CODEN:LUTFD2/(TFRT-5327)/1-047/(1985)

RÖRELSEDETEKTERING I BILDSEKVENSER

SVEN-OLOF JANSSON

INSTITUTIONEN FÖR REGLERTEKNIK
LUNDS TEKNISKA HÖGSKOLA
SEPTEMBER 1985

Department of Automatic Control
Lund Institute of Technology
P.O. Box 118
S-221 00 Lund Sweden

Document name
MASTER THESIS

Date of issue
September 1985

Document Number
CODEN: LUTFD2/(TFRT-5327)/1-047/(1985)

Author(s)
Sven-Olof Jansson

Supervisor
Lars Nielsen

Sponsoring organisation

Title and subtitle
Rörelsedetektering i bildsekvenser
(Motion detection in image sequences)

Abstract

An experimental system for motion detection in image sequences has been developed. Based on the vision system of the frog the motion detector consist of simple and local operators put together to a more complex algorithm. A main issue has been to describe moving corners. The convexity of corners is determined by area measurements in concentric zones. The experiments raised a need to make the operations depending on the intensity distributions. A new technique for image segmentation is based on classification of local histograms.

Key words

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title

ISBN

Language
Swedish

Number of pages
47

Recipient's notes

Security classification

INNEHÅLLSFÖRTECKNING

1. INTRODUKTION

Ett experimentsystem har byggts upp med vilket vi kan undersöka tidssekvenser av bilder. Systemet är baserat på ett existerande bildlaboratorium beskrivet i [1].

Inspiration

Inspirationskällan för arbetet har varit en artikel där funktionen hos grodans synsystem har studerats, [2]. Neurofysiologiska mätningar har gjorts på levande grodor. Artikeln ger följande beskrivning.

Rörelsedetekteringen är en primitiv mekanism hos grodan.

> The frog "will starve to death surrounded by food if it is not moving. His choice of food is determined only by size and movement. He will leap to capture any object the size of an insect or worm, providing it moves like one."

Vid stimulering med föremål av varierande storlek och form har mätningar i synnerven hos levande grodor visat att det finns fyra separata typer av operatorer verkande på den bild som alstras på näthinnan i grodans öga : "sustained contrast detection", "net convexity detection", "moving edge detection" samt "net dimming detection". Totalt finns cirka 0.5 miljoner operatorer. Varje enskild operator har ett begränsat område på näthinnan där den är känslig. Detta område kallas "receptive field". De olika typerna av operatorer har olika storlek på "receptive field", upplösning i resultatbilden samt överföringshastighet. Överlappningen av "receptive fields" är stor.

Groddetektorerna

a) Sustained contrast (kontrast detektering)

Om en skarp kant förs in i detektorns utbredningsområde sänds en signal. Denna sänds oavsett om kanten rör sig eller stannar inom området. Resultat fås vid både liten och stor kontrast mellan kant och bakgrund.

b) Net convexity (konvexitet detektering)

Detektorn ger information i ett något större område om hörn eller små objekt, mindre än detektorns utbredning, passerar genom eller förs in och stannar i området. Reaktionen är större ju större krökning föremålet har. Ingen reaktion sker vid excitation med raka kanter.

c) Moving edge (rörlig kant detektering)

Detektorn reagerar på rörliga kanter och resultatet är proportionellt mot hastigheten i rörelsen. Reaktionen slutar om rörelsen avstannar.

d) Net dimming (detektering av intensitets förändringar)

Stimuleringen sker genom en plötslig minskning av ljusmängden i detektorns "receptive field". Resultatet är relaterat till storleken på den delyta där förändring sker samt hur fort det sker. Känsligheten är större i fältets centrum än i dess ytterområden.

Sammanfattningsvis säger artikeln [2] :

> "Fundamentally, it shows that the eye speaks to the brain in a language already highly organized and interpreted".

De olika detektorerna i grodans öga är operatorer som strukturerar informationen i bilden för senare analys i grodans hjärna.

Målet med examensarbetet

Målet har varit att skapa en modell för detektering och beskrivning av rörliga objekt ("flugor"). Modellen är sammanfogad av idéer från artikeln. Grundprinciper har varit

att enbart reagera i ytor i bilden som motsvarar rörelse;

att med enkla algoritmer separera varierande information till skilda beslutsbilder;

att kombinera detta till en mer komplex algoritm för att detektera och beskriva rörelsen för ett objekt i bilden.

Vid implementeringen av våra operatorer har vi inte försökt att direkt imitera det biologiska systemet. Vi började med att söka en net dimming operator som skulle

tala om för oss var i bilden vi troligtvis kunde finna rörliga objekt. För att klassifiera ett föremåls form gjordes en lokal konvexitetsoperator. Dessa operatorer har utgjort grunden för det experimentsystem som utvecklats.

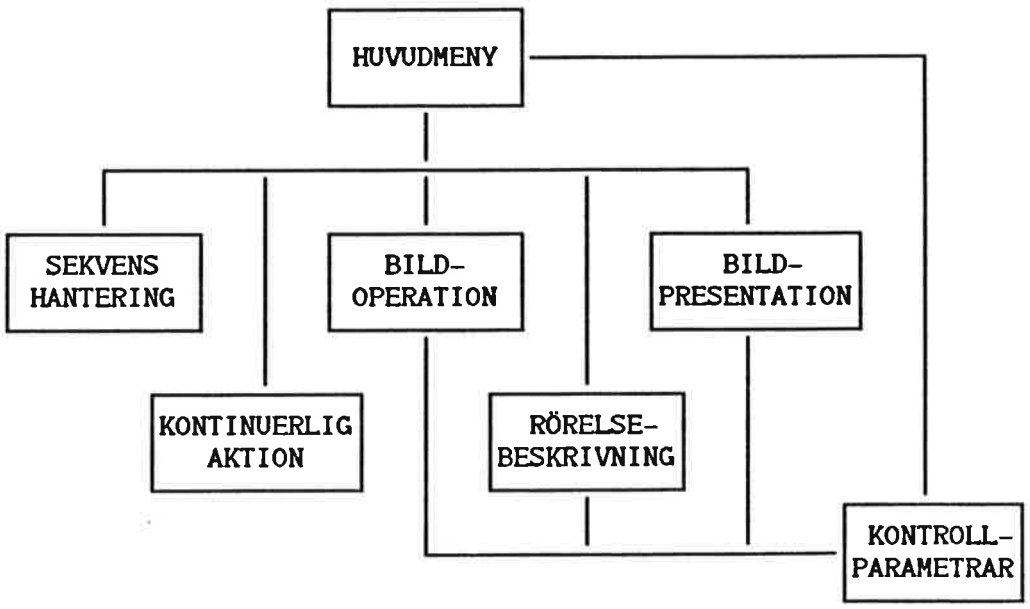
I kapitel 2 beskrivs användarsnittet i programmet översiktligt. Dokumentation av program och operationer ges i kapitel 3, appendix A och appendix B. Ett experiment finns presenterat i kapitel 4. Där kommenteras även egenskaper hos bilderna som särbehandlats vid utvecklingen av operatorerna.

2. ANVÄNDARSNITT

Programmet är uppbyggt kring en försöksuppställning bestående av terminal, kamera och monitor. Övriga hjälpmedel är ett pekdon (mus) samt tillgång till skrivare.

Användarsnittet består av ett antal menyer där alternativen antingen leder till en undermeny eller att en funktion utförs. Kommandon matas in via tangentbord. Vid vissa kommandon krävs svar på frågor för att funktionen/övergången till submenyn skall utföras. Vid inmatning av värden vid sådana frågor ges vanligen ett standardvärde. Menyernas utseende finns listade i appendix A.

Huvuddragen av snittet mot användaren ses i figur 1. Huvudmenyn, som erhålls när programmet startas, innehåller val till programmets fem huvuddelar, sekvenshantering, kontinuerlig aktion, bildoperation, rörelsebeskrivning och



Figur 1 Blockschema över användarsnittet.

bildpresentation. Vidare finns en ingång för inställning av styrparametrar för de algoritmer som används vid bildoperationerna samt specialkommandon för hantering av kamera och monitorn samt presentation av resultat.

I sekvenshantering, i huvudmenyn kallad ImageSeqHandler, finns kommandon för att spela in en ny bildsekvens eller bakgrundsbild samt för att lagra och hämta en befintlig sekvens. Här finns också möjlighet att spela upp en sekvens utan att utföra operationer.

Med hjälp av de tre alternativen, kontinuerlig aktion, bildoperation och rörelsebeskrivning studeras operatorernas inverkan på en lagrad sekvens. Funktionen kontinuerlig aktion, FreeRun, medför att den totala algoritmen beräknas för hela sekvensen. Resultatet presenteras på monitorn för varje bild. För att i detalj studera operationerna på en bild i sekvensen väljs undermenyn bildoperation, LookAtOne. Resultat kan här fås från delstegen i algoritmen. Med musens hjälp pekas intressanta delytor av bilden ut. På dessa områden kan operationer utföras eller karakteristika såsom t.ex. histogram över intensitetsfördelning erhållas. Informationen presenteras huvudsakligen på monitorn men ges även via terminalen och skrivare. Musens funktioner finns kommenterade i programpaketet Mouse, se appendix B. Rörelsebeskrivning, MotionDescript, beräknar läge och form för det objekt som rör sig i bilden utifrån resultatet från grodoperatorerna. Resultatet markeras på monitorn och värden från beräkningar ges på terminalen. Mellanresultat kan även lagras på fil. En ingång för att utföra rörelsebeskrivningen finns också i LookAtOne.

Det finns en funktion, kontrollparametrar, för att ändra globala värden som styr algoritmen. Denna funktion kan nås från de menyer där operationer på bilder i sekvensen utförs och är där benämnd ControlStatus.

För dokumentation finns en undermeny, bildpresentation eller PresentImages, där bilder från deloperationer väljs ut och presenteras i full skala på monitorn. Bilderna kan överlagras med resultatet från den totala algoritmen.

3. IMPLEMENTERING

Programmet har utvecklats för att utgöra ett flexibelt experimentsystem. Databasen har i stort sett varit fix. Algoritmer och operationer har utvecklats och testats under utvecklingen. Syftet har varit att finna och beskriva hörn hos rörliga föremål. I detta kapitel beskrivs först de operationer och bildtyper som används. Därefter följer en kort beskrivning av programmets databas.

Rörelsedetektorn

Vi använder olika bilder och operationer. En sammanställning av dessa ges i figur 2. De tretton bildtyperna representerar den lagrade sekvensen och resultaten från operatorerna. Operationerna är differensbildning, DiffIm, kantdetektering, SusCon, detektering av intensitetsförändringar, NetDim, samt konvexitetsdetektering, NetCon.

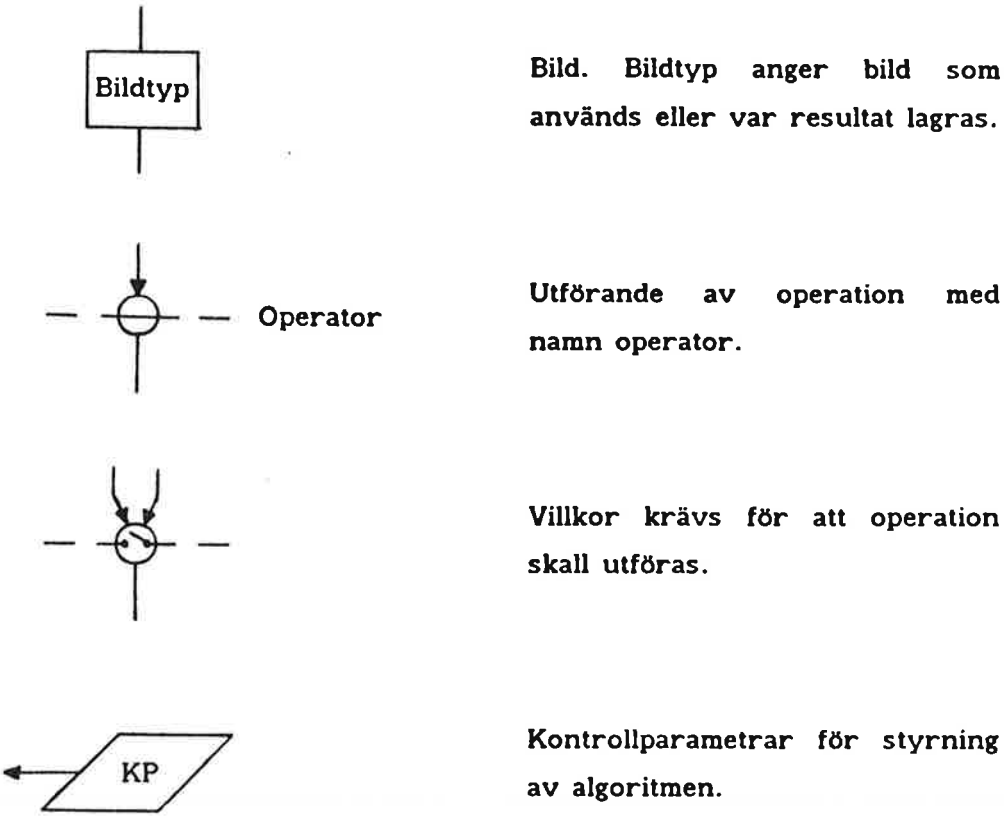
backgr : Bakgrundsbild.

frog : Aktuell bild i sekvensen.

Bildstorleken för backgr och frog är 128 x 128 punkter. Bakgrundsbilden är konstant för hela sekvensen. Grundinformationen för rörelsedetekteringen är

Bildtyp	Operator
backgr, frog, timed, backd, timet, backt,	DiffIm
susconf, susconb, suscft, suscbt, diffsuscftbt,	SusCon
netdim,	NetDim
netcon	NetCon

Figur 2 Sammanställning av använda bildtyper och operatorer.



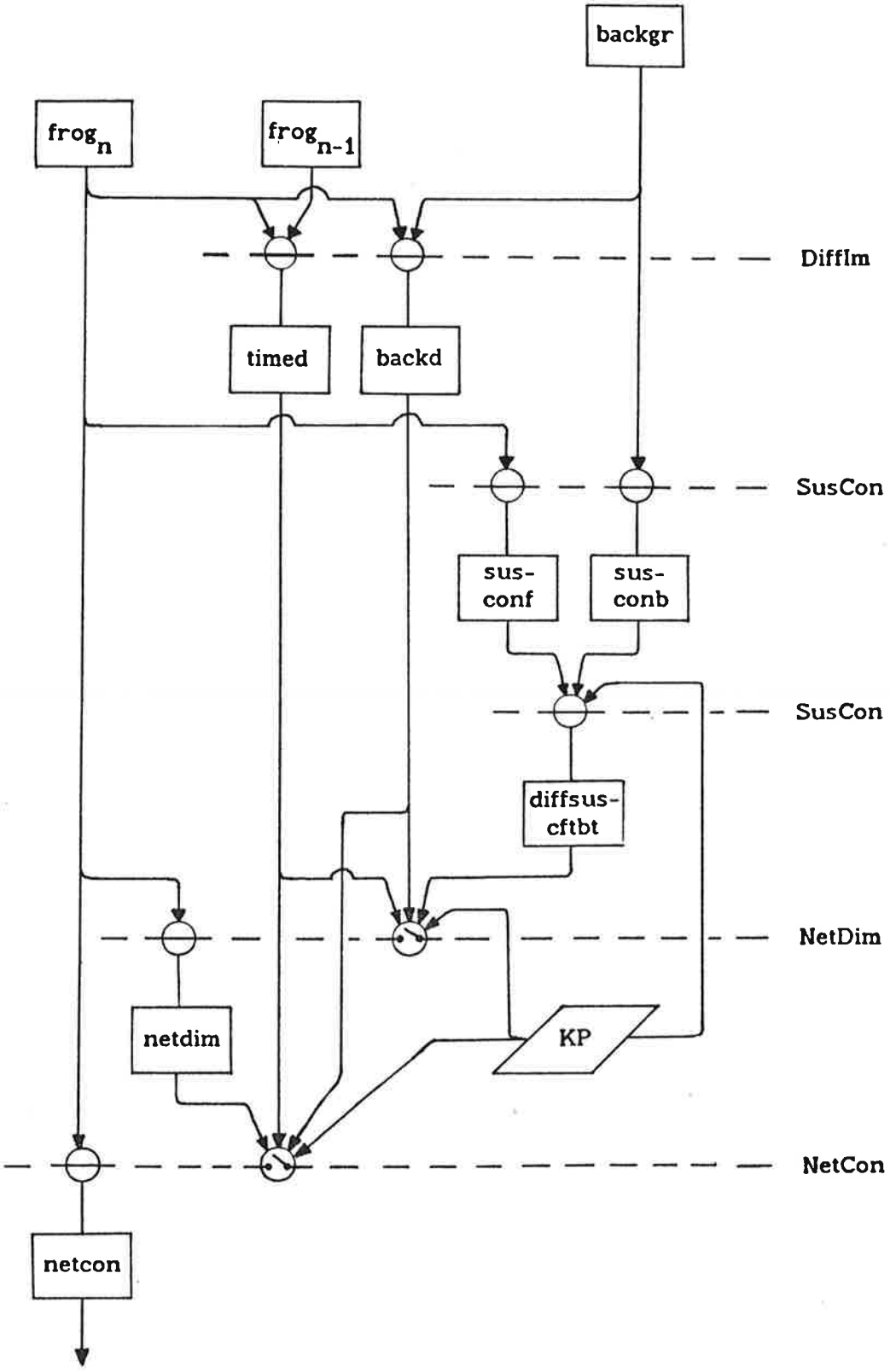
Figur 3 Förklaringar till symboler använda i figur 4.

aktuell bild, föregående bild och bakgrundsbild. Den totala algoritmen finns illustrerad i figur 4. Förklaringar till symbolerna i figur 4 ges i figur 3.

timed, backd : Skillnadsbild mellan aktuell bild och föregående bild respektive mellan aktuell bild och bakgrundsbild.

timet, backt : Tröskelsatt skillnadsbild. Resultatet lagras endast vid presentation.

Bilderna timed, backd, timet och backt är resultat från operatorn Diffim. Tröskelsättningen av timed och backd ger två binära bilder där punkter med intensitetsändring markeras. Ändringen härrör sig till föremål i bilden, reflektioner eller skuggning. Nivån för tröskelsättningen ges av en global styrparameter. Storleken för bilderna är 128 x 128 punkter.



Figur 4 Flödesschema över algoritmen för rörelsedetektering.

susconf, susconb : Kantbild av aktuell bild respektive bakgrundsbild.

suscft, suscbt : Tröskelsatt kantbild. Resultatet lagras endast vid presentation.

diffsuscftbt : Skillnadsbild mellan de tröskelsatta kantbilderna.

Bilderna är resultat från operatorn SusCon. Operatorn består av två delsteg. I susconf och susconb lagras resultat från gradientberäkning (Sobel) av respektive ursprungsbild. Kantbilderna tröskelsätts med hjälp av kontrollparameter. En "exclusive-or" operation utförs mellan de tröskelsatta kantbilderna. Resultatet är en binär bild, diffsuscftbt, som representerar skillnader i kanter. Storleken för de fem bilderna är 128 x 128 punkter.

netdim : Bild över områden med förändringar i intensitetsfördelningen.

Bilden netdim består av 16 x 16 punkter där varje punkt är resultat från en operator, NetDim, av storlek 14 x 14 punkter. I definitionsområdet kontrolleras ändringar av intensiteterna genom beräkning av antalet sammanfallande punkter i de tröskelsatta differensbilderna timed och backd samt beräkning av antalet punkter i diffsuscftbt. Värdena måste var för sig vara större än en motsvarande given styrparameter. NetDim beräknar även tröskelvärden från lokal segmentering av bilden (se Levelize i paket LocalSeg).

netcon : Bild av rörliga lokala konvexiteter.

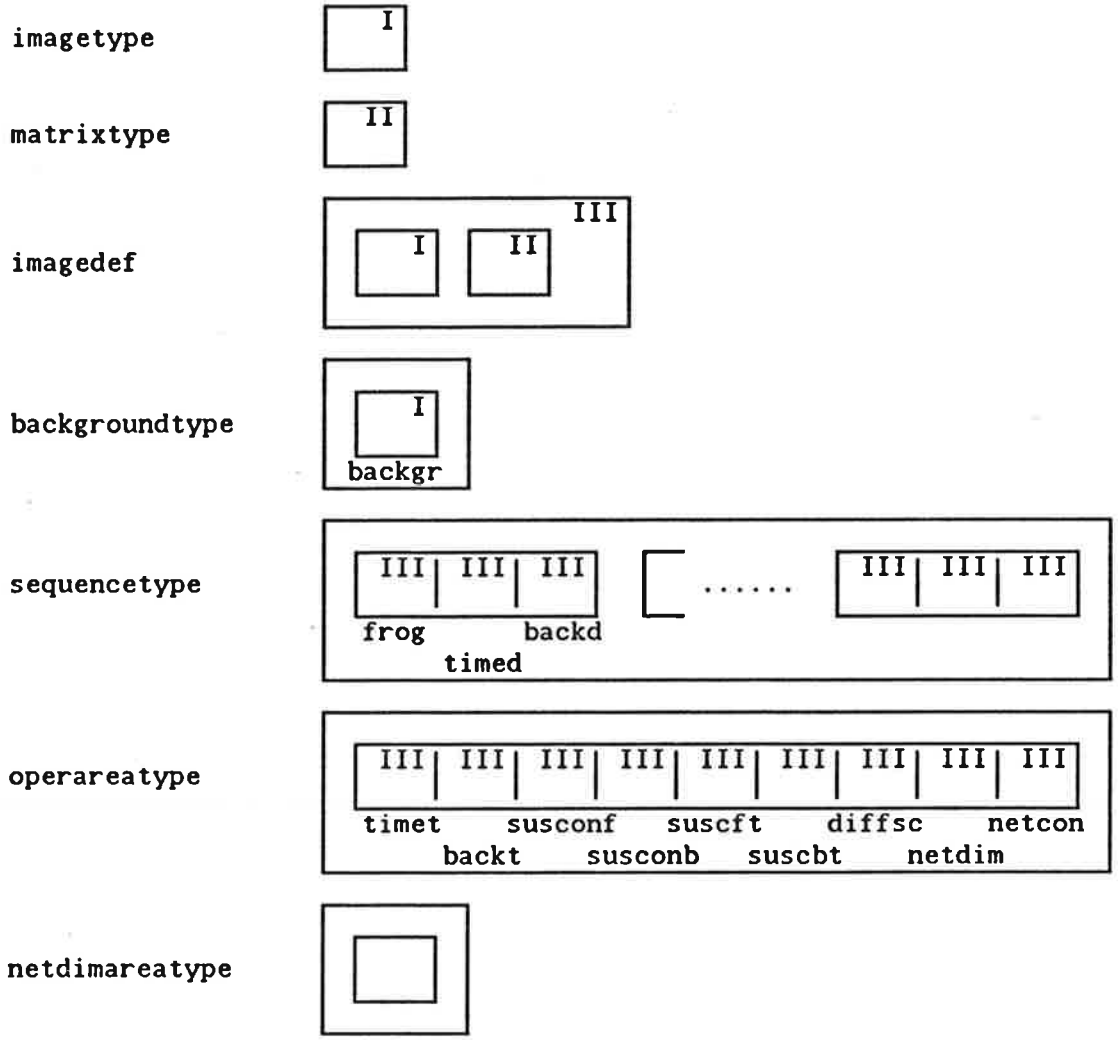
Bilden netcon består av 64 x 64 pixel där varje punkt är ett resultat från en operator, NetCon, av storlek 7 x 7. I dess definitionsområde bestäms ett lokalt krökningsmått genom segmentering av ursprungsbilden. I den erhållna binära delbilden mäts konvexiteten med hjälp av koncentriska zoner. Resultatet kodas och lagras i netcon. Villkoren för operation är dels en kontroll av timed och backd, samma som för NetDim men i ett 7 x 7 område, dels att segmenteringen i området måste motsvara den segmentering som erhålls från NetDim samt att intensitetsskillnader inom området överstiger en kontrollparameter.

Operationerna net dimming och net convexity verkar på bilder av storlek 128 x 128 pixel. Deras operatorer överlappar alltså varandra. Definitionsområdets storlek för de båda operatorerna, 14 x 14 respektive 7 x 7, motsvarar iden med "receptive fields".

Kommentarer till programmet

Programkoden är uppdelad i paket. Varje paket består av en paketspecifikation samt en paketkropp. I specifikationen är konstanter, datatyper, variabler samt rutiner deklarerade. Rutinerna finns sedan kodade i paketkroppen. En preprocessor sammanfogar de olika paketen till en modul som kompileras och länkas på vanligt sätt. Paketbegreppet finns beskrivet utförligare i [1]. I tabell 1 finns en uppräkning av de paket som används för generering av programmet. Paketspecifikationerna finns listade i appendix B.

I figur 5 ges databasen för de i programmet använda bilderna. Grundstommen för lagring av bilder och resultat är datatyperna imagetype och matrixtype, båda bestående av en matris med 128 x 128 element. Imagetype används för representation av bilden. Varje matriselement representeras med 8 bitar. Matrixtype är en hjälpvariabel för operationerna innehållande heltalsvariabler. Bakgrunden lagras i en fristående variabel av backgroundtype, enbart innehållande imagetype. Övriga bilder är uppbyggda av datatypen imagedef, vilken innehåller båda matriserna. Bildsekvensen är en vektor där varje element består av de tre bilderna frog, timed och backd. Vektorn är definierad i sequencetype. Längden på vektorn bestäms av konstanten numofimages. I operareatype är de övriga nio bilderna definierade. Vid operationerna NetDim och NetCon används inte hela matriserna för lagring av data. utan endast 16 x 16 respektive 64 x 64 element. För operationen NetDim finns en variabel av netdimareatype för lagring av delresultat från den lokala segmenteringen. Den består av en matris, storlek 16 x 16, där varje element är en vektor av fem heltal.



Figur 5 Sammanställning av bilddatabasen.

PROGRAMPAKET

Bibliotek	Paketnamn	Beskrivning
MachDep	FrogFile	Filhantering för lagring av bildsekvenser.
RasterMem	RastProc	Rutiner för överföring av bilder till bildminnet.
Interact	Mouse	Rutiner för mushantering.
MainDef	FrogImDef	Grundläggande bilddefinitioner.
Controler	Control	Definition av globala kontrollparametrar.
Operators	DiffIm	Rutiner för skapande av differensbilder.
	SusCon	Rutiner för utförande av kontrastdetektering.
	NetDim	Rutiner för lokalisering av intensitetsförändringar.
	NetCon	Rutiner för bestämning av lokal konvexitet.
	LocalSeg	Rutiner för lokal segmentering.
MenuItems	FrogProg	Huvudmeny och initiering.
	SeqHand	Submeny för hantering av bildsekvenser.
	LookAt1	Submeny för operationer på en bild
	FreeRun	Rutin för beräkning av algoritmen över hel bildsekvens.
	MotDesc	Rutiner för rörelsedetektering.
	Present	Submeny för presentation av blder i fullskärmsskala.
	SpecShow	"Specialrutin" för presentation av delresultat.

SUPPORTPAKET

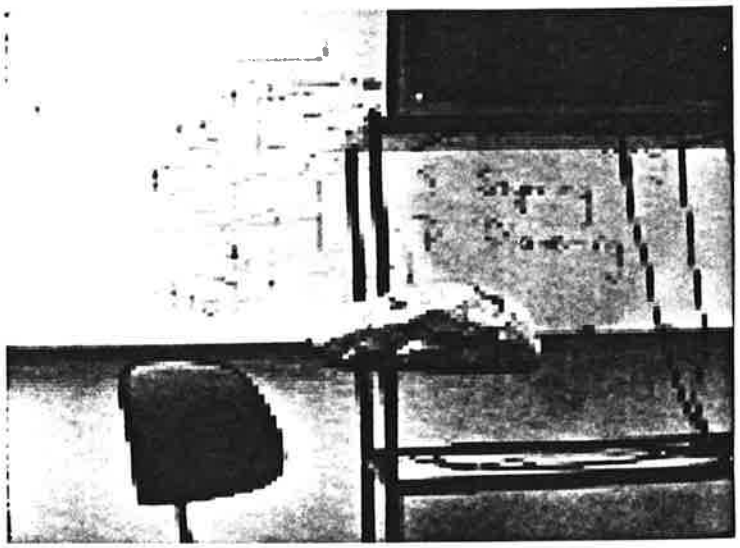
Bibliotek	Paketnamn	Beskrivning
MachDep	Extern	Deklarationer av externa maskinberoende procedurer.
RasterMem	Raster	Grundläggande rutiner för hantering av bildminnet.
	RasterReg	-"-
	LookUp	-"-
Interact	InitInter	Grundläggande rutiner för hantering av mus och terminal.
	Interact	-"-
	Screen	-"-

Tabell 1 Sammanställning av paket för programgenerering.

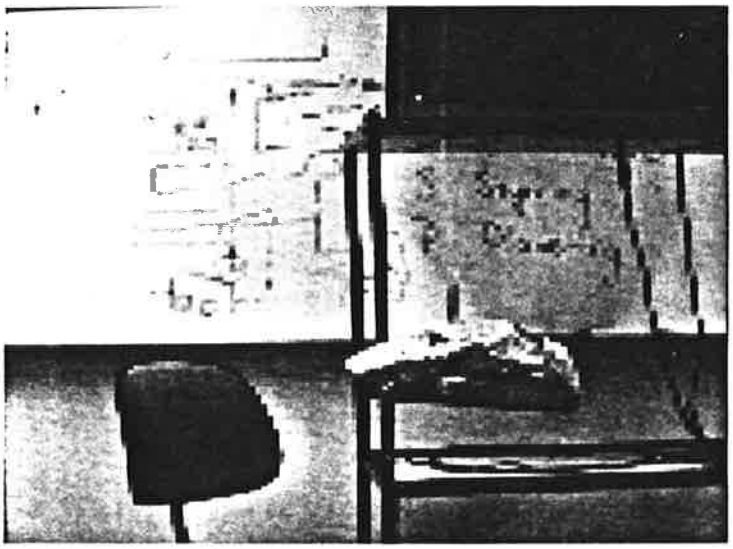
4. EXPERIMENT

Vår "fluga" i experimenten är ett objekt bestående av ihopskrynklat papper. Föremålet förs omkring i rummet. Bakgrunden är av skiftande komplexitet. Ljus kommer från fönster och/eller lampor i taket.

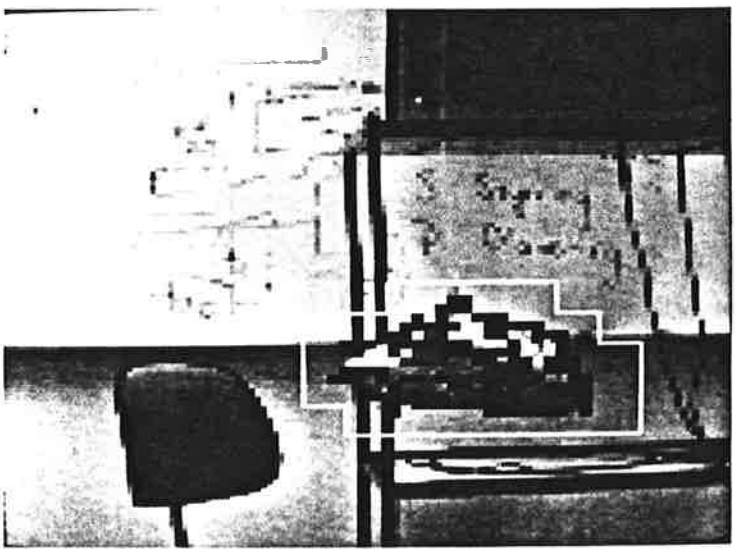
En del av en typisk sekvens ses i figurerna 7, 8 och 10. I figur 9 och 11 ses resultatet från algoritmen efter operation på figur 8 respektive 10. Den ljusa ramen i resultatbilderna ringar in det område där net dimming operatorn detekterat tillräckligt stor intensitetsförändring. I området har sedan net convexity operatorn beräknat föremålets lokala form, dess konvexitet. De mörka ytorna markerar detta resultat. Besluten är här kodade och ordnade i stigande konvexitet. Resultaten markeras på skärmen med en färgskala.



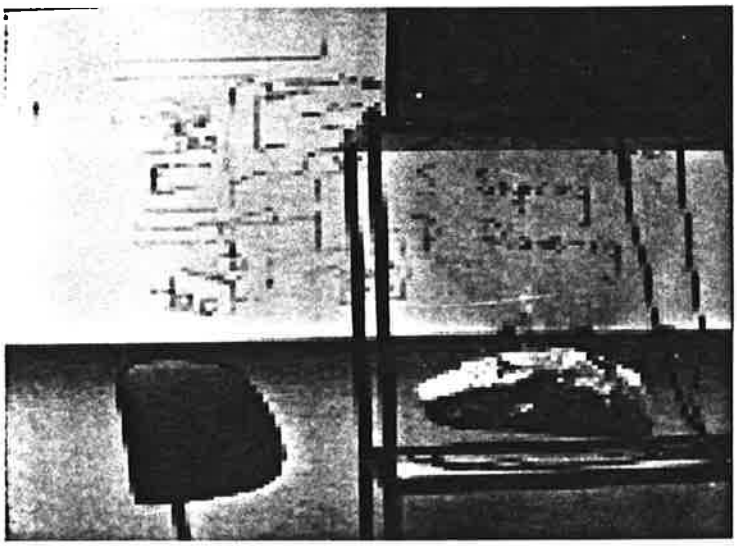
Figur 7 Sekvensbild 1.



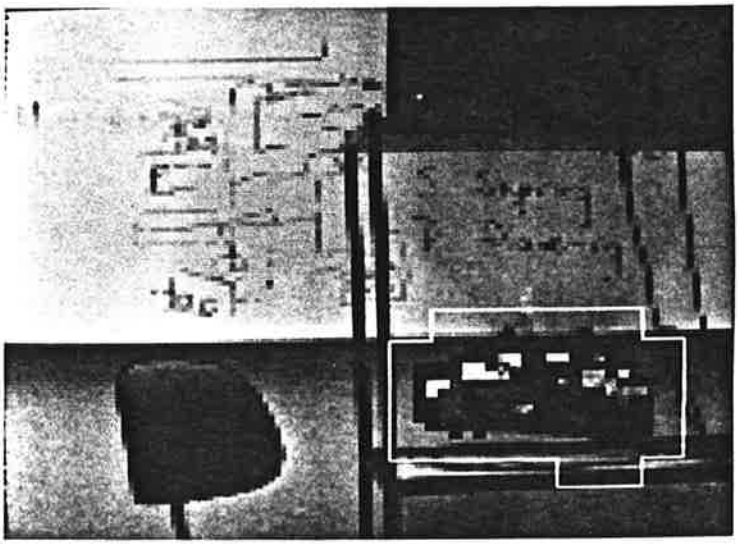
Figur 8 Sekvensbild 2.



Figur 9 Resultatbild för sekvensbild 2.

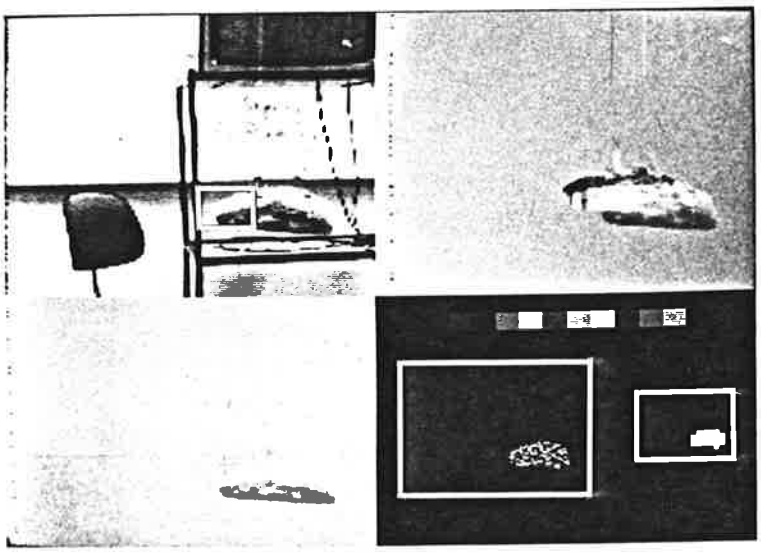


Figur 10 Sekvensbild 3.

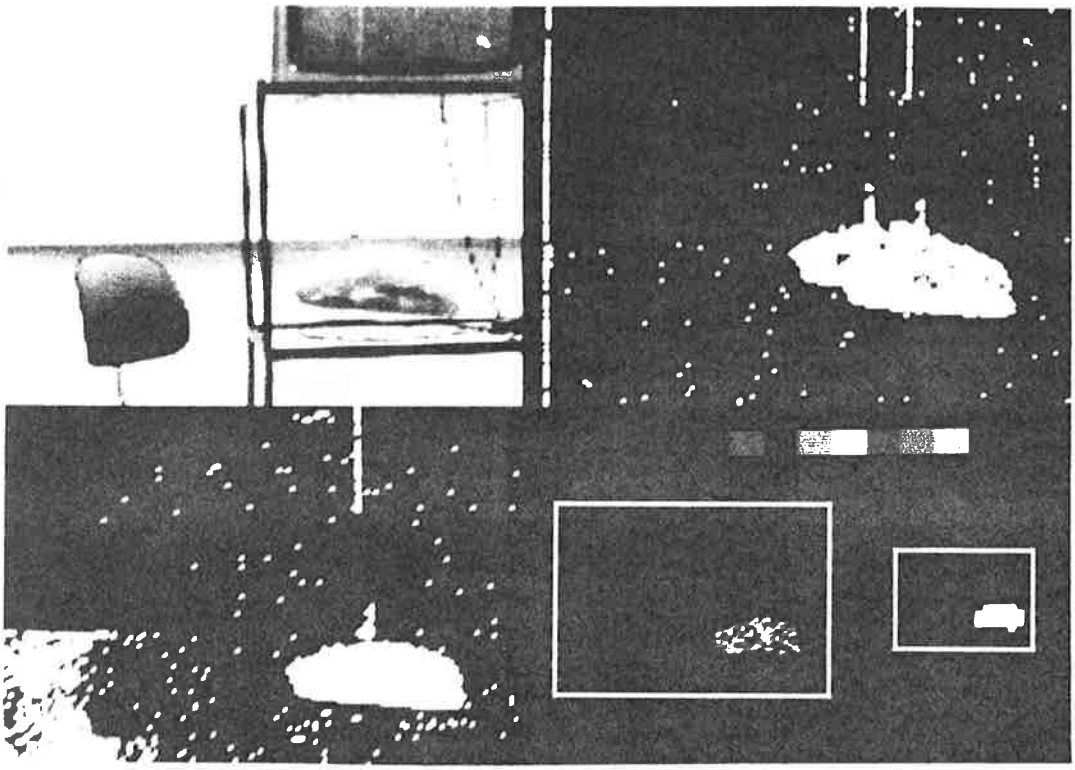


Figur 11 Resultatbild för sekvensbild 3.

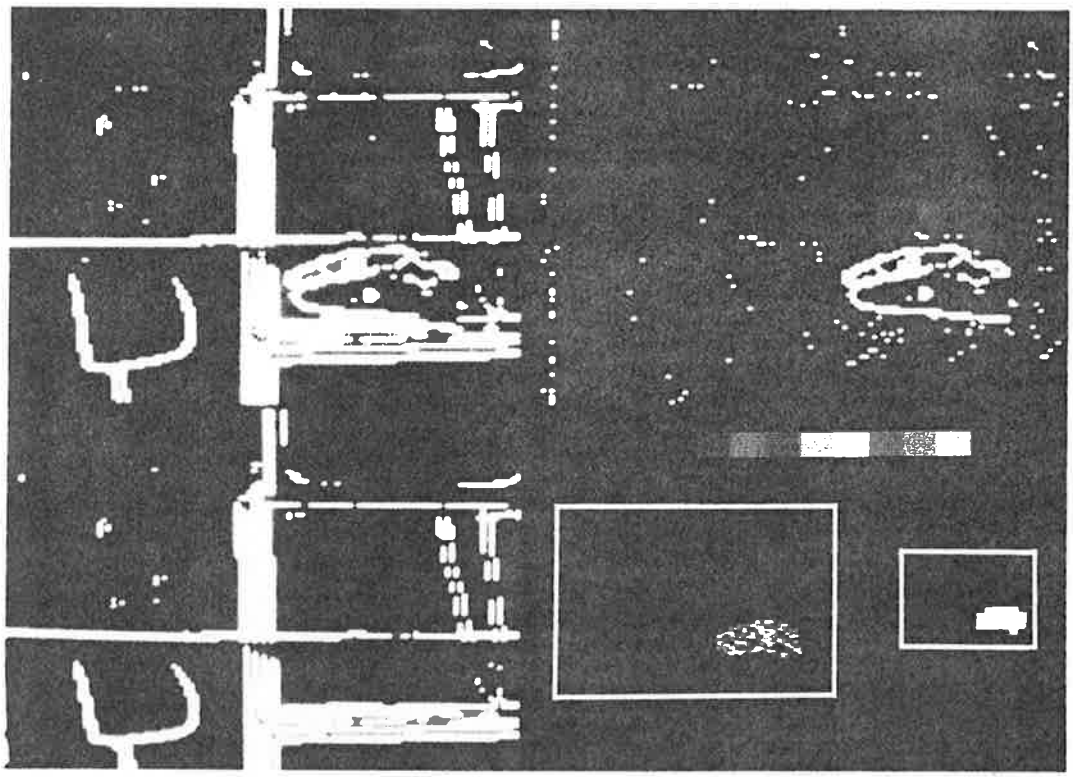
För att studera hur dessa resultat är uppbyggda används submenyn för bildoperation, LookAtOne. Skärmen delas här in i fyra delar, se figur 12. I övre vänstra hörnet presenteras aktuell bild, frog, och som i detta fall är motsvarar bilden i figur 10. Skillnadsbilderna timef och timeb visas överst till höger respektive nederst till vänster. De ljusa partierna refererar till en ökning i intensitet och de mörka svarar för en minskning. Det fjärde området beskriver resultaten av net dimming, till höger, och net convexity. Dessa resultat kan också presenteras genom överlagring i de två översta bilderna. Över dem ges en färgskala för tolkningen av de kodade resultaten. De tröskelsatta differensbilderna timet och backt ses i figur 13. Varje ljus punkt markerar en absolut förändring av intensiteten större än en given tröskel. I timet överst till höger ses att signal fås från föremålets nuvarande läge samt från det läge den hade i föregående bild. I backt ses nederst till vänster ett område där långsamma variationer av intensiteterna gett utslag. Resultatet av kantdetekteringen visas i figur 14. Bilderna till vänster är suscft och suscbt. Till höger ses skillnadsbilden diffsuscftbt. Den högra kanten av flugan har inte gett någon signal. Kantbilden ger även bruspunkter vid de kontraster som finns i bilden. Resultaten i bilderna 13 och 14 visar att ingen av operationerna ensam kan lokalisera hela objektet. Den sammansatta algoritmen lyckas ändå beskriva hela föremålet.



Figur 12 Differensbilder och resultat för sekvensbild 3.



Figur 13 Tröskelsatta differensbilder för sekvensbild 3.



Figur 14 Tröskelsatta kontrastbilder för sekvensbild 3.

Med hjälp av musen plockas ett delområde ut för detaljgranskning. Det valda området är markerat i figur 12. Vi väljer att illustrera med ett problem vi har tvingats specialstudera. Flugan är belyst ovanifrån varför övre delen är ljus medan den undre delen uppträder mörk. Bakgrunden är inte utsatt för direkt ljus och ses därför som grå. Detta ger tre områden med olika intensiteter. Vi kallar detta halvmåneeffekt. Effekten kan även ses i intensitetsvärdena för det valda området vilka ges i figur 15. Centrum är punkten (90, 76). Värdena är angivna inom net dimmings definitionsområde, 14 x 14, och motsvarar aktuell bild, föregående bild samt deras differens. I aktuell bild ses föremålet som två områden, ett med intensiteter över 200 och ett med värden under 100. Bakgrundens värden varierar mellan 110 och 140. Texten i figuren gäller intensitetsvariationer i en 5 x 5 omgivning kring centrumpunkten. Value of net dimming är det kodade resultatet från operatorn NetDim. Värdet 7 anger detektering av ändring i intensiteter inom området. Value of net convexity anger resultat från operatorn NetCon. Vår lösning för att få korrekt segmentering vid skapandet av den binära bilden i operatorn NetCon är som följer.

I definitionsområdet för net convexity, 7 x 7, vill vi bestämma om området innehåller 1, 2 eller 3 intensitetsområden samt vid 2 eller 3 vilket delområde som är störst. Det sker genom analys av intensitetsfördelningen för definitionsområdet. Histogrammet för området kring (90, 76) ges i figur 16. Avvikande ytterpunkter i det totala intensitetsintervallet tas bort. I vårt fall ändras intervallet från 72-207 till 90-206. Differenserna mellan intervallets percentiler normaliseras med intensitetsskillnaden för det totala intervallet. I figuren anger Q-värden percentilerna och M-värden anger de normaliserade differenserna. Storheterna M_{ij} används för att bestämma fördelningens utseende. Därefter används percentilerna Q_i för att bestämma intensitetströsklar för segmenteringen. Vid jämn fördelning är alla värdena M_{ij} lika stora, dvs. här 0.16 eftersom vi använder 6 intervall. Ett mindre värde på något M_{ij} motsvarar en liten differens mellan percentilerna vilket ses som en topp i histogrammet. En tröskel för M_{ij}-värdena, för att bestämma om en topp finns eller ej, har via experiment valts till 0.10. De tröskelsatta värdena används för klassificering med avseende på antal toppar (proceduren Levelize i paket LocalSeg). Resultatet för exemplet är en topp mellan Q_1 och Q_3, där $M_{12} = 0.06$ och $M_{23} = 0.04$, och en från Q_5 och uppåt, där $M_{56} = 0.05$. Den första består av två M-värden och anses

```
Center     y :  90   x :  76

Number in sequence     :  7

   40   51   91   89   94   95   97   97   94   98  105  113  121  169
   30   45   90   95  100  101  104  112  105  116  138  198  187  168
   31   45   94  102   99  105  109  111  129  188  203  204  208  213
   28   45  101  107  113  119  123  155  201  202  204  203  215  215
   30   41  106  112  111  128  190  207  205  206  203  200  220  213
   29   39  113  113  116  140  206  207  207  203  203  186  191  135
   30   37  114  113  118  126  200  203  188  176  151  133   96  103
   33   38  118  117  120  125  143  148  139   92   99   83   99   88
   29   37  119  119  124  124  111  110   89   90   86   83   70   90
   36   33  127  122  124  123  119   75   72   92   76   79   71   86
   33   36  124  126  132  132  131   66   71   72   79   72  102   74
   30   29  131  120  128  135  131  100   57   69   75   98  105   71
   35   36  132  131  132  132  130  124   58   55   57   71   60   61
   34   35  130  128  137  135  134  138  121   75   67   62   58   56

In 5x5  )  Max : 207  Min :  89  Diff : 118  Middle : 148

Number in sequence     :  6

   85   92   92   89   99   90   89   86  101   69   84   78   81   88
   87   70   88  106   94   94   82   78   73   74   85   92   70   72
   70   78   75   96   99   88   74   75   71   93  123  138  203  114
   84  113   75   82   80   77   64   67   74   91  112   95   95   83
  110   81   61   60   59   61   67   62   67   89   89   80   68   65
   64   60   58   64   51   55   54   56   55   73   95   88   55   62
   57   52   54   55   57   54   54   51   50   58   60   59   52   54
   45   54   63   52   54   54   53   52   50   49   46   47   50   55
   30   35  121  120  123  124  122  121  119  115  106   94   81   62
   31   29  121  123  123  125  130  132  128  127  129  125  120  120
   32   36  131  122  126  128  132  126  129  128  127  126  129  122
   28   35  126  130  129  132  134  132  132  132  132  121  124  123
   31   29  129  126  131  135  137  134  136  135  135  135  128  127
   34   31  132  132  131  137  131  134  131  135  129  128  130  125

Difference

  -45  -41   -1    0   -5    5    8   11   -7   29   21   35   40   81
  -57  -25    2  -11    6    7   22   34   32   42   53  106  117   96
  -39  -33   19    6    0   17   35   36   58   95   80   66    5   99
  -56  -68   26   25   33   42   59   88  127  111   92  108  120  132
  -80  -40   45   52   52   67  123  145  138  117  114  120  152  148
  -35  -21   55   49   65   85  152  151  152  130  108   98  136   73
  -27  -15   60   58   61   72  146  152  138  118   91   74   44   49
  -12  -16   55   65   66   71   90   96   89   43   53   36   49   33
   -1    2   -2   -1    1    0  -11  -11  -30  -25  -20  -11  -11   28
    5    4    6   -1    1   -2  -11  -57  -56  -35  -53  -46  -49  -34
    1    0   -7    4    6    4   -1  -60  -58  -56  -48  -54  -27  -48
    2   -6    5  -10   -1    3   -3  -32  -75  -63  -57  -23  -19  -52
    4    7    3    5    1   -3   -7  -10  -78  -80  -78  -64  -68  -66
    0    4   -2   -4    6   -2    3    4  -10  -60  -62  -66  -72  -69

In 5x5  )  Max : 152  Min : -30  Diff : 182  Middle :  61

Value of net dimming ( 91, 77)  :   7
Value of net convexity          :   4  e1 : 0  e2 : 2
```

Figur 15 **Intensitetsvärden i en 14 x 14 omgivning.**

```
      Q1  :  112, Q2  :  119, Q3  :  124, Q4  :  140, Q5  :  200
M01 : 0.19, M12 : 0.06, M23 : 0.04, M34 : 0.14, M45 : 0.52, M56 : 0.05
Number of peaks  :  12  , DecLev :   101.0  ,  154.7

 72   90  101  113  124  136  148  159  171  182  194  206  207
  **   **   **   **   **   **   *         *    **   **   **
  **        **   **   *    **                      **   *
            **   **                               **
            **   **                               **
                 **
                 **
                 *
```

Figur 16 Intensitetsfördelning i en 7 x 7 omgivning.

därför vara störst. Beräkning av nivåerna för segmentering, DecLev, sker genom en viktning av percentilerna kring den största toppen. Så har exempelvis nivån 101.0 erhållits som (90 + 112)/2 (Levelize i LocalSeg).

I figur 17 ges den lokalt segmenterade bilden tillsammans med beslut gjorda av konvexitetsoperatorn. Utskrift av besluten ges för nio angränsande punkter. Vi tar här endast upp beslutet för 7 x 7 omgivningen till punkt (90, 76). De med * markerade värdena har klassifierats som bakgrund till flugan. Definitionsområdet delas in i fyra koncentriska zoner. I respektive zon beräknas antalet markerade punkter. Utgående från resultatet från dessa zoner beräknas två besluts parametrar, e1 och e2, beskrivande konvexiteten respektive precisionen i beslutet. En stor konvexitet ger ett högt värde på e1 och god precision motsvaras av ett lågt värde för e2. Det kodade resultatet ges av Descision.

```
( 90, 76)                          Levels : 101.0 , 157.7

   *107   *113   *119   *123    155    201    202

   *112   *111   *128    190    207    205    206

   *113   *116   *140    206    207    207    203

   *113   *118   *126    200    203    188    176

   *117   *120   *125   *143   *148   *139     92

   *119   *124   *124   *111   *110     89     90

   *122   *124   *123   *119     75     72     92

Decision :  4                        (e1,e2)  : ( 0, 2)
Zone3 : 13    Zone2 : 10    Zone1 : 5    Zone0 : 0
```

Figur 17 Utdrag ur utskrift över beslut från konvexitetsoperatorn.

5. SLUTSATSER

Ett experimentsystem för studium av tidssekvenser av bilder har utvecklats. Baserat på kännedom om grodans synsystem har vi använt enkla sammankopplade operatorer för att detektera rörelser. Genom att göra operatorerna lokala finns möjlighet att implementera dem i parallell hårdvara. Vi har vid experimenten sett ett behov av att göra bildsegmenteringen beroende av intensitetsfördelningen. Det vi kallar halvmåneeffekten hanteras genom klassificering av lokala histogram. Klassificeringen av hörn görs genom beräkning av areor i koncentriska ytor. Detta ger ett nytt sätt att beskriva rörliga hörn.

Referenser

[1] L. Nielsen (1985). Simplifications in Visual Servoing. Ph D thesis CODEN: LUTFD2/(TFRT-1027), Department of Automatic Control, Lunds Institute of Technology, Sweden.

[2] J.Y. Lettvin, H.R. Maturana, W.S. McCulloch and W.H. Pitts (1959). What the Frog's Eye Tells the Frog's Brain. Proceedings of the IRE, November, pp. 1940-1951.

A. Menyer

Listning av programmets menyer.

```
---------------  MENU  ---------------

i  ImageSeqHandler   , m  MotionDescript
f  FreeRun           , l  LookAtOne
p  PresentImages

z  Zpecial           , h  ColorHandler
r  Recorder          , c  Camera

$  ControlStatus     , x  Exit

Command  >
```

```
--------  Image Sequence Handler  ------

g  GrabSequence    , v  VideoSequence
s  SaveSequence    , r  RestoreSequence
b  BackGround      , z  ReztoreNoBackG

Use [return] to exit  >
```

```
------------------------  Look At One  -----------------------

$  StatusControl     , ^  Curser            , x  Exit

d  OperNetDim        , c  OperNetCon        , m  MarkResults
t  ThreshDiffs       , s  SusContrast       , f  RefreshImages
v  ValueOffPos       , o  OperOnPosition    , w  OperOnWindow
h  HistoOffNetCon    , *  MotionDesc        , r  ResetOperImages

p  PrintWindow       , n  PrintNetDim       , i  PrintHistogram
3  Print NetConDecisions 3x3

[ Return ]   =   Menu    >
```

```
--------- Motion Description ---------

m  MotDesc          , f  MotDescFiling
c  ComputeOnFile    , $  ControlStatus

Use [return] to exit  >


---------- Present  Images ----------

%  ChangeIndex      , $  ControlStatus

f  Frog             , g  Background
r  SusConFrogThr    , o  SusConBackThr
s  Diff SusConFrogThr and SusConBackThr

t  TimeDiff         , b  BackDiff
i  SliceTimeDiff    , a  SliceBackDiff

d  NetDim           , c  NetCon
m  MarkNetDim       , n  MarkNetCon

h  ColorHandler

Use [return] to exit  >


---------- Control Parameters ----------

Sustained contrast : threshold   =    30
                   : nopixels    =     4
Time, back diff    : threshold   =    10
                   : nopixels    =     4
Net dimming        : boundary    =    25
Net convexity      : imagenoice  =    20
                   : max e2      =     3
Marking, netcon    : min inten   =     4

c  ChangeParam   ,  r  ResetParam

Use [return] to exit  >
```

B. Paketspecifikationer

Listning av paketspecifikationer för de i tabell 1 sammanställda programpaketen.

```
package FROGIMDEF is

-- Definitions and procedures to handle

     sequence, operator and result images.

-- Global constants, types and varibles defined are

     imagesize,    : the different sizes used for the image
     smallimsize,    representation,
     miniimsize

     imagevariant : names on all types of images used,

     imagetype    : matrix storing the imageintensities as
                    byte values,

     matrixtype   : matrix storing other information as
                    integer values,

     ImageIndex   : informs wich number in Sequence
                    to be handled,

     Sequence     : represents sequence and difference
                    images,

     BackGround   : represents the background image,

     OperArea     : represents operator and result images,

     NetDimArea   : stores values from the NetDimOperator.

-- Global constants used are

     numofimages  : defines the number of images in the
                    varible Sequence,

   must be defined in the mainprogram.

-- Refers to package

     Screen.

-------------------------------------------------------------

.CONST

{ seq image }
imagesize      = 128;

{ oper image }
smallimsize    =  64;
miniimsize     =  16;
```

```
.TYPE

imagevariant   = ( frog,                    { for var Sequence }

                  timed, backd,             { for DiffIm.pak }
                  timet, backt,

                  susconf, susconb,         { for SusCon.pak }
                  suscft,  suscbt,
                  diffsuscftbt,

                  netdim,                   { for NetDim.pak }

                  netcon,                   { for NetCon.pak }

                  backgr );               { for var BackGround }


imageline      = packed array[ 1..imagesize ] of bytes;
imagetype      = array[ 1..imagesize ] of imageline;

matrixline     = array[ 1..imagesize ] of integer;
matrixtype     = array[ 1..imagesize ] of matrixline;

imagedef       = record
                    Image  : imagetype;
                    Matrix : matrixtype;
                 end;

indextype      = 0..numofimages;

{ seq image }
sequencetype   = array[ indextype,frog..backd ] of imagedef;

{ backgr image }
backgroundtype = imagetype;

{ oper image }
operareatype   = array[ timet..netcon ] of imagedef;

netdimareatype = array[ 1..miniimsize, 1..miniimsize ] of
                 array[ 1..5 ] of integer;

.VAR

ImageIndex     : indextype;

Sequence       : sequencetype;

BackGround     : backgroundtype;

OperArea       : operareatype;

NetDimArea     : netdimareatype;
```

```
.FORWARD

procedure ResetSequence;                                    forward;
  { Clears the varible Sequence }

procedure ResetBackGround;                                  forward;
  { Clears the varible BackGround }

procedure ResetOperNetDimArea;                              forward;
  { Clears the varibles OperArea and NetDimArea }

procedure ChangeIndex;                                      forward;
  { Set a new ImageIndex }

.END
```

```
package CONTROL is

-- Definition of global parameters for

     algorithms and operators,
     result and image presentation

   and procedures to

     display and change the values of the parameters.

-------------------------------------------------------------

.VAR

{ Control variables }

  { SusCon }
    SusConThresh,
    SusConNOPixels,

  { TimeDiff and BackDiff }
    DiffThresh,
    DiffNOPixels,

  { NetCon }
    NoiceInFrogIm,
    Maxe2,

  { NetDim }
    NetDimBound,

{ Presentation variables }

  { NetCon }
    MinMarkValue    : integer;

.FORWARD

procedure ControlParam( var Changed : boolean );     forward;

procedure PresentParam;                              forward;

.END

-------------------------------------------------------------
```

```
package DIFFIM is

-- Procedures to

     create and threshold difference images.

-- Global constants, types and variables used are

     imagesize, imagevariant, ImageIndex, Sequence,
     OperArea

   defined in package FrogImDef,

     DiffThresh

   defined in package Control.
```

```
.FORWARD

procedure TimeDifference;                                  forward;
  { Calculates the difference between two frog images
    using current ImageIndex and ImageIndex-1.
    Puts the result in image timed with values 0 - 255
    and in matrix timed with values -255 - 255. }

procedure BackGroundDifference;                            forward;
  { Calculates the difference between frog image using
    current ImageIndex and the Background image.
    Puts the result in image backd with values 0 - 255
    and in matrix backd with values -255 - 255. }

procedure SliceTimeDiff( Ask : boolean );                  forward;
  { Eliminates noice in the time difference image.
    If Ask is true the user will be prompt for a value
    of Slice else the value is taken from DiffThresh.
    Puts the result in image timet. }

procedure SliceBackDiff( Ask : boolean );                  forward;
  { Eliminates noice in the back difference image.
    If Ask is true the user will be prompt for a value
    of Slice else the value is taken from DiffThresh.
    Puts the result in image backt. }

.END
```

```
package SUSCON is

-- Definitions of operators and procedures for

     sustained contrast ( edge detection ).

-- Global constants, types and variables used are

     imagesize, imagevariant, ImageIndex, Sequence,
     BackGround, OperArea

   defined in package FrogImDef,

     SusConThresh

   defined in package Control.

-------------------------------------------------------------

.FORWARD

procedure {Sobel}SusContrast;                          forward;
  { Calculates the contrast image on image frog
    in variable Sequence using current ImageIndex.
    The result is put in OperArea[ susconf ]. }

procedure {Sobel}SusConBackg;                          forward;
  { Calculates the contrast image on image backgr in
    BackGround. The result is put in OperArea[ susconb ]. }

procedure SusConFrogThresh( Ask : boolean );           forward;
  { Thresholds the contrast image OperAre[ susconf ].
    The result is put in OperArea[ suscft ].
    If Ask is true a prompt will be given to allow
    inputvalue for the thresholding. }

procedure SusConBackThresh( Ask : boolean );           forward;
  { Thresholds the contrast image OperAre[ susconf ].
    The result is put in OperArea[ suscft ].
    If Ask is true a prompt will be given to allow
    inputvalue for the thresholding. }

procedure DiffSusConFrogTBackT;                        forward;
  { Calculates the difference image between
    OperArea[ suscft ] and OperArea[ suscbt ].
    The result is put in OperArea[ diffsuscftbt ]. }

procedure OperAllSusCon;                               forward;
  { Procedure to do the total operation of sustained
    contrast. The value of ask is false. }

.END

-------------------------------------------------------------
```

```
package NETDIM is

-- Definition of

     the NetDimming-operator.

-- Global types and variables used are

     miniimsize, imagevariant, ImageIndex, Sequence,
     OperArea, NetDimArea

   defined in package FrogImDef,

     DiffThresh, NetDimBound

   defined in package Control.

-- Refers to packages

     LocalSeg, RastProc.

-----------------------------------------------------------

.FORWARD

procedure NetDimming;                                  forward;
  { Operates NetDimming on diffimage,
    puts result in OperArea[ netdim ]. }

procedure MarkNetDimInDiff( Size : integer );          forward;
  { Marks the net dimming result.
    If Size = 1 then the result is written on 256x256 large
    images in upper left and right corners else if Size = 2
    it is written on images of size 512x512. }

.END

-----------------------------------------------------------
```

```
package NETCON is

-- Definition of

     the NetConvexity-operator.

-- File defined is

     LPFile : allows output of intensities and descisions
              directly to a lineprinter.

-- Global types and variables used are

     smallimsize, imagevariant, ImageIndex, Sequence,
     OperArea, NetDimArea

   defined in package FrogImDef,

     vectortype

   defined in package LocalSeg,

     DiffThresh

   defined in package Control.

-- Refers to packages

     LocalSeg, RastProc.

-------------------------------------------------------------

.VAR

LPFile       : text;

.FORWARD

procedure NetConvexity( Row, Col, Size : integer;
                        ToPrinter      : boolean ); forward;
  { Operates NetConvexity on a specified window, where
    Row and Col points out center, on Sequence, using
    current ImageIndex. Puts result in OperArea[ netcon ].
    If ToPrinter is true the result is sent to a printer. }

procedure MarkNetConInDiff( Size : integer );       forward;
  { Marks the result of the netconvexity. If Size = 1
    then in the two images frog and diff at the place
    were they have been written, else if Size = 2
    then into a presentation image of 512x512 pixel. }

.END

-------------------------------------------------------------
```

```
package LOCALSEG is


-- Definitions of procedures for

     handling local segmention of 7x7 a imagearea to
     a binary image.

-- Global type defined is

     vectortype : array which represent the 49 intensity
                  values of the real-life image.


-------------------------------------------------------------

.TYPE

vectortype    = array[1..49] of integer;

.FORWARD

procedure SortVector( var Vec : vectortype );        forward;
  { Sorts the variable Vec, containing intensity values,
    in increasing order. }

procedure CreateM06(     Vec             : vectortype;
                     var M01, M12, M23,
                         M34, M45, M56   : real );    forward;
  { Creates six intervals with equal number of pixels.
    On return M01 to M56 contains the value of each interval
    divided by the total interval (Vec[4] - Vec[46]). }

procedure Levelize(      V               : vectortype;
                         M01, M12, M23,
                         M34, M45, M56   : real;
                     var Level1, Level2  : real;
                     var NOPeak          : integer );
                                                      forward;
  { Creates descision levels for the segmentation.
    On return Level1 is the low level for thresholding
    and Level2 is the high level. NOPeak tells how many
    peaks were detekted and if any of them were greater
    than the others }

.END

-------------------------------------------------------------
```

```
package FROGPROG is

-- The main menu of

     an experimental program to implement a visual system
     inspired by the structure of the frogs eye.

-- File variables declared in the programline are

     CoordFile    : described in package MotDesc,

     ImageFile    : described in package FrogFile,

     IntenFile    : described in package LookAt1,

     LPFile       : described in package NetCon.

-- Global constants and types defined are

     numofimages  : described in package FrogImDef,

     filenametype : described in package FrogFile.

-- Refers to packages

     Control, Extern, FreeRun, LookAt1, MotDesc, Present,
     RasterReg, RastProc, Screen, SeqHand, SpecShow.

-------------------------------------------------------------

.PROGRAM

program MAIN( input, output,
              CoordFile, ImageFile, IntenFile, LPFile );

.CONST

numofimages             =  10;

.TYPE

filenametype            = packed array[ 1..60 ] of char;

.MAIN

MainMenu;

.END

-------------------------------------------------------------
```

```
package SEQHAND is

-- A menu to

     get and store new and restore sequences.

-- Global constantes, types and variables used are

     imagevariant, ImageIndex

   defined in package FrogImDef,

     numofimages

   described in package FrogImdef,
   defined in package FrogProg.

-- Refers to packages

     DiffIm, Extern, FrogImDef, FrogFile, RasterReg,
     RastProc, Screen.

-------------------------------------------------------------

.FORWARD

procedure SeqHandler;                                  forward;

.END

-------------------------------------------------------------
```

```
package LOOKAT1 is

-- procedures to examine one image in the sequence

-- File defined is

     IntenFile : for output of intensity information
                 directly to a lineprinter.

-- Global types and varibles used are

     miniimsize, imagevariant, ImageIndex, Sequence,
     OperArea, NetDimArea

   defined in package FrogImDef,

     vectortype

   defined in package LocalSeg.

-- Refers to packages

     RastProc, Control, DiffIm, FrogImDef, LocalSeg,
     Motdesc, Mouse, NetCon, NetDim, Screen, SusCon.

------------------------------------------------------------

.VAR

IntenFile : text;

.FORWARD

procedure LookAtOne;                                  forward;

.END

------------------------------------------------------------
```

```
package FREERUN is

-- A procedure to

     operate on a whole the sequence at one time.

-- Global constants, types and variables used are

     miniimsize, imagevariant, ImageIndex, OperArea

   defined in package FrogImDef,

     numofimages

   described in package FrogImDef,
   defined in package FrogProg.

-- Refers to packages

     FrogImDef, NetCon, NetDim, RastProc, Screen.

-------------------------------------------------------------

.FORWARD

procedure FreeRun;                                    forward;

.END

-------------------------------------------------------------
```

```
package MOTDESC is

-- A procedure for motion description in a sequence.

-- File defined is

     CoordFile : used for output of coordinate values.

-- Global constants, types and variables used are

     smallimsize, miniimsize, imagevariant, ImageIndex,
     OperArea

   defined in package FrogImDef.

-- Refers to packages

     Control, FrogImDef, NetCon, NetDim, RastProc, Screen.

-------------------------------------------------------------

.VAR

CoordFile : text;

.FORWARD

procedure MotionDescript;                              forward;
  { Menu }

procedure GetCoord;                                    forward;
  { Subtracts those charecteristic points, which
    makes up the object, from OperArea[ netcon ]. }

procedure PackCoord;                                   forward;
  { Purges equal points. }

procedure MotDescOne( Monitor : boolean );             forward;
  { Performs the motion descript algorithm.
    If Monitor is true the result is presented in the
    images frog and timed ( written in size = 256 ) }

.END

-------------------------------------------------------------
```

```
package PRESENT is

-- A menu to

     present images and results in size 512x512 pixel.

-- Global types and variables used are

     miniimsize, imagevariant, ImageIndex

   defined in package FrogImdef.

-- Refers to packages

     Control, DiffIm, FrogImDef, NetCon, NetDim, RastProc,
     Screen, SusCon.

-------------------------------------------------------------

.FORWARD

procedure PresentImages;                              forward;

.END

-------------------------------------------------------------
```

```
package SPECSHOW is

-- A procedure to
     operate on an image step by step.
-- Global constants, types and variables used are
     miniimsize, imagevariant, ImageIndex
   defined in packege FrogImDef.
-- Refers to packages
     FrogImDef, NetCon, NetDim, RastProc, Screen.

-------------------------------------------------------------

.FORWARD
procedure SpecialShow;                                forward;
.END

-------------------------------------------------------------
```

```
Package FROGFILE is

-- Definition of procedures

     to save and restore image sequences.

-- File defined is

     ImageFile :  to read and write Sequence and Background
                  image information from/to a selected file.

-- Global constants, types and varibles used are

     imagevariant, imageline, Sequence, BackGround

   defined in package FrogImDef,

     numofimages

   described in package FrogImDef,
   defined in package FrogProg,

     filenametype

   must be defined in the main program as

     packed array[ 1..60 ] of char;

-- Refers to package

     Extern.
```

```
.VAR

ImageFile            : file of imageline;

.FORWARD

procedure SaveSequence;                                    forward;
  { Saves Sequence on file. }

procedure RestoreSequence( var Restored : boolean );  forward;
  { Restores Sequence from file.
    At exit Restored is true if a file was restored. }

procedure RestoreNoBack( var Restored : boolean );     forward;
  { Same as RestoreSequence except no backgr has been saved.
    At exit Restored is true if a file was restored. }

.END
```

```
package RASTPROC is

-- Procedures to

     read from and write to the Raster Memory,
     program the Look Up Table.

-- From packages Raster and LookUp used procedures are

     RasterRead, RasterWrite, DrawPlane, RasterCol,
     WritePixel and SetColorMap, BlackAndWhite256.

-- Global constants, types and varibles used are

      imagesize, smallimsize, miniimsize, imagevariant,
      ImageIndex, Sequence, BackGround, OperArea

   defined in package FrogImDef.

-------------------------------------------------------------

.FORWARD

procedure ReadImage( ImVar : imagevariant );          forward;
  { Reads Raster memory to Sequence, using current ImageIndex,
    or to BackGround depending on the value of ImVar. }

procedure WriteImage( ImVar : imagevariant;
                      Size  : integer     );          forward;
  { Writes to Raster memory from Sequence, using current
    ImageIndex, BackGround or from OperArea depending on the
    value of ImVar. If Size = 1 the images are written in size
    up to 256x256 pixels. If Size = 2 the size is 512x512. }

procedure SetDot( y, x, Col : integer );              forward;
  { Writes one pixel in color Col at coordinate (y,x) }

procedure DrawLine( y1,x1, y2,x2, Col : integer );    forward;
  { Draws a line in color Col between (y1,x1) and (y2,x2) }

procedure TestColors;{ yStart, xStart, Size }         forward;
  { Creates referencearea for the 10 darkest intensities.
    Value of placement and size as parameters. }

procedure MarkColors;                                 forward;
  { Colors the 10 darkest intensities }

procedure ColorHandler;                               forward;
  { Menu to mark and reset colors and to create testcolors. }

.END

-------------------------------------------------------------
```

```
package MOUSE is


-- Definitions of procedures to

     use a mouse to move a curser.

-- This package is based on procedures in the packages Extern,
   InitInter and Interact to handle input from mouse and
   keyboard. From package Interact used procedure is

     GetMouse.

-- Global types used are

     imagetype

   defined in package FrogImDef,

     buttontype

   defined in package Interact.

-- Refers to packages

     Interact, Raster, RastProc.


-------------------------------------------------------------

.CONST

blackdot        =   0;
whitedot        = 255;

.VAR

BUFrog, BUTimed : imagetype;

.FORWARD

procedure WriteCurser( r, c : integer );            forward;
  { Writes two cursers at position (r, c) and stores the
    overwritten area in backup images.
    The images the cursers are active in should be written
    in upper left and upper right corners and have the size
    256x256 pixel. The values off r and c should be in the
    range off 14 to 118 in correspondens to the defined
    imagesize = 128. }

procedure UnWriteCurser( r, c : integer );          forward;
  { Erases the two cursers at position (r, c) and rewrites
    the curser area from stored backup images.
    The images the cursers are active in should be written
```

```
    in upper left and upper right corners and have the size
    256x256 pixel. The values off r and c should be in the
    range off 14 to 118 in correspondens to the defined
    imagesize = 128. }

procedure CurserHandler( var Row, Col : integer );  forward;
  { Moves the two cursers. At exit Row and Col returns
    the new location of the curser. The three buttons
    on the mouse has the following meaning :
    left button              =  exit,
    center button            =  move cursers,
    right button             =  write current position,
    center and right button  =  draw window size 14x14. }

.END
```
