Forord

Denne hovedoppgaven tar for seg redigering i sammenheng med et totalsystem for søk, gjenfinning og gjenbruk av digital video. Oppgaven tilhører fagfeltet multimedia og bygger delvis på min sommerjobb ved Norsk Regnesentral, sommeren 1994, hvor jeg deltok i implementeringen av en digital videospiller. Oppgaven ble gitt som et samarbeid mellom Norsk Regnesentral (NR) og Institutt for datateknikk og telematikk (IDT), i tilknytning til et prosjekt med tittelen *Leveranse av video over ATM* (LAVA).

Veiledere ved IDT har vært amanuensis Roger Midtstraum og stipendiat Rune Hjelsvold, og jeg vil takke begge for god oppfølging og godt samarbeid under arbeidet med denne oppgaven. Ved Norsk Regnesentral har jeg fått god hjelp og støtte av forsker Gisle Aas. Jeg vil også takke forskningssjef Erling Maartmann-Moe for muligheten til å skrive denne oppgaven ved NR, noe som har gitt meg mange gode faglige og utenomfaglige erfaringer.

En takk fortjener også Jan Braathen ved Silicon Graphics, som har vært svært behjelpelig med informasjon og utlån av utstyr. Det samme gjelder Johan Elvemo og Reinhold Rathjen ved Institutt for drama, film og teater ved AVH. Til sist vil jeg rette en stor takk til min far for hans gode kommentarer og idéer.

Arild Eistein Bekkadal

Sammendrag

Ved å sette redigering inn i en databasesammenheng kan vi utnytte de fordeler som digital video gir oss. Redigering av digital video lagret på harddisk gir redigereren større frihet til å manipulere videomaterialet sammenlignet med den analoge redigeringsprosessen.

Gjennom arbeidet med denne oppgaven har jeg studert redigeringsverktøy og -prosesser, og satt dette inn i en databasesammenheng. Jeg har vurdert løsninger for et redigeringsverktøy som skal kunne benyttes i et totalsystem, bestående av videodatabase og verktøy for søk og registrering. Noen av de løsningene jeg har foreslått er utprøvd gjennom implementasjonen av en prototyp.

En essensiell fordel som oppnås ved å sette redigering inn i en databasesammenheng er at behovet for duplisering av materiale reduseres vesentlig. Vi kan redigere på referanser til, i stedet for direkte på, de fysiske videosekvensene. Dette gir oss stor frihet til å manipulere de ulike sekvensene, ved å endre referansene i stedet for å kopiere materialet.

Ved å sette redigeringsverktøyet inn i et totalsystem vil vi kunne utnytte de ressurser som ligger i utstrakte muligheter til søk, gjenfinning og gjenbruk av videomateriale i en database. Redigeringen setter imidlertid spesielle krav til de andre verktøyene vedrørende samhandling. Det kreves en tett interaksjon mellom videospiller og redigeringsverktøy for å kunne fremskaffe oppdaterte grensesnitt i begge applikasjoner. Ved redigering får vi dessuten generert informasjon om strukturen til videomaterialet, som vi kan legge direkte inn i databasen, for dermed å begrense kravet til strukturoppdeling i ettertid.

Innhold

Figurer

Kapittel 1

Innledning

Først i den senere tid har video kommet inn i dataverden. Dette har tidligere vært begrenset av spesielle krav til høy transport- og lagringskapasitet, relativt eksisterende teknologi. Digital video lagret på harddisk gir oss muligheter til å behandle videomateriale på nye måter sammenlignet med analog video. Først og fremst kommer dette av at vi har direkte tilgang til hver enkelt ramme i en videosekvens, i motsetning til analog video som er preget av linearitet ved søk og visning. Vi ser nå en utvikling som går mot utstrakt bruk av kontinuerlige medier innen datateknologi.

Ved Norsk Regnesentral (NR) har gruppen for Interaktive Medier (IMEDIA) drevet med forskning og utvikling innen multimedia informasjonssystemer, fokusert på integrasjon av ulike medier og informasjonsteknologi. IMEDIA jobber aktivt innen feltene interaktive teknologier, brukergrensesnitt, elektronisk publisering og nettbaserte informasjonstjenester. Ved Institutt for datateknikk og telematikk (IDT) ved NTH drives det også forskning innen multimedia. Databasegruppen jobber med prosjekter for beskrivelse av videofilm, søk i videomateriale og videodatabaser. Denne oppgaven er gjennomført i et samarbeid med disse to gruppene.

Målet med denne oppgaven er å vurdere funksjonalitet for et redigeringsverktøy for digital video. Verktøyet skal kunne inngå i et totalsystem som også omfatter videodatabase og verktøy for registrering og søking. Oppgavens formål er ikke å ta opp konkurransen med eksisterende, kommersielle redigeringsverktøy, men å sette redigering inn i en databasesammenheng.

Grunnlaget for oppgaven vil være kunnskap om redigering, redigeringsprosesser og -miljø, i tillegg til datateknologi. Det skal vurderes hva som kreves av et redigeringsverktøy generelt, fordi måten man redigerer på og virkemåten til eksisterende verktøy danner grunnlaget for å kunne si noe om redigering i databasesammenheng. Spørsmål som oppgaven skal søke å finne svar på er:

- Hva kreves av et redigeringsverktøy?
- Hvordan kan et redigeringsverktøy tilpasses et totalsystem for digital

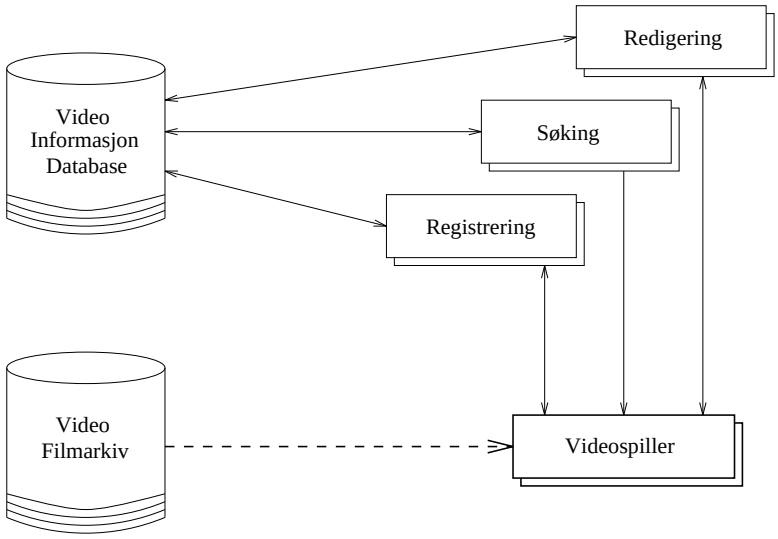
video?

- Hvordan påvirker et redigeringsverktøy totalsystemet?
- Hvilken nytte kan verktøyet ha av søking og registrering, og hva bør det inneholde av søking og registrering?
- Hvordan kan en videodatabase tilrettelegges for best mulig å kunne støtte et redigeringsverktøy?

Det skal utvikles en prototyp som demonstrerer ulike sider ved et redigeringsverktøy i databasesammenheng. Dette gjøres i tilknytning til LAVA-prosjektet.

1.1 LAVA – Leveranse av Video over ATM

CoMMedia[1] er et prosjekt startet etter initiativ fra Televerkets Forskningsinstitutt (TF) og UNINETT i 1992. Aktivitetene i dette prosjektet er knyttet til distribuerte multimedia-anvendelser. En bredere beskrivelse av CoMMedia finnes i [Hje94d]. Et forprosjekt til CoMMedia anbefalte en nasjonal satsning innen distribuert multimedia. Rapporten anbefaler at det gjennomføres prosjekter innen CoMMedia der brukere, universiteter og forskningsinstitutt deltar sammen.



Figur 1.1: Modulene i LAVA demonstrator

Et område som nevnes i denne rapporten er multimedia tjenester over høykapasitets datanett. Ut fra dette har man høsten 1994 startet et flerårig prosjekt med benevnelsen LAVA. Målet til dette prosjektet for 1994 var

[1] Cooperation, Communication and Multimedia

å lage en brukerorientert demonstrator for video på forespørsel over supernettet. De som samarbeidet om dette var SINTEF-DELAB, Institutt for datateknikk og telematikk (IDT), NORUT, Norsk Regnesentral (NR), NRK, TF og UNINETT. Prosjektet har fire hovedområder; brukergrensesnitt, nettverksløsning, lagring og redigeringsverktøy. Demonstratoren kan deles opp i seks moduler som er illustrert i figur 1.1.

Min oppgave faller inn i dette prosjektet som en del av det å utvikle et redigeringsverktøy. Et enkelt verktøy skal utvikles for å få en bedre forståelse for hvilke krav slike verktøy stiller til resten av systemet.

1.2 Terminologi

Innen fagfeltet data finnes mange engelske ord og uttrykk som vi ofte bruker i daglig tale. Det samme gjelder innen redigering av film og video. En del av disse ordene har gode ekvivalenter på norsk, mens mange oversettelser ofte kan oppleves som kunstige. Jeg vil i denne oppgaven hovedsaklig bruke norsk terminologi, men jeg vil anvende engelske ord og uttrykk der jeg finner det naturlig. Dette gjelder ord som enten er godt innarbeidet i daglig tale eller som jeg finner unaturlig å oversette. En oversikt over ord og uttrykk finnes i vedlegg A.

1.3 Oversikt over innholdet

Kapittel 2 presenterer en del teori rundt film, video og redigering generelt. Dette danner grunnlaget for å kunne vurdere redigering satt i en databasesammenheng. I kapittel 3 presenterer og vurderer jeg ulike datatekniske aspekter vedrørende digitale videosystemer. I tillegg blir digital video satt i sammenheng med NRK Nyhetsavdelingen og deres eksisterende produksjons- og redigeringsprosesser.

Kapittel 4 ser på eksisterende redigeringsverktøy og vurderer ulike sider ved slike verktøy. Her setter jeg også redigering inn i en databasesammenheng, og foreslår og vurderer ulike løsninger. Kapittel 5 beskriver konstruksjon og implementasjon av prototypen som er utviklet i denne oppgaven. I kapittel 6 kommer en vurdering av prototypen og de ulike løsningene. I dette kapitlet er også en konklusjon hvor jeg trekker frem de viktigste resultatene av arbeidet med denne oppgaven.

Det finnes et eget vedlegg til denne rapporten som inneholder en detaljert implementasjonsbeskrivelse i tillegg til kildekode.

Kapittel 2

Redigering av film og video

Redigering av film og video er et komplekst arbeid. Framgangsmåten og resultatet vil være avhengig av målet til redigereren og målgruppen for produktet. En romantisk kjærlighetsfilm, en dokumentarfilm fra Rwanda, dagsrevyen på NRK og en hjemmevideo fra sommerferien på Sørlandet vil alle ha ulike tilnærmingsmåter når det gjelder redigering. [Pud94] refererer til den sovjetiske filmskaperen Kuleshow, som uttalte seg om redigering:

> *Kuleshow holdt fast ved at film-kunst ikke startet ved opptakene av de ulike scenene, og skuespillernes prestasjoner. Dette er bare forberedelse av materialet. Film-kunst starter idet produsenten begynner å kombinere og sette sammen de ulike filmbitene.*

2.1 Film og filmterminologi

For å kunne diskutere metoder og muligheter rundt film og video er det viktig å forstå en del begreper og uttrykk.

[Merok93] ser nærmere på film og filmterminologi og foreslår en oppdeling i fire ulike elementer. I følge denne oppdelingen er filmens grunnelementer *ramme, innstilling, scene* og *sekvens*. Filmens minste enhet er et enkeltbilde. Dette kalles en **ramme** etter den engelske betegnelsen *frame*. En lengde uavbrutt film kalles en **innstilling**. Dette er en sekvens av rammer tatt opp med ett og samme kamera i ett kontinuerlig opptak. Den engelske betegnelsen er *shot* og vi kan også bruke den norske betegnelsen *tagning*. En innstilling blir ofte kalt den *minste filmiske enhet* i den betydning at det er den minste meningsbærende enheten i en film. En **scene** er sammensatt av innstillinger som hører sammen i tid og rom. Flere scener som tilsammen gir et meningsfyllt innhold kalles en **sekvens**. En film består dermed av innstillinger satt sammen til scener og sekvenser.

En film er altså en rekke rammer vist etter hverandre [Merok93]. At vi ikke oppfatter dette som en rekke enkeltbilder skyldes øyets treghet. Det er også en forutsetning at rammene vises i samme posisjon. I tillegg må rammene ha

en viss kontinuitet, dvs. at det kan ikke være en rekke tilfeldig valgte rammer, men rammer som tilsammen beskriver handlinger. Frekvensen rammene vises med kalles gjerne *rammerate* og er varierende etter hvilken standard som følges. Vanlig film viser 24 bilder/s, den europeiske fjernsynsstandarden PAL krever 25 bilder/s og den amerikanske fjernsynsstandarden NTSC har 30 bilder/s.

2.2 Redigeringsnivåer

Generelt kan vi dele redigering inn i tre ulike nivåer; *mekanisk*, *praktisk* og *artistisk* nivå (basert på [Mil90]). På mekanisk nivå kan vi si at redigering består av fire elementer.

1. Tidspunkt for bytte fra en innstilling til en annen.
2. Utførelsen av og hastigheten til overgangen.
3. Rekkefølgen av innstillinger og lengden av hver innstilling.
4. Kontinuiteten i bilde og lyd.

På praktisk nivå vil redigering dreie seg om å skape en "flytende bildekomposisjon". Det som er irrelevant kan fjernes, og man kan forkorte eller forlenge tiden som brukes til en handling. På praktisk nivå kan man også "overvinne" mekaniske problemer. Hvis en person beveger seg utenfor rekkevidden til kameraet, kan man flytte kameraet til en ny posisjon og senere redigere dette sammen slik at handlingen ikke avbrytes.

På artistisk nivå består redigering i å skape en handling, uttrykke meninger og fremkalle følelser. Her har man uendelig mange muligheter og innfallsvinkler.

- Redigering kan flytte interessepunktet til ulike deler av en scene, eksempelvis fremheve ting som skjer i bakgrunnen av en scene.
- Redigering kan utheve eller holde tilbake informasjon.
- Rekkefølgen og varigheten av innstillinger vil påvirke hvordan publikum reagerer.
- Redigering kan gi frihet til å bevege seg i tid og rom.
- Redigering kan skape forhold og relasjoner som har eller ikke har eksistert.
- Redigering kan endre innholdet i en handling for å skape eksempelvis humor, redsel, nervøsitet osv.

En redigerer har mange tilnærmingsmåter og må foreta mange valg. De valgene som blir tatt gir en direkte virkning på sluttproduktet. Gjennom redigering kan man skape et mer presist bilde av virkeligheten eller man kan skape en helt egen verden.

2.3 Redigeringsteknikker

Tre teknikker innen redigering er ofte brukt [Mil90].

- Kontinuitetsklipping
- Relasjonsklipping
- Dynamisk klipping

Kontinuitetsklipping består i å sette sammen bilder og lyd slik at det skaper en kontinuerlig handling. Teknikken har som formål å gi god kontinuitet i det som skal formidles. Kontinuitet i bilde og lyd, samt en flytende historie er viktig i de fleste redigeringssituasjoner. Innen filmredigering bruker man gjerne mye tid på å bestemme rekkefølgen av innstillinger, finne de beste tagningene osv. Ved fjernsynsredigering er ofte tiden en kritisk faktor, slik at det viktige her blir å få god kontinuasjon i handling eller presentasjon.

Relasjonsklipping består i å sette sammen ulike sekvenser for å skape en relasjon mellom disse, eksempelvis en sekvens hvor en person går mot et hus etterfulgt av en sekvens med den samme personen inne i et hus. Vi antar da at denne personen har gått inn i huset og er deretter i et rom i det samme huset. Teknikken kan brukes til å skape relasjoner som ikke har noen direkte sammenheng i virkeligheten. To ulike scener kan være filmet på forskjellige steder til forskjellig tid. Ved å sette disse sammen kan man skape en sammenheng mellom dem, slik at tilskueren oppfatter det som deler av samme virkelighet.

Sammensettingen av to ulike innstillinger skaper umiddelbart en sammenheng mellom disse. Dette foregår på to plan, nemlig fysisk og intellektuelt. Tilskuerens øye blir fysisk oppmerksom på at en endring har skjedd og tilskueren vil umiddelbart søke etter meningen bak de nye bildene. Responsen blir dermed en sammenheng mellom hva tilskueren har sett og opplevd i sammenheng med de nye inntrykkene. *Parallell klipping* setter sammen innstillinger fra to ulike steder, og kan illustrere to ting som skjer samtidig. Dette kan bygge opp spenning og dramatikk. *Kryss-klipping* bytter mellom ulike synsvinkler av samme handling, og brukes som oftest til å variere det visuelle inntrykket.

Dynamisk klipping består i å sette sammen innstillinger på en kreativ måte som kan gi tilskueren assosiasjoner til forhold som ikke uttrykkes direkte. Et eksempel er en innstilling med et knust vindu, etterfulgt av en innstilling av en gråtende gutt som kan gi assosiasjoner til en spesiell

årsak/virkning-sammenheng. Tilskueren oppfatter situasjonen som at gutten gråter fordi han har vært uheldig og knust vinduet. Dynamisk klipping kan skape dramatikk, følelser, stemninger og abstrakte idéer som ellers ikke kan uttrykkes direkte. Dette er en type sammenklipping som har som formål å uttrykke idéer som ikke finnes i kildematerialet.

2.4 Overganger mellom innstillinger

Det finnes flere ulike måter å gjøre overganger, eller transisjoner mellom innstillinger på.

Klipp er den enkleste transisjonen. Den gir en umiddelbar overgang som har en momentan effekt på tilskueren. Et klipp er svært effektfullt, noe som er denne teknikkens styrke. Denne teknikken, som de fleste andre, må brukes med forsiktighet. Et umotivert klipp kan ofte forvirre tilskueren.

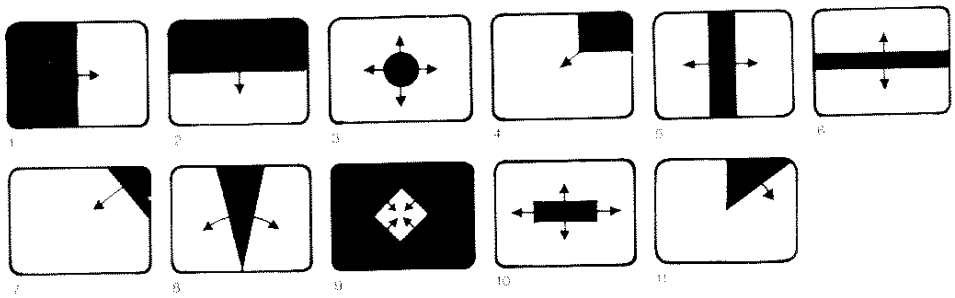
Fade vil si en glidende overgang fra eller til et helt ensfarget, ofte svart, bilde. *Fade-inn* gir en myk introduksjon til en handling. *Fade-ut* kan gi en rolig avslutning. Ved sammensetting av to innstillinger vil *fade-ut/inn* gi en pause i handlingen. Stemning og tempo bestemmes av fadenes relative hastighet og tiden mellom fade-ut og fade-inn. Denne effekten brukes ofte til å illustrere bevegelse i tid eller rom. Det er også mulig å bruke en kombinasjon av fade og klipp. Ved klipp/fade-inn skaper man en kraftig introduksjon til neste innstilling. Ved fade-ut/klipp-inn skapes et visuelt sjokk. Dette brukes gjerne mellom to statiske innstillinger, dvs. innstillinger som er visuelt stillestående.

Overtoning betyr at to innstillinger glir over i hverandre. Den ene innstillingen fades ut, mens den andre fades inn. Dette gir en rolig, flytende transisjon mellom to innstillinger. En hurtig overtoning (ca.1-2 sek. i følge [Mil90]) vil vanligvis gi inntrykk av at de to handlingene som mikses foregår samtidig. En sakte overtoning (ca.3-5 sek.) antyder endringer i tid eller rom. Generelt vil en overtoning gi en sammenlignende effekt. Den vil:

- Peke ut likheter eller ulikheter.
- Sammenligne tiden mellom de to innstillingene. Man kan bruke overtoning til å illustrere at tiden går.
- Sammenligne tid og posisjon. En serie med innstillinger kan ved hjelp av overtoning illustrere progresjon, f.eks. et hus som blir bygd eller gresset som gror.
- Relatere områder til hverandre.

Wipe er en visuell effekt for å gi pene transisjoner. Denne effekten brukes ofte innen reklamefilm. Den kan brukes til å skjule at to innstillinger egentlig ikke hører sammen. Wipe gir en effekt av å avdekke, finne frem eller skjule ulike elementer (f.eks. ulike gjenstander) i en film. To metoder av wipe

er *uncover* og *pushover*. Ved uncover wipe avdekkes den nye innstillingen gradvis. Ved pushover blir en av innstillingene dratt/dyttet ut av skjermen for å gjøre plass til den andre. Wipe kan utføres i mange forskjellige mønstre (se eksempler i figur 2.1).



Figur 2.1: Ulike wipe-mønstre (sakset fra [Mil90])

2.5 Digitale videoeffekter

Rammesekvenser kan digitaliseres og behandles av en datamaskin. Dermed har man direkte aksess til de enkelte delene av rammene. Avansert bildebehandling og datagrafikk kan utnyttes til å skape effekter som kan brukes i redigeringen. Dette kan være tradisjonelle transisjonseffekter som wipe og fade, eller det kan være andre spennende effekter.

Det finnes svært mange digitale videoeffekter som benyttes innenfor moderne redigering av film og video, og nye blir stadig funnet opp. De mange effektene har ulik betydning. Noen effekter har en direkte verdi i produksjonsfasen slik at de er med på å skape stemninger, følelser, forsterke handlingen osv. Andre effekter er spennende for å kunne finne nye presentasjonsmåter og skape et visuelt spennende uttrykk. Musikkvideoer inneholder ofte en hel mengde videoeffekter som er med på å skape et helt spesielt visuelt uttrykk.

Jeg vil her beskrive noen digitale videoeffekter, for å gi et intrykk av mangfoldet innen dette området. Dette er eksempler tatt fra [Mil90].

Gjennombrudd/smelting Et bilde løser seg opp og glir inn i et annet bilde.

Terning Roterende terning med ulike bilder på hver side.

Sylinder Et bilde danner en sylinder.

Ekko Flere lag dannes av et eneste bilde.

Eksplosjon Bildet brytes opp i flere fragmenter som forsvinner fra hverandre.

Folding Venstre og høyre bildekant foldes innover til en vertikal strek. Bildet kan også foldes utover.

Bildeplassering Plassering av et bilde hvor som helst i en ramme.

Forstørrelsesglass Forstørrelse av en del av et bilde.

Mosaikk Bildet deles opp i små, like store firkanter. For hver firkant tar man gjennomsnittet av fargetonene fra det originale bildet. Dette brukes ofte til å skjule identiteten til mennesker.

Rotasjon Bildet roteres rundt aksene i et 3-D rom.

2.6 Temporale effekter

Muligheten for å spille av film/video i ulike hastigheter er en effekt som ofte benyttes. Dette gjelder både film, analog og digital video.

Hurtigspoling er avspilling med en hastighet høyere enn vanlig avspillingshastighet. Dette kan benyttes sette opp farten på bevegelser, skape komiske effekter, korte ned tiden til en handling og hurtig illustrere utviklingen til en normalt treg prosess (f.eks. blomster som gror) osv.

Sakte avspilling, eller slow motion, vil si avspilling med en hastighet lavere enn vanlig avspillingshastighet. Effekten kan brukes til f.eks. å se nøye på hurtige bevegelser (f.eks. muskelbevegelser til en sprinter). I film oppnås sakte avspilling ved å foreta opptakene med mer enn 24 rammer/s for senere å avspille med normal hastighet. Sakte avspilling for video oppnås ved å senke hastigheten til videospilleren, slik at den viser færre rammer pr. sekund.

Frysing av bildet betyr at en enkel ramme spilles av over et visst tidsrom. Dette kan gi et inntrykk av å forhindre avslutningen til en handling.

Revers avspilling betyr at rammene avspilles i motsatt rekkefølge. Dette kan skape en komisk eller magisk effekt. Eksempel på dette er en knust bygning som gjenopprettes eller en julegave som pakkes inn igjen.

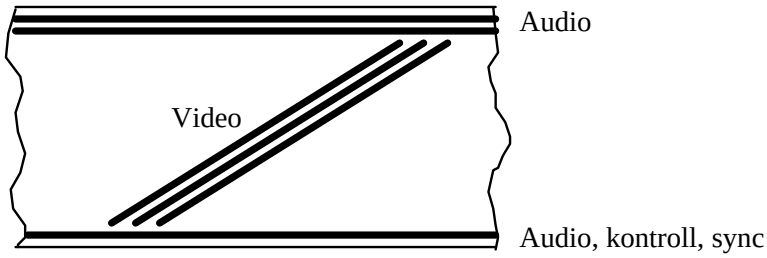
Syklisk avspilling er en bestemt sekvens som repeteres. Dette kan fremheve en handling eller også skape komiske effekter.

2.7 Redigering av videobånd

Videobåndredigering skiller seg vesentlig fra redigering av film. En filmredigerer jobber med en klippekopi. Denne kopien klippes fysisk, og de delene som skal være med i filmen beholdes og skjøtes sammen igjen. Filmredigering kan betegnes som **ikke-lineær**. Det vil si at man kan klippe inn nye sekvenser hvor som helst i filmen.

Bildeinformasjonen på et videobånd lagres på skrå over båndet (se figur 2.2). Både avspillings- og innspillingshodene sitter på en roterende *tønne*. Dette er

en teknikk som kalles *helical scanning* og blir brukt i alle moderne videobånd-systemer [Mil90]. Teknikken gjør at man ikke fysisk kan klippe et videobånd. Ved klipping vil man skjære over flere videospor. Redigering av videobånd må derfor skje via kopiering. En videoredigerer tar råmaterialet og kopierer de aktuelle innstillinger og sekvenser inn på et master-bånd. Redigering av videobånd kan betegnes som **lineær**. Alt materiale som skal legges til i en video må legges til etter det som allerede er redigert sammen.



Figur 2.2: Eksempel på strukturen i et videobånd

2.7.1 Redigeringsmetoder

For redigering av videobånd kan kan man benytte ulike metoder. Man kan *redigere med videokameraet* mens man tar opp. Dette er en usikker metode som krever stor nøyaktighet i planlegging og opptak. Metoden har den fordelen at den sikrer maksimal bildekvalitet på produktet i og med at det ikke skjer noen form for kopiering.

En annen mulighet er *manuell redigering* ved avspilling. Man trenger da en master-videospiller og en eller flere kilde-videospillere. Råmaterialet spilles over fra kilde-spillerne til master-spilleren. Redigereren plukker ut klipp fra kilde-spillerne, spoler slik at både kilde- og master-spilleren står i riktig posisjon, og kopierer det aktuelle opptaket. Dette krever mye spoling og gjennomsyn av råmaterialet, dvs. at denne metoden tar mye tid.

Den tredje måten er *datastyrt redigering*. Dette er den vanligste metoden å redigere videobånd på i dag. Det kommer stadig ny programvare både for profesjonelt bruk og for hjemmebruk/amatørvideo. Prinsippet her er at sammensetting av videosekvenser blir styrt av en datamaskin. Dette er egentlig den samme som forrige metode bortsett fra at her styrer datamaskinen de involverte videospillerne. Datastyrt redigering gjør at klippene blir veldig nøyaktige og hele prosessen forenkles.

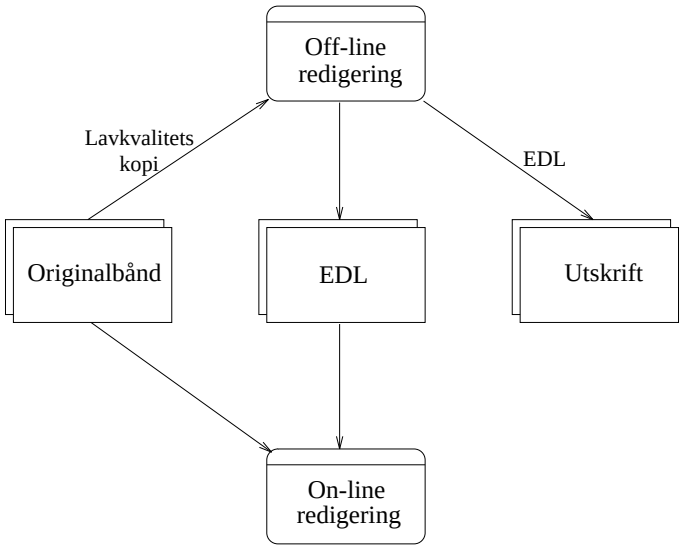
2.7.2 Generasjonstap

Et problem med redigering av videobånd er at man må kopiere materialet fra bånd til bånd. For analoge videobånd vil hver kopiering redusere kvaliteten. Direkte opptak kalles *første generasjon* opptak [Mil90]. Kopierer vi denne får

vi *andre generasjon* opptak. Kvaliteten vil her være noe redusert i forhold til første generasjon. Det er introdusert mer bakgrunnsstøy, større uklarhet i bildet og ulike forstyrrelser. Et *tredje generasjons* opptak (dvs. kopi av andre generasjon) vil ha ytterligere redusert kvalitet. Dermed vil man søke å unngå mye kopiering i redigeringsprosessen, helst ikke ut over tredje generasjon. Med digitale videobånd vil man unngå at kvaliteten reduseres ved kopiering. Digital BETACAM [Sony] fra Sony er et slikt digitalt videosystem som nylig er tatt i bruk av blant annet NRK.

2.7.3 On-line og off-line

Redigering kan utføres enten *on-line* eller *off-line*. On-line redigering er direkte redigering på råmaterialet. On-line redigeringsutstyr er fullkvalitets utstyr som skal levere et tilfredsstillende sluttresultat. Hva som er tilfredsstillende kan være forskjellig fra bruker til bruker. On-line utstyr er ofte veldig dyrt og redigering med dette vil gjerne legge bånd på store ressurser i en organisasjon.



Figur 2.3: Sammenhengen mellom on-line og off-line

Off-line redigering skjer ikke med originalmaterialet, men med en kopi av dette. Kopien kan være videobånd av mindre format, eller det kan være digitalisert video. Denne kopien inneholder tidskode (se avsnitt 2.8) og båndnummer for opptakene. Redigeringen utføres på denne kopien og resultatet blir en liste med redigeringspunkt. Dette ender i en editliste (EDL - Edit Decision List) som brukes til den endelige redigeringen av originalmaterialet (se avsnitt 2.9).

Off-line utstyr er ofte billigere enn on-line utstyr. Off-line redigering kan dermed gjøres med mindre ressurser og er ofte mer effektivt enn on-line

redigering. Effektiviteten til off-line redigering ligger i det at man sparer tid og ressurser.

2.8 Tidskode

Et system som gjør at sekvenser i videomaterialet kan identifiseres på en enkel og grei måte er tidskoding. SMPTE[1] er en tidskodestandard som identifiserer hvert enkelt bilde i en film. SMPTE representeres tekstlig som `tt:mm:ss:rr`, hvor `tt` = timer, `mm` = minutter, `ss` = sekunder og `rr` = rammer. Kodingen kan enten baseres på det aktuelle tidspunktet på dagen hvor opptaket er gjort, eller den kan baseres på avstanden i tid fra begynnelsen av båndet. SMPTE kodes på båndet med enten *longitudinal time code*(LTC) eller *vertical interval time code*(VITC) [Mil90]. LTC spilles inn på et lineært lyd-spor. Denne kodingen kan bare leses av ved hastigheter nær normal avspillingshastighet. VITC spilles inn i videosignalet, mellom rammene. Det er denne kodingen som blir foretrukket til redigering av videobånd.

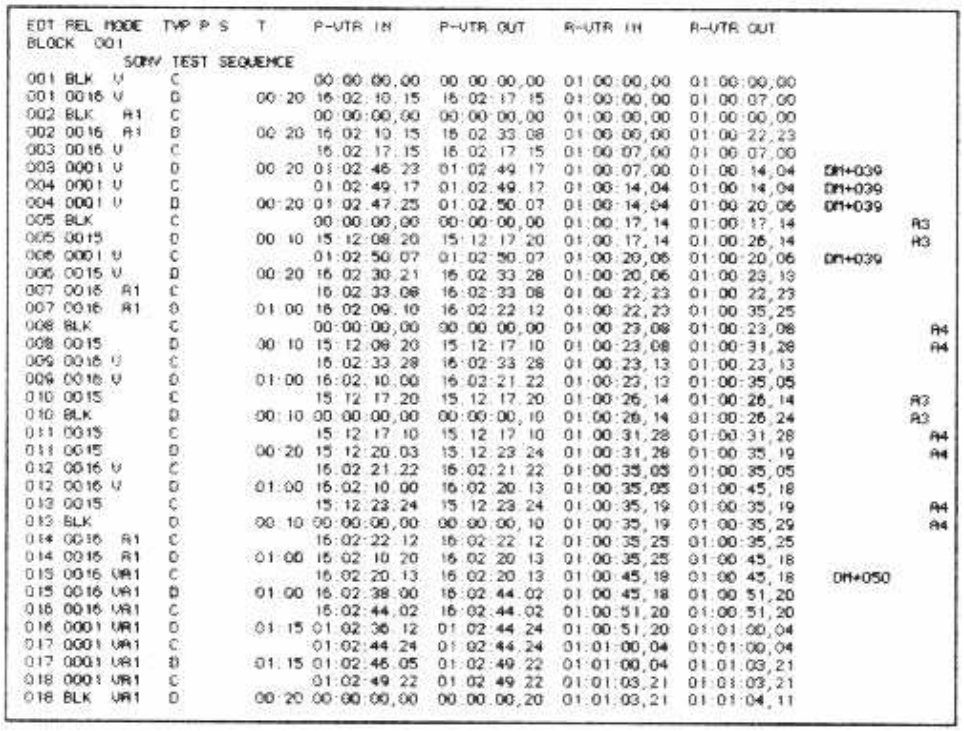
2.9 Editliste

Resultatet av en off-line redigering er en video med ikke fullgod bildekvalitet og en tilhørende editliste. Editlisten er en beskrivelse av den redigeringen som er utført. Beskrivelsen består i hvilke sekvenser som er benyttet, fra hvilke originalbånd de er hentet, tidskoder for de ulike sekvensene, effekter osv. Editlisten betegnes gjerne som EDL (Edit Decision List).

Meningen med en editliste er at man skal kunne ta denne og "*mate*" den inn i et on-line utstyr for å få utført en fullkvalitets redigering av originalmaterialet. For at vi skal kunne gjøre dette er vi avhengig av at editlisten har et format som er kompatibelt med on-line utstyret. Dette betyr at editlisten må inneholde de koder som er nødvendige for at on-line utstyret skal kunne utføre den redigeringen som er utført off-line. Det finnes ingen entydig standard for editlister. Leverandører har sine egne "*standarder*". Eksempler på proprietære formater for editlister er *CMX, AMPEX, GVG (Grass Valley Group)* og *Sony*. Det kan også være kompatibilitetsproblemer mellom de enkelte leverandørers editlister og redigeringsutstyret. Dette er sannsynligvis et område som vil utvikle seg etterhvert som teknologien modnes [White94].

Hvis vi, som et eksempel, ser på en editliste fra Sony, figur 2.4, så består den av en linje for hver videosekvens som produksjonen inneholder. Hver linje inneholder sekvensnummer, båndnummer, informasjon om sekvensen er lyd eller video, type overgangseffekt og eventuelt tidsparameter for denne, inn- og utpunkt på originalbåndet og inn- og utpunkt på masteren. Dette er informasjon som definerer den redigeringen som er utført og som kan

[1] Definert av The Society of Motion Picture and Television Engineers.



EDIT REL MODE TYP P S T P-VTR IN P-VTR OUT R-VTR IN R-VTR OUT
BLOCK 001
SONY TEST SEQUENCE

001	BLK	V	C		00:00:00,00	00:00:00,00	01:00:00,00	01:00:00,00			
001	0016	V	D	00:20	16:02:10:15	16:02:17:15	01:00:00,00	01:00:07,00			
002	BLK	A1	C		00:00:00,00	00:00:00,00	01:00:00,00	01:00:00,00			
002	0016	A1	D	00:20	16:02:10:15	16:02:33:08	01:00:00,00	01:00:22,23			
003	0016	V	C		16:02:17:15	16:02:17:15	01:00:07,00	01:00:07,00			
003	0001	V	D	00:20	01:02:46:23	01:02:49:17	01:00:07,00	01:00:14,04	DM+039		
004	0001	V	C		01:02:49:17	01:02:49:17	01:00:14,04	01:00:14,04	DM+039		
004	0001	V	D	00:20	01:02:47:25	01:02:50:07	01:00:14,04	01:00:20,06	DM+039		
005	BLK		C		00:00:00,00	00:00:00,00	01:00:17,14	01:00:17,14			A3
005	0015		D	00:10	15:12:08:20	15:12:17:20	01:00:17,14	01:00:26,14			A3
006	0001	V	C		01:02:50:07	01:02:50:07	01:00:20,06	01:00:20,06	DM+039		
006	0015	V	D	00:20	16:02:30:21	16:02:33:28	01:00:20,06	01:00:23,13			
007	0016	A1	C		16:02:33:08	16:02:33:08	01:00:22,23	01:00:22,23			
007	0016	A1	D	01:00	16:02:09:10	16:02:22:12	01:00:22,23	01:00:35,25			
008	BLK		C		00:00:00,00	00:00:00,00	01:00:23,08	01:00:23,08			A4
008	0015		D	00:10	15:12:08:20	15:12:17:10	01:00:23,08	01:00:31,28			A4
009	0016	V	C		16:02:33:28	16:02:33:28	01:00:23,13	01:00:23,13			
009	0016	V	D	01:00	16:02:10:00	16:02:21:22	01:00:23,13	01:00:35,05			
010	0015		C		15:12:17:20	15:12:17:20	01:00:26,14	01:00:26,14			A3
010	BLK		D	00:10	00:00:00,00	00:00:00,10	01:00:26,14	01:00:26,24			A3
011	0015		C		15:12:17:10	15:12:17:10	01:00:31,28	01:00:31,28			A4
011	0015		D	00:20	15:12:20:03	15:12:23:24	01:00:31,28	01:00:35,19			A4
012	0016	V	C		16:02:21:22	16:02:21:22	01:00:35,05	01:00:35,05			
012	0016	V	D	01:00	16:02:10:00	16:02:20:13	01:00:35,05	01:00:45,18			
013	0015		C		15:12:23:24	15:12:23:24	01:00:35,19	01:00:35,19			A4
013	BLK		D	00:10	00:00:00,00	00:00:00,10	01:00:35,19	01:00:35,29			A4
014	0016	A1	C		16:02:22:12	16:02:22:12	01:00:35,25	01:00:35,25			
014	0016	A1	D	01:00	16:02:10:20	16:02:20:13	01:00:35,25	01:00:45,18			
015	0016	VA1	C		16:02:20:13	16:02:20:13	01:00:45,18	01:00:45,18	DM+050		
015	0016	VA1	D	01:00	16:02:38:00	16:02:44:02	01:00:45,18	01:00:51,20			
016	0016	VA1	C		16:02:44:02	16:02:44:02	01:00:51,20	01:00:51,20			
016	0001	VA1	D	01:15	01:02:36:12	01:02:44:24	01:00:51,20	01:01:00,04			
017	0001	VA1	C		01:02:44:24	01:02:44:24	01:01:00,04	01:01:00,04			
017	0001	VA1	D	01:15	01:02:46:05	01:02:49:22	01:01:00,04	01:01:03,21			
018	0001	VA1	C		01:02:49:22	01:02:49:22	01:01:03,21	01:01:03,21			
018	BLK	VA1	D	00:20	00:00:00,00	00:00:00,20	01:01:03,21	01:01:04,11			

Figur 2.4: Eksempel på editliste fra Sony (sakset fra [White94])

gjenskape produksjonen fra originalbåndene.

Det er helt avgjørende for at redigeringen skal bli vellykket at originalbåndene har tidskode som samsvarer med den tidskoden som benyttes ved off-line redigering. Dette er en forutsetning for at editlisten skal ha sin funksjon, og krever grundige forberedelser i forkant av off-line redigeringen. Hvis off-line redigeringen utføres på et digitalt redigeringsverktøy må tidskode registreres ved digitalisering. En annen avgjørende faktor er samsvar mellom båndnummer. Editlisten må inneholde de korrekte båndnummer i forhold til originalbåndene. Dette må også kontrolleres/registreres før redigeringen finner sted. En editliste med ukorrekte båndnummer er verdiløs. I følge [White94] kan editlisten være et perfekt eksempel på "*garbage in, garbage out*".

Kapittel 3

Digitale videosystemer

3.1 Multimedia

Multimedia er et begrep som samler mange informasjonskilder som grafikk, animasjon, bilder, lyd og video til et bredt spekter av applikasjoner. Datateknologi, kommunikasjon og kringkasting ser ut til å være de teknologiske drivkreftene innen utviklingen av multimediasystemer. Multimediasystemer kan deles inn i to grupper. Den ene er såkalt *stand-alone*-systemer, dvs. systemer som opererer uten tilknytning til omverden. Den andre gruppen er distribuerte systemer, hvor man har større muligheter til blant annet å utveksle informasjon og applikasjoner. Multimedia informasjonssystemer, datastøttet samarbeid, konferansesystemer, opplæringssystemer og multimedia-på-forespørsel (on-demand) er noen områder som er i stor utvikling.

Det mest karakteristiske ved multimediasystemer er innholdet av kontinuerlige medier som lyd, video og animasjon. Disse mediene krever kontinuerlig dataoverføring over lengre perioder, og dette setter store krav til både maskinvaren, programvaren og nettet som brukes i overføringene. Synkronisering, stor lagringskapasitet, stor overføringskapasitet, hurtig aksess og effektive I/O-operasjoner er noen av kravene [Furht94]. I tillegg trenger man spesielle indekserings- og gjenfinningsmetoder for behandling av multimediainformasjonen.

3.1.1 Digital video

Digital video er en viktig del av multimedia. Med begrepet *digital video* beskriver vi lagringsformatet, men det er også av betydning hvilket lagringmedium vi benytter. Ved å få video inn i datamaskinen, åpner det seg nye muligheter for å behandle videomateriale. Den store fordelen med video lagret på en harddisk er at man får vilkårlig aksess til hver enkelt ramme. Dette gir nye muligheter til presentasjon, søk, redigering og bildebehandling, i forhold til analog video. Det finnes også systemer for å lagre

digital video på bånd [Sony], men dette gir oss fremdeles den begrensningen at tilgangen til materialet er sekvensiell. Fordeler med et digitalt format er:

- Man unnår generasjonstap. Kopier får samme kvalitet som originalinnspillingen.
- Bedre gjengivelse av video. Man er ikke avhengig av finjusteringer av bildet.
- God bildekvalitet ved spoling.
- Man unngår hvit støy i lyd og bilde.
- Muligheter for nye grensesnitt gjennom datamaskinen.
- Digital video på en datamaskin kan integreres med andre verktøy og kan dermed benyttes i dokumenter eller presentasjoner.

Alt dette gjør at digital video åpner nye muligheter. Men det stiller også store krav til behandlingen av videodataene. Digital video tar stor lagringsplass, og krever også høy overføringskapasitet. Den europeiske fjernsynsstandarden PAL definerer en bildeoppløsning på 720×576 piksler. For en fargekoding med 24 bit pr. piksel blir dette ca. 1.2 MByte pr. ramme. Med 25 rammer/s krever dette en dataoverføring på omtrent 30 MByte i sekundet. Dette vil igjen si at vi trenger rundt 54 GByte for å kunne lagre en 30 minutters Dagsrevy-sending med full kvalitet. I tillegg kommer lagrings- og overføringskapasitet for lyden.

3.1.2 Komprimeringsmetoder

Lyd, video og stillbilder gir store mengder data i digitale multimediaapplikasjoner. Derfor trenger vi ulike komprimeringsmetoder for å overkomme store lagringskapasitetskrav, relativt trege lagringsmedier som ikke kan overføre multimediadata i sanntid og båndbredden i dagens nettverk som ikke tillater sanntids video-overføringer av full kvalitet [Furht94].

Komprimering av bilde data bygger på tre ulike typer redundans i et bilde. Dette er *kodings*redundans, *interpikselredundans* og *psykovisuell* redundans [Gon92]. Et digitalt bilde representeres som et sett piksler, hvorav hver piksel er representert ved et antall bit. Ved å benytte et lite antall bit til å kode de pikselverdiene som forekommer mest i bildet, kan vi redusere den totale datamengden som skal til for å representere et bilde digitalt. Dermed har vi utnyttet den *kodingsredundans* som finnes i et digitalisert bilde, og kan representere et bildet med en mindre datamengde enn ved å kode hver pikselverdi med et fast antall bit. Et annet faktum er at to piksler som er naboer ofte har pikselverdier som er relativt like. Dette kan utnyttes ved f.eks. å kode differansen mellom to pikselverdier, i stedet for full hver verdi for seg selv. Differansen mellom to nabopiksler vil med stor sannsynlighet kunne kodes med færre bit enn hele pikselverdien. Dette kalles

gjerne *interpikselredundans*. *Psykovisuell redundans* betyr at øyet er mer følsomt for små variasjoner i lys og kontraster enn små variasjoner i farger. Dermed kan vi kode farger med mindre nøyaktighet enn intensiteten i bildet.

Komprimeringsmetodene som er spesielt utviklet for digital video, utnytter også en *temporal redundans*. Temporal redundans betyr at det ofte er kun små variasjoner mellom to påfølgende rammer i en video. Dette gjør at videorammer kan kodes med referanser til de nærliggende rammene i stedet for koding som selvstendige rammer.

Vi kan gruppere komprimeringsmetodene i *tapsfrie* (lossless) og *informasjonstapende* (lossy). En tapsfri metode vil kunne gjengi eksakt det originale materialet. De informasjonstapende metodene vil ikke kunne gjengi originalmaterialet eksakt, men gi en representasjon av dette med mindre nøyaktighet. De informasjonstapende metodene vil derimot gi mindre komprimeringsfaktor[1], og er derfor de mest brukte når det gjelder bilde- og videokomprimering. Informasjonstapende metoder utnytter i stor grad psykovisuell redundans. Jeg vil her presentere noen komprimeringmetoder som blir benyttet til komprimering av video.

JPEG (Joint Photographic Expert Group) er en standard for komprimering av stillbilder, men har også blitt en populær standard for komprimering av film [Ske93]. En nyttig egenskap ved JPEG er at bildekvaliteten kan bestemmes ved å justere komprimeringsparametrene. Dette betyr at man kan velge forholdet mellom lagringskapasitet og bildekvalitet avhengig av applikasjonen som skal benytte JPEG. En annen viktig egenskap er at man kan få en raskere dekoding mot en dårligere bildekvalitet. Dette er viktig i applikasjoner som krever rask dekomprimering og som tåler at bildekvaliteten ikke er optimal. Dette kan i mange tilfeller gjelde digital video. JPEG definerer fire ulike komprimeringsmodi hvorav en er tapsfri. Den mest brukte modus er den sekvensielle DCT-baserte[2] modus. En inngående beskrivelse av JPEG komprimering finnes blant annet i [Taw94].

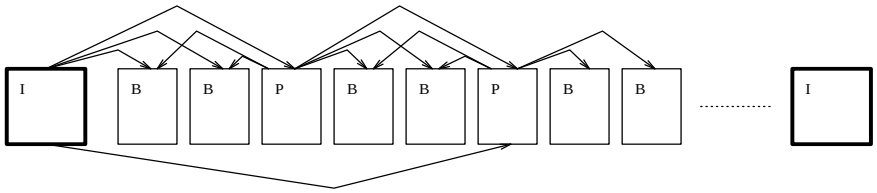
Dataraten for JPEG er avhengig av den ønskede bildekvaliteten. JPEG koder hvert bilde for seg, noe som gjør JPEG mer fleksibel enn andre kodingsstandarder. En slik uavhengig kodet ramme kalles en *Intra-ramme*, og video i JPEG format er en sekvens av Intra-rammer. Video kodet i JPEG kalles gjerne *Motion JPEG (MJPEG)*, men det finnes ingen standard filformat til dette. Som oftest utnytter MJPEG det faktum at man trenger å lese kodingsparametrene kun en gang for en sekvens av likt kodete bilder.

MPEG (Motion Picture Experts Group) er en standard for komprimering av film-sekvenser. MPEG utnytter at forskjellen mellom to påfølgende rammer i en video-sekvens ofte er veldig liten. MPEG definerer en algoritme for prediksjon av neste ramme i en sekvens. Dersom den virkelige rammen er ulik prediksjonen, blir differansen lagret som den nye rammen. En rammesekvens består av følgende typer rammer:

[1] Komprimeringsfaktor = komprimerte data/ukomprimerte data
[2] Discrete Cosinus Transform

- **Intra-rammer (I) :** Uavhengig kodede rammer (som JPEG rammer). Rammene er ikke avhengige av andre rammer.
- **Predikterte rammer (P) :** Predikterte rammer kodet med referanser til forrige I- eller P-ramme.
- **Bidireksjonale rammer (B) :** Predikterte rammer kodet med referanse til forrige og neste P-ramme.

Vanligvis er det 2 B-rammer mellom hver P-ramme i en sekvens. I-rammer finnes med variable intervaller. Som regel er det 9 P-rammer mellom to påfølgende I-rammer. En MPEG rammesekvens er illustrert i figur 3.1. MPEG definerer også en standard for koding av lyd, som kalles **MPEG-AUDIO**.

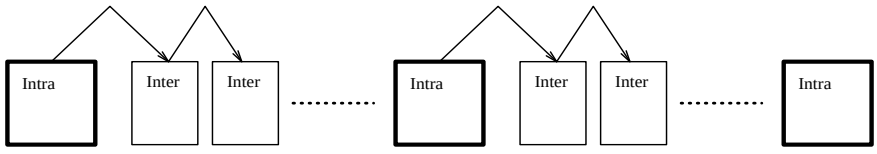


Figur 3.1: En sekvens av MPEG rammer

H.261(px64) er en standard som i likhet med MPEG er konstruert for videokomprimering. Standarden er optimalisert for videobasert telekommunikasjon ved en bitrate på et multippel av 64 kbit/s [Furht94]. H.261 koder differansen mellom to påfølgende rammer. En rammesekvens består av to typer rammer:

- **Intra-rammer :** Uavhengig kodede rammer.
- **Inter-rammer :** Rammer kodet som en subtraksjon av denne ramme fra forrige ramme.

I figur 3.2 er en H.261 rammesekvens illustrert.



Figur 3.2: En sekvens av H.261 rammer

3.1.3 Synkronisering

Multimedia systemer inneholder både lyd og bilder som har et bestemt forhold til hverandre. Digital video består av en sekvens av bilder med tilhørende lyd, og det er derfor viktig at bilde og lyd er synkronisert.

Vi kan dele synkroniseringsmetoder inn i *parallelle* og *serielle*. Seriell synkronisering behandler dataraten i en enkel datastrøm. For video er det viktig at bildene blir avspilt med en konstant hastighet. Dette betyr at bildene synkroniseres serielt. Parallell synkronisering trenger vi når vi har flere datastrømmer som skal synkroniseres i forhold til hverandre, f.eks. bilde- og lydstrøm. I de fleste multimediasystemer trenger vi både seriell og parallell synkronisering. I noen tilfeller kan også parallell synkronisering gjøres om til seriell ved at ulike datastrømmer kan lagres i en felles fil.

Vi skiller også mellom *punktsynkronisering* og *kontinuerlig synkronisering*. Punktsynkronisering krever at et punkt i en datastrøm sammenfaller med et punkt i en annen strøm. Kontinuerlig synkronisering er synkronisering av to aktiviteter over et visst tidsrom. Synkronisering kan utføres med ulik presisjon, avhengig av hva applikasjonen krever. F.eks. er det ved synkronisering av et intervju viktig at lyden synkroniseres med leppebevegelsene. Dette vil kreve en presisjon på bedre enn 0.1 sekunder [Ske93].

3.2 Totalsystem for digital video

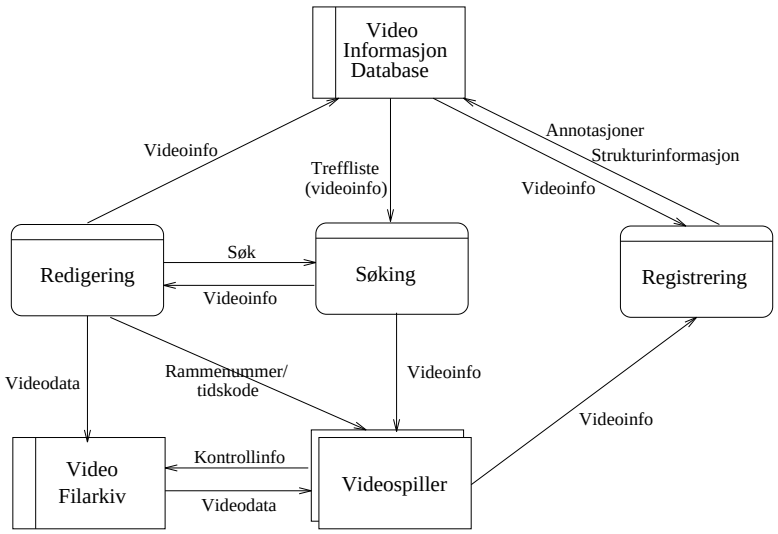
Fordelene ved å ha video på et digitalt format er beskrevet i avsnitt 3.1.1. Vi ønsker å utnytte disse fordelene til lagring, søk, gjenfinning og gjenbruk av videomateriale. Til dette kan vi tenke oss et totalsystem som støtter de ulike sidene ved behandling av videoinformasjon. Et slikt system kan bestå av fire elementer:

- Videodatabase
- Registrering
- Søking
- Redigering

Et slikt totalsystem er illustrert i figur 3.3. Demonstratoren som utvikles i LAVA-prosjektet (se avsnitt 1.1) har som hensikt å prøve ut de ulike sidene ved et slikt totalsystem.

3.2.1 Videodatabase

Brukere av en videodatabase vil ønske direkte aksess til den videoinformasjonen som er lagret der. Dette kan ha ulike årsaker. Det kan være for gjenbruk



Figur 3.3: Et totalsystem for digital video

av tidligere lagret video, søk etter bakgrunnsmateriale for en ny produksjon eller i en læresituasjon, forskning på video eller film, en interaktiv videotjeneste osv. For å kunne støtte dette må en videodatabase oppfylle minst tre betingelser. Den må kunne lagre videomateriale, levere video og den må ha strukturerte beskrivelser av videoinformasjonen som muliggjør søking og gjenfinning.

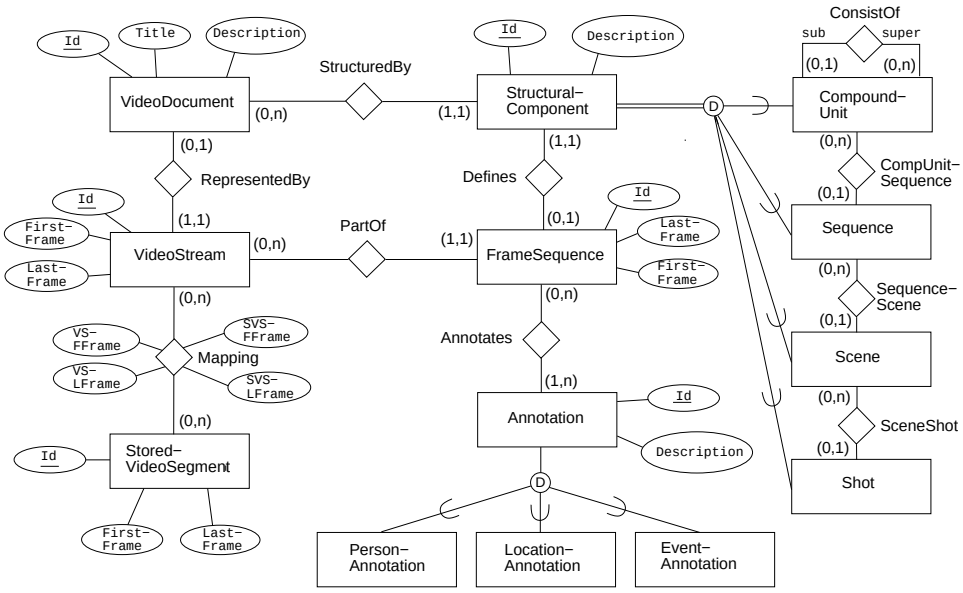
En slik database er beskrevet i [Hje94][3]. I denne databasen er videodata lagret i kontinuerlige grupper av rammer som kalles **StoredVideoSegment** (se figur 3.4). Et **VideoDocument** er representert ved en **VideoStream** som igjen er mappet ned til en eller flere StoredVideoSegments. Denne mappingen gjør at en VideoStream kan betraktes som *en* kontinuerlig strøm av video selv om den består av flere fysisk adskilte segmenter.

Struktureringen støtter tanken om deling og gjenbruk av videomateriale. Et StoredVideoSegment kan brukes i flere VideoStreams og kan dermed også inngå i flere VideoDocuments. Et viktig poeng ved denne organiseringen er at den gir minst mulig replisering av data. Hvis en videosekvens er inneholdt i to videodokumenter trenger den bare lagres på *en* fysisk plass i lageret og kan mappes slik at den tilhører begge dokumentene.

Beskrivelsene av videomaterialet er strukturert på to ulike nivåer, *strukturindeksering* og *tematisk indeksering*.

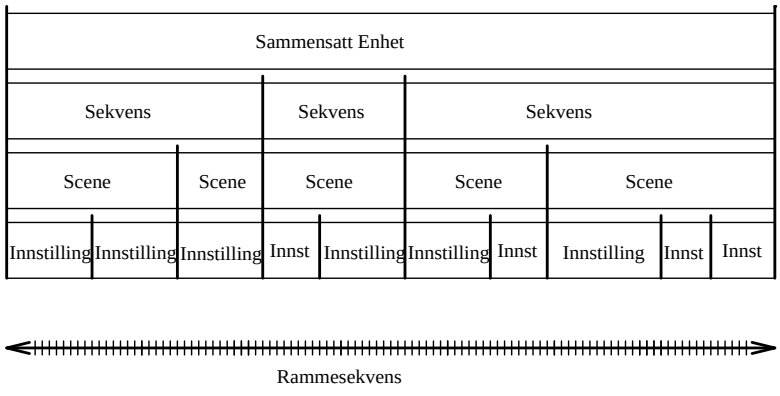
Strukturindekseringen er inspirert av filmteori og deler en video inn i sine enkelte strukturkomponenter. Dette bygges rundt en **StructuralComponent** som assosieres med en rammesekvens. En slik strukturkomponent spesialiseres så til en **CompoundUnit, Sequence, Scene** eller **Shot**.

[3] I skrivende stund kalles denne databasen VideoSTAR.



Figur 3.4: En videodatabase-modell beskrevet i EER-notasjon (sakset fra [Hje94])

Den innbyrdes strukturen mellom disse komponentene er slik at en CompoundUnit kan bestå av en eller flere CompoundUnits eller et antall Sequences. En Sequence kan igjen bestå av et antall Scenes som igjen består av et antall Shots. Se også figur 3.5.



Figur 3.5: Innbyrdes struktur mellom struktur-komponentene

Strukturindekseringen sier ikke alt om innholdet i vieomaterialet. Derfor behøver vi også tematisk indeksering. Den tematiske indekseringen forsøker å strukturere selve innholdet i en video. Vi kan beskrive hva en video inneholder ved å knytte tekstlige beskrivelser til bestemte rammesekvenser. Vi kan på denne måten få en oversikt over hva videoen inneholder, hvor de ulike personer vises, hvor ulike hendelser opptrer osv. Modellen baseres på

en **Annotation** som assosieres med en rammesekvens. En Annotation kan være av tre ulike typer, nemlig **PersonAnnotation, LocationAnnotation** eller **EventAnnotation**. Dette vil til sammen gi en viss beskrivelse av innholdet i videomaterialet. Det kan være applikasjonsavhengig hvilke annotasjonstyper som er ønskelige. Det kan være tilfeller hvor disse tre annotasjonstypene ikke er tilstrekkelig til å beskrive fullt ut en bestemt type film/video.

Med denne videodatabasen har vi en strukturering av videomaterialet som støtter avansert søking, deling og gjenbruk. Hvor godt videomaterialet blir beskrevet er avhengig av den som gjør annotasjonene. Annoteringen bestemmer hvor stor nytte vi vil ha av databasen senere. Som et eksempel kan vi ha en reportasje om kongen på besøk i Hammerfest, hvor vi også har innslag med midnattsol. Den som gjør annotasjonene synes kanskje ikke at midnattsola er spesielt viktig i denne reportasjen og utelater den fra annoteringen. Derimot vil en annen bruker, som skal lage en reportasje om nettopp midnattsola, synes at dette er en viktig sekvens. Hvis ikke annotasjonen er gjort, kan ikke denne brukeren lett finne igjen det spesielle innslaget uten å gå gjennom videomaterialet på nytt.

Som påpekt av forfatterne er ikke modellen nødvendigvis idéell for alle applikasjoner. Noen applikasjoner vil ha behov for å utvide modellen med f.eks. flere annotasjonstyper, mens andre bare utnytter deler av modellen.

3.2.2 Innlasting av videodata

En viktig oppgave som ikke er vist i figur 3.3 er innlasting av videodata i systemet. I dag benyttes stort sett videokamera med analog tape som opptaksmedium. Dette betyr at video må digitaliseres ved innlasting. Digitaliseringen er en prosess som i de fleste tilfeller vil kreve store ressurser i form av tid og maskinkapasitet. Den hurtighet og kvalitet som digitaliseringen kan utføres med er viktige forutsetninger for hvor stor glede vi vil ha av et slikt totalsystem. Det er en stadig utvikling også i opptaksutstyr. Jeg har i avsnitt 2.7.2 nevnt DigitalBETACAM som er et opptaks- og avspillingssystem for digital tape. Det jobbes også med å få harddiskbaserte kameraer ut på markedet [Ler94]. Kameraer som kan ta opp video direkte til harddisk vil effektivisere innlasting av video i databasen. Video tatt opp på et harddiskbasert kamera vil kunne overføres direkte med høy hastighet og uten tap av kvalitet.

3.2.3 Registrering

En *registreringsprosess* er nødvendig for å få lagt inn informasjon om de videoene som finnes i databasen. Dette er ofte en tidkrevende og omstendelig prosess. I dagens fjernsynsarkiv gjøres stort sett registreringen manuelt og er utsatt for feil [Merok93]. Ved hjelp av et verktøy som støtter registrering kan registreringen bli mer effektiv og pålitelig.

Et registreringsverktøy kan f.eks. bestå av en videospiller i tillegg til en spesiell editor hvor man skriver inn beskrivelser og nøkkelord. Et slikt verktøy er beskrevet i [Bek94] og [Holte94]. Verktøyet utnytter den informasjon som videospilleren har om videomaterialet, i tillegg til at det støtter annoteringer i modellen som er beskrevet i avsnitt 3.2.1.

De fleste av dagens arkivsystem for film-materiale registrerer informasjon i ettertid. Dette kan føre til tap av viktig informasjon som er tilgjengelig på et tidligere stadium i produksjonsfasen. Gjennom hele produksjonen genereres informasjon som er viktig for resultatet, men som går tapt og i mange tilfeller må gjenskapes ved registrering. Å kunne registrere denne informasjonen på et tidligere tidspunkt vil dermed være gunstig for en videodatabase. Registreringsverktøy kan også støtte dette, f.eks. med registrering/logging ved digitalisering. [Dav91] foreslår et *datakamera* som kan registrere informasjon om bevegelser i omgivelsene, kameraposisjon, start og stopp av kamera osv. Dette vil ytterligere forenkle arbeidet med registreringen.

3.2.4 Søk

En prosess for *søk* i databasen er viktig for gjenfinning og gjenbruk av videoinformasjon. En bruker av systemet må ha mulighet til å utføre avanserte søk som utnytter den informasjon som ligger i databasen. Dette kan benyttes til å finne igjen video for avspilling eller videre behandling (redigering, presentasjon o.l.). Søkeverktøyet kan også støtte brukere som er ukjente med omgivelsene, slik at det gjør det enklere å finne frem i videoinformasjonen.

3.2.5 Redigering

[Mac89] introduserer konseptet med en virtuell produksjon av video. Idéen er at man skal kunne sette sammen en videoproduksjon ved å gi referanser til videomaterialet. Dette står i motsetning til den tradisjonelle, lineære kopieringen som man er vant til fra redigering av videobånd. En slik idé kan vi utnytte i et totalsystem for digital video.

Med et integrert *redigeringsverktøy* kan vi utnytte mulighetene som digital video gir oss til gjenbruk og nye produksjoner. Idéen om virtuell produksjon støtter også best mulig utnyttelse av lagringskapasiteten. Som kjent krever digital video stor lagringskapasitet. Ved å kunne bruke et redigeringsverktøy til å skape virtuelle produksjoner kan vi spare lagringsplass. Skal et slikt system fungere må det gi oss muligheten til å søke i videomateriale og kunne sette sammen sekvenser til en ny video som igjen kan lagres.

3.2.6 Målgrupper

Hvem er det så som kan tenkes å dra nytte av en slik totalløsning med et redigeringsverktøy integrert med en database? Jeg vil her nevne tre grupper

som kan utnytte et slikt system og hvordan.

Fjernsynsselskaper oppnår et forenklet bruk av arkivmateriale. Redigeringsverktøyet kan brukes til for-redigering eller til å lage skisser/oversikter for et program. Med et system som kan levere kringkastingskvalitet video vil man ha en fullstendig integrert og digitalisert produksjonsprosess, som fullstendig vil forandre hverdagen i et fjernsynsselskap. Fjernsynsselskaper med distrikstskontor vil kunne opparbeide en stor felles distribuert database som støtter enkel gjenbruk av materiale.

Reklameselskaper kan også bruke et slikt system til for-redigering. Her er ofte ikke kravet til bildekvalitet like høyt som hos TV-selskaper, slik at en heldigitalisert produksjonsprosess krever mindre utvidelser av dagens system.

Presentasjoner F.eks. lærere som skal lage en presentasjonsfilm om et spesielt emne. Distribuerte systemer kan lagre en mengde videoer som er tilgjengelig for sammensetting på en enkel måte.

3.3 NRK Nyhetsavdelingen

Her vil jeg forsøke å beskrive en produksjons- og redigeringsprosess, med NRK Nyhetsavdelingen som eksempel. En bredere beskrivelse av hele produksjonsprosessen finnes i [Holte94].

3.3.1 Produksjonsprosessen

En generell produksjonsprosess kan skisseres som fem aktiviteter.

Planlegging Idéer, skisser, bakgrunnsinformasjon, arkivmateriale, tidsplan, produksjonsplan.

Opptak Opptak av nytt materiale.

For-redigering Off-line redigering på materiale med tidskode (gjerne VHS- eller S-VHS format). Resulterer i en EDL.

Redigering Redigering basert på EDL fra forrige punkt eller manuell redigering fra kildemateriale (hvis for-redigering ikke er utført).

Etterarbeid Lydpålegg, arkivering o.l.

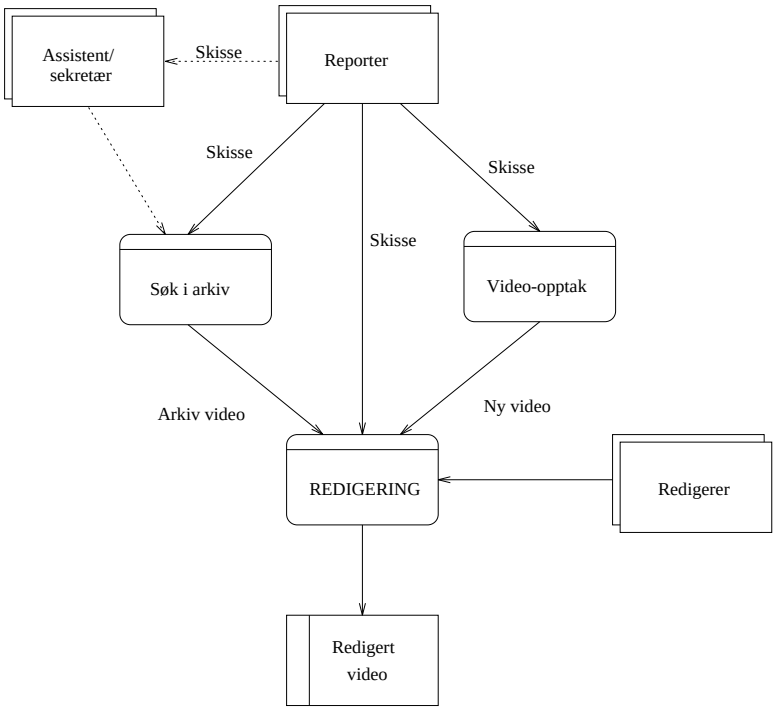
Aktivitetene vil variere fra produksjon til produksjon og fra avdeling til avdeling. Denne korte oversikten kan derfor ikke betraktes som fullstendig.

Produksjonen av et *nyhetsinnslag* starter med at en *reporter* har en idé til en nyhetsreportasje. Reporteren skaffer seg kunnskap om den aktuelle nyheten

gjennom søk i Basys[4], aviser, intervjuer o.l. Ut fra dette bakrunnsmaterialet lager reporteren en plan eller skisse over hvordan innslaget skal bli. Skissen inneholder hva som ønskes av arkivbilder, nytt videomateriale, lyd osv.

Ut fra denne skissen må det skaffes videomateriale til reportasjen. Dette kan bestå av arkivmateriale og nye opptak. Arkivmateriale finnes ved søk i NRKs SIFT[5]-database for Fjernsynsarkivet etter beskrivelser av program og innslag. Programbeskrivelser som ser ut til å omfatte det ønskede materialet, leveres så til *Fjernsynsarkivet* for å få utlånt arkivert video.

Selve redigeringen skjer så på bakgrunn av reporterens skisse, arkivmateriale og nye opptak. *Redigereren* og reporteren setter seg sammen for å sette materialet sammen til en reportasje. Produksjonssprosessen kan summeres opp i en figur (se 3.6).



Figur 3.6: Dataflyt i produksjonsprosessen

3.3.2 Søk etter arkivmateriale

Søk etter arkivmateriale for bruk i reportasjer er i dag en omstendelig prosess. Det starter med et søk i SIFT-databasen for Fjernsynsarkivet. SIFT-databasen inneholder beskrivelser av programmer og innslag som er sendt

[4]Basys er et intern datasystem som benyttes i nyhetsredaksjonene i radio og fjernsyn. Her legges blant annet inn meldinger fra ulike nyhetsbyrå [Holte94].

[5]SIFT (Søk i FriTekst) er et databasesystem fra Statens Datasentral (SDS).

[Holte94]. For hvert program eller innslag finnes en fritekstlig beskrivelse i tillegg til noe informasjon, som tittel, hvilken serie programmet hører til, når det er sendt, tidsforbruk, båndnummer, båndtype osv. Beskrivelsene i databasen er underlagt spesielle skriveregler, som f.eks. at en person som opptrer i bildet beskrives med etternavn i STORE bokstaver. Dette gjør at det ligger en del meta-informasjon i teksten som kan nyttes til å finne igjen spesielle ting man vil søke etter. Databasen krever dermed også en del opplæring og trening, både av de som skal registrere informasjon og de som skal søke i basen.

Når man har funnet passende beskrivelser, sendes disse til Fjernsynsarkivet som returnerer det aktuelle videomaterialet. Man må så se igjennom videokassetten(e) for å få bekreftet at materialet er det som man er ute etter. Hvis videomaterialet ikke samsvarer med beskrivelsene eller ikke er tilfredsstillende på andre måter, resulterer dette i nye søk og gjennomsyn av videokassettene. Søk i SIFT-databasen kan utføres av reporteren selv eller av en redaksjonssekretær.

Prosessen *Søk i arkiv* i figur 3.6 består altså av følgende punkter:

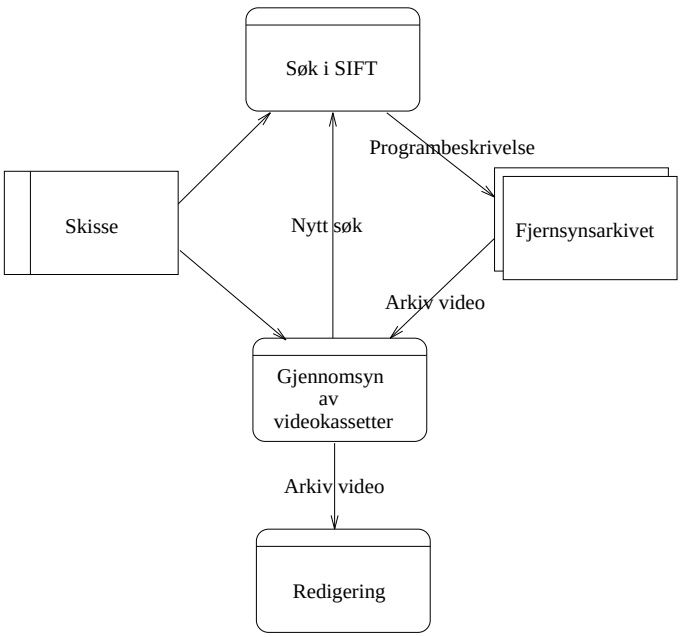
1. Søk i SIFT-databasen etter passende programbeskrivelse.
2. Send programbeskrivelse til *Fjernsynsarkivet*, og få returnert videokassett.
3. Se igjennom materialet for å verifisere innholdet.
4. Gjenta søk om nødvendig.

Dette illustreres i figur 3.7. *Punkt 3* kan utføres enten av reporteren eller av redigereren. Det antydes at ca. halve redigeringstiden går med til spoling og gjennomsyn av videokassetter [Holte94]. Hvis gjennomsyn foretas først i redigeringsrommet og dette fører til nye søk, vil produksjonen forsinkes betraktelig. Dette kan være kritisk i og med at både reportere og redigerere jobber med absolutte tidsfrister. En annen side av saken er at i stedet for å jobbe kreativt med en reportasje risikerer både reporter og redigerer å bruke unødvendig mye tid på søk etter materiale. I følge redigerere ved NRK er mislykkede søk er noe som forekommer relativt ofte. Det er vanlig at reporteren kommer med 3-4 videokassetter pr. sekvens med arkivmateriale som skal brukes.

Man har liten eller ingen garanti for at det materialet som man finner ved søk i SIFT-databasen er det som passer best til den aktuelle reportasjen. Det er tidkrevende å se igjennom mange ulike videokassetter for å se etter om en videosekvens er bedre egnet enn en annen.

3.3.3 Redigeringsprosess

Redigereren og reporteren setter seg sammen for å redigere et nyhetsinnslag basert på skissen fra reporteren og videomaterialet. Både programtekst og



Figur 3.7: Søking etter arkivmateriale i NRK

video redigeres. Reporterene har ulike tilnærmingsmåter når det gjelder redigeringsprosessen. Noen har sett igjennom alt materialet og laget et detaljert forslag til klippeliste, mens andre kommer til redigeringsrommet med en bunke videokassetter og en vag idé om hvordan innslaget skal være.

Gjennom redigeringsprosesen er redigereren ofte kritikeren. Reporteren har på forhånd en idé om hvordan reportasjen skal være, mens redigereren ser alt med *nye øyne* og kan lettere være kritisk til måten en reportasje blir fremstilt på. Samarbeidet mellom redigerer og reporter er en kreativ prosess som ender i en ferdig reportasje.

Selve redigeringsprosessen består i at man spoler til man finner et ønsket innpunkt på kildematerialet og på masteren. Deretter kopieres video fra kildekassetten til masterkassetten. Prosessen er beskrevet som *datastyrt redigering* i avsnitt 2.7. Redigering i nyhetsavdelingen består av kun enkle klipp. Utstyret tillater ikke andre overganger og effekter. For mer avanserte effekter sendes materialet til *Båndfilmsentralen* for videre behandling.

3.3.4 Redigeringsverktøy

For å motivere for at et digitalt redigeringsverktøy kan være til hjelp for NRK Nyhetsavdelingen vil jeg beskrive et enkelt scenario. Scenariet tar utgangspunkt i at Fjernsynsarkivet er organisert i en videodatabase som beskrevet i avsnitt 3.2.1 og [Hje94] (et digitalt fjernsynsarkiv er beskrevet i [Holte94]).

I en slik situasjon vil man kunne søke i videoinformasjonen etter bestemte sekvenser, få opp på skjermen en *treffliste* og umiddelbart se igjennom materialet for å verifisere innholdet. Trefflisten består av videosekvenser som tilfredsstiller søkekriteriene. Man vil enkelt kunne se igjennom en sekvens og vurdere om den egner seg bedre til den aktuelle reportasjen enn en annen sekvens fra trefflisten. Et redigeringsverktøy integrert i denne pakken kan benyttes til for-redigering. Dette betyr at redigering skjer med det digitaliserte videomaterialet. Redigeringen gir som resultat en EDL (med tidskoder) som kan benyttes ved den endelige redigeringen.

Vi kan se for oss to situasjoner:

1. Kun arkivmateriale er digitalisert og beskrevet i videodatabasen.
2. Alt materiale (arkiv og råmateriale) blir digitalisert og beskrevet i videodatabasen.

I *situasjon 1* vil man kunne sette sammen en skisse over arkivmaterialet i reportasjen. Dette vil være tidsbesparende ved at man hurtigere finner fram til det materialet som kan benyttes og ved at man enkelt finner tidskoder for hvor på båndet materialet befinner seg. Ved *situasjon 2* vil man kunne sette sammen hele reportasjen i en for-redigeringsprosess og senere bare "*mate*" et on-line redigeringsutstyr med EDLen og de nødvendige kassetter. Dette vil være svært tidsbesparende i og med at man kan foreta redigeringen samtidig med søking etter materialet.

Hvem er det så som kan tenkes utføre denne for-redigeringen? Det kan enten være en reporter eller en redigerer. En reporter som utfører en fullstendig for-redigering vil få mer innflytelse på det endelige resultatet. Imidlertid er det ulike syn på hva rollene i en slik situasjon bør være. Mange reportere er kun interessert i det journalistiske arbeidet og vil overlate mest mulig av redigeringsarbeidet til redigereren. En EDL fra reporteren kan eventuelt betraktes som en skisse eller forslag til redigereren. Dette kan benyttes som erstatning for tekstlige skisser på et ark. Dermed får reporteren anledning til å lage mer fullstendige skisser og kan lettere se sammenhengen i reportasjen. En redigerer som utfører søk og for-redigering vil kunne bruke mer tid til selve redigeringen i stedet for til spoling og gjennomsyn av en mengde videokassetter. Ved en heldigitalisert produksjonsprosess vil man kunne gjøre redigering og søk fullstendig integrert. Dette vil lette arbeidssituasjonen for de fleste redigerere. Trenger man en spesiell sekvens, kan man bare søke spesielt etter dette, og slipper å se gjennom en rekke bånd for å se om det *kanskje* er en sekvens som passer inn.

Kapittel 4

Redigeringsverktøy

4.1 Eksisterende verktøy

Det finnes mange ulike verktøy for redigering av digital video på markedet i dag. Mange ulike kvalitetsklasser og prisklasser er representert. Det varierer fra billige lavkvalitets hjemmevideo redigeringsverktøy til høykvalitets profesjonelle verktøy. Jeg vil her presentere en del av egenskapene og funksjonaliteten til noen verktøy, for å gi et innblikk i hva kommersielle produkter kan tilby.

4.1.1 AVID

Avid Technology Inc. er en internasjonal leverandør av integrerte løsninger for opptak, redigering og avspilling av digitale medier. I Norge leveres utstyret av Danmon. Avid har en rekke løsninger for film, video og lyd. Blant brukerne av Avid finnes BBC, CNN, TV4 og Institutt for drama, film og teater ved AVH i Trondheim.

For ikke-lineær redigering av film og video har Avid en serie med betegnelsen Media Composer. Dette er løsninger som støtter både *off-line* redigering (Media Composer 400, 800 og Film Composer) og *on-line* redigering (Media Composer 1000, 4000, 8000). Alle systemene i denne serien har en del likhetstrekk, som f.eks. at de kjøres på Apples Machintosh Quadra maskiner, med to multi-sync monitorer. I tillegg kommer en del andre maskinvare-moduler, avhengig av hvilken løsning (kvalitet og prisklasse) man velger (eks. 14" eller 20" monitorer, JPEG eller Advanced JPEG komprimeringskort, Avid Effects Module, Avid Enhancement Board osv.). Det er høye krav til maskinvaren. Media Composer 800-8000 og Film Composer benytter Macintosh Quarda 950 m/32 MByte RAM og 1 GByte harddisk.

Avid tilbyr valg av bildekvalitet/oppløsning (AVR - Avid Video Resolutions) for å gi brukeren valgmuligheter i å oppnå et kompromiss mellom bildekvalitet og lagringskapasitet. Med Avids laveste oppløsning kan man lagre

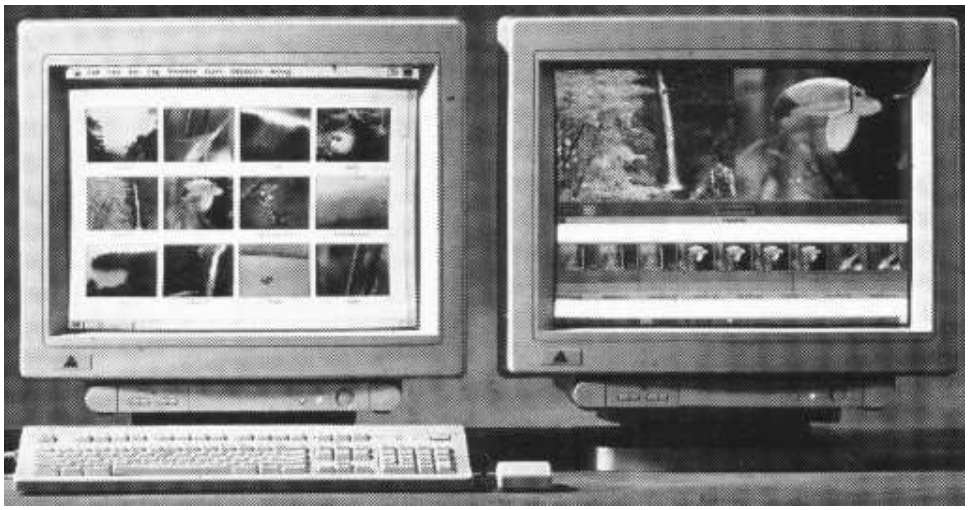
inntil 2 timer video pr. GByte.

Film Composer
Systemene i Media Composer familien har mange liketstrekk når det gjelder funksjonalitet. Jeg vil her behandle *Film Composer*[1] for å gi et innblikk i hvilke funksjoner og muligheter et slikt profesjonelt verktøy har.

Avids Media Composer systemer benytter seg av to monitorer. En *source*-monitor hvor alt råmaterialet blir presentert (som også kalles *bin*-monitor) og en *recorder*-monitor.

En *bin* er et lager av en mengde videosekvenser. Dette er råmaterialet som er utgangspunktet for redigeringen. En bin kan inneholde mange videosekvenser og en videosekvens kan lagres i flere ulike bin. De ulike videosekvensene kan representeres på skjermen på to måter. Det kan være et icon/bilde med en kort tekst/navn (*frame mode*) eller man kan bruke kun en tekstlig beskrivelse (*text mode*). Bildet som er knyttet til sekvensen i *frame mode* er det første bildet i sekvensen. En videosekvens kan flyttes over i recorder-monitoren for visning og redigering ved å klikke og dra med musen.

Recorder-monitoren har et fullstendig eget grensesnitt, med to "vinduer" for videosekvenser. Det ene vinduet brukes til å spille av videosekvenser, valgt fra en bin. Det andre brukes til å vise den redigerte videoen (masteren)[2].



Figur 4.1: Avid Film Composer

Det finnes en rekke funksjoner koblet til knapper i grensesnittet til AVID. Det er en knapperad for råmaterial-vinduet og en for master-vinduet. Disse inneholder blant annet:

[1] Institutt for drama, film og teater ved AVH benytter dette systemet til off-line redigering av film og video.

[2] I Avid kalles denne masteren en sekvens (for å gjøre forvirringen komplett).

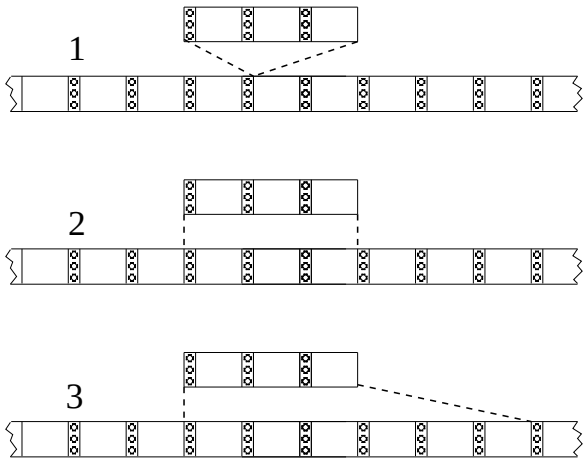
- Play/pause.
- Setting av innpunkt og utpunkt.
- Direktehopp til innpunkt og utpunkt.
- Enkeltramme fram og tilbake.
- Ti rammer fram og tilbake.
- Tidslinje/slider.
- Fullskjerm visning.

Råmaterialet redigeres inn i masteren ved at man velger et ut- og et innpunkt i videosekvensen, og minst et innpunkt i masteren. Råmaterialet kan så redigeres inn i masteren på tre ulike måter (se også figur 4.2).

Splice (1): Rydd plass og sett inn.

Overwrite (2): Skriv over det som ligger på masteren fra før.

Replace (3): Tilpass den valgte råsekvens til et valgt tidsrom i masteren.



Figur 4.2: De ulike editerings-muligheter i AVID

Redigeringen kan skje på kun video, kun lyd eller begge deler. Man kan også redigere ut sekvenser fra masteren. Dette kan man gjøre ved å løfte eller klippe ut. Løfter man ut lyd eller video blir det tilbake en tom sekvens der hvor man tar ut. Klipper man ut, vil de resterende sekvensene "skjøtes sammen" slik at man unngår diskontinuitet i masteren.

Film Composer tilbyr en god del effekter som man kan bruke på overgangene mellom klippene. Dette er blant annet ulike former for *wipe* og *overtoning*. Transisjonseffektene kan styres vha. flere parametre, f.eks. hvor mange rammer effekten skal benyttes på. Når man definerer en effekt, settes denne

sammen av de involverte sekvenser og lagres som en egen sekvens. Dette gjør at man slipper å behandle transisjonseffekter ved avspilling (on-the-fly) og man får en rask visning av masteren.

Masteren visualiseres ved hjelp av en tidslinje. På denne tidslinjen merkes av hvilke klipp som inngår og hvor lenge de ulike klippene varer. Ulike transisjonseffekter er også avmerket på denne tidslinjen. Klippene identifiseres vha. tekst/navn eller et icon/bilde. Det er ikke mulig å redigere de ulike klippene direkte på selve tidslinjen. Hvis inn- og utpunkt skal endres må dette skje ved å klippe ut og sette inn en ny sekvens.

Film Composer har også innebygd prosjekthåndteringsfunksjoner som består i å lagre prosjekter i ulike *bin*. Hvert klipp kan tilknyttes en mengde opplysninger og det finnes en søkefunksjon som gjør at man kan søke blant denne informasjonen. Dette baserer seg på *logging* som må gjøres ved digitalisering av materialet. Avid definerer et logg-format for informasjon som tilknyttes hvert klipp/hver sekvens. Dette logg-formatet består av en del *global* informasjon (global headings) og en del *lokal* informasjon (column headings). Den globale informasjonen defineres for en hel digitaliseringssesjon og er den samme for alle sekvenser som blir digitalisert i løpet av denne sesjonen. Felter i *global headings* er VIDEO_FORMAT (NTSC, PAL), AUDIO_FORMAT (samplingsfrekvens - kHz), TAPE (båndnavn), og FPS (antall rammer pr. sekund ved digitalisering). Den lokale informasjonen kan være individuell for hver digitalisert sekvens. Felter i *column headings* er blant annet Name, Start, End, Creation Date, Duration, Scene, Take, Descript, Comments osv. All logget informasjon kan senere benyttes til søk for å finne igjen ulike videosekvenser. Den lokale informasjonen inneholder mange felter og det er opp til brukerne av systemet å benytte de ulike feltene mest hensiktsmessig. Nytten av søk i dette systemet er avhengig av den som digitaliserer og logger informasjonen om materialet. Hvor mye informasjon er nyttig? Hvor mye informasjon kan man ha bruk for senere? Alle feltene i loggen kan man vise i bin-monitoren i *text mode*.

Avid definerer *redigeringsprosessen* ved hjelp av fire punkter:

Innlasting av materiale Valg av kildemateriale, digitalisering, logging.

Enkel redigering Sette sammen sekvenser til en helhet. Bruk av enkle klipp.

Avansert redigering Legge på effekter, avansert lydredigering osv.

Resultat "Ta ut" digital video på en videokassett og få ut en EDL.

For at man skal få ut en EDL er man avhengig av at man har logget informasjon om båndnummer/navn, tidskode på båndet osv. Dette er viktig informasjon som er en forutsetning hvis man skal bruke systemet til for-redigering, for senere å foreta fullkvalitets redigering på annet redigeringsutstyr. Det finnes flere standarder for EDLer som Avid støtter, men alt er ikke nødvendigvis kompatibelt. Utstyret som skal benyttes til fullkvalitets redigering må som oftest "læres opp" til å forstå de ulike EDLene.

"Opplæringen" gjelder hvordan utstyret skal tolke de ulike kodene i en EDL. Standarder som Avid støtter er blandt annet *ampex, cmx* og *Sony*. Se avsnitt 2.9 for nærmere beskrivelse av EDL.

Film Composer er et profesjonelt verktøy som tar utgangspunkt i prosessen for klipping/redigering av film. Dette gir seg utslag i at man foretar de samme operasjonene på skjermen som en analog filmredigerer ville ha gjort fysisk. Man tar kildematerialet, klipper (markerer) et inn- og et ut-punkt, klipper (markerer) et inn-punkt i masteren og klipper inn kildematerialet. På denne måten vil en filmredigerer kjenne seg igjen i redigeringsprosessen, samtidig som man utnytter de fordeler digital teknikk har til å behandle materialet på en enkel måte.

4.1.2 Integrated Video

Integrated Video er et redigeringsverktøy for redigering av presentasjoner og video. Systemet kjøres på en Silicon Graphics arbeidsstasjon, f.eks. Indy. Integrated Video tilbyr en integrert pakke for digitalisering, redigering og avspilling.

Integrated Video støtter tre videoformater:

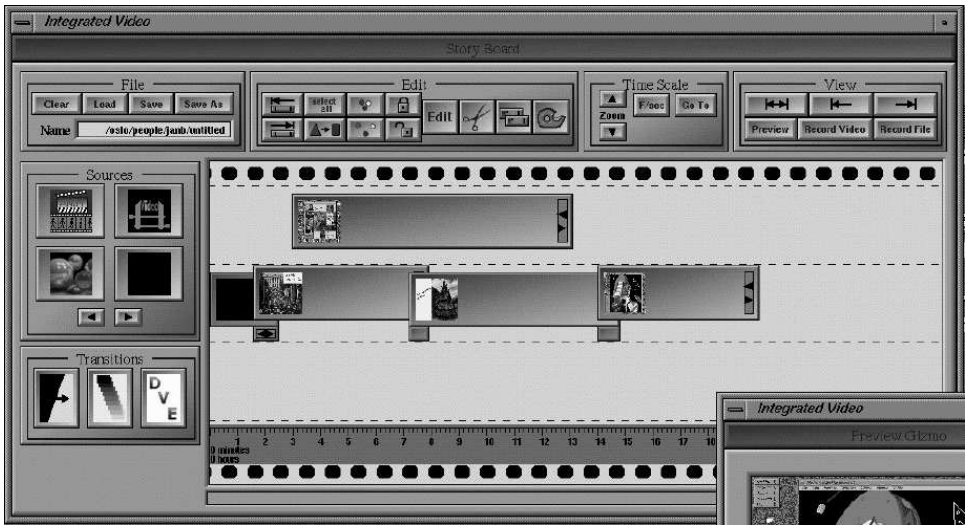
- IR Videofile (Integrated Research Videofile)
- SGI Moviefile
- Apple Quicktime

og 15 raster bildeformater (bl.a. SGI RGB, Tiff, Mac Pict, Aurora)

Videoen redigeres i et *Story Board* (se figur 4.3) med tidslinje. Denne modulen består av tre nivåer med ulik prioritet. Her kan man legge rammer over hverandre og bestemme hvilke som har høyest prioritet. Til å sette sammen en ferdig video brukes klikk-og-dra prinsippet. Videoen kan settes sammen av tre elementer:

- Stillbilde
- Animasjon
- Video

Vil man sette inn en av disse elementene, klikker man på et ikon som representerer det man vil sette inn, drar dette inn i tidslinjen og slipper det på ønsket plass. Deretter kan man velge hvilket bilde, animasjon eller video som skal settes inn på ikonets plass i tidslinjen. Dette gjøres gjennom en fildialogboks som velger fra ønsket fil. Man kan også styre ulike parametre som varighet, start og sluttpunkt fra en dialogboks.



Figur 4.3: Story board i Integrated Video

Et *stillbilde* kan "dras ut" slik at det vises over et bestemt tidsrom, dvs. over flere rammer. Dette kan brukes til f.eks. å legge tekst, logo eller figurer i bildet. En *animasjon* er en sekvens av stillbilder som ligger fysisk separat lagret på disk. Man velger inn en sekvens av stillbilder og rekkefølgen på disse. Ved valg av *video* har man den begrensningen at man ikke kan klippe ut en sekvens av videoen. Man kan ta inn en videosekvens og manipulere den, men ikke klippe den opp i flere sekvenser. Dette gjør at man må redigere og dele opp videoskvensene ved digitalisering eller man kan benytte annen programvare til dette. Alle elementene redigeres ved hjelp av SMPTE tidskode eller man kan benytte absolutte rammeverdier. Dette gjelder både start- og sluttpunkt og varighet for innslagene. I tillegg til disse elementene kan man legge inn et helt ensfarget bilde eller et bilde med farge-striper (for TV-produksjon).

En video i *Story Board* er en sammensetning av objekter av de elementene som nevnt ovenfor. Dette kan lagres som skript som senere kan hentes inn for videre redigering. Videre kan den ferdige videoen overføres/spilles inn på eksternt videoutstyr eller lagres på disk. Ved lagring på disk kan man velge mellom å lagre videoen som enkeltbilder eller som en videosekvens. Man har de samme formatene å velge mellom som nevnt tidligere. Lagrer man resultatet som en videosekvens kan man også velge mellom ulike komprimeringsmetoder. F.eks. ved lagring som SGI Moviefile kan man velge JPEG komprimering. Det finnes også mulighet for forhåndsvisning av videosekvensen man har i tidslinjen. Systemet går her gjennom hele sekvensen og behandler bilde for bilde, som igjen settes sammen til en sekvens som kan vises i en forhåndsviser (se figur 4.4) eller lagres på disk. Dette kan ofte ta relativt lang tid, spesielt hvis man har med ulike effekter.

Integrated Video har en del funksjoner for editering av objekter på tidslinjen.

Figur 4.4: Forhåndsvisning i IntegratedVideo.

Mange av disse er lik de vi ofte finner i andre typer vindusbaserte editorer. Et objekt markeres ved å klikke på det før man klikker på en editeringsfunksjon. Dette inkluderer:

Cut/Copy/Paste Velkjente funksjoner for klipp, kopiering og liming av objekter.

Edit properties Man kan endre parametre for de ulike objektene.

Align beginnings/ends Det er lett å få objekter som ligger over hverandre til å starte og/eller slutte på samme ramme.

Select all Marker alle objektene i tidslinjen.

Change type Et objekt kan endres til å være av et annet element (still-bilde, animasjon eller video) uten at det mister de andre parametrene som lengde og start- og sluttpunkt.

Group/ungroup Flere objekter kan grupperes slik at de kan behandles som et enkelt objekt.

Lock/unlock Deler av tidslinjen kan låses slik at man unngår tilfeldig og uønsket sletting og innsetting.

Alle objekter kan reduseres eller *dras ut* i tid. Dette skjer ved klikk-og-dra prinsippet eller ved *edit properties* (se ovenfor). Det som skjer er at animasjon og video vil endre hastighet, dvs. enkeltbilder blir duplisert eller forkastet. Under redigeringen kan tidsskalaen strekkes og forminskes. Rammeraten kan hele tiden endres slik at man tilpasser sluttresultatet til den rammeraten man ønsker. Man kan jobbe med den rammeraten man vil. Hvis man endrer rammeraten underveis i redigeringen, kan man velge om videosekvesen skal tilpasses i tid eller i rammer.

Når det gjelder transisjonseffekter har Integrated Video *overtoning* og en rekke *wipes* og *digitale videoeffekter* (DVE). I tillegg tilbyr systemet muligheten til å legge inn egne wipes og DVEer. Integrated Video har også en hel masse bildebehandlingsfunksjoner som jeg ikke vil komme inn på her. Dette er funksjoner som manipulerer enkeltbilder/rammer eller hele sekvenser.

Det krever mye diskplass å jobbe med Integrated Video. En liten, uformell test viser at en videosekvens på omtrent ni sekunder ga en datamengde på 4,7 MByte for lagring som skript og 16,1 MByte for lagring som SGI Movie (JPEG-komprimert). Sekvensen bestod av en animasjon, en videosekvens og to stillbilder med en overtoning og en DVE. Sekvensen var lagt ut i to nivåer.

Først og fremst er Integrated Video godt egnet for presentasjoner. Det har et svært enkelt og intuitivt brukergrensesnitt som gjør det enkelt å komme i gang med redigeringen. Integrated Video har mange funksjoner som vi er vant til å bruke i andre vindusbaserte programmer og man blir raskt kjent med systemets muligheter. Disse punktene gjør at det er enkelt å bruke for folk som ikke har spesielt mye erfaring fra redigering.

Integrated Video tilbyr ikke redigering av lyd, noe som gjør redigering av video med lyd, spesielt synkronisert lyd, ganske tungvint. Først må man gjøre ferdig videoredigeringen og deretter benytte annen programvare for å legge på lyd. Silicon Graphics tilbyr (selvfølgelig) dette i programmet MovieMaker. Andre programmer som tilbys i samme pakke er MovieConvert som konverterer mellom ulike videoformater og SoundEditor som gjør at man kan mikse og redigere lyd.

Et annet problem ved å redigere video med Integrated Video er at man ikke kan klippe i en videosekvens som man skal legge inn i StoryBoard. Man kan kun legge inn hele sekvenser. Dette gjør at man også her må benytte MovieMaker hvis man ikke har delt opp videoen tilstrekkelig ved digitalisering.

4.1.3 Andre verktøy

Det finnes mange ikke-lineære, digitale redigeringsverktøy på markedet i dag. De tilbyr både ulik billedkvalitet og funksjonalitet. Noen verktøy er beregnet på "hjemmemarkedet" mens andre sikter mot profesjonelt bruk.

Digitale verktøy er på vei inn i både profesjonelle redigeringsmiljø og i miljø som tradisjonelt ikke har drevet med videoredigering. Da tenker jeg mest på miljøer som bruker video til presentasjoner og reklame, f.eks. bedrifter og læreinstitusjoner. Det er ingen tvil om at det kommer til å skje en rivende utviklig innen digital redigering i de nærmeste årene.

I [Avg94] beskrives en del systemer både for MacIntosh og PC. Dette er **Montage, Matrox Studios** og **D/Vision Pro** for PC og **Digital Film Deluxe, VideoCube, Media 100** og **VideoVision Studio** for Mac. [Col94] beskriver **Adobe Premiere, VideoFusion** og **Avid VideoShop** for Mac. [Sch94] beskriver også **Adobe Premiere** for Mac. I tillegg kan en kvalitetstest av ulike systemer finnes i [Doh94]. Dette er omfattende beskrivelser og vurderinger. De fleste relevante sider ved mener jeg er beskrevet i avsnittene om AVID og Integrated Video, så jeg finner det ikke naturlig å gå mere i detalj på flere av de kommersielle verktøyene.

Felles for disse systemene er elementer som:

- Redigering på tidslinje.
- Enkel oversikt over kildemateriale.
- Avspilling av master og kildemateriale.
- Ulike redigeringsmuligheter (klipp og lim).
- Et utvalg wipe, overtoninger og DVEer.

Det virker som om de fleste er enige om at enkel bruk og redigering på ei tidslinje er det viktigste for et digitalt redigeringsverktøy. Et annet viktig punkt er at mye av redigeringen kan skje direkte på tidslinjen, ved bl.a. klikk-og-dra med musa. Dette kalles en interaktiv tidslinje og bl.a. fremhevet som den fremste egenskapen til **D/Vision Pro** [Avg94]. Dette er altså et viktig punkt å behandle ved konstruksjon av et redigeringsverktøy for digital video.

4.2 Generell funksjonalitet

For et redigeringsverktøy vil det være stor forskjell i bruksmønster og metoder, avhengig av om vi har profesjonelle brukere eller amatører. Hva som kreves av et slikt verktøy vil avhenge av den tilsiktede bruken av det og hvilken erfaring brukerne har fra før. Jeg vil her behandle en del funksjonalitet som kan være ønskelig i et ikke-lineært redigeringsverktøy. Dette er idéelle krav og ønsker, som i en del tilfeller ikke er helt nødvendige. Jeg vil se på hva vi kan forvente oss, og i et senere kapittel plukke ut noen punkter som illustreres gjennom implementasjonen av en prototyp.

Hovedoppgaven til et redigeringsverktøy er å sette sammen ulike videosegmenter til en ny videoproduksjon. Det må kunne presentere et sett

med videosekvenser og la brukeren velge fra dette for å sette sammen en ny videosekvens. Jeg vil videre bruke betegnelsen *master* om den videoen som er under produksjon, dvs. det som til slutt blir den ferdig redigerte videoen.

4.2.1 Kildemateriale

Vi må ha tilgang til et sett med kildemateriale og ut fra dette kunne velge sekvenser som kan settes sammen til en ny produksjon. I en analog redigeringssituasjon vil kildematerialet ligge på ulike videokassetter. Med video på datamaskinen kan vi presentere kildematerialet på andre måter. En tekstlig beskrivelse eller et lite bilde/icon for hver sekvens av kildemateriale, kan illustrere de sekvenser man har til rådighet. Den tekstlige beskrivelsen kan bestå av flere punkter som fritekst, navn/tittel, tid for opptak osv. Dette er beskrivelser som sier noe om innholdet i en sekvens, og som kan være til hjelp når vi skal plukke ut sekvenser som skal inn i masteren. I Avid (se avsnitt 4.1.1) har man valget mellom en tekstlig beskrivelse eller et bilde med tilhørende navn på sekvensen. I databasesammenheng vil kildematerialet være de sekvenser som ligger lagret i databasen.

Med utgangspunkt i databasemodellen som er presentert i avsnitt 3.2.1, vil det være nyttig å kunne presentere annotasjonene som er gjort til de ulike sekvensene. Dette er strukturer som er spesielt egnet for å beskrive videomateriale. Ved å presentere dette for redigereren, vil det forenkle letingen etter de sekvensene som er aktuelle for bruk i redigeringen.

4.2.2 Avspilling

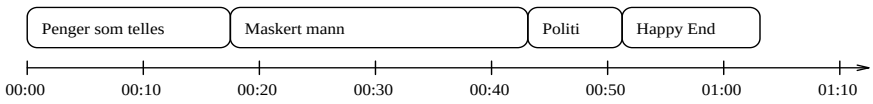
Det er viktig at redigeringsverktøyet gir oss muligheten til å spille av både kildemateriale og masteren underveis i produksjonen. Videoredigering er hovedsaklig en visuell prosess. Dette betyr at muligheten for avspilling av kildemateriale og master er et ubetinget krav. En avspillingsmekanisme gjør at vi kan se igjennom kildematerialet vi har til rådighet og også se gjennom resultatet av redigeringen som til enhver tid er utført.

Avspillingen må inneholde ulike funksjoner som letter søk og gjennomsyn av materiale. Slike funksjoner er frysing av bilde og avspilling i ulike hastigheter, både framover og bakover i videoen.

4.2.3 Tidslinje

Alle ikke-lineære redigeringssystemer som er beskrevet i avsnitt 4.1 benytter en tidslinje som visualiserer masteren. Dette er en linje med tidsanvisning som angir hvilke sekvenser masteren består av og tidsangivelser for de ulike sekvenser (illustrert i figur 4.5).

Denne tidslinjen må inneholde en visualisering av de sekvenser som er redi-



Figur 4.5: Slik kan en tidslinje se ut

gert sammen og de enkelte sekvenser bør være identifisert. På samme måte som ved identifikasjonen av kildemateriale kan denne identifikasjonen være gjennom tekst eller bilde. Tidslinjen bør avbilde hvor lang tid de ulike sekvenser tar og hvor lang tid som er brukt i masteren. Det er på dette punktet man virkelig får utnyttet fordelen med ikke-lineær i forhold til lineær redigering. Ved hjelp av en tidslinje får man hele tiden et visuelt bilde av hvordan masteren er satt sammen, mens ved analog redigering må man se igjennom hele videoen for å danne seg et inntrykk av dette. I tillegg ser man enkelt hvilke sekvenser masteren er satt sammen av. Dette er ikke mulig ved analog redigering.

Tidsanvisningen kan basere seg på f.eks. SMPTE tidskode, antall rammer, sekunder e.l. Presisjonen til denne vil være applikasjonsavhengig. Hvilke krav setter brukerne til nøyaktighet i redigeringen? Tidsangivelser v.h.a. tidskode er det mest brukte i dagens systemer.

4.2.4 Innsetting

Når man vil sette en videosekvens fra kildematerialet inn i masteren er det flere ting man må ha mulighet til. Tradisjonell redigering krever at man har muligheten til å sette et innpunkt i kildesekvensen og i masteren. Innpunktene forteller oss hvor kildesekvensen settes inn i masteren, men det forteller ingenting om hvor mye av kildesekvensen som blir satt inn. For å utnytte potensialet til ikke-lineær redigering, bør vi ha muligheten til også å sette utpunkt i kildematerialet. Dette kalles gjerne *tre-punkts redigering.*

Kildesekvensen må kunne settes inn hvor som helst i masteren og man må kunne velge mellom å sette inn og overskrive det som allerede er i masteren. Å sette inn en sekvens (ved å rydde plass) vil være til nytte når man trenger en ekstra sekvens inne i det som man allerede har redigert. Overskrivning vil være svært praktisk hvis man ønsker å bytte ut en sekvens med en annen (kanskje nyere opptak), men la resten av materialet være uberørt.

4.2.5 Redigering

For å få utnyttet konseptet med tidslinje, er det en fordel at det meste av redigeringen skjer direkte på tidslinjen (se avsnitt 4.1.3). Tidslinjen gir et enkelt visuelt bilde på hva videoen inneholder og det er naturlig at det meste av manipuleringene med videosekvensene foregår direkte på denne. Ved å ha en enkel tidslinje og redigering direkte på denne, kan man forenkle forståelsen

av redigeringsprosessen, slik at det blir mere intuitivt å redigere. Dette vil også føre til en lavere inngangsterskel for det å jobbe med redigering.

Det som er viktig når det gjelder selve redigeringsoppgavene er at man kan merke av sekvenser for senere å manipulere disse. Man kan tenke seg flere redigeringsfunksjoner som kan være nyttige i ulike sammenhenger. Det er ikke mulig å sette noen faste krav til hva dette må være. I avsnittet ovenfor blir innsetting behandlet. Det motsatte, nemlig sletting, er også en nødvendig operasjon. Andre operasjoner kan være kopiering, duplisering, markere en sekvens for så å forkaste resten, ulike bildemanipulasjonsteknikker osv.

4.2.6 Effekter

En del typer overgangseffekter er beskrevet i avsnitt 2.4. Det er ikke et absolutt krav for et redigeringsverktøy å tilby så mange som mulig av disse. Antall effekter og hvilke effekter som er aktuelle, bestemmes av målgruppen for systemet og hvilken type redigering systemet er ment for. F.eks. ved grovredigering, hvor man vil benytte andre verktøy senere i produksjonsfasen, vil man sannsynligvis ikke ha behov for noen spesielle effekter i det hele tatt.

4.2.7 Resultat

Redigeringsverktøyet må kunne gi som resultat en ferdig redigert video. Denne må kunne lagres for senere avspilling. Det må også være mulig å hente inn den redigerte videoen i redigeringsverktøyet for å kunne redigere videre eller bruke materialet til andre produksjoner. Når en produksjon lastes inn i redigeringsverktøyet er det viktig at det kan identifisere alle sekvenser, overganger og effekter som var resultatet av tidligere redigering. Dette er viktig slik at man kan endre det som allerede ligger i videoen og ikke bare "bygge videre".

En fleksibel løsning på dette er at man definerer et lagringsformat som både registreringsverktøyet og videospilleren kan lese. En annen løsning er å benytte to ulike formater. Da må man skille mellom skript for redigeringsverktøyet og avpillingsformat for videospilleren.

For et verktøy som skal brukes til for-redigering er det viktig at det kan gi som resultat en EDL. Dette er beskrevet i avsnitt 2.9. I og med at det ikke finnes noen definert standard for slike lister, vil det måtte tilpasses det on-line utstyret som benyttes.

4.2.8 Brukergrensesnitt

Som alle andre applikasjoner bør et redigeringsverktøy ha et intuitivt grafisk brukergrensesnitt. Hvis redigeringsarbeidet er tenkt utført ved hjelp av

mus/tastatur bør grensesnittet inneholde "knapper" for de mest brukte og nyttige funksjonene, slik at disse er enkle å aktivere. Hvordan brukergrensesnittet er utformet kan være avhengig av spesielle brukerkrav og hvordan brukeren er vant til å jobbe fra før. F.eks. en profesjonell redigerer vil gjerne ha mest mulig fysiske knapper og spaker som kan benyttes i redigeringen. Dette er nærmere beskrevet i avsnitt 4.4.1

4.2.9 Lydredigering

Et fullverdig redigeringsverktøy må også ha mulighet for redigering av lyd med pålegg av flere lydspor o.l. Dette har mange interessante sider, men det er ikke innenfor rammen av denne oppgaven å se på redigering av lyd. Jeg vil derfor bare slå fast at dette er et emne som må utforskes, men som jeg her ikke vil behandle videre.

4.2.10 Oppsummering

For å summere opp, kan alle punktene samles i ei liste. Generell funksjonalitet som et digitalt redigeringsverktøy bør tilby er:

1. Gi tilgang til et sett videosekvenser (kildemateriale).
2. Mulighet for valg av sekvenser som skal settes sammen til en ferdig video.
3. Avspillingsmekanisme for kildemateriale og master, med mulighet for frysing av bilde og avspilling ved ulike hastigheter.
4. Visualisering av masteren ved hjelp av tidslinje.
   (a) Visualisering av de sekvenser som er redigert sammen.
   (b) Identifikasjon av de enkelte sekvenser.
   (c) Identifikasjon av lengden til de enkelte sekvenser og masteren.
5. Mulighet for valg av inn- og utpunkt i kildematerialet.
6. Mulighet for valg av innpunkt i masteren.
7. Mulighet for sletting av sekvenser i masteren.
8. Et intuitivt grafisk brukergrensesnitt.
9. Hurtig tilgang til de mest brukte og nyttige funksjonene.
10. Tilby ulike overgangseffekter.
11. Gi som resultat en ferdig redigert video.
12. Mulighet for redigering av lyd.
13. Kunne lagre det ferdige resultatet.

4.3 Redigering i databasesammenheng

Når vi ønsker å bruke et redigeringsverktøy i databasesammenheng stiller det oss overfor en del nye spørsmål og utfordringer. Jeg vil i dette kapitlet drøfte noen problemstillinger som er spesielle for redigering av digital video fra en database.

4.3.1 Representasjon

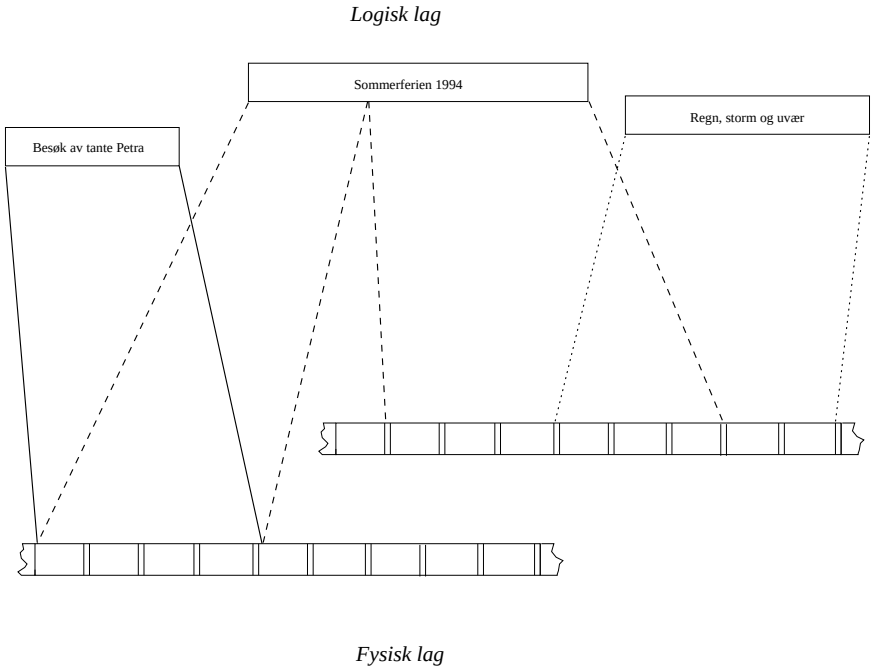
Redigering av videobånd skjer ved kopiering fra kildebånd til et masterbånd. Dette vil si at det foregår ved duplisering av videoinformasjonen. Etter en redigering har vi det samme materialet lagret på minst to fysisk atskilte steder. I en databasesammenheng ønsker vi å kunne kontrollere dupliseringen av informasjon. Vi vil bruke en så liten del av lagringsmediet som mulig. Det er to grunner til dette. Den første årsaken er plassbehov. Digital video inneholder mye data og forutsetter relativt stor lagringskapasitet (se avsnitt 3.1.1). Ved å kontrollere dupliseringen av data kan vi holde lagerbehovet på et minimum. For det andre vil vi få problemer med konsistens og oppdatering hvis den samme informasjonen blir spredd utover på forskjellige steder. En database med minimal duplisering av videoinformasjon blir billigere og enklere å håndtere.

Ferdig redigerte videofilmer kan lagres som referanser til kildematerialet. [Duda] foreslår en algebra for å sette sammen videosekvenser. Andre måter å representere en sammensatt video på, kan være en form for skript eller redigeringslister (EDL). Vi kan betrakte alle videoer som *virtuelle*. Dette gjør at vi skiller mellom fysiske og logiske videosekvenser. Dette er også gjort i databasemodellen fra avsnitt 3.2.1. Her kalles en fysisk lagret videosekvens for *StoredVideoSegment* og den logiske strukturen representeres av *VideoStream* og *VideoDocument*.

En logisk video kan bestå av en eller flere referanser til de fysiske videosekvensene. En videosekvens som vi spiller av vha. videospilleren behøver ikke å lagres fortløpende på harddisken. Denne todelingen er illustrert i figur 4.6.

4.3.2 Endringer

I en database med flere brukere må vi ha regler for hvordan vi kan gjøre endringer og redigeringer i videomaterialet. Siden de fysiske videosekvensene kan være inneholdt i flere logiske produksjoner, kan vi ikke tillate redigering direkte på disse. All redigering må foregå på det logiske plan, altså av de virtuelle videosekvensene. Selv om det er en 1:1 mapping mellom logisk og fysisk lag kan vi ikke tillate redigering direkte på de fysiske sekvensene. Dette vil senere gi oss problemer hvis vi vil sette sammen en video fra flere fysiske sekvenser.



Figur 4.6: Mapping mellom fysisk og logiske sekvenser

Hver produksjon vil dermed starte opp med å definere en ny logisk sekvens. Alt som settes inn i denne er referanser til de fysiske sekvensene. Hvis vi ønsker å sette inn videosekvenser fra andre virtuelle produksjoner, må vi først avbilde dette ned til de fysiske sekvensene. En virtuell video kan altså ikke inneholde referanser til andre virtuelle videoer. Dette kravet kan illustreres med et lite eksempel. Anta at video A består av de fysiske sekvensene X og Y (A=X,Y). Vi ønsker å redigere video B som skal være lik video A etterfulgt av sekvens Z. Hvis B består av referanse til A pluss en referanse til Z (B=A,Z), vil video B være avhengig av at video A ikke redigeres videre. Bestemmer vi oss for at at A=Y,Z så vil også B endres til B=Y,Z,Z istedet for B=X,Y,Z som var den opprinnelige meningen. Konklusjonen er at vi må hele tiden gi referanser ned til det fysiske lag, selv om vi bruker *redigerte sekvenser* som kildemateriale. Ønsker vi å sette B=A,Z, må dette avbildes ned til fysisk lag slik at resultatet blir B=X,Y,Z.

Ønsker vi å tillate referanser til logiske videosekvenser, trenger vi et mer utfyllende regelverk for å gjøre endringer. En videosekvens som er inneholdt i en annen, kan ikke redigeres med mindre man ønsker at redigeringen skal gjenspeile seg *oppover* i hierarkiet. En video A som er inneholdt i B kan ikke redigeres videre. Ønsker vi i dette tilfellet å redigere på A må vi kopiere A og operere på flere versjoner av *samme* video. Dette fører til en mer kompleks database og mange *spesialtilfeller* ved oppdatering og søk.

4.3.3 Effekter

En annen utfordring er hvordan overgangseffekter skal lagres. Det som er aktuelt her er overgangseffekter mellom sekvensene. Andre effekter som krever avansert bildebehandling (se avsnitt 2.5) er interessante som en del av et redigeringsverktøy, men vil ikke bli spesielt behandlet i denne oppgaven. En overgang vil i utgangspunktet være et klipp (se avsnitt 2.4). Dette trenger ingen spesiell representasjon. Det er ellers viktig å kunne representere effekter på en måte som gjør at avspilling av masteren kan skje effektivt.

Hvis vi vil unngå å duplisere data, må det finnes en måte å representere overgangseffektene på i et skript eller redigeringsliste. Dette kan være en form for koding. En effekt kan f.eks. gis koden DIS for dissolve (overtoning) med parametre for antall rammer fra begge sekvensene. For at en slik koderepresentasjon skal fungere er vi avhengige av at effektene kan utføres/behandles i sanntid ved avspilling. Dette er gjerne et svært tøft krav da effektene ofte krever en del bildebehandling. En overtoning krever f.eks. at intensiteten for rammene fra en sekvens blir mindre for hver ramme, samtidig som intensiteten fra den andre sekvensen øker for hver ramme. Et felles bilde for to og to rammer må beregnes, med parametre for intensitet, før det kan legges ut på skjermen. Dette vil ofte ta for lang tid hvis det ikke finnes støtte for å gjøre dette i maskinvaren.

En annen mulighet når det gjelder lagring av overgangseffekter er å lagre overgangene separat som egne sekvenser. Dette vil si at overgangene behandles/beregnes når effekten blir valgt av brukeren. Deretter blir hele overgangssekvensen lagret for seg selv med en referanse i skriptet, som for en *vanlig* videosekvens. Dette er normalt det mest fornuftige tatt i betraktning de datamengder som ellers må behandles i sanntid. Å spille av en effekt som er lagret separat krever bare at man hopper fra sekvens til sekvens, som for et vanlig klipp. En slik løsning fører derimot til et større forbruk av lagringplass. Dette er allikevel ikke et stort problem fordi en video vil inneholde betydelig mer sekvenser som ikke berøres av overgangseffekter. [Mil90] antyder at en halv times fjernsynsshow kan inneholde rundt 200 overganger mellom innstillinger. De fleste av disse vil vanligvis være klipp. Antar vi at $\frac{1}{4}$ av overgangene benytter ulike former for effekter og har en gjennomsnittlig varighet på 2 sekunder (antydet i avsnitt 2.4), får vi at overgangseffekter vil oppta $50 * 2s = 100s$. Resten av showet vil utgjøre $(30min * 60s) - 100s = 1700s$, slik at forholdet mellom disse vil være $\frac{100}{1700} = 0.06$, dvs. overgangene vil utgjøre ca.6% av det lagrede materialet. Dette betyr at "*merkostnadene*" i lagringskapasitet ved denne løsningen ikke vil ha så veldig stor betydning.

4.3.4 Avspilling

Når vi vil spille av skript må vi ha en metode for å tolke skriptene for å få spilt av de riktige videodata. Bruker vi en tredelt modell med videodatabase, videospiller og redigeringsverktøy som grunnlag, kan vi gjøre tolkningen av skriptene i en av de tre delene av systemet.

Redigeringsverktøyet kan tolke skriptene og sørge for å få sendt de riktige videodata til videospilleren til riktig tid. Dette krever egne rutiner som overvåker sending av videodata og holder rede på tid og rammer som avspilles. Dette er kanskje den mest tungvinte metoden i og med at funksjonalitet må dupliseres. Redigeringsverktøyet må innhente informasjon om hva som skjer i videospilleren for å holde rede på progresjonen i skriptet, bytte av sekvens osv. Dessuten vil det ikke være naturlig å la redigeringsverktøyet overvåke prosesser som skjer i database og videospiller.

Tolkingen av skript kan også legges til *databasesiden*. Når databasen får beskjed om at det er et skript som skal spilles av, kan den samle alle videosekvenser og mate videospillere med en kontinuerlig videostrøm. Dette gjør jobben svært enkel for videospilleren som bare spiller av den strømmen den får tilsendt. Denne løsningen gjør at databasen har full kontroll over datastrømmen og de involverte videoene. Derimot vil ikke denne metoden egne seg for redigering fra flere atskilte databaser. Har man en video som er redigert sammen av sekvenser fra flere atskilte databaser over et nett, er det ikke mulig å styre tolkingen av et skript fra en database (videoserver).

Den siste muligheten er at *videospilleren* har muligheten for å lese skript og selv sørge for å be om de riktige videodata fra databasen. Dette er den enkleste og mest fleksible metoden å implementere i og med at videospilleren selv har full kontroll med hvilke rammer som spilles av og hvilke kommandoer som blir gitt. Ulempene med denne løsningen er at det fører til økt kompleksitet i videospilleren. I stedet for bare å vise den videostrømmen som den får tilsendt, må den håndtere referanser til ulike videofiler.

4.3.5 Annotasjoner

Ved redigering av video fra en database, setter vi gamle videosekvenser inn i en ny sammenheng. Redigering kan gi materialet en annen mening enn den opprinnelige (se avsnitt 2.2). Derfor kan det være nyttig å gi også den nye produksjonen en beskrivelse, selv om kildematerialet er beskrevet fra før. Ved å gjøre nye annotasjoner på den ferdige produksjonen kan vi se kildematerialet fra flere synsvinkler. En og samme bildesekvens vil bli beskrevet i forskjellig kontekst og vil kunne gi forskjellig mening. Vi kan oppnå at betydninger ved videomaterialet som ikke ble fanget opp ved første annotering, nå legges inn i databasen. Dette vil igjen gi oss muligheten til bredere søk etter videoinformasjon.

For å kunne gjøre slike annotasjoner mest mulig effektivt er det en forutsetning at brukeren kan hente inn de annotasjonene som allerede er gjort på materialet. Dette er viktig slik at vi slipper å ha flere "lag" med de samme annotasjonene for *en* fysisk videosekvens. Redigeringen setter dermed krav til registreringsverktøyet. Når man skal gjøre annotasjoner til en ny produksjon, må det være mulig å hente opp de annotasjoner som allerede ligger i databasen, for de fysiske sekvenser som er inneholdt i den nye produksjonen. Ved oppstart av en annoteringssesjon må altså registreringsverktøyet kunne

utføre følgende:

1. For hver fysiske sekvens i den nye produksjonen:
   (a) Sjekk hvilke logiske videoer denne er inneholdt i.
   (b) Hent fram annotasjonene til disse.
2. Muliggjør ytterligere annotasjoner på det samme materialet.

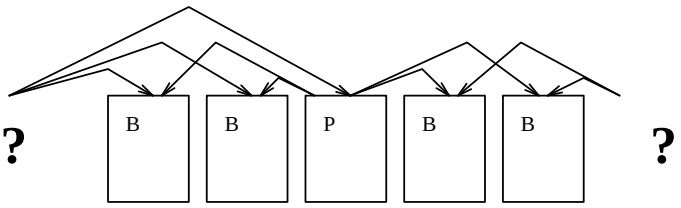
Ved redigering vil man implisitt få kunnskap om hvordan en video er oppdelt i innstillinger. Alt kildemateriale inneholder strukturinformasjon i databasen, som sier hvordan en video er delt opp i sekvenser, scener og innstillinger. For hver sammensetting i redigeringsverktøyet, skapes et nytt innstillingsskifte (se avsnitt 2.1). Dette gjør at vi vil inneha en fullstendig oversikt over alle innstillinger ved redigering. Vi kan dra nytte av dette ved å overføre denne strukturinformasjon til databasen. En slik oppdeling vil senere lette arbeidet for strukturering i scener og sekvenser.

Fordi scener og sekvenser sier noe om innholdet i videoen, får vi ingen implisitt kunnskap om dette i redigeringsprosessen. Denne strukturingen må altså gjøres ved hjelp av registreringsverktøyet. Det er dermed en fordel at registreringsverktøyet støtter grupperingen av innstillinger til scener og sekvenser. Det kan arumenteres for at en redigerer kjenner til inndelingen i scener og sekvenser og at han kan legge inn denne informasjonen i databasen. Dette kan føre til en sterkere interaksjon mellom registrerings- og redigeringsverktøyet, og man får gjort alt i en prosess i stedet for å registrere i ettertid. Derimot vil det ikke være naturlig for en redigerer å registrere slik informasjon. En redigerer er hovedsaklig opptatt av å gjøre en kreativ redigeringsjobb, og er ikke interessert i å bruke mye tid på registrering og annet arbeid. Dette synet bekreftes av Christian Grotnes i NRK.

4.3.6 Komprimering

Det er viktig å se på hvordan de ulike komprimeringsmetodene påvirker redigering mot en database. Hovedoppgaven til et redigeringsverktøy er jo at det skal kunne sette sammen utvalgte videosekvenser til en ny produksjon. For å få til dette er det nødvendig å ha tilgang til hver enkelt ramme av videosekvensene. Av de komprimeringsmetodene som er nevnt i avsnitt 3.1.2, er det bare JPEG som komprimerer hver enkelt ramme for seg selv. Dette gjør JPEG til et spesielt godt egnet i redigeringssammenheng fordi det er enkelt å gi referanser til hver enkelt ramme innen en videosekvens.

Redigering av MPEG og H.261-komprimert video er ikke så lett som bare å gi referanser til de enkelte rammene. Begge disse komprimeringsmetodene koder ikke hver ramme som *en* uavhengig enhet. Kodingen av en ramme er avhengig av andre rammer i samme sekvens (se avsnitt 3.1.2). For å få en fullstendig video er vi avhengig av at for hver gang vi hopper til en ny fysisk sekvens, må vi kunne dekomprimere og beregne de rammene vi vil



Figur 4.7: Problemer ved avspilling av vilkårlig MPEG-sekvens

benytte. Problemet er illustrert i figur 4.7. Den enkleste løsningen på dette problemet er å tillate redigering av kun uavhengige sekvenser, dvs. en sekvens mellom to I-rammer for MPEG og mellom to Intra-rammer for H.261. Dette vil imidlertid ikke være tilfredsstillende i alle sammenhenger. Ofte vil man redigere på en mer nøyaktig måte, og velge sekvenser med nøyaktighet ned til den enkelte ramme. Dette kan løses på to ulike måter. Den ene muligheten er å ha en mekanisme for å beregne de *løse* rammene, dvs. rammer som ikke har de nødvendige referanser til uavhengig kodete rammer. Ved avspilling må dette kunne skje raskt, slik at det ikke blir noe opphold når vi skifter fra en fysisk sekvens til en annen. Det stiller store krav til maskinvaren for å kunne gjøre disse beregningene hurtig nok. Den andre muligheten er at man beregner en ny videosekvens på grunnlag av kildematerialet, som inneholder de nødvendige I- eller Intra-rammer, slik at det blir en selvstendig enhet. Denne nye videosekvensen må da lagres separat, og dette bryter med intensjonen om ikke å duplisere videomateriale.

4.3.7 Koding

Ved digitalisering av video har man mange muligheter til å påvirke kodingen av videoen. Vi kan ha flere typer komprimering i tillegg til at det er mulig å sette ulike parametre for hver type av komprimering, eksempelvis bildestørrelse og rammerate. Ved å tillate video med ulik koding, skaper dette problemer ved redigering og avspilling av dette materialet. Vi har hovedsaklig tre ulike måter å løse problemet på.

1. Late som det ikke er et problem[3], dvs. spille av videoen slik den er. Følger: Rammerate og bildestørrelse kan endre seg underveis i avspillingen.

2. Ikke tillate redigering av video med ulik koding. Følge: All video må kodes likt ved digitalisering.

3. Transformere videoen til en bestemt koding ved redigering. Følger: Dette setter store krav til redigeringsverktøyet som må komprimere/-ekspandere bildestørrelse, regne om rammerate og duplisere evt. kaste rammer.

[3] Kjent reaksjonsmønster fra atferdspsykologi

Enda større utfordringer vil vi støte på ved å tillate videoer med ulike komprimeringsmetoder i databasen. Dette betyr at vi må ha mulighet for sanntid dekomprimering av de ulike typene. I dagens maskiner er det mest vanlig å gjøre dekomprimering ved hjelp av egne videokort, dvs. dedikerte prosessorer som gjør dekomprimeringen. Et eksempel på dette er Parallax videokort som komprimerer og dekomprimerer JPEG. En database med flere ulike komprimeringsmetoder vil dermed kreve flere typer videokort og at det er mulig å bytte mellom disse kortene uten tidstap.

Konklusjonen på dette er at det enkleste er å ha en homogen database hvor all video er kodet likt. Dette er en forutsetning for implementeringen av en prototyp i denne oppgaven.

4.4 Profesjonelle redigerere vs. amatører

Profesjonelle og amatører vil ha ulike forventninger og krav til et redigeringsverktøy. Jeg vil her forsøke å beskrive noen av de ulike kravene.

4.4.1 Profesjonell bruk

En profesjonell redigerer som er vant med å bruke annet redigeringsutstyr vil ønske seg en så liten overgang som mulig fra det utsyret han har brukt tidligere [Eng94]. F.eks. vil mange redigerere ønske å styre datamaskinen med fysiske spaker og ratt, som er vanlig på analogt redigeringsutstyr. Et fullstendig datamaskingrensesnitt vil for en profesjonell redigerer ofte representere noe helt nytt som det vil ta tid å venne seg til. I tillegg vil man føle at det gamle utstyret er raskere å jobbe på fordi man kjenner det svært godt. Det kan også være en annen side av denne saken, nemlig at det analoge utstyret/grensesnittet virkelig *er* spesielt godt egnet til å redigere med. Dette er avveininger som må tas ved implementasjon av et brukergrensesnitt for profesjonellt bruk. Følelsene for og mot et *datamaskingrensesnitt* vil variere fra person til person.

Ellers setter profesjonelle brukere store krav til bilde- og lydkvalitet. Det er vanskelig å sette noe mål på hva som er god nok bildekvalitet. I NRK bruker man en *jurygruppe* som fastsetter hva som er akseptabelt. Et verktøy som tenkes brukt til for-redigering har ikke like høye krav til bildekvalitet, i og med at man senere i produksjonen vil benytte on-line utstyr.

Nedenfor følger ei liste over ulike krav som profesjonelle redigerere setter til et diskbasert redigeringsverktøy. Kravene er stort sett hentet fra interne notat fra NRK ([Eng94], [Gro94] og [Van94]).

1. Et grensesnitt som er *minst mulig datamaskin*, dvs. mest mulig fysiske spaker og ratt, som ved analoge redigeringsomgivelser.
2. Mulighet for hurtig tilgang til de viktige funksjonene (ikke for mye menyer).

3. Overgangseffekter (overtoning og wipe) i sanntid.
4. Avansert redigering av lyd.
5. Mulighet for flere lydspor.
6. Enkel og visuell tilgang til å foreta justeringer av klippepunkt, forlenging og forkorting av videosekvenser.
7. Mulighet for å gjøre mest mulig redigering direkte på tidslinjen.
8. Mulighet for sakte avspilling (slow-motion) med lyd.
9. Justeringsmuligheter for bildeparametre, f.eks. fargetonekorreksjon.
10. Høy driftssikkerhet.

4.4.2 Amatører

En amatørredigerer vil vanligvis ikke ha så mange preferanser i forhold til hvordan utstyret bør være, fordi han ikke har så mye erfaring fra redigering. Nettopp derfor er hovedkravet for amatører at verktøyet er enkelt å bruke. Verktøyet bør ha et enkelt grensesnitt og en funksjonalitet som er lett å lære. Det er viktig for en amatør at det er lett å komme i gang med redigeringen. Verktøyet bør være enkelt og intuitivt å jobbe med, men bør ikke inneholde begrensninger i funksjonalitet, som gjør at man ikke kan jobbe med dette utover *opplæringsfasen*. I tillegg vil gjerne amatørbrukere ønske seg muligheten til å legge inn tekst, symboler og kanskje grafikk i bildet for å illustrere videoen, eksempelvis ved redigering av presentasjonsvideoer.

Et eksempel på et verktøy som er lett å komme i gang med, er Integrated Video (beskrevet i avsnitt 4.1.2). Dette verktøyet har et enkelt og lettforståelig grensesnitt, men tilbyr også mer avansert funksjonalitet.

Kapittel 5

Konstruksjon

I dette kapitlet vil jeg se på konstruksjon og implementasjon av et redigeringsverktøy som del av et totalsystem for digital video. Beskrivelsene vil her dreie seg om en prototyp som skal illustrere noen punkter jeg anser for viktige. Det jeg vil ta for meg i dette kapitlet baserer seg på den eksisterende infrastruktur ved IDT og NR, og jeg vil forholde meg til de begrensninger som ligger i at dette er tenkt som en del av en demonstrator i LAVA-prosjektet.

5.1 Begrensninger

Jeg vil her kort beskrive de begrensninger som vil gjelde for denne prototypen.

I tilknytning til LAVA-prosjektets deltagere er det utviklet to ulike digitale videospillere. Den ene er utviklet av Sigurd Skeide [Ske93] ved IDT og den andre utviklet av Gisle Aas og undertegnede ved Norsk Regnesentral [Aas94]. Den spilleren som vil bli brukt i denne oppgaven er sistnevnte. En av grunnene til dette er at redigeringsverktøyet skal ende som en del av en demonstrator i LAVA-prosjektet. Dette medfører samkjøring med NR og DELAB, som vil bruke den samme spilleren til søkeverktøyet. Videospilleren er nærmere beskrevet i vedlegg B.

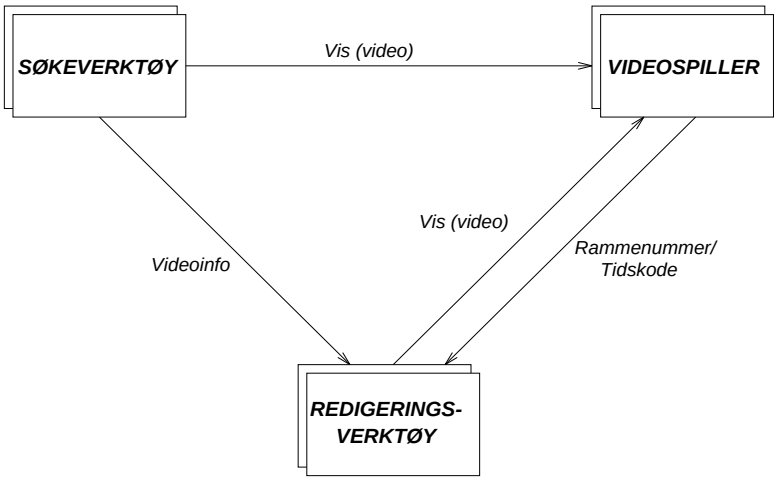
Både ved IDT og NR har man ved utviklingen av applikasjoner i tilknytning til video basert seg på **JPEG** komprimering. Dette gjør at jeg ikke vil se videre på problemstillinger i tilknytning til andre komprimeringsmetoder i dette kapitlet. Videospilleren er avhengig av å kjøre på en Sun arbeidsstasjon med installert **Parallax** videokort. Videokortet utfører komprimering og dekomprimering av JPEG.

Jeg vil også benytte funksjonaliteten til det **søkeverktøyet** som utvikles ved DELAB innenfor LAVA-prosjektet. Dette er et søkeverktøy mot en SYBASE database.

5.2 Samspill med andre verktøy

Redigerigsverktøyet skal, som tidligere nevnt, være en integrert del av et totalsystem for digital video. I en slik intergrert løsning har vi en database, en videospiller, et søkeverktøy mot databasen og et registreringsverktøy. Mange av redigeringsverktøyets egenlige oppgaver kan utføres av videospilleren og søkeverktøyet. Jeg vil her behandle hvilke krav som settes til de andre verktøyene for å kunne utnytte disse ressursene i redigeringsverktøyet.

Samspillet mellom redigeringsverktøyet, søkeverktøyet og videospilleren er illustrert i figur 5.1. Registreringsverktøyet vil ikke inngå som en del av LAVA-demonstratoren og er dermed utelatt fra denne figuren.



Figur 5.1: Samspillet mellom verktøyene

5.2.1 Videospilleren

For at man skal kunne ha en mulighet til å spille av den produksjonen man jobber med, er det nødvendig med en kommunikasjon mellom redigeringsverktøy og videospilleren. Det som må styres fra redigeringsverktøyet er avspilling av enkeltsekvenser og flere videosekvenser etter hverandre. Det siste er viktig for å kunne få avspilt den produksjonen man jobber med, dvs. masteren. Det er altså viktig at videospilleren kan spille en form for skript. Et skript vil da være en samling referanser til ulike videosekvenser som kan avspilles som en video.

Videospilleren må kunne styres med "vanlige" kommandoer som *play, rewind, fastforward, pause* osv. Dette betyr at spilleren i sin helhet må kunne styres fra søke- eller redigeringsverktøyet, hvis det er ønskelig.

I tillegg vil redigeringsverktøyet gjerne få tilbakemelding om hvor i avspillingen videospilleren befinner seg. Aktuell informasjon er hvilken video og

hvilket rammenummer den befinner seg på. Dette er viktig for å kunne redigere video samtidig med at man ser den på skjermen. For å få satt inn- og utpunkt i en videosekvens må man ha muligheten til å stille inn videoen på ønskede punkter og markere inn- eller utpunkt i redigeringsverktøyet. Redigeringsverktøyet må da kunne spørre videospilleren om hvor denne befinner seg i filmen. Returverdi må være av typen tidskode (se avsnitt 2.8) eller rammenummer, som bestemmer entydig hvilken ramme som avspilles.

5.2.2 Søkeverktøy

Søkeverktøyet kan overta funksjonen med å presentere kildemateriale for brukeren. Jeg antar her at videomaterialet er digitalisert og registrert ferdig i forkant av redigeringsprosessen. Dermed er utgangspunktet en videodatabase med video og beskrivelser av materialet.

Vi kan betrakte alt som ligger i databasen som kildemateriale for redigering. Søkeverktøyet benyttes til søk i videomaterialet og resultatet av et søk presenteres i en *treffliste* på en dertil egnet måte. Brukeren må kunne se igjennom videosekvenser fra trefflista og overføre de ønskede sekvenser til redigeringsverktøyet. Dette betyr at søkeverktøyet må inneha funksjonalitet for å overføre informasjon om videomaterialet. Dette kan skje på to måter. Redigeringsverktøyet kan be om å få informasjon om videosekvensen som er avmerket i trefflista, eller søkeverktøyet kan ha en egen funksjon for eksplisitt å sende informasjon til redigeringsverktøyet. I dette prosjektet vil den siste metoden bli benyttet.

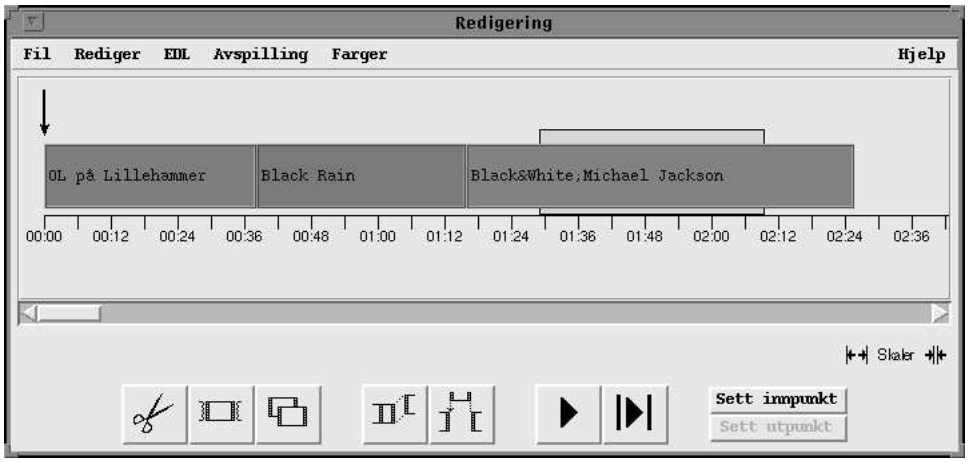
I f.eks. Avid er det mulig å velge inn- og utpunkt samtidig som man ser igjennom kildematerialet. Det er imidlertid ikke naturlig å legge inn den typen redigeringsfunksjonalitet i hverken søkeverktøyet eller videospilleren, i og med at de skal kunne benyttes som selvstendige verktøy også når man ikke driver med redigering.

5.3 Redigeringsverktøy

Her vil jeg presentere konstruksjonen av et redigeringsverktøy for digital video. Konstruksjonen er gjort ut fra forutsetningen om at verktøyet skal være integrert i et totalsystem. En annen er en viktig forutsetning at dette verktøyet ikke skal forsøke å ta opp konkurransen med dagens kommersielle systemer. Redigeringsverktøyet skal kunne brukes i en grovredigeringssituasjon. **Uthevede ord** refererer til diskusjonen i avsnitt 4.2.

5.3.1 Brukergrensesnitt

Brukergrensesnittet består av en menylinje, en tidslinje og et sett knapper. All funksjonalitet skal være tilgjengelig gjennom menylinjen øverst i grensesnittet. De mest brukte funksjonene presenteres som knapper nederst i



Figur 5.2: Grensesnittet til redigeringsverktøyet

brukergrensesnittet. Hensikten med dette er at de mest brukte funksjonene skal være lett tilgjengelig for brukeren.

5.3.2 Funksjonalitet

Utgangspunktet for redigeringen er **kildemateriale**, som tilbys via søkeverktøyet. Brukeren kan benytte søkeverktøyet til å søke etter video i databasen. Dette vil resultere i en treffliste over videosekvenser som tilfredsstiller søkekriteriene. Brukeren kan se igjennom sekvensene og overføre de ønskede sekvensene, dvs. informasjon om disse, til redigeringsverktøyet. Informasjon som er nødvendig om kildematerialet er

- Tittel – For identifikasjon av hver sekvens.
- Referanse – Identifikasjon av hvor videosekvensen er lagret.
- Startpunkt – Starten av videosekvensen (tidskode).
- Sluttpunkt – Slutten av videosekvensen (tidskode).
- Båndnummer – For utskriving av EDL.

Denne informasjonen overføres til redigeringsverktøyet og behandles videre der.

Som alle andre ikke-lineære redigeringssystemer skal også dette verktøyet benytte en **tidslinje** i redigeringen. Når informasjon om en videosekvens blir sendt fra søkeverktøyet, legges det inn en representasjon av dette på tidslinjen. Representasjonen av en sekvens er et rektangel. For å skille mellom de ulike sekvensene benyttes fargekoding på rektanglene, i tillegg til at hvert rektangel inneholder navnet til videosekvensen. *En* farge representerer *en* tittel, noe som gjør det lett å skille mellom to sekvenser.

Tidslinjen skal inneholde en tidsskala, representasjon av de ulike sekvenser og en peker som avbilder en *nåtilstand* i masteren. *Nåpekeren* skal kunne flyttes/dras bortover tidslinjen. Samtidig skal videospilleren få informasjon om hvilken video og tidskode som pekeren peker på, slik at videospilleren hopper til det aktuelle stedet i videoen.

Brukeren skal kunne benytte musa til å merke av områder på tidslinjen. Etterhvert som brukeren merker av med musa tegnes det opp et rektangel som viser det som er avmerket. Man kan så manipulere på det avmerkede området. Funksjoner som kan utføres på et avmerket område er *klipp, kopier* og *avspilling*. I tillegg tilbys en funksjon vi kan kalle *behold*, som gir brukeren mulighet til å merke av et område innenfor *en* videosekvens for så å forkaste det resterende av denne sekvensen.

Det skal være en mulighet for **avspilling** av de sekvenser som er lagt inn på tidslinjen. Brukeren kan velge mellom å spille fra *nåpekeren* eller å spille av en avmerket videosekvens. Redigeringsverktøyet sender da informasjon om det som skal spilles av til videospilleren.

Sekvenser blir **satt inn** sist på tidlinjen. Dette blir altså utgangspunktet for en redigering. Etter sekvensen er satt inn, kan brukeren manipulere med sekvensen vha. de funksjoner som er skissert ovenfor.

Dette verktøyet vil ikke tilby andre **overgangseffekter** enn vanlig klipp. Siden verktøyet ikke er beregnet på annet enn grovredigering har det i utgangspunktet ikke behov for andre effekter. Dessuten krever overgangseffekter en god del bildebehandling, slik at dette ikke er praktisk mulig å implementere innenfor rammen av denne oppgaven.

5.3.3 Lagring

Resultatet fra en redigering er hovedsaklig en EDL. I tillegg skal masteren lagres på et format som gjør at den kan hentes inn igjen i redigeringsverktøyet og redigeres videre. Masteren må også kunne lagres på et format som kan spilles av videospilleren. I og med at videospilleren i dag (dessverre) ikke har mulighet for å spille skript, setter ikke dette noen krav til hvordan lagringsformatet skal være. Jeg velger derfor å lagre ferdig redigert video på et format som kan tas inn i redigeringsverktøyet igjen for videre redigering. For å gjøre dette mest mulig fleksibelt vil jeg ta utangspunkt i en enkel EDL og lagre masteren på dette formatet. På den måten slipper vi å skille mellom lagring av EDL og et eget skriptformat for masteren. Videospilleren kan senere utvides til å tolke dette formatet, dvs. lese en EDL og spille av de aktuelle videoreferansene.

5.3.4 EDL

Som nevnt i avsnitt 2.9, finnes ingen entydig standard for editlister. Dermed vil jeg definere min egen standard som er enkel, og tjener kun illustrasjons-

formål. Jeg har ikke gjort noen bestrebelser på å finne et format som kommuniserer med on-line utstyr. Dette kan være en av utvidelsesoppgavene, hvis dette verktøyet skal videreutvikles.

Redigeringsverktøyet skal altså kunne lese og lagre på dette formatet:

```
AEB.EDL.0.1
Editliste : <tittel>
------------------------
<båndnummer> <innpunkt> <utpunkt> <filreferanse>
<båndnummer> <innpunkt> <utpunkt> <filreferanse>
<båndnummer> <innpunkt> <utpunkt> <filreferanse>
    :
    :
```

`AEB.EDL.0.1` er en kode for å forsikre redigeringsverktøyet, ved innlesning, at det leser en EDL det forstår. `innpunkt` og `utpunkt` skal være på SMPTE format (se avsnitt 2.8). `filreferanse` må være på et format som gjør at videospilleren kan spille videoen (se avsnitt 5.3.5).

5.3.5 Grensesnitt mot andre verktøy

Redigeringsverktøyet kommuniserer med videospilleren via programmet `vsend`. Dette programmet gir muligheten til å styre videospilleren fra eksterne programmer som er uavhengige av videospilleren. `vsend` kan sende et utvalg styringskommandoer til videospilleren. Dette er nærmere beskrevet i vedlegg B.

De kommandoene som det er aktuelt å sende fra redigeringsverktøyet til videospilleren er hovedsaklig `Where` og `File`.

```
vsend File <filref> <posisjon>
```

benyttes for å starte avspilling av en video ved en gitt posisjon, og med parameteren `-stop` kan man gå til en bestemt ramme av videoen og stoppe der.

```
vsend Where <filref>
```

benyttes for å spørre hvor videoen befinner seg i øyeblikket. Dette utnyttes til å oppdatere tidslinjen og nåpekeren i redigeringsverktøyet.

Søkeverktøyet sender videoinformasjon ved at det kaller opp funksjonen `newentry` i redigeringsverktøyet. Kallet ser slik ut:

```
newentry <tittel> <båndnr> <båndtype> <innpunkt> <utpunkt>
        <fokus-offset> <fokus-lengde> <filref> <bånd-offset>
```

Parametrene til dette kallet tilsvarer de ulike parametrene i tabellen `entries` (se avsnitt 5.4.3 for nærmere beskrivelse). Denne løsningen er mulig fordi begge applikasjonene implementeres i Tcl/Tk. Kommandoen `send` i Tcl ordner denne typen kommunikasjon mellom ulike applikasjoner [Oust94].

5.4 Implementasjon

Jeg vil her beskrive de overordnede trekkene ved implementasjonen av en protype for redigering av digital video. En beskrivelse av detaljene ved implementasjonen finnes i et eget vedlegg.

5.4.1 Programmeringsspråk

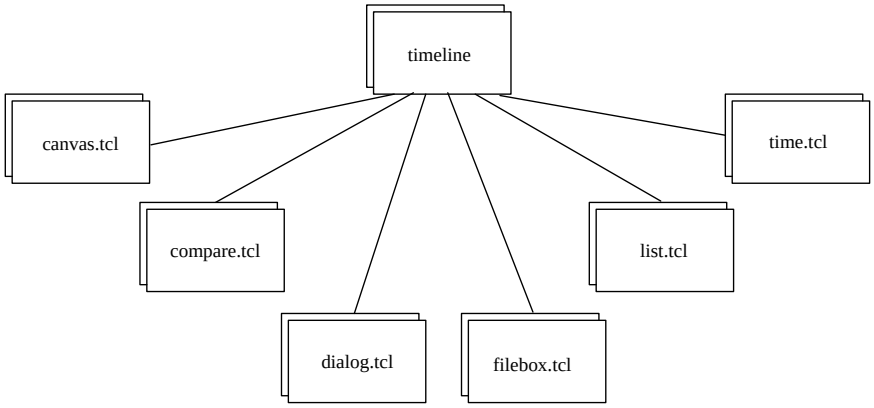
Jeg vil til denne prototypen programmere i **Tcl/Tk**[1] (se [Oust94] og [Welch94]). **Tcl** er et skript-språk utviklet av John Ousterhout. **Tk** er en verktøykasse (toolkit) for å bygge grafiske brukergrensesnitt. Tk bygger *Motif*-lignende grensesnitt.

Tcl/Tk er spesielt gunstig i denne oppgaven av to grunner. For det første tilbyr Tcl/Tk en enkel mekanisme for kommunikasjon mellom applikasjoner. Alle Tcl/Tk-applikasjoner kan sende et skript til en annen Tcl/Tk-applikasjon slik at det blir utført der. Dette gjør Tcl/Tk spesielt godt egnet til applikasjoner som vil kommunisere med flere andre applikasjoner. For det andre er det svært enkelt å implementere enkle grensesnitt i Tcl/Tk. Tk tilbyr høynivå programmering av vindusbaserte applikasjoner. Dette gjør at Tcl/Tk er meget godt egnet til å prøve ut idéer for et enkelt redigeringsverktøy.

5.4.2 Filstruktur

Redigeringsverktøyets hovedfunksjoner er implementert i filen `timeline`. Denne filen inneholder hoveddelen av redigeringsverktøyet. I tillegg har jeg implementert en del hjelpefunksjoner for å få abstrahert en del funksjonalitet som er svært generell. Disse finnes i filene `canvas.tcl`, `compare.tcl`, `list.tcl` og `time.tcl`. Et generell fildialogboks er implementert i `filebox.tcl` og et generelt dialogvindu er implementert i `dialog.tcl`. Alle `.tcl`-filene inkluderes i `timeline`. I tillegg til dette kommer filene som definerer bitmaps til de ulike knapper i grensesnittet. Dette er `copy.xbm`, `cut.xbm`, `delete.xbm`, `insert_at.xbm`, `insert_last.xbm`, `keep.xbm`, `play.sel.xbm`, `play.xbm`, `scale_down.xbm` og `scale_up.xbm`.

[1] Uttales tikkl-ti-kei



Figur 5.3: Filstrukturen

5.4.3 Oversikt

Den interne representasjonen av tidslinjen baserer seg på to strukturer. Rekkefølgen av sekvensene på tidslinjen bestemmes av listen `entry_order`. Denne listen inneholder nøkler til alle sekvensene i den rekkefølge de opptrer på tidslinjen. Nøklene benyttes som *pekere* til tabellen `entries`.

En videosekvens på tidslinjen tilsvarer et innslag i tabellen `entries`. En sekvens gis en nøkkel, som kun er på formatet `.e<heltall>`. Til hver sekvens hører parametrene:

`title`	Tittelen til en sekvens.
`tape_nr`	Båndnummer for originalbåndet.
`tape_type`	Båndtype. Dette er viktig for NRK-bruk fordi man vil kontrollere at det finnes avspillingutstyr til den aktuelle båndtypen.
`tape_in`	Innpunkt i sekvensen, format tt:mm:ss:rr.
`tape_out`	Utpunkt i sekvensen, format tt:mm:ss:rr.
`focus_offset`	For avspilling av kun en del av en digitalisert video.
`focus_length`	For avspilling av kun en del av en digitalisert video.
`length`	Lengden til sekvensen i antall rammer (gitt en rammerate på 25 rammer/s).
`fileref`	Referanse til hvilken fil sekvensen er fra.
`tape_offset`	For å vise riktig tidskode i videospilleren.
`button_id`	En identifikator for knappen som representerer sekvensen på tidslinjen.

Et innslag er i sin helhet definert ved `entries(.e<heltall>.<parameter>)`, hvor `<parameter>` er av de som er nevnt ovenfor. Eks. første sekvens som legges inn defineres av `entries(.e0.title), entries(.e0.tape_in), entries(.e0.tape_out) osv.`.

Til å tegne opp tidslinjen brukes et `canvas`. Sekvensene legges ut på tidslinjen representert ved knapper (buttons). Utleggingen av knapper tar ut-

gangspunkt i et origo (`origin_x, origin_y`) som representerer tid 0. Knappene legges så ut med tidsaksen mot høyre, fra origo.

Variabelen `scale_factor` holder rede på lengen av tidsenhetene bortover tidslinjen. For å bestemme lengde på tidslinjen og de enkelte knappene brukes en viktig konvensjon:

$$Et\ sekund = \frac{25\ piksler}{scale\_factor}\ i\ x-retningen$$

som betyr at $scale\_factor >= 1$.

Tidskoden til et vilkårlig punkt x langs x-aksen i canvaset finnes dermed av formelen

$$int2time((x - origin\_x) * scale\_factor)$$

hvor `int2time` er en funksjon som regner om fra eksakte rammenummer til tidskode, basert på 25 rammer/s.

Redigeringsfunksjonene klipp og kopier er implementert ved hjelp av et *utklippstavle*-konsept (clipboard). Referanser til de sekvensene som klippes ut eller kopieres, legges inn i en liste som kalles `clipboard_list`. Dette er en liste av samme type som `entry_order`. Listen inneholder nøkler til de `entries` som er kopiert eller klippet ut. Ved kopiering opprettes nye `entries` og nøklene til disse legges inn i `clipboard_list`. Ved klipp flyttes de aktuelle nøkler fra `entry_order` til `clipboard_list`. Det som er lagt inn i `clipboard_list` kan legges tilbake på tidslinjen ved hjelp av to funksjoner. Enten kan sekvensene legges inn sist på tidslinjen, eller de kan legges inn ved posisjonen til *nåpekeren*.

En detaljert beskrivelse av implementasjonen, i tillegg til kildekoden, finnes i et eget vedlegg til denne rapporten.

Kapittel 6

Oppsummering

6.1 Systembetraktninger

I dette kapitlet vil jeg vurdere prototypen og dens funksjonalitet. Spesielt vil jeg fokusere på sider som ikke er spesielt gustige og ting som kunne ha vært gjort anderledes.

6.1.1 Totalvurdering

Prototypen som er utiklet i denne oppgaven er et enkelt redigeringsverktøy, som gir brukeren mulighet til enkelt å manipulere videosekvenser fra en database. Det er et oversiktlig verktøy som er relativt lett å bruke. Verktøyet gjør det enkelt å manipulere på ulike videosekvenser ved funksjonene *klipp, kopier* og *behold*. Tidslinjen er enkel og oversiktlig, og det er lett å skille mellom de ulike videosekvensene ved hjelp av fargekodingen. Fargekodingen er spesielt nyttig når videosekvensene er relativt korte og man ikke kan se tittelen inne i rektangelet som representerer sekvensen.

Verktøyet egner seg til grovredigering, dvs. redigering som ikke krever svært stor nøyaktighet. Grunnen til at det er noe vanskelig å oppnå svært nøyaktig redigering ligger i navigeringsfunksjonene i selve videospilleren og kommuniksjonen mellom videospiller og redigeringsverktøy. Dette er områder som kan forbedres.

6.1.2 Kommunikasjon mellom applikasjonene

En av forutsetningene for redigeringsverktøyet som er utviklet i denne oppgaven var at det skulle inngå som en del av et totalsystem for digital video. Dermed er man delvis avhengig av funksjonaliteten til de andre verktøyene i totalsystemet. Dette setter også en del begrensninger for hva som kan implementeres. Både søkeverktøyet og videospilleren skal fungere som selvstendige verktøy, også utenfor en redigeringssituasjon.

Den største utfordringen med å implementere disse verktøyene som selvstendige applikasjoner er kommunikasjonen mellom dem. Redigeringsverktøyet er avhengig av å få informasjon fra søkeverktøyet om de enkelte videosekvenser. Fra videospilleren er det nødvendig at redigeringsverktøyet mottar informasjon om hva som avspilles i øyeblikket, i tillegg til at man kan ha styring over videospilleren fra redigeringsverktøyet.

Kommunikasjonen mellom redigeringsverktøyet og søkeverktøyet er her realisert gjennom programmeringsspråket Tcl. Språket tilbyr enkle rutiner for kommunikasjon mellom to applikasjoner som er implementert i Tcl. For applikasjoner som er avhengige av mye kommunikasjon mellom dem er det svært enkelt å realisere denne mekanismen i Tcl.

Videospilleren er implementert i C++, så her har vi ikke den samme enkle muligheten til kommunikasjon. Til videospilleren hører et hjelpeprogram som heter `vsend`. Dette benyttes her til å realisere kommunikasjonen mellom videospiller og redigeringsverktøyet. Et problem med dette er at `vsend` er en uavhengig applikasjon, som må startes opp for hver gang en kommando skal sendes fra eller til videospilleren. Dette betyr at det tar en viss tid fra man sender en kommando fra redigeringsverktøyet til det når fram til videospilleren. Dermed blir det vanskelig å få en synkron skjermoppdatering.

Det er ønskelig å ha en peker på tidslinjen som hele tiden oppdateres til å vise hvor i masteren videospilleren befinner seg ved avspilling. Slik det er i dag må redigeringsverktøyet benytte `vsend` til å spørre videospilleren eksplisitt hvor den befinner seg. Dette tar lang tid og er ikke gjennomførbart i praksis. For det første vil videospilleren ha spilt et godt stykke videre før *svaret* kommer tilbake til redigeringsverktøyet. For det andre vil vi med denne tilnærmingsmåten, ha en egen løkke i redigeringsverktøyet som hele tiden sender forespørsler til videospilleren, samtidig som den sjekker hendelser i brukergrensesnittet. Dette virker vanskelig gjennomførbart. Et slikt problem kunne blitt løst med kommunikasjon via. *X properties*, men dessverre tilbyr ikke Tcl dette. Det er mulig å lenke inn egne C-rutiner i Tcl slik at det er mulig å implementere en egen rutine for å kommunisere via. X properties. Dette er også planlagt som en utvidelse av Tcl når versjon 4.0 av språket blir lansert[1]. Med muligheten for *properties-kommunikasjon* kunne videospilleren ha skrevet til en property for hver gang den har skiftet ramme (evt. etter et visst antall rammer) og redigeringsverktøyet kunne ha fått melding om når denne er endret slik at pekeren kunne ha blitt oppdatert.

6.1.3 Tidslinjen

Ikke-lineær redigering med bruk av tidslinje er en stor fordel i forhold til tradisjonell lineær redigering. Dette gjelder først og fremst muligheten til å sette inn sekvenser på en vilkårlig plass i masteren i tillegg mulighetene for enkelt å kunne markere en vilkårlig sekvens for senere å manipulere denne.

[1] Sannsynligvis i løpet av 1995

Man kan allikevel ønske seg mere funksjonalitet som gjelder redigering direkte på tidslinjen. At mest mulig funksjonalitet kan utføres direkte på tidslinjen fører til at brukeren beholder fokus på et område, istedet for å lete etter knapper, menyer o.l. Dette vil igjen føre til en forenkling av selve redigeringsprosessen. Det bør være mulig å *ta tak i* sekvenser med musa og dra dem ut i tid. Slik prototypen er i dag må man slette en sekvens og sette inn på nytt hvis en sekvens skal forlenges.

En annen utvidelse av fuksjonaliteten til tidslinjen vil være å kunne representere hver enkelt sekvens med et bilde/ikon istedet for bare en tittel. Dette setter krav til både programmeringsspråket og databasen. Man må for hver videosekvens kunne finne et representativt ikon. Dette kan f.eks. plukkes ut av brukeren og da krever dette at man har et ikon lagret i databasen som er tilknyttet hver sekvens. Det vil være naturlig å knytte et slikt ikon til hver logiske videosekvens siden det er disse som vil være resultat av et søk i databasen. Et ikon vil kunne være en god illustrasjon for en sekvens. I [Ben93] foreslår man en metode for å plukke ut et representativt bilde ut av en sekvens med bider. Dette kan eventuelt benyttes til å plukke ut et representativt ikon, slik at man ikke behøver å lagre dette i databasen som en egen enhet. For begge metodene setter det krav til programmeringsspråket som benyttes i implementasjonen av verktøyet. Tk tilbyr i utgangspunktet ikke fremvisning av andre bildeformater enn vanlig *X bitmap*. Dette er ikke et godt egnet format til å fremstille detaljerte bilder. I versjon 4.0 av Tcl/Tk vil man få nye muligheter til å vise fram bilder kodet i *gif*-format som er bedre egnet til dette formålet. Dessuten finnes flere utvidelser (extensions) til Tcl/Tk som kan benyttes til fremvisning av ulike bildeformat.

6.1.4 Interaktivitet

En stor forskjell fra denne prototypen og f.eks. Avids Film Composer er det faktum at man i prototypen ikke kan sette inn- og utpunkt i kildematerialet før man legger en sekvens inn i masteren. Dette fører til at man først må legge inn hele den ønskede sekvens og deretter manipulere denne slik at man starter og slutter på riktig sted. Grunnen til at dette ikke er implementert i denne prototypen er hovedsaklig at det ikke er naturlig å legge denne typen funksjonalitet inn i hverken søkeverktøyet eller videospilleren. Det kan imidliertid realiseres i redigeringsverktøyet ved at sekvensen legges midlertidig inn i en *dialogboks* hvor man har muligheten til å forkorte, eller forlenge sekvensen, før den legges inn på tidslinjen. Det må i så fall vurderes om dette blir forrvirrende for brukeren i forhold til den ulempen med ikke å ha denne funksjonaliteten i det hele tatt.

6.1.5 Avspilling

En av de aller største ulemper med dagens system er at videospilleren som jeg har basert arbeidet på, og som er en del av LAVA-demonstratoren, ikke kan spille *skript*. Betegnelsen skript er her benyttet om en rekke referanser

til fysisk adskilte videosekvenser som skal avspilles fortløpende. Dette gjør at det ikke er mulig å spille av masteren i sin helhet. Det er kun mulig å spille av sekvens for sekvens. Dette er helt klart en stor ulempe med dette systemet. Som også beskrevet i avsnitt 4.2.2 er det en viktig funksjonalitet å kunne spille av hele den produksjonen man jobber med. Dette er kanskje den viktigste utvidelsen mot et fullverdig redigeringsvektøy.

6.2 Konklusjon

I arbeidet med denne oppgaven har jeg utført en studie av redigering og redigeringsverktøy. Jeg har forsøkt å knytte dette opp mot et totalkonsept for digital video, med støtte i en videodatabase. Ulike sider ved digital video og redigering er vurdert, og forskjellige løsninger er foreslått. I avsnitt 4.2 er generell funksjonalitet til et redigeringsverktøy vurdert, og i avsnitt 4.3 er redigeringsverktøy i en databaseanvendelse evaluert. I tillegg er en prototyp beskrevet i kapittel 5. Denne prototypen er utviklet for å utprøve ulike sider ved et redigeringsverktøy for digital video. Prototypens funksjonalitet er deretter vurdert i avsnitt 6.1. Basert på de ovennevnte vurderinger og utprøvnger har jeg kommet frem til følgenede hovedkonklusjoner:

Redigeringsverktøyets primære oppgave er å sette sammen videosekvenser til en ferdig redigert video. Verktøyet må derfor kunne tilby et kildemateriale som kan benyttes i redigeringen. Videre må det kunne utnytte et tidslinjekonsept for redigering og interaktivitet med brukeren. Redigeringsverktøyets ulike funksjoner må omfatte innsetting og sletting, foruten flere andre brukerspesifikke funksjoner som mest naturlig fastsettes i samarbeid med de som anvender systemet. Redigeringen skal resultere i en ferdig redigert video, som man kan spille av og ta ut en redigeringsliste fra, for å kunne behandle videoinformasjonen videre.

Det er viktig å skille mellom fysisk lagrede videosekvenser og ferdig redigerte videoer, som må betraktes som *virtuelle*. Ferdig redigerte videoer må her kunne representeres og lagres som referanser til de fysiske videosekvensene. Derimot vil det være naturlig å lagre overgangseffekter som separate sekvenser og kun gi referanser til disse i de virtuelle videoene. Dette setter spesielle krav til databasen vedrørende tolkningen av disse virtuelle videoene, for å kunne utnytte informasjonen om de ulike referansene og dermed gi et riktig bilde av annotasjoner og strukturer som er gitt til de enkelte videoene. En virtuell video setter også spesielle krav til tolkning ved avspilling. Dette er her foreslått gjort enten i en database eller i en videospiller, med angitte fordeler og ulemper ved begge metodene. Videre vil redigering sette krav til homogenitet i databasen, når det gjelder komprimering og koding. Dette bør være enhetlig for all video i databasen.

I et totalsystem, som skissert i avsnitt 3.2, vil det være naturlig å knytte redigeringsverktøyet sammen med de andre verktøyene for å få et optimalt totalkonsept. Det vil være hensiktsmessig å la søkeverktøyet foreta presentasjonen av kildematerialet. Søkeverktøyet må derfor inneha tilstrekke-

lig funksjonalitet til å kunne overføre informasjon om de ulike sekvensene av kildematerialet til redigeringsverktøyet. Videospilleren vil benyttes til avspilling av de enkelte sekvensene i redigeringsverktøyet og av hele masteren, dvs. resultatet av redigeringen. I tillegg stilles det krav til en *tett* kommunikasjon mellom redigeringsverktøy og videospiller for å få visualisert selve redigeringsprosessen på en fornuftig måte. Det er gjennom hele redigeringsprosessen viktig med hyppig *oppdatering* av såvel videospiller som redigeringsverktøy (tidslinje) for å kunne avbilde den samme *tilstand* i begge applikasjoner. Når det gjelder registreringverktøyet, vil jeg konkludere med at det ikke er naturlig å knytte dette direkte til redigeringsverktøyet.

Ved redigering av video fra en database oppnås implisitt kunnskap om strukturen til videomaterialet. Dette gjelder materialets oppdeling i innstillinger, noe som kan overføres direkte til databasen, slik at man forenkler senere registrering og strukturinndeling. Dette fører til at vi genererer kunnskap i redigeringsprosessen som vi tar vare på for å forenkle den totale arbeidsmengde med informasjonen i en videodatabase. En følge av dette er at registreringsverktøyet bør inneha tilstrekkelig funksjonalitet til å kunne innhente informasjon fra databasen om alle fysisk lagrede videosekvenser. Dette må kunne kombineres slik at man for en virtuell video kan trekke ut informasjon om struktur og annotasjoner.

6.3 Videre arbeid

I forrige avsnitt presenteres konklusjoner som er trukket på grunnlag av vurderinger og diskusjoner gjennom denne rapporten. Noen av disse har kommet fram ved implementasjon og utprøvning av prototypen utviklet i denne oppgaven. En del idéer er testet ut, men på langt nær alle. I det videre arbeidet bør idéene knyttet til videodatabasen utprøves i større grad. Det som er testet ut her er basert på den eksisterende infrastruktur og har ikke fanget opp det som retter seg sepesielt mot databasespørsmål.

Prototypen i denne oppgaven har primært blitt utviklet for å få en bedre forståelse for ulike krav som stilles, og for å teste ut noen idéer. Hvis dette verktøyet er tenkt benyttet i praktisk bruk, må det videreutvikles i henhold til forslag gitt i denne rapporten og i samarbeid med brukerne av systemet. Et slikt redigeringsverktøy vil i stor grad kunne formes etter spesielle brukerkrav.

Vedlegg A

Ord og uttrykk

Her vil jeg presentere noen ord og uttrykk som er brukt i denne oppgaven. Den forklaringen jeg gir er ikke generell, men tar utgangspunkt i temaet for oppgaven. Flere av uttrykkene blir brukt i andre sammenhenger, og kan ha andre forklaringer enn det som blir skissert her.

Annotasjon Beskrivelse av videomaterialet som er tilknyttet en bestemt rammesekvens. Bruker også betegnelsen *tematisk indeksering*.

Dissolve Den engelske betegnelsen på overtoning.

Editliste Liste med nøyaktige angivelser av de videosekvenser som er brukt i en redigering. Kalles gjerne EDL (Edit Decision List).

Fade Glidende overgang fra eller til et ensfarget bilde.

Generasjonstap Forverring av bilde- og lydkvalitet ved kopiering.

H.261 Standard for videokomprimering.

Hovedoppgave er det du ser på akkurat nå. Kalles ofte diplom.

Ikke-lineær redigering Muligheter for å sette inn videosekvenser på vilkårlige punkter i en video.

Innstilling En lengde uavbrutt film. En sekvens med rammer tatt opp i *ett* kontinuerlig opptak.

JPEG Standard for komprimering av stillbilder. Brukes også til komprimering av video.

Klipp Betegner en direkte overgang fra en videosekvens til en annen.

Klippeliste Annen betegnelse for editliste.

Klipping Brukes ofte om redigering. Kommer fra filmredigering, hvor man fysisk klipper i filmmaterialet.

Lineær Linjeformet, sekvensiell.

Lineær redigering Nye videosekvenser kan kun innsettes etter det som allerede er redigert.

Motif Verktøy for programmering av vindusgrensesnitt.

MPEG Standard for videokomprimering.

Multimedia Samling av mange informasjonskilder.

Overtoning To innstillinger *fades* over i hverandre.

Parallax Videokort som komprimerer og dekomprimerer JPEG video.

Ramme Et enkeltbilde av en videofilm.

Rammerate Frekvensen rammene vises med, måles i rammer/s.

Scene Flere innstillinger som hører sammen i tid og rom.

Sekvens Flere scener satt sammen til et meningsfylt innhold. I denne rapporten har jeg også brukt betegnelsen *videosekvens* om et sett rammer, uavhengig av strukturell inndeling.

Shot Den engelske betegnelsen på innstilling.

SIFT Databasesystem fra Statens Datasentral. Brukes i NRK.

SMPTE En standard for tidskodeangivelse. Representeres tekstlig som `tt:mm:ss:rr`, hvor tt=timer, mm=minutter, ss=sekunder og rr=rammer.

Story Board En *tidslinje* med flere nivåer, dvs. flere innstillinger kan legges oppå hverandre.

Tagning Annen betegnelse for innstilling.

Temporal Noe som har med tid å gjøre.

Tidskode Benyttes til identifikasjon av enkeltrammer i en video.

Tidslinje Grafisk fremstilling av en editliste. Benyttes i harddiskbaserte redigeringsverktøy.

Videosekvens Betegnelsen er her brukt om et stykke video, uavhengig av strukturell inndeling.

Wipe Overgangseffekt som drar en ny innstilling inn i eller ut av skjermen, eller gradvis avdekking av en innstilling.

Vedlegg B

VoDoo

VoDoo er et enkelt *video-på-forespørsel*-system (video-on-demand). Dette består av en videospiller og en videoserver. Systemet baserer seg på video som består av JPEG-komprimerte rammer og kjører på Sun arbeidsstasjoner med Parallax videokort. Det er kun arbeidsstasjonen som kjører videospilleren som trenger Parallax, videoserveren kan kjøre på en hvilken som helst unix-maskin.

B.1 Videospilleren

Videspilleren kalles **vshow** (se figur B.1) og kan startes opp med følgende kall fra kommandolinjen:

```
vshow  [vodoo://<vertsmaskin>/]<filnavn> [<start> [<stop>]]
```

Hvis `<vertsmaskin>` utelates vil `vshow` prøve å åpne `<filnavn>` lokalt. Ellers vil applikasjonen forsøke å koble seg opp mot en videoserver på `<vertsmaskin>` og be om å få avspilt filen derifra. Det er også mulig gjennom `<start>` og `<stop>` å spesifisere posisjoner innenfor den angitte fil, hvor avspillingen vil starte og stoppe. `<start>` og `<stop>` må angis på formatet `tt:mm:ss:rr`, hvor `tt=`timer, `mm=`minutter, `ss=`sekunder og `rr=`antall rammer (gitt en rammerate på 25 rammer/s).

Knappene nederst i grensesnittet benyttes til å styre videospilleren. De representerer funksjonaliteten til en vanlig, analog videospiller. Mulige funksjoner er *avspilling/pause, enkeltramme forover, enkeltramme bakover, hurtigspoling forover* og *hurtigspoling bakover*. Ved å dobbeltklikke på knappene for enkeltramme, oppnår man sakte avspilling (slow motion) henholdsvis bakover eller forover. *Slideren* (*glider*) nederst i høyre hjørne gir mulighet for hurtig å posisjonere seg innenfor en videofil. Ved å flytte på slideren kan man hoppe direkte til en ønsket del av filen (videofilmen). Man har også mulighet til å styre alle disse funksjonene fra tastaturet. Se [Aas94] for mer informasjon.

Figur B.1: Videospilleren vshow

B.2 Kontroll fra eksterne applikasjoner

`vshow` kan styres fra andre applikasjoner gjennom en protokoll, basert på *X properties*. Dette utføres av et lite hjelpeprogram som heter `vsend`. `vsend` benytter disse parametrene:

`Play/Pause`	Veksler mellom avspillings- og pausemodus for videospilleren.
`Step`	Går til neste ramme i videoen.
`StepBack`	Går til forrige ramme i videoen.
`Wind`	Spoler videoen forover.
`Rewind`	Spoler videoen bakover.
`Quit`	Avslutter videospilleren.
`Buttons`	Veksler mellom å vise eller ikke vise knappene i videogrensesnittet.
`Timing`	Veksler mellom å vise eller ikke vise tidskode for videoen i videobildet.
`Version`	Returnerer versjonsnummer til videospilleren.
`Where [<filref>]`	Returnerer tidskode for hvor i videoen avpillingen befinner seg. `<filref>` er referanse til hvilken fil videoen avspiller. Hvis denne ikke stemmer med den faktisk avspilte videoen, returneres en feilmelding. Hvis `<filref>` ikke er oppgitt returneres den aktuelle filreferansen i tillegg til tidskoden.
`File [-o <offset>] [-t <tittel>] [-stop] <filref> [<posisjon>]`	Gir videospilleren melding om å bytte film eller forflytte seg innen en film.

For `vsend File`-kommandoen gjelder disse valgfrie parametre:

`-o <offset>`	Benyttes til visning av tidskode i videobildet. Tidskoden som da vises er 'lengde fra starten av videoen + offset'.
`-t <tittel>`	Brukes til å sette tittelen på viduet til videospilleren.
`-stop`	Betyr at videoen settes direkte i pausemodus. Uten denne parameteren begynner videoen å spille fra angitt sted.
`<posisjon>`	Brukes til å hoppe et stykke ut i filmen. Uten denne parameteren starter videoen fra begynnelsen. Man kan her angi en tidskode hvor man starter spillingen eller man kan angi start og stopp.

Alle tidsangivelser skrives på formatet `tt:mm:ss:rr` (se ovenfor) og alle filreferanser benytter samme notasjon som for `vshow`.

Referanser

[Avg94] Avgerakis George, Irving Terry. *Nonlinear Showdown*, Digital Video, november 1994.

[Bek94] Bekkadal Arild, Wasskog Trond Arve. *Grensesnitt for digital video*, Prosjektoppgave, Institutt for datateknikk og telematikk, NTH, april 1994.

[Ben93] Bender Walter, Teodosia Laura. *Salient Video Stills*, Proceedings of ACM Multimedia 93.

[Col94] Collins Mike. *Frame work*, Personal computer world, september 1994.

[Dav91] Davenport G, Smith T.G.A, Pincever N. *Cinematic Primitives for Multimedia*, IEEE Computer Graphics & Applications, july 1991.

[Dav] Davenport G, Evans R, Halliday M. *Orchestrating Digital Micromovies*, Technical paper, MIT Media Lab.

[Doh94] Doherty Richard. *Nonlinear Showdown, part 2*, Digital Video, desember 1994.

[Duda] Duda Andrzej, Weiss Ron, Gifford David K. *Content-Based Access to Algebraic Video*, Technical paper, MIT Laboratory for Computer Science.

[Elm89] Elmasri R, Navathe S. B. *Fundamentals of Database Systems*, Addison-Wesley Publishing Company, 1989.

[Eng94] Engeset Solbjørg. *Vurdering av Avid NewsCutter, Quantel NewsBox og O.L.E. Heavyworks*, Teknisk notat, NRK TKAV, 27.06.94.

[Furht94] Furht Borko. *Multimedia Systems: An Overview*, IEEE Multimedia, spring 1994.

[Gib94] Gibbs S, Breiteneder C, Tsichritzis D. *Data Modeling of Time-Based Media*, Proceedings of the 1994 ACM SIGMOD, International Conference on Management of Data.

[Gid94] Gidney E, Chandler A, McFarlane G. *CSCW for Film and TV Preproduction*, IEEE Multimedia, summer 1994.

[Gon92] Gonzalez R, Woods R. *Digital Image Processing*, Addison-Wesley Publishing Company, 1992.

[Gro94] Grotnes Christian. *Brukertest av Heavyworks*, Teknisk notat, NRK FAKE, 06.09.94.

[Ham93] Hamakawa Rei, Rekimoto Jun. *Object Composition and Playback Models for Handling Multimedia Data*, Proceedings of ACM Multimedia 93, side 273-281.

[Hje94] Hjelsvold Rune, Midtstraum Roger, *Digital Video Archives*, Proceedings of Norsk Informatikkonferanse 1994, Molde, Norway, november 1994.

[Hje94b] Hjelsvold Rune. *Video Information Contents and Architecture*, Proceedings of the 4th International Conference on Extending Database Technology, Cambridge, UK, mars 1994.

[Hje94c] Hjelsvold Rune, Midtstraum Roger. *Modelling and Querying Video Data*, Proceedings of the 20th International Conference on Very Large Databases, Santiago, Chile, september 1994.

[Hje94d] Hjelsvold Rune. *Experiences with distributed multimedia – the Norwegian Commedia Project*, OII Spectrum, Volume 1 No.5, februar 1994.

[Holte94] Holte Tormod. *Digitalt fjernsynsarkiv*, Hovedoppgave, Institutt for datateknikk og telematikk, NTH, 1994.

[Ler94] Lerbæk Flemming. *Digital nyhetsfangst med Avid NewsCutter*, Audiovisuelle medier, Nr.5, august 1994.

[Lit94] Little T.D.C, Venkatesh D. *Client-Server Metadata Management for the Delivery of Movies in a Video-On-Demand System*, Technical Report, Boston University, 1994.

[MM92] Maartmann-Moe, Erling. *Multimedia*, 2.utgave, Universitetsforlaget, 1992.

[Mac89] Mackay Wendy E., Davenport Glorianna. *Virtual Video Editing in Interactive Multimedia Applications*, Communications of the ACM, Vol.32, No.7, july 1989.

[Mat93] Matthews James, Gloor Peter, Makedon Fillia. *Video Scheme: A Programmable Video Editing System for Automaton and Media Recognition*, Proceedings of ACM Multimedia 93, side 419-426.

[Merok93] Merok Petter. *Datamodeller for digitale film- og videoarkiver*, Hovedoppgave, Institutt for datateknikk og telematikk, NTH, 1993.

[Mil90] Millerson Gerald. *The Technique of Television Production*, Twelfth edition, Focal Press, 1990 (Reprinted 1993).

[Oust94] Ousterhout John K. *Tcl and the Tk Toolkit*, Addison-Wesley Publishing Company, 1994.

[Pen94] Pentland A, Picard R, Davenport G, Haase K. *Video and Image Semantics: Advanced Tools for Telecommunications*, IEEE Multimedia, spring 1994.

[Pud94] Pudovkin V. I. *Film Technique*, Lear Publishers Inc, New York, 1949.

[Rab89] Rabiger Michael. *Directing - film techniques and aestetics*, Focal Press, Boston, 1989.

[Sch94] Schunck Brian G. *Four Steps to Mac Multimedia*, IEEE Multimedia, spring 1994.

[Ske93] Skeide Sigurd. *High Quality Video Server*, Hovedoppgave, Institutt for datateknikk og telematikk, NTH, 1993.

[Sony] Sony Corporation. *Digital BETACAM DVW-510P*, Produktinfo, MK2773KYP9401P1-004, 1994.

[Van94] Vangberg Tore. *Noen viktige punkter for diskbasert videored.utstyr*, Teknisk notat, NRK FAKE, 25.03.94.

[Taw94] Wasskog Trond Arve. *General video-client* Hovedoppgave, Institutt for datateknikk og telematikk, NTH, 1994.

[Welch94] Welch Brent. *Practical programming in Tcl and Tk*, Draft, Prentice Hall, 1994.

[White94] White Charlie. *Heavy Metal Video: EDLs, Edit Suites and Nonlinear Editors*, Digital Video, desember 1994.

[Aas94] Aas Gisle. *VoDoo - a simple "Video on demand" system for workstations*, URL: http://www.nr.no/vodoo/, 1994.