3 ganger	56 frames	10-33, 196-209, 275-305

7.3.2.6 Video6

Video6 har filnavn "Image14.avi". I denne videoen er snittet plassert slik at det skal gi dårligst mulig resultat hva gjelder "ut-av-planet" bevegelser. Snittet er snudd 90 grader i forhold til de foregående videoene slik at nyrens bredde i stedet for høyde blir avbildet. I denne posisjonen vil steinen bevege seg opp og ned i pasientens lengderetning og dermed inn og ut av ultralydsnittet. Dette fører til mange korte intervaller i planet. Det er verdt å merke seg at steinen likevel befinner seg innenfor det samme området hver gang den kommer tilbake i planet. Faktisk er ikke avstanden mellom ytterpunkter i noen retninger lengre enn brennpunktets størrelse på 20x20 piksler.



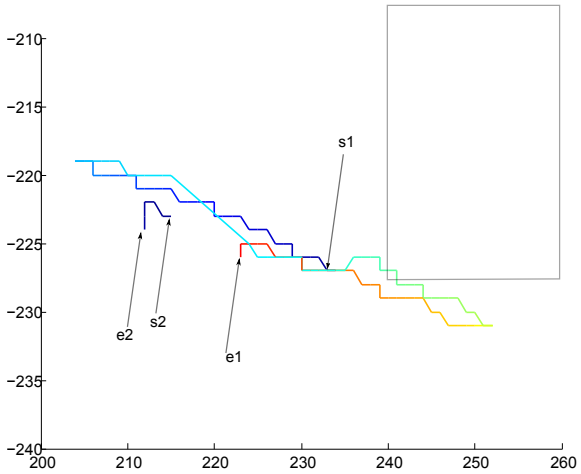
Figur 7.6: Steinens bevegelse i Video6. Totalt er opptaket på 305 frames og det loopes ikke. Figuren er fargekodet med mørk blå i startpunkt. Rektangelet i bildet representerer 20x20 piksler, noe som tilsvarer omtrent 1 cm, altså det samme som brennpunktet.

Tabell 7.8: Faktatabell Video6

Filnavn	Lengde	Ute av plan	Tid ute av plan	Ute av plan intervall
Image14.avi	305 frames	5 ganger	251 frames	1-173, 192-204, 210-246, 260-265, 280-305

7.3.2.7 Video7

Video7 har filnavnet "Image015.avi". Steinen i videoen er i planet i to intervaller. Først et langt sammenhengende før den på slutten forsvinner og dukker opp i noen frames før filmen er over. Bevegelsen skjer innenfor 48 piksler i x- og 12 piksler i y-retningen. Steinen er klar og tydelig gjennom hele opptaket og skiller seg bra fra omgivelsene. Startpunktene ligger i (234, -227) og (215, -223).



Figur 7.7: Steinens bevegelse i Video7. Totalt er opptaket på 260 frames og det loopes ikke. Figuren er fargekodet med mørk blå i startpunkt. Rektangelet til høyre i bildet representerer 20x20 piksler, noe som tilsvarer omtrent 1 cm, altså det samme som brennpunktet. s1 og s2 og e1 og e2 representerer start og endepunkt for intervallene hvor steinen er i planet.

Tabell 7.9: Faktatabell Video7

Filnavn	Lengde	Ute av plan	Tid ute av plan	Ute av plan intervall
Image015.avi	260 frames	2	17 frames	228-235,241-251

7.3.3 Sanntidsegenskaper

Tabell 7.10 viser en oversikt over beregninger utført ved analyse av kjøretiden. I tillegg til snitt og makstid er standardavvik og varians beregnet for å få et bilde av fordelingen til kjøretiden på de forskjellige samplene. Dette kan brukes for å argumentere for om kjøretiden er forsvarlig på en gjennomsnittlig basis.

Tabell 7.10: Sanntidsegenskaper

	Løkketid med maks belastning	Løkketid uten belastning
Antall samples	107	292
Snitt tid	37ms	35ms
Makstid	67ms	70ms
Standardavvik	12ms	11ms
Varians	1.4844e-004ms	1.3284e-004ms

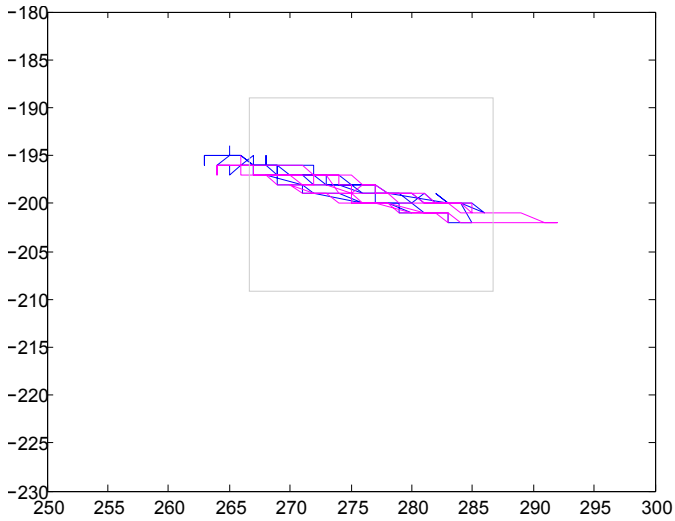
7.3.4 Sluttresultat

Syv tester ble utført på syv forskjellige videoer, alle omtalt tidligere i kapittel 7.3.2. Figurene i de påfølgende delkapitlene viser sammenhengen mellom tracket stein og faktisk posisjon under en avspilling av videoene. Faktisk "ut-av-planet" bevegelse refererer også til antall hendelser under én avspilling, mens trackingens registrerte bevegelser er hentet fra test hvor avspillingen ble loopet.

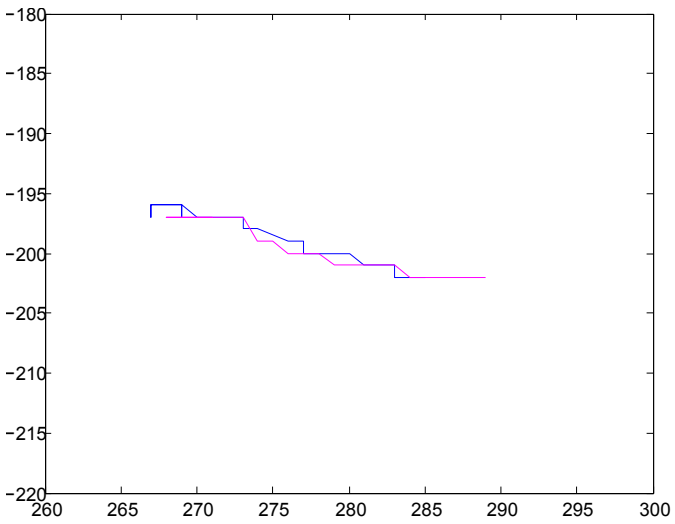
7.3.4.1 Video1

Som nevn i kapittel 7.3.2.1 er steinen i planet gjennom hele filmen. Dette er ikke det samme som algoritmen registrer. Programmet registerte "ut-av-planet" bevegelse i totalt 7 tilfeller i løpet av de 22,28 sekundene trackingen foregår. Den samlede tiden ute av planet er 0,7010 sekunder. Dette har imidlertid ingen innvirkning på resten av forløpet. Steinen ble fulgt videre fordi bevegelsen i løpet av disse intervallene var så liten av den ikke gikk utenfor søkevinduets området. På denne måten ble steinen funnet igjen med en gang. Algoritmen registrerte "ut-av-planet" bevegelse fordi programmet har for strenge krav til segmentering, grenseverdien blir altså beregnet som så høy at også steinen blir segmentert vekk. Figur 7.8 viser at forløpet til steinen, beregnet av trackingalgoritmen, avviker lite fra faktisk forløp. ESWL simulatoren registerte 35 skudd i trackingperioden og beregnet at 37 puls-skudd ville blitt sendt ved konstant puls-frekvens.

Figur 7.9 viser et utsnitt av testforløpet og faktisk tilstand ved samme frame i videofilen. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla. Steinens trackede posisjon er tilnærmet tilsvarende som den korrekte posisjonen.



Figur 7.8: Plott av tracking samt faktisk bevegelse til nyrestein i video 1. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla. Rektangelet skal representere brennpunktet og er 20x20 piksler stort. Det er liten variasjon mellom faktisk og tracket verdi. Plottet er et utsnitt av én gjennomkjøring av filmen. Steinen er i planet hele opptaket.

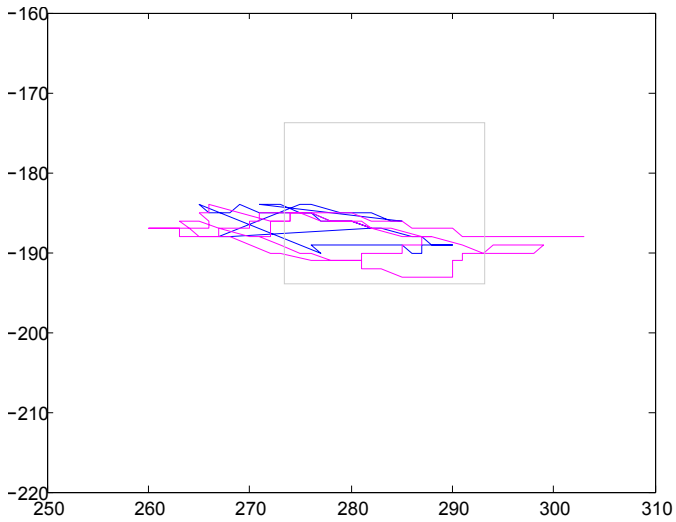


Figur 7.9: Et begrenset utsnitt av reel bevegelse og testresultat for video1. Begrensningen er gjort for bedre å sammenligne de to forløpene. For begge tilfeller er posisjonene frame nummer 1 til 50 plottet. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla.

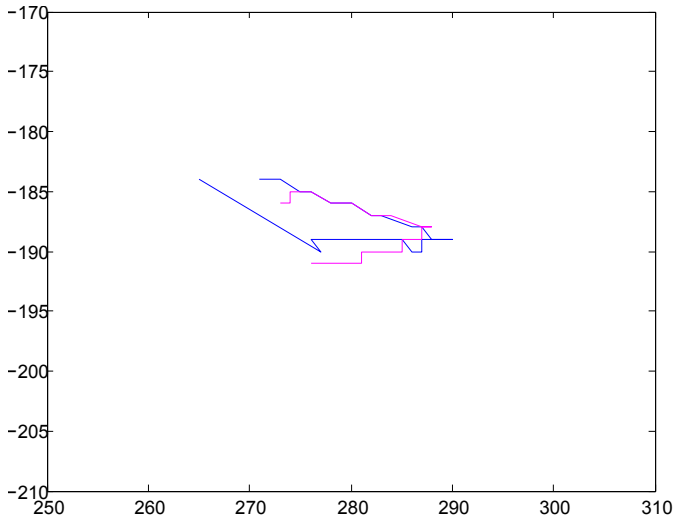
7.3.4.2 Video2

Også i video2 er steinen i planet under hele opptaket. Tracking algoritmen registrerer derimot steinen ute av planet 9 ganger på grunn av streng segmentering. Testkjøringen varer i 21.31 sekunder og steinen er totalt 13.47 sekunder ute av planet.Algoritmen segmenterte vekk steinen da den befant seg i høyre halvdel av reelt bevegelsesområde. Det er mange lyse objekter i området rundt steinen, dette gjør at steinen ikke like ofte er en del av de 5% lyseste intensitetene og derfor blir segmentert vekk. For å få et godt resultat måtte "region of interest" plasseres slik at ikke de lyseste områdene ble med i beregningen av histogrammet. Dette er ikke ødeleggende for noe annet en effektiviteten. Plassering av brennpunktet i venstre bevegelsesområdet ville ha eliminert dette problemet. Et faremoment med objekter med høy lysintensitet rundt steinen, er at Mean-Shift algoritmen kan missforstå disse som nyresteiner. Dette inntraff ikke under test av tracking i video2. Ved tap av stein under tracking greide algoritmen å finne tilbake til steinen i alle tilfeller. 13 sjokkbølgeskudd ble registrert av simulatoren. På tilsvarende periode ville 35 skudd blitt avfyrt med dagens behandlingsmetode hvorav 22 ville bommet på steinen ved tilsvarende plassering av brennpunkt.

Figur 7.11 viser et utsnitt av testforløpet og faktisk tilstand ved samme frame i videofilen. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla. Trackingen følger korrekt bane godt i øvre del av grafen. Etter at bevegelsesretningen snur og steinen beveger seg tilbake blir avviket stort, men holder seg fortsatt innenfor brennpunktets størrelse på 1cm eller 20x20 pixler.



Figur 7.10: Plott av tracking samt faktisk bevegelse til nyrestein i video 2. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla. Rektangelet skal representere brennpunktet og er 20x20 piksler stort. Det er liten variasjon mellom faktisk og tracket verdi. Plottet er et utsnitt av én gjennomkjøring av filmen. Steinen er i planet hele opptaket.

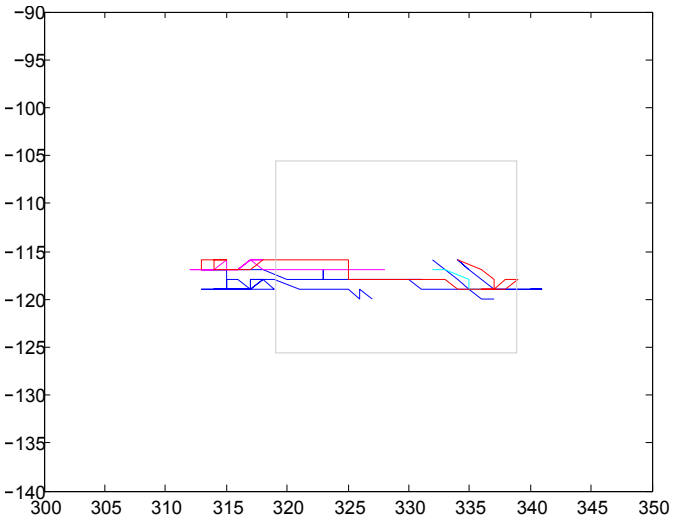


Figur 7.11: Et begrenset utsnitt av reel bevegelse og testresultat for video2. Begrensningen er gjort for bedre å sammenligne de to forløpene. For begge tilfeller er posisjonene frame nummer 197 til 254 plottet. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla.

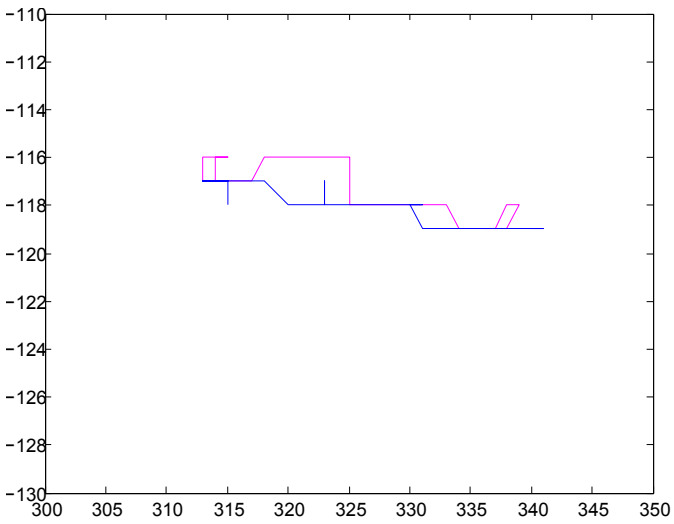
7.3.4.3 Video3

Testkjøringen av video3 viste som nevn i kapittel 7.3.2.3 at steinen er ute av ultralydplanet to ganger. Programmet registerte 8 tilfeller, med en total tid på 3,65 sekunder ute av planet. Testkjøringen varte i 23,44 sekunder. De trackede posisjonene har et avvik på 3-5 piksler i forhold til faktisk verdi, noe som er godt innenfor brennpunktet. I og med at steinens bevegelse totalt sett ikke overgår brennpunktstørrelsen i særlig stor grad, har heller ikke tap av objekt noen innvirkning på resultatet i dette tilfellet. Søkevinduet finner tilbake til steinen så fort den dukker opp igjen i ultralydplanet. Simulatoren registrerer 20 pulser i løpet av testkjøringen og beregner at 39 pulser ville blitt avfyrt uten tracking.

Figur 7.13 viser et utsnitt av testforløpet og faktisk tilstand ved samme frame i videofilen. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla.



Figur 7.12: Plott av tracking samt faktisk bevegelse til nyrestein i video 3. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla, rødt og cyan for hver av de 3 intervallene hvor steinen går faktisk inn i planet. Rektangelet skal representere brennpunktet og er 20x20 piksler stort. Det er liten variasjon mellom faktisk og tracket verdi. Plottet er et utsnitt av én gjennomkjøring av filmen.

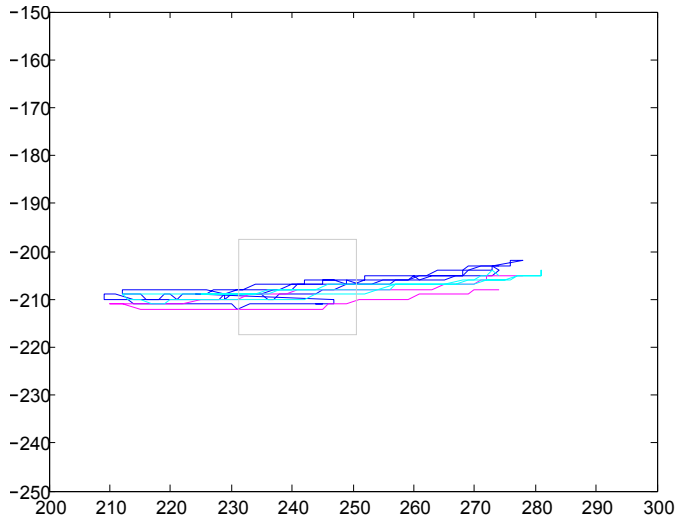


Figur 7.13: Et begrenset utsnitt av reel bevegelse og testresultat for video3. Begrensningen er gjort for bedre å sammenligne de to forløpene. For begge tilfeller er posisjonene frame nummer 158 til 168 plottet. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla.

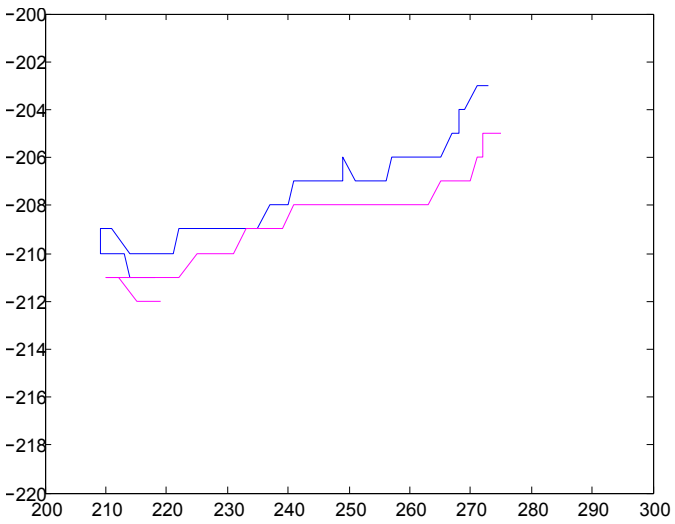
7.3.4.4 Video4

Video4 består av en tydelig stein som skiller seg meget godt ut fra omgivelsene. I kapittel 7.3.2.4 er antall faktiske "ut-av-planet" bevegelse identifisert til å inntreffe en gang. Trackingalgoritmen registrerer 8 tilfeller. Til sammen er steinen ute av søkevinduet i 3, 45 av de 29,68 sekundene testkjøringen varer. Som vist i figur 7.14 beveger steinen seg på utsiden av brennpunktet med over 3 ganger brennpunktstørrelsen. I slike tilfeller blir trackingen viktig for å kunne avfyre pulser bare når steinen er i fokus. Det blir registrert 17 ESWL pulser med tracking og 49 uten. Altså ville 65 % av skuddene bommet på steinen ved normal behandling.

Figur 7.15 viser et utsnitt av testforløpet og faktisk tilstand ved samme frame i videofilen. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla. Bevegelsen i de til tilfelle kan sees på som tilnærmet lik. Særlig området lengst til høyre i figuren er et eksempel på dette.



Figur 7.14: Plott av tracking samt faktisk bevegelse til nyrestein i video 4. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla og cyan for hver av de 2 intervallene hvor steinen går faktisk inn i planet. Rektangelet skal representere brennpunktet og er 20x20 piksler stort. Det er liten variasjon mellom faktisk og tracket verdi. Plottet er et utsnitt av én gjennomkjøring av filmen.

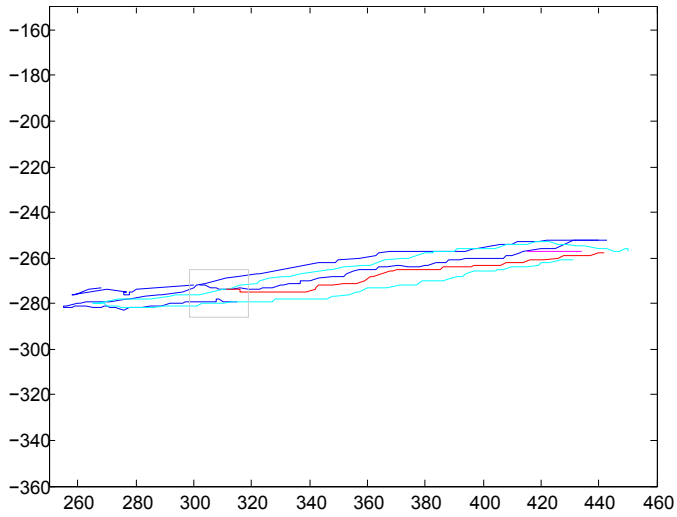


Figur 7.15: Et begrenset utsnitt av reel bevegelse og testresultat for video4. Begrensningen er gjort for bedre å sammenligne de to forløpene. For begge tilfeller er posisjonene frame nummer 50 til 108 plottet. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla.

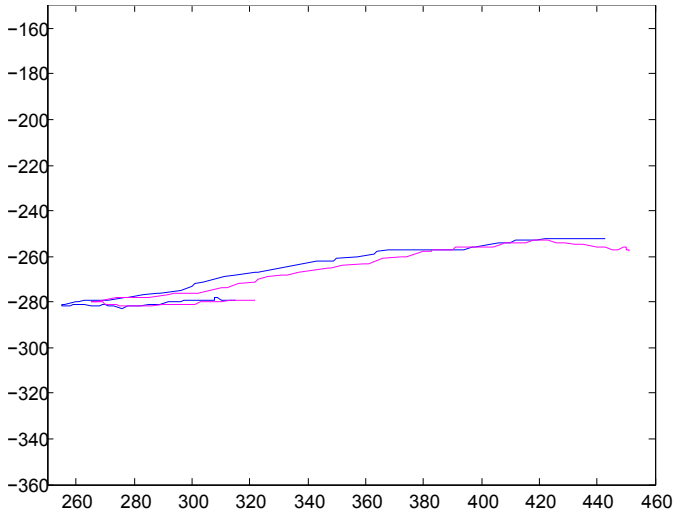
7.3.4.5 Video5

Video5 er et forstørret utsnitt av steinens bane. Det er større bevegelser enn på tidligere testvideoer og farten på bevegelsene er også tørre enn tidligere tester. Trackingen greier å følge steinen hele banen. Ved et tilfelle, når filmen startes på nytt og søkevinduet ikke flyttes og re-initieres, greier algoritmen å finne tilbake til steinen etter over 2,5 sekunder og 200 pikslers bevegelse i x-retning. Også i alle andre "ut-av-planet" bevegelser gjenoppdages steinen og trackes videre. I opptaket er steinen ute av planet 3 ganger. Algoritmen registrerer 11 bevegelser ut av plan på til sammen 9,93 sekunder. Testkjøringen har en varighet på 38,31 sekunder. 11 pulser simuleres på samme tid som 63 ville blitt avfyrt med dagens metoder. 52 skudd mot nyrevevet ble unngått.

Figur 7.17 viser et utsnitt av testforløpet og faktisk tilstand ved samme frame i videofilen. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla. Den trackede steinen avviker minimalt fra reel forløp og kan sees på som tilnærmet lik.



Figur 7.16: Plott av tracking samt faktisk bevegelse til nyrestein i video 5. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla, rød og cyan for hver av de 3 intervallene hvor steinen går faktisk inn i planet. Rektangelet skal representere halvparten av brennpunktet og er 20x20 piksler stort. Det er liten variasjon mellom faktisk og tracket verdi. Plottet er et utsnitt av én gjennomkjøring av filmen.

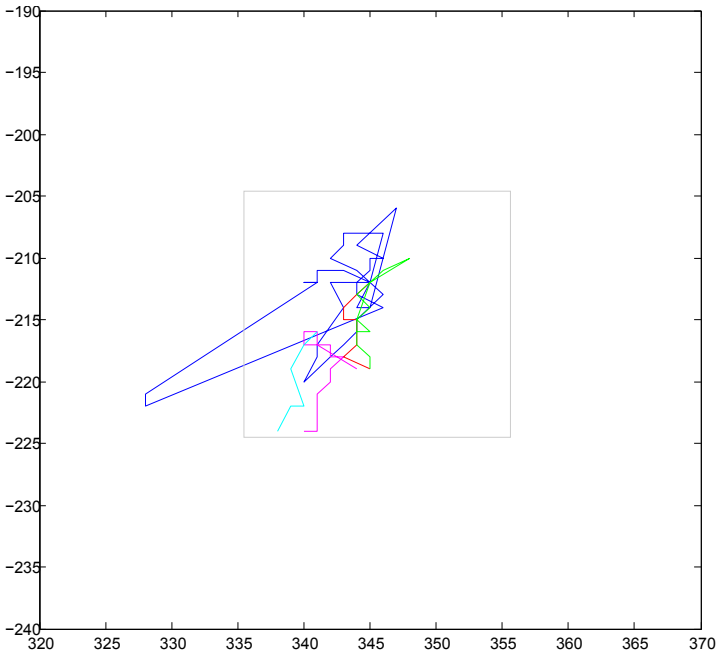


Figur 7.17: Et begrenset utsnitt av reel bevegelse og testresultat for video5. Begrensningen er gjort for bedre å sammenligne de to forløpene. For begge tilfeller er posisjonene frame nummer 77 til 198 plottet. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla.

7.3.4.6 Video6

I kapittel 7.3.2.6 er det beskrevet på hvilken måte video 6 skiller seg fra resterende testopptak. Siden steinen er så sjeldent i planet vil det være mange andre objekter som kan lure programmet og flytte søkevinduet vekk fra punktet hvor steinen i korte øyeblikk går inn og ut av planet. Under testkjøringen lot det seg ikke gjennomføre å bruke programmet på konvensjonell måte for å tracke steinen. Hver gang steinen forsvant ut av plan var det andre objekter som ble missoppfatet som stein og tracket. Når så steinen kom tilbake var ikke søkevinduet nære nok til å kunne reoppdage den. Løsning en ble å skru av dynamisk grenseverdisegmentering og stille grenseverdien konstant høy. Slik ble de uønskede objektene segmentert vekk mens steinen fortsatt var av en slik lysintensiv karakter at den ble værende i bildet. I figur 7.18 kan det vises det hvordan steinen ble oppdaget innenfor et området som var mindre enn brennpunktet nesten hver gang. Bare i et tilfelle ble et uønsket objekt fulgt av søkevinduet, men når dette objektet forsvant ut av planet var vinduet fortsatt nære nok nyresteinen til å kunne refokusere og tillate en simulert ESWL puls. Faktisk"ut-av-planet" bevegelse er fem ganger i løpet av én gjennomkjøring av filmen. I løpet av 18,74 sekunder med testkjøring ble det registrert 8 tilfeller.

8 skudd ble simulert. Med dagens behandlingsteknikk ville 31 vært avfyrt hvorav 74% ville vært bom.

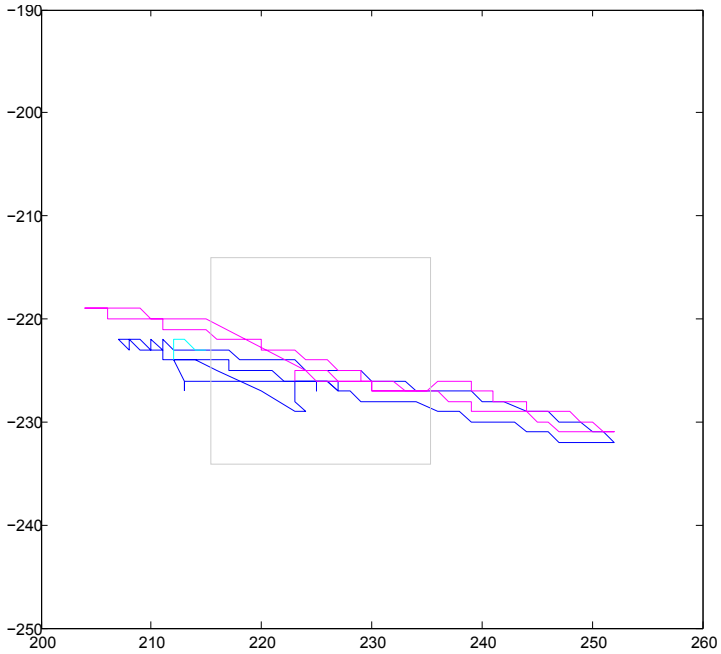


Figur 7.18: Plott av tracking samt faktisk bevegelse til nyrestein i video 5. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla, rød, grønn og cyan for hver av de 4 intervallene hvor steinen går faktisk inn i planet. Rektangelet skal representere brennpunktet og er 20x20 piksler stort. Plottet er et utsnitt av én gjennomkjøring av filmen.

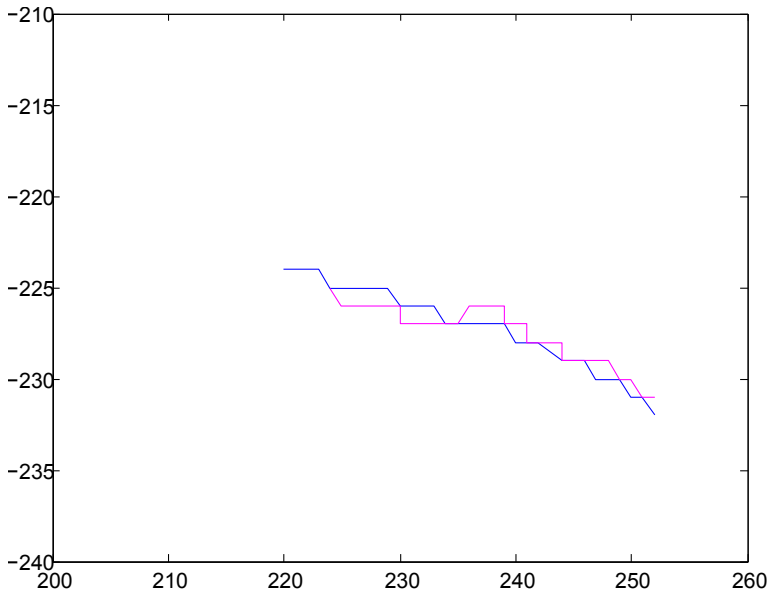
7.3.4.7 Video7

I video 7 er steinen ute av planet to ganger. Programmet registerte 6 tilfeller, med en total tid på 3,65 sekunder ute av planet i løpet av en testkjøring på 27,08 sekunder . De trackede posisjonene har et avvik på 5-7 piksler i forhold til faktisk verdi, noe som er godt innenfor brennpunktet. Søkevinduet finner tilbake til steinen så fort den dukker opp igjen i ultralydplanet.Simulatoren registrerer 15 pulser i løpet av testkjøringen og beregner at 45 pulser ville blitt avfyrt uten tracking.

Figur 7.20 viser et utsnitt av testforløpet og faktisk tilstand ved samme frame i videofilen. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla. Den trackede steinen avviker aldri med mer enn 1 piksel fra reel posisjon.



Figur 7.19: Plott av tracking samt faktisk bevegelse til nyrestein i video 7. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla og cyan for hver av de 2 intervallene hvor steinen går faktisk inn i planet. Rektangelet skal representere brennpunktet og er 20x20 piksler stort. Det er liten variasjon mellom faktisk og tracket verdi. Plottet er et utsnitt av én gjennomkjøring av filmen.



Figur 7.20: Et begrenset utsnitt av reel bevegelse og testresultat for video7. Begrensningen er gjort for bedre å sammenligne de to forløpene. For begge tilfeller er posisjonene frame nummer 90 til 150 plottet. Tracket stein er gjengitt med blått og faktisk bevegelse er lilla.

Kapittel 8

Diskusjon

Oppgavens hovedmål var å utvikle et system for tracking av nyrestein i et ultralydbilde. Systemet ble utviklet i henhold til en kravspesifikasjon, definert i kapittel5.2. I motsetning til for eksempel røntgen som avbildningsteknikk, består ikke ultralyd av skadelig stråling. Derfor er det ønskelig å bruke denne avbildningsteknikken fremfor og ikke bare som et substitut for røntgen under ESWL behandling. Kravspesifikasjonen inneholdt punkter om sanntidskrav for systemet, behandling av støy, robusthet mot "ut-av-planet" bevegelse og at metoden skulle kunne skille mellom stein og andre objekter i området rundt. På grunnlag av disse spesifikasjonene ble det utviklet en dynamisk grenseverdisegmenteringsalgoritme og Mean-Shift metoden ble valgt som trackingteknikk for systemet.

Testvideoene ble beskrevet i kapittel 7.3.2. Av resultatene kan det leses "at-ut-av-planet" bevegelsen er en av de mest dominerende utfordringene for et trackingsystem. Det viser seg at de gangene steinen forsvinner ut av planet, kommer den ofte tilbake på samme sted eller i området rundt samme sted. Dette skyldes hovedgrunnen for steinbevegelse, respirasjon, som ved konstant puls er en periodisk og stort sett uniform mekanikk. Den valgte tracking algoritmen har den egenskapen at den ved tap av tracket objekt, blir stående på siste registrerte koordinater, klar til å reagere på et nytt objekt. Hvis segmenteringen er utført korrekt, vil dette objektet være steinen. Dette er bevist gjennom testresultatene. "Ut-av-planet" bevegelse ble behandlet på en god måte i alle testtilfellene. Også i video 6, hvor ultralydproben var plassert for å maksimere"ut-av-planet" bevegelse, ble steinen funnet igjen. Her viste det seg derimot at den dynamiske grenseverdisegmenteringen feilet og manuell fremgangsmåte måtte brukes.

Det må tas høyde for en viss usikkerhet i beregning av den trackede steinens posisjon. Programmet er konstruert slik at det beregner tracket steins midtpunkt ut fra sentrum at trackingvinudet. Formen på vinduet spiller inn på midpunktets

plassering. Teoretisk skal rektangeler som ligger over hverandre, uansett form, ha sentrum i samme punkt. Men formen på kvadratet kan ha en innvirkning på hvordan søkevinduet plasserer seg over steinen. Et søkevindu som er mindre enn steinen vil kunne "flyte" fritt innenfor steien fordi tyngdepunktet er maksimalt over hele det hvite feltet uansett hvilken retning det søkes. Et søkevindu som er større eller like stort vil sentrere seg med steinens intensistetstyngdepunkt nøyaktig i midten. Denne effekten har vist seg å ikke ha noe å si for den totale prestasjonen til systemet, da trackingen ikke avviker nevneverdig fra reel bane. Dette gjelder selvsagt så lenge søkevinduet finner steinen og den ikke er segmentert ut eller er ute av planet. Figur 7.20, 7.17 , 7.15, 7.13, 7.11 og 7.9 viser hvordan avviket ikke overgår mer enn 5 piksler i noen av testene. Her er søkevinduene valgt vilkårlig uten tanke på å oppnå hverken perfekt eller unøyakt posisjonering over steinen.

Det er tross tilfredsstillende resultater verdt å merke at testvideoene ble generert av en røntgenlege med spesialkompetanse på området. Testvideoene var på mange måter av god kvalitet sammenlignet med hva som kan forekomme under behandling. De egnet seg godt til konsepttesting, men er ikke tilstrekkelig for å bekrefte systemets kvalitet for bruk under faktisk behandling. For dette trengs det mer omfattende testing in vitro. Mer om dette i kapittelet om videre arbeide.

Støy i bildet viste seg å ha liten eller ingen innvirkning i de forskjellige testscenarioene så lenge et filter på standardstørrelse $7x7$ piksler ble brukt. Det viktigste poenget med å filtrere bildet var å fjerne små lysintensive punkter slik at disse punktene ikke påvirket den generelle beregningen av tyngdepunkt i Mean-Shift algoritmen. Disse lyse punktene kan også bli missforstått som steinfragmenter og forflytte søkevinduet. Muligheten for å endre filterstørrelsen blir allikevel værende i sluttsystemet for å kunne håndtere eventuelle forskjeller ved bruk av testvideo fra andre ultralydmaskiner og modeller. Andre lysintensive objekter rundt steinen hadde ikke en fremtredende effekt på trackingen. Steinen ble alltid funnet tilbake til. I video 6 forflyttet søkevinduet seg etter et annet objekt, men da dette objektet forsvant ut av planet og steinen kom til bake i planet, ble steinen på nytt funnet og tracket. Når søkevinduene ble initiert ble de valgt slik at de hadde en mest mulig lik størrelse som nyresteinen. Hvis vinduet hadde blitt valgt større kunne objekter i nærheten hatt mer å si for resultatet.

Mean-Shift og forbehandlingen fungerte godt sammen for å takle de identifiserte utfordringene ved å tracke nyrestein i ultralydbilde. Hadde ikke forbehandlingen vært utført hadde resultatet vært noe annet, på grunn av ultralydbildets gjentagende lokale toppunkter i intensivitet. Den dynamiske grenseverdisegmenteringen som ble utviklet tidligere i oppgaven var avhengig av en spesiell parameter for å kunne gi et godt resultat. Verdien for prosent av histogramareal som skulle inneholde den lysintensive steinen, var generelt et godt mål for å kunne segmentere vekk mørkere objekter enn nyresteinen. Med unntak av to til feller i video2 og

video6. Hvis nivået er for lavt vil steinen bli segmentert vekk i for stor grad. I motsatt tilfelle vil ikke steinen skille seg noe ut fra omliggende anatomi. I det siste tilfellet kan man risikeres at ESWL pulser blir avfyrt mot andre objekter enn steinen og føre til å forverre resultatet i forhold til dagens behandlingsmetoder. Det kan oppstå en situasjon som er totalt motsatt av det ønskede resultatet, alle skudd bommer på steinen. Hvis nivået blir satt for lavt kan steinen bli segmentert vekk i for stor grad. Dette vil allikevel bare gå ut over behandlingstiden fordi færre skudd blir avfyrt. Fra tabell 7.2 kan det leses at en grenseverdi på 10 % vil gi lavest tid ute av planet. Men denne verdien tok med så mange andre lysintensive objekter at trackingen på et tidspunkt ble obstruert av et annet objekt. Søkevinduet fulgte dette objektet istedenfor steinen. 4 % gir faktisk en lavere tid ute av planet enn ved 5 prosent, men forskjellene er så små at det kan være andre faktorer som for eksempel initiering av søkevindu, som spiller inn. Testene fra kapittel 7.3.4 er utført med en 5 % grense, men 4 % kan også være et godt valg da denne verdien har marginalt lengre perioder med steinen i planet.

For at systemet skal fungere som i praksis og brukes under behandling, er det stilt visse krav til behandlingstid. Disse sanntidskravene ble omtalt i kapittel 7.1 og 7.2.3 . Systemet må rekke å gjøre beregninger før neste puls skal avfyres. Her er det i ytterste tilfelle snakk om 500 ms, noe som ikke utgjør noe problem for algoritmen. Det er derimot to tilfeller av strengere krav som gjelder for beregningstiden til systemet. For det første må systemet rekke å behandle en frame før neste frame er klar til uthenting. Dette er et mykt sanntidskrav siden steinen beveger seg lite mellom hver frame og derfor kan en senere frame være like nyttig for beregning. Det andre kravet kan sies å være fast. Når steinens posisjon er beregnet må det gå så liten tid at dette fortsatt er steinens posisjon når ESWL pulsen avfyres. Kravet kan sies å være fast og ikke hardt fordi det kan aksepteres eventuelle ikke treff på steinen, så lenge det ikke skjer for ofte. For definisjon av myke og faste sanntidskrav, se Burns et al[4]. Alle ikke treff som unngås er en forbedring fra dagens teknikk. Fra tabell 7.10 kan det leses at gjennomsnittlig kjøretid på en løkke med belastning i form av tracking og simulator er 37ms. Med en framerate på 25 bilder i sekundet gir dette 40ms mellom hver nye frame. I snitt overholder programmet det myke sammtidskravet. Når et søkevindu er like stort som nyresteinen, vil det gi en grense på én steinlengde per frame hva gjelder bevegelse av nyrestein, før det går ut over systemets evne til å tracke steinen. Siden kode for Mean-Shift og simulering av sudd ligger like etter hverandre i programflyten, vil tiden alltid være mindre enn løkketiden for en kjøring og derav vil også det andre kravet holde.

Alle testvideoene viser at nyresteinen kan trackes. Programmet kjennestegnes av å ha en streng segmenteringsfunksjon. Steinen blir segmentert bort for ofte, selv om den fortsatt er i planet. Dette vil gå ut over tiden behandlingen tar, altså det vil avfyres færre skudd enn hva som er mulig. Det er allikevel en nødvendig sikkerhet

for å unngå å treffe andre objekter enn steinen med ESWL pulser. De gangene steinen forsvinner fra søkevinduet, blir den alltid funnet tilbake av algoritmen. I tilfellet med større variasjon og hvor steinen eventuelt ikke blir funnet igjen eller det går for lang tid før dette skjer, inneholder programmet en timer som tar tiden på hvor lenge steinen er utenfor søkevinduet. Denne timeren kan brukes til å utløse en alarm som forteller operatøren at det er på tide å re-initiere søkevinduet eller flytte pasienten for å fokusere brennpunktet på et mer egnet område.

Fordelen med et system for tracking av nyrestein er åpenbar. I de syv gjennomkjøringene av de syv forskjellige videoene ble det totalt avfyrt 119 ESWL pulser. Skuddfrekvensen var 100 skudd i minuttet. Med tilsvarende frekvens og på en tilsvarende tid som for testtrackingen, ville 299 pulser blitt avfyrt. Når vi har rimeliggjort at hvert skudd treffer steinen under tracking, vil dette si at 60,2 % av skuddene ville ha truffet i nyrevevet rundt steinen dersom tracking ikke hadde vært brukt.

Kapittel 9

Konklusjon

I denne oppgaven har et system for tracking av nyrestein i et ultralydbilde blitt utviklet. Systemet har også fått et brukergrensesnitt som gjør det håndterlig for en operatør. Først og fremst for test og simulering, men også med tanke på videre arbeide med systemet for å kunne brukes under ESWL behandling. På denne måten kan ultralyd brukes som avbildningsteknikk fermfor rønteng. Røntgenavbildning er en teknikk som utsetter pasienten for skadelige stråling samt gjør det vanskelig å avgjøre om steinen er i brennpunktet fra én posisjon. Utfordringene for et trackingsystem ble identifisert til å være "ut-av-planet" bevegelse, støy, lav oppløsning og andre lysintensive objekter i området rundt nyresteinen. Mean-Shift algoritmen ble valgt som trackingalgoritme fordi ultralydbildets karakter som en intensitetsdistribusjon er nøyaktig hva algoritmen bruker som input. Algoritmen var allikevel avhengig av forbehandling i form av filtrering og grenseverdisegmentering for å kunne identifisere og skille steinen fra omliggende lysintensive objekter. En dynamisk grenseverdisegmentering ble utviklet for å håndtere variasjonen i intensitet mellom forskjellige filmer og i løpet av opptak.

Systemet ble implementert i OpenCV og Qt og det ble stilt krav til sanntidsegenskapene ved systemet. Det ble testet og begrunnet for at systemet overholdt sine to sanntidskrav. Kravene ble definert som svak og fast, det vil si at det tolereres overtredelser uten at det går ut over prestasjonen til systemet.

Syv testvideoer ble undersøkt og steinens faktiske bevegelser ble registrert. Det ble gjennomført testkjøringer som ble sammenlignet med de faktiske bevegelsene. Trackingen fungerte meget tilfredsstillende med bevegelser innenfor de faktiske bevegelsesområdene for steinene. "Ut-av-planet" bevegelse ble håndtert godt av algoritmen ettersom steinen kom tilbake i planet i samme området som den forsvant. Dette var tilfelle i alle testvideoene, men det blir anbefalt videre testing i mer naturlig omgivelser for å sikrere kunne validere robustheten til trackingen. I de syv

gjennomkjøringene av de syv forskjellige videoene ble det totalt avfyrt 119 ESWL pulser. Skuddfrekvensen var 100 skudd i minuttet. Med tilsvarende frekvens og på en tilsvarende tid som for testtrackingen, ville 299 pulser blitt avfyrt. Når vi har rimeliggjort at hvert skudd treffer steinen under tracking, vil dette si at 60,2% av skuddene ville ha truffet i nyrevevet rundt steinen dersom tracking ikke hadde vært brukt.

I kapittel 2.2.5 nevnes det at pasienter kan oppleve blod i urinen, indre blødninger, smerter i forbindelse med indre blødninger og store blåmerker under og etter ESWL behandling. For å begrense skadene er både effekten i sjokkbølgen og skuddfrekvens begrenset. Når man er garantert å treffe steinen med hvert skudd vil man ikke bare begrense plagene hos pasienten, men også kunne få en mer effektiv behandling ved å fjerne begrensinger på effekt og frekvens.

Den dynamiske grenseverdisegmenteringen gav gode resultater i alle filmene unntatt video6 hvor manuell fremgangsmåte måtte brukes. Noen av filmene med høy intensitet i området rundt steinen var avhengig av å plassere "region of interest" på en slik måte at ikke de lyseste områdene ble med i beregningen av histogrammet. Bare da var steinen tydelig og kunne trackes på vanlig måte. Dette tyder på at det ikke finnes en generell prosentverdi som kan brukes for alle videoer. En videre utvikling må kunne gi operatøren mulighet til å bestemme hvor mange prosent som skal segmenteres vekk i bildet.

Problemstillingen til denne oppgaven innebefattet et literturstudie for først å undersøke muligheten for å lage et tracking system. Dette litteraturstudiet førte frem til en implementasjon med overbevisende testresultater. Det anbefales derfor videre arbeid på bakgrunn av denne oppgaven.

Kapittel 10

Videre arbeide

Det totale målet med å bruke tracking for å vite hvor nyresteinene er, vil til slutt være å kunne forbedre dagens ESWL behandling. Både ved å fjerne røntgen som avbildningsteknikk og eliminere plager under og etter behandling. Videre arbeid vil i all hovedsak dreie seg om å validere trackingen i mer naturlige omgivelser. Og bruke koden sammen med en faktisk ESWL maskin for å avfyre pulser til riktig tid og til slutt test in vitro, altså på mer eller mindre naturlige stoffer i kontrollerte omgivelser.

Lage testsystem: Et testsystem bør ha muligheten for å simulere respirasjon med forskjellige hastigheter og retninger. Et styringssystem for bevegelse av en modell som simulerer vev med en kunstig nyrestein kan være et godt eksempel på et slikt system. Testsystemet må ha mulighet til å settes i kontakt med en ESWL maskin for å utsette modellen for sjokkbølger.

Kobling mot ESWL maskin: Systemet som er utviklet til nå simulerer bare en ESWL puls når det er mulig. I virkeligheten må det være en kobling mellom systemet og ESWL maskinen. En live videostrøm må hentes fra en ultralydmaskin. Her må det undersøkes hvordan videoen kan hentes ut og sannsynligvis lages egne drivere for å kunne gjøre programmet generelt nok til å kunne bruke flere typer ultralydmaskiner. I koden hentest video inn fra en fil. Det er på tilsvarende sted frames kan hentes fra en live videostrøm. Neste steg vil være å styre avfyring av pulser fra ESWL maskinen. Dersom systemet er lukket og det ikke går an å styre fra et interface, kan det være nødvendig å utvikle et system som for eksempel bruker en servo for å direkte kontrollere styringspanelet.

Test: Selv om systemet gav overbevisende resultater under kontrollert test, kreves mer testing for å bekrefte om systemet kan brukes under behandling. Det må undersøkes:

- Hvordan ultralydproben kan plasseres under behandling for at steinen skal være mest mulig i planet.
- Kan skuddfrekvens og effekt økes for å gjøre behandlingen mer effektiv.
- Sanntidsegenskapene etter systemutvidelse.

Tillegg A

Brukermanual

Komme i gang

På medfølgende DVD ligger både programkode som kan kompileres for å kjøre programmet og en installasjonsfil som legger inn alt som er nødvendig for å bruke programmet.

For å kompilere og kjøre programmet anbefales det å installere Qt Creator og OpenCv først. OpenCV kan lastes ned fra denne lenken: http://opencv.willowgarage.com/wiki/ Det anfalles å installer OpenCv til denne banen: C:\OpenCV2.1\. På denne måten trengs det ikke gjøre noen endringer med Qt prosjektet KidneyTrack for å bruke det. Qt kan lastes ned fra denne lenken: http://qt.nokia.com/products/ og trenger ikke gjøres noen endringer med for at KidneyTrack skal kunne åpnes.

For å bruke programmet direkte installeres programmet med "KidneyTrack 0.5 beta.msi" som ligger på DVD. Etter endt installasjon ligger det et ikon på skrivebordet med navnet "KidneyTrack". Dobbeltklikk på ikonet for å starte programmet.

Bruk

Når "KidneyTrack" er startet kommer det først opp en dialogboks. Her kan man bla seg frem til den ønskede filen som skal analyseres. Medfølgende på DVD ligger alle syv testvideoene som er brukt under denne rapporten. Disse videoene er av riktig format for å kunne åpnes programmet. Etter at video er valgt åpnes programmet med første frame fra filmen i venstre avspillingsvindu.

- I feltet "Movie Controls " kan filmen startes og stoppes, samt at posisjon kan endres med glidebryteren.

- "Tracking Parameters" inneholder glidebrytere for å posisjonerer arbeidsområdet og for å endre filterstørrelsen. Resultatet av endringene kan sees i høyre avspillingsvindu.
- "Dynamisk threshold" inneholder data fra dynamisk grenseverdisegmentering. Her er det også mulighet for å skru av den automatiske segmenteringen og endre grenseverdien med glidebryteren nederst i feltet.
- I "Simulator" feltet kan simulatoren kontrolleres. Brennpunkt størrelsen kan settes til å tilsvare naturlig brennpunkt størrelse og det kan også plasseres til ønskelig posisjon ved å bruke glidebryterne. Skuddfrekvensen kan reguleres til ønsket verdi ved å endre skudd per minutt variabelen. Når alle parametrene er stilt inn startes simulatoren med start knappen.

Loggfilen er komplett når programmet avsluttes og den finnes i programmappen.

Tillegg B

Bug rapport

B.1 Liste over kjente feil

Codec Programmet støtter i utgangspunktet bare de codecene som er støttet av OpenCV. I noen tilfeller vil programmet kunne spille av formater som h.264, divX, Xvid osv på Windows Xp plattformen, men vil feile når filmen kommer til siste frame.

Loggfeil Denne feilen gjelder for Windows 7. Når programmet startes på vanlig måte i Windows 7 vil ikke loggfilen genereres og fylles ut med loggdata.

Glidebryter Når man trykker et sted i slider baren for å flytte slideren dit uten å trykke på selve slideren og dra den dit, vil ikke posisjonsvariabelen i programmet opptatteres og filmen vil starte på samme sted som den ble pauset.

B.2 Løsninger på kjente feil

Codec Bare bruke støttede codecer. I videre arbeid anbefales det å skrive om programmet til å bruke ffmpeg biblioteket for å hente inn film. På denne måten vil alle codecer støttes så lenge de er installert i Windows.

Loggfeil Problemet løses ved å kjøre programmet som administrator. Da får man rettigheter av OSet til å opprette en fil.

Glidebryter Dra slideren i stedet for å klikke seg bort. En løsning i koden er eventuelt å opptatere slider pos hver gang slider beveger seg og ikke bare når den klikkes og dras.

Tillegg C

Programkode

I dette kapittelet ligger noen av headerfilene fra programmet. De er lagt med som et enkel oppslagsverk og som støtte til implementasjonsdelen av rapporten. En komplett programkode ligger på vedlagt DVD.

C.1 mainwindow.h

```
1
2  /*************************************************************
3  *
4  * FILENAME mainwindow.h
5  *
6  * Forfatter    : Jens Kr Tøraasen
7  *
8  * Beskrivelse  : Denne filen inneholder stort sett hele
9  *                programmet. Det vil si metode for å
10 *                 tegne videobildet i guiet, koble sammen
11 *                tracking koden med guiet med SLOT og
12 *                 SIGNAL mekanismen deffinert i Qt,selve
13 *                 tracking algoritmen og tilhørende forbehandling
14 *                 og simulator
15 *
16 *                 Programstrukturen er i ettertid sett til
17 *                å være lite oversiktlig for å arbeide
18 *                videre med for andre personer. Har prøvd
19 *                å løse dette så godt som mulig med kommentarer
20 *                og beskrivelse, men den beste løsningen
21 *                hadde nok vært å dele opp koden i mere filer
22 *                og klasser.
```

```
23 ************************************************************/
24 #ifndef MAINWINDOW_H
25 #define MAINWINDOW_H
26 #include <QMainWindow>
27 #include <highgui.h>
28 #include <cv.h>
29 #include <stdlib.h>
30 #include <stdio.h>
31 #include <ctype.h>
32 #include <time.h>
33 #include <sys/types.h>
34 #include <QWidget>
35 #include <QGraphicsRectItem>
36 #include <QGraphicsScene>
37 #include <QPushButton>
38 #include <QTimer>
39 #include "videoimage.h"
40 #include <QFileDialog>
41 #include <QDebug>
42 namespace Ui {
43     class MainWindow;
44 }
45 class MainWindow : public QMainWindow {
46     Q_OBJECT
47 public:
48 /*
49  *  Konstruktør og dekonstruktør for GUI klassen
50  */
51 MainWindow(QWidget *parent = 0);
52 ~MainWindow();
53 /*
54  *  Metode for å konvertere IplImage fra OpenCv til et pixmap
55  *  som er formatet som støttes i Qt. Når metoden kalles settes
56  *  det konverterte pixmapet inn i et pixmap item som i sin tur
57  *  er koblet til et graphics view i mainwindow.ui.
58  *
59  *  Koden er en modifisert utgave av kode for å utføre samme ↩
        type
60  *  konvertering, skrevet av Stig Hornang og Andreas L. Carlsen
61  *
62  *  Metoden tar inn en IplImage peker og setter pixmapet inn i ↩
        et
63  *  QGraphicsPixmapItem ved hjelp av:
64  *          trackViewGPI->setPixmap( trackViewPixmap )
65  *
66  *  For å bruke metoden kammer man den fra det stede i koden ↩
        hvor
67  *  bildet oppdatteres ved hjelp av:
68  *          MainWindow::redrawTrackView( frame );
```

```
69   *
70   *
71   *  Siden metoden er til for konvertere fra OpenCv til Qt er det en
72   *  en del ting som må sette opp på forhånd.
73   *
74   *  Først lager vi et VideoImage, som er en egendeffinert klasse.
75   *  Metoden er kommentert i videoimage.h.
76   *          trackViewScene = new VideoImage( cvSize(frame_width,frame_height),
77   *                                           this->ui->mainView->pos(),  this );
78   *  Deretter setters VideoImaget inn i et Graphics View, i dette tilfellet mainView
79   *          this->ui->mainView->setScene( trackViewScene );
80   *  Til sist legger vi GraphicsPixmapItemet som redrawTrackView jobber mot inn i
81   *  trackViewScene. En ganske kronglette prosess med andre ord
82   *
83   *          trackViewGPI = new QGraphicsPixmapItem();
84   *          trackViewScene->addItem( trackViewGPI );
85   *
86   */
87      void redrawTrackView( IplImage* frame );
88  /*
89   * Metoden fungerer på tilsvarende måte som redrawTrackView
90   */
91      void redrawDataView( IplImage* frame );
92  /*
93   *  Metoden fungerer på tilsvarende måte som redrawTrackView, bortsett fra at vi
94   *  i selve metoden bare jobber med en kanal i IplImage.
95   */
96      void redrawHistView( IplImage* hist );
97  /*
98   *  Metoden simulerer skudd av ultralydpulser. Sjekker at det har gått nok tid siden
99   *  forrige skudd og øker teller for totalt skudd med 1. Metoden inneholder også kode
100  *  for å skrive til loggfilen
101  */
102     void skyt();
103 /*
104  *  Metoden konverterer QRect til CvRect
105  */
106     CvRect qrect2cvrect(QRect qrect);
107 protected:
108     void changeEvent(QEvent *e);
```

```
109 private:
110     Ui::MainWindow *ui;
111 /*
112  *  Variabel som brukes av OpenCv for å hente frames fra filen ↩
        som capture peker mot
113  */
114     CvCapture* capture;
115 /*
116  *  Roi er området av bildet som skal forbehandles (filter og ↩
        threshold) før trackingen
117  *  oppdatteres. Området kan ikke endres etter at det er ↩
        initiert første gang. En naturlig
118  *  utvidelse vil være å gjøre dette dynamisk
119 */
120 int roiWidth;
121 int roiHeight;
122 int roiX;
123 int roiY;
124 /*
125 *  Grenseverdien for threshold
126 */
127 int thresholdValue;
128 /*
129 *  Forteller om dynamisk thresholding skal være på eller om det ↩
       skal styres manuelt. 1=på, 0=av
130 */
131 int dyn_tresh;
132 /*
133 *  Filteret som brukes er et averaging filter på størrelse ↩
       filterSize*filterSize
134 */
135 int filterSize;
136 /*
137 *  Variabel for å kontrollere playback av video og dermed hele ↩
       programmet. 1= play, 0 = stop
138 */
139 int pause;
140 /*
141 *  Totalt antall frames i videoen som behandles
142 */
143 int frames;
144 int frameNumber;
145 /*
146 *  Setter opp IplImage peker som skal brukes i tracking ↩
       algorimen. Frame, processed og hist_image
147 *  er bilder som vises i programmet. De andre brukes bare til ↩
       midlertidige beregninger internt i
148 *  algoritmen
149 */
```

```
150 IplImage* frame;
151 IplImage* procesed;
152 IplImage* midl;
153 IplImage* processed_en_kanal;
154 IplImage* hist_en_kanal;
155 IplImage *image;
156 IplImage *hist_image;
157 /*
158 *  Brukes av dynamsik thresholding algoritmen. bin_total er ↩
    arealet under hele histogram-grafen, bin_prosent
159 *  er arealet av de forhåndsdeffinerte prosent lyseste verdiene,↩
     bin_vol_limit er bin_total - bin_prosent.
160 *  Til sist brukes bin_top_sum når arealet under grafen summeres↩
     opp fra den lyseste intensiteten og nedover
161 *  helt til bin_tip_sum er større eller lik bin_vol_limit. ↩
    max_value brukes er histogramets lyseste intensitet
162 */
163 int bin_total, bin_prosent, bin_vol_limit, bin_top_sum;
164 float max_value;
165 /*
166 *  Forteller om det trackede objektet er i det simulerte ↩
    brennpunktet slik a firkanten rundt brennpunktet blir grønn
167 *  Variabelen settes til 0 igjen med en gang den grønne ↩
    firkanten er tegnet. Hvis objektet er i brennpunktet på neste
168 *  iterasjon settes den igjen til 1 og firkanten blir tegnet ↩
    grønn. Hvis ikke vil variabelen være 0 og firkanten rød
169 */
170 int iBrennpunkt;
171 /*
172 *  Når steinen går ut av fokus, det vil si går i z-retning i ↩
    forhold til xy-planet som er ultralydsnittet, vil variabelen
173 *  settes til 1 og bli 0 når stenen er i fokus. Dette brukes ↩
    bare i logg.
174 */
175 int fokus_intervall;
176 /*
177 *  Variablene har nesten samme funksjon som fokus_intervall, men↩
     brukes bare til å fortelle når skuddene starter i et skudd
178 *  intervall. Brukes bare i logg.
179 */
180 int changed;
181 int forrige_brennpunkt;
182 /*
183 *  track_window deffinerer området objektet som skal trackes ↩
    ligger innenfor og brukes av cvMeanShift metoden.
184 */
185 CvRect track_window;
186 /*
```

```
187 *  Rester fra test av cvCamShift. Lar den være med i tilfelle ↩
       noen ønsker å kommentere inn metoden igjen i koden
188 */
189 CvBox2D track_box;
190 /*
191 *  Struktur hvor cvMeanShift plasserer det nye beregnede ↩
       søkevinduet (->rect) og summen av alle pixler i vinduet (->↩
       data)
192 */
193 CvConnectedComp track_comp;
194 /*
195 *  Variabler for det simmulerte brennpunktets plassering og ↩
       størrelse
196 */
197 CvRect brennpunkt;
198 int brennpunktX;
199 int brennpunktY;
200 int brennpunktHeight;
201 int brennpunktWidth;
202 /*
203 *  Brukes av av delen av koden som sjekker om steinen er i ↩
       brennpunktet. Det er steinens sentrum som må være innenfor.
204 */
205 CvPoint stenSentrum;
206 /*
207 *  Variabler for histogram som brukes av den dynamiske ↩
       thresholdingen. hist_size er størrelsen på histogrammet,
208 *  mens *hist er en peker til OpenCv sin histogram struktur
209 */
210 int hist_size;
211 CvHistogram *hist;
212 /*
213 *  Timer variable som brukes til å lagre på hvilket prosessor ↩
       tick enkelte operasjoner utføres. Brukes så til beregning
214 *  av når og hvor lang tid operasjoner skjer og tar. ↩
       program_start_time og program_time_start_track er de eneste
215 *  variabelen som bare oppdatteres en gang.
216 */
217 int64 now_time;
218 int64 program_start_time;
219 int64 end_time;
220 int64 program_time_start_track;
221 int64 sten_ut_av_plan_start_time;
222 /*
223 *  Hvor mange pulser sendes avgårde per minutt. Er en del av ↩
       simuleringen.
224 */
225 int skudd_per_min;
226 /*
```

```
227 *  Hvor mange milisekunder det går mellom hvert skudd. Er en del↩
        av simuleringen.
228 */
229 float skudd_frekvens;
230 /*
231 *  Flagg som for å kontrollere riktig rekkefølge på innhold i ↩
       loggfil
232 */
233 int forst_skudd;
234 int forste_track;
235 /*
236 *  Variabler for å beregne og logge effektiviteten i algoritmen
237 */
238 int total_skudd;
239 int total_ut_av_plan;
240 /*
241 *  Her lagres antall sekunder siden programmet ble startet
242 */
243 float program_time_sek;
244 /*
245 *  Videoens bredde og høyde
246 */
247 int frame_width;
248 int frame_height;
249 /*
250 *  Kontroller om logging skal utføres. Den starter også selve ↩
       simuleringen av skudd. I utgangspunket er
251 *  denne mekanismen bare ment til å starte simuleringen og ikke ↩
       stoppe den. Dette er fordi man skal få tid
252 *  å plassere brennpunktet i steinens bane før man begynner test↩
        av effektiviteten.
253 */
254 int loggOn;
255 /*
256 *  Variable og datastrukturer som trengs for å konvertere et ↩
       OpenCv bilde til en Qt bilde og vise det i Qt
257 *  Bruk er forklart i kommentarene til redraw****View metodene
258 */
259 uchar *trackViewData, *dataViewData,*histViewData;
260 QImage trackViewImage, dataViewImage, histViewImage;
261 QPixmap trackViewPixmap, dataViewPixmap, histViewPixmap;
262 QGraphicsPixmapItem *trackViewGPI, *dataViewGPI, *histViewGPI;
263 VideoImage * trackViewScene;
264 VideoImage *dataViewScene;
265 VideoImage *histViewScene;
266 /*
267 *  Timer som brukes for å hente ut frames fra filen i gitte ↩
       intervaller
268 */
```

```
269 QTimer *timer;
270 /*
271 *  Filen som loggen dumpes til.
272 */
273 FILE *file;
274 FILE *pos_plot;
275 public slots:
276 /*
277 *  Denne metoden utfører alt arbeidet. I tillegg til å ta seg av mean shift utførere den også forbehandling og simmulering
278 *  Det er denne metoden som burde deles opp for å få en mer forståelig kode. Den er forklart så godt som mulig med kommentarer
279 *  i mainwindow.cpp filen
280 */
281 void OpenCVMeanShift();
282 /*
283 * Håndterer hva som skal skje når Play trykkes. Altså setter pause = 1.
284 */
285 void StartButtonClicked();
286 /*
287 * Håndterer hva som skal skje når Pause trykkes. Alså setter pause = 0.
288 */
289 void StopButtonClicked();
290 /*
291 * Håndterer hva som skal skje når Start trykkes i simulator feltet. Altså setter loggOn variabelen til 1.
292 */
293 void startLoggButtonClicked();
294 /*
295 * Håndterer hva som skal skje når På trykkes i dyn thresh feltet. Altså setter dyn_tresh variabelen til 1.
296 */
297 void dynThreshOnClicked();
298 /*
299 * Håndterer hva som skal skje når Av trykkes i dyn thresh feltet. Altså setter dyn_tresh variabelen til 0.
300 */
301 void dynThreshOffClicked();
302 /*
303 *  Metoder for å kontrolere hva som skjer når slidere beveges. Altså endre korrekt variabel i henhold
304 */
305 void moveVideoSlider(int pos);
306 void moveThresholdSlider(int pos);
307 void moveBrennpunktXSlider(int pos);
308 void moveBrennpunktYSlider(int pos);
```

```
309 void moveRoiXSlider(int pos);
310 void moveRoiYSlider(int pos);
311 void moveFilterSlider(int pos);
312 /*
313 *  Metoder som endrer størrelsen på brennpunkt når den reguleres↩
        fra spin boxene
314 */
315 void endreBrennpunktXBox(int verdi);
316 void endreBrennpunktYbox(int verdi);
317  /*
318 *  Oppdattere skuddfrekvensen når antall skudd per min endres i ↩
       spin boxen i simulator feltet
319 */
320 void endreSkuddFrekvens(int verdi); signals:
321 /*
322 *  Emitter data tilbake til guiet i henhold til metodikken i Qt ↩
       rammeverket.
323 */
324 void updateVideoSlider(int pos);
325 void updateThresholdSlider(int pos);
326 void updateSkuddLcd(int verdi);
327 };
328 #endif // MAINWINDOW_H
```

Referanseliste

[1] Store medisinske leksikon Anton Hauge. Ekstracellulærvæske. url-http://www.snl.no/.sml_artikkel/ekstracellulærvæske, 2010.

[2] Justin Bowra and Russel E McLaughlin. *Emergency ultrasound made easy.* Churchill Livingstone, 2006.

[3] Gary Bradski and Adrian Kaehler. *Learning OpenCV, Computer Vision with the OpenCV library.* O'Reilly, 2008.

[4] Alan Burns and Andy Wellings. *Real-time Systems and Programming Languages.* Addison Wesley, 2001.

[5] Marius S. Eltervåg. Nokia har kjøpt trolltech. url-http://www.amobil.no/artikler/nokia_har_kjopt_trolltech/47851, Januar 2008. Oppdatert: 29. jan 2008 09:40.

[6] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing.* Pearson Education, 2008.

[7] Michael Grasso. Extracorporeal shockwave lithotripsy. urlhttp://emedicine.medscape.com/article/444554-overview, 2008.

[8] Richard Gunderman. *Essential Radiology: Clinical Presentation Pathophysiology Imaging.* Thieme, 2nd edition, 2006.

[9] Aglika Gyaourova, Chandrika Kamath, and Sen-ching Cheung. Block matching for object tracking. LAWRENCE NATIONAL LABORATORY LIVERMORE, October 2003.

[10] Tom Harris. How x-rays work. http://health.howstuffworks.com/x-ray.htm, 2002.

[11] Michael Isard and Andrew Blake. Condensation conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28, 1998.

[12] J. Larry Jameson, Dennis L. Kasper, Dan L. Longo, and Eugene Braunwald. *Harrison's Principles of Internal Medicine*. McGraw-Hill Professional, 2008.

[13] Maciej Orkisz, Maurice Bourlion, Gerard Gimenez, and Thierry A. Flam. Real-time target tracking applied to improve fragmentation of renal stones in extra-corporeal lithotripsy. *Machine Vision and Applications*, 11:138–144, 1999.

[14] Store medisinske leksikon Per Holck. Epitel. url-http://www.snl.no/.sml_artikkel/epitel, 2010.

[15] Store medisinske leksikon Per Holck. Urinleder. url-http://www.snl.no/.sml_artikkel/urinleder, 2010.

[16] Milan Sonka, Vaclav Hlavac, and Roger Boyle. *Image Processing, Analysis and Machine Vision*. Thomson, 2008.

[17] Norsk Helseinformatikk Terje Johannessen. Nyrestein - en oversikt. url-http://nhi.no/sykdommer/kirurgi/nyrer -og-urinveier/nyrestein-en-oversikt-2593.html, 2009.

[18] Johan Thelin. *Foundations of Qt Development*. Apress, 2007.

[19] Jenho Tsao and Jia-Hong He. Ultrasonic renal-stone tracking with mesh regularization. In *Proceedings of the 29th Annual International Conference of the IEEE EMBS*, 2007.

[20] Deepak Turaga and Mohammed Alkanhal. Search algorithms for block-matching in motion estimation. url-http://www.ece.cmu.edu/ ee899/project/deepak_mid.htm, 1998.

[21] Kjell Tveter and Store medisinske leksikon Wahlqvist, Rolf. Nyrestein. url-http://www.snl.no/.sml_artikkel/nyrestein, 2010.

[22] Cinnamon VanPutte, Jennifer Regan, and Andyyy Russo. *Seeley's Essentials of Anatomy and Physiology*. McGraw-Hill, 7.th. edition, 2010.

[23] Fai Yeung, Stephen F Levinson, Dongshan Fu, and Kevin J parker. Feature-adaptive motion tracking of ultrasound image sequences using a deformable mesh. *IEEE Transactions on Medical Imaging*, 17:945–956, 1998.