QUERY BASED VISUAL ANALYSIS: SPATIO-TEMPORAL REASONING IN COMPUTER VISION

Hilary Buxton[†] *and Nick Walker*[‡]

Department of Computer Science[†],
Queen Mary College,
London E1 4NS.

Imperial Cancer Research Fund Laboratories[‡],
Lincolns Inn Fields,
London WC2A 3PX.

Abstract

Two experimental systems for query based visual analysis are described. The first simulates an image sequence of moving, dividing cells with simple rules and monitors significant visual events. The second processes single raw images of real cells. Both invoke appropriate processing using explicit knowledge to respond to user queries. We propose that this selectivity is an essential feature for any system to analyse raw image sequences of moving, dividing cells as the computational expense of allowing all possible processing to proceed is enormous. Processing as required by the query allows adaptive strategies (e.g. different resolutions and focal processing) to be utilised and gives an effective attentional control structure to the system.

1. Introduction

The interpretation of visual data is a classically underdetermined problem and appropriate knowledge, both general physical constraints and specific object knowledge, must be used at all levels of the visual system. Here we are concerned with image motion analysis and dynamic interpretations in computer vision systems. Many algorithms have been developed to extract information from a time-series of images captured by a moving camera (eg Buxton and Buxton, 1983 and Buxton, Buxton, Murray and Williams, 1984). The basic steps in this spatio-temporal processing are first, measurement of the visual motion or optical flow, second, an interpretation of the flow data in terms of the relative translational and rotational motion and the visible surface slant, tilt and depth, and finally, obtaining a fully dynamic, symbolic interpretation of the surface and motion information in terms of the objects and their changing 3D spatial relationships. However, a great deal of research remains needs to be done to adequately model reasoning about visual evidence consistent with models of objects in the domain and their interactions using logic and knowledge based techniques.

Typically, "low-level" vision research emphasises "bottom-up" mappings (as in the three stages above which derive an image based description, then a surface based description, and then an object based description) and uses a numerical algorithmic approach based on a full mathematical formalism or "computational theory". "A set of evidence", a visual representation of some kind, eg. edge features detected in an image, is processed to obtain "an interpretation", another visual representation, eg. surfaces in depth consistent with the edge feature evidence. The mapping between the levels is driven by knowledge of the relationships between parameters in the two representations and is coded as a set of constraints that should be satisfied in the processing. Typically, this is general knowledge about the imaging of structures in the physical world. So, for instance, there is the "continuity constraint" used in many visual processing algorithms that states that

AVC 1987 doi:10.5244/C.1.29

neighbouring points in the 2D image should have smoothly varying depth values almost everywhere in the 3D interpretation as they are usually projections of points which lie on the same surface in the scene. In some sense, then, this codes "deep knowledge" of vision into the processing schemes. The problem remains, however, that for a flexible visual system, this quantitative modelling of the processing needs to interact with more abstract (often qualitative) knowledge concerning objects and events in the domain.

In the "AI approach to computer vision", described above, the computational theory is an account of both the human perceptual mechanism and a basis for many computer algorithms. The implementation of these fully explicit formalisms could, in principle, use constraint satisfaction, connection graphs, an extended logic, production rules etc rather than the numerical algorithms. The form of the implementation is chosen to match the task imposed and is usually experimental. Using mathematical formalisms seems to have brought a deeper understanding to the science of early visual processing (at least for the initiated!). The precision of expression tends to ensure that equivalent formalisms are obviously equivalent, that formalisms for different parts of the system can be combined, and that fundamental distinctions can be communicated without confusion with the particular processing algorithms used in a demonstation. Extended logics (Turner, 1984) are becoming popular in AI and offer a similar degree of formalism for higher-level reasoning about space and time.

The proposal is, then, to incorporate quantitative visual algorithms in a flexible system that includes qualitative reasoning about space and time using an explicit knowledge base. The target application is the interpretation of time-lapse sequences of moving, dividing cells where representations of the concepts of motion, shape, collision, normal division, ancestor relationships etc will be needed to answer queries about events in the image database. Current AI vision research typically results in algorithms to do processing but stops short of representation of higher level knowledge of visual events or any explicit representation of the mapping of one level onto another. Explicit semantics right down the system in the form of metalevel knowledge of these mappings (a declarative form of the computational theory) would allow higher level knowledge of the task or query to control the processing. It is clearly not enough to just stick knowledge on top of a fully extracted database as seen in most current attempts at integrating domain rules with image processing as this does not give flexibility of control (processing on demand) down through the lower levels of the system. In particular, if a formal spatio-temporal logic were used uniformly thoughout the system, problems of consistency and completeness could be addressed. Within the framework of a proof based system this would mean that a given query sets up a model for instantiation in the representational databases of the system. The control could be backward-chaining through the levels allowing instantiation at that level if the fact/relationship had already been extracted but would allow typical sub-goal proof at lower levels if required.

For visual analysis it is likely to be most effective to precompute some initial database with a set of low-level algorithms which would still require their own explicit metalevel knowledge representation since selected query driven control is desirable even below this level, for example, when a higher resolution analysis of the image data is required. We will call this database the Processed Image Database (PID) and control could be forward-chaining using lowest-level metaknowledge to assert a set of facts in the lowest level of the Maintained Image Database (MID). The higher level MID facts would represent qualitative relationships between entities in the domain while lower level MID facts would represent quantitative relationships between entities such as edge fragments. To establish a fact at a highish level might require only reasoning using the declarative metalevel knowledge of the mappings but at lowish levels it could be more effective to use the metalevel knowledge to call an algorithm to extract the fact from the lower level MID facts. The metalevel knowledge is in the form of axioms of the space-time logic either domain independent or domain specific. The queries in this system invoke further processing using domain specific knowledge of what the query means in terms of the highest-level MID facts. This in turn sets up a model for proof down through the lower levels using the metalevel knowledge of the mappings.

Section 2 briefly describes an experimental system implemented on a SUN3 workstation at the Imperial Cancer Research Fund Biomedical Computing Group using the PROPS2 expert system language. PROPS2 allows the representation of facts and rules in 'pseudo-English' as well as many other extensions to the earlier ICRF PROPS systems which explore the use of a triple control structure combining 'forward' or data-driven reasoning, 'backward' or goal-driven reasoning, and an explicit task agenda. Such systems have proved very flexible in building knowledge-based systems when combined with a user interface which brings the power

of the inference engine to the user at any time via an interpreter. The interpreter processes all input, whether commands, queries, answers or data for the system.

Section 3 then describes a simple prototype query based processing system for single images that will be developed into a full system to analyse image sequences of these moving, dividing cells. Finally, Section 4 contains a discussion of the proposed extension and an outline of the issues that may be addressed in such a project.

2. Simulation, Analysis and Query in a Toy World of Moving, Dividing Cells

The challenge here was to design a toy vision system for analysis and query of a simulated world of moving, dividing cells as a basis for the full system by incorporating some of the key features. The main question addressed was the control of subparts of the system together with subsidiary questions of the knowledge representation in this domain. The requirement was for a procedural command sequence of update cycles for the simulation subsystem plus a freely forward chaining analysis subsystem which automatically asserts a history of significant events in the cell world to set up the visual database. Also, a query subsystem was required to allow further processing of the database as needed by the particular query. The last requirement was the most exacting for the PROPS shell as the basic strategy for queries is backward-chaining on an incomplete database with user queries to establish a conclusion. Here, however, the information in the database would be complete (it is assumed that the user does not supply data in this simple vision system) and the basic free forward-chaining strategy would assert all possible conclusions without some further control. Full truth-maintenance of this kind, we suggest, would be computationally infeasible in the full system and leads to a requirement to impose selective query-driven processing as discussed in the section on Query Based Control.

In the following subsections, first, the System Structure is described in terms of the knowledge bases that comprise the simulation, analysis and query subsystems; second, the System Performance is described for a typical session in which a starting set of cells undergoes a few update cycles and the database is used to answer a set of queries; and third, alternative solutions to the problem of Query Based Control are discussed.

2.1. System Structure

The complete system is arranged in a simple modular set of knowledge bases that are easily extensible. The simulation subsystem is procedurally controlled in cycles that update the world which consists of a set of cells with known size, position, age and type and then updates the database to note which cells are currently too close as neighbours and need to be moved on the next cycle (see Figure 1):

```
if command cycle
then update world
and update database

if command update world
then age cells
and move cells
and grow cells
and divide cells

if command update database
then retract C1 | C2 are too close | Diff

if command update database
and cell C1 is size S1|xpos X2|ypos Y2| Anything
and cell C2 is size S2|xpos X2|ypos Y2| Anything
and C1 isnt C2
and neighbour limit = Limit
and Diff = ((X1-X2)*(X1-X2)) + ((Y1-Y2)*(Y1-Y2))
and Diff < Limit
then C1 | C2 are too close | Diff
```

Note that the variable terms (denoted by capitals) are instantiated by pattern matching in the database and are unified within a single rule. As usual in a production system, the 'if' part of the rule must be satisfied by assertions in the database and the 'then' part of the rule can retract and assert new data as well as firing new rules.

The aging, moving, growing and dividing of the cells is straightforwardly rule based and proceeds pseudo-parallel for each of the current set of cells. The rules are currently very simple, for example, growing cells:

```
fact growth increment type healthy = 1
fact growth increment type abnormal = 2

if command grow cells
and cell C is size S | Anything
then grow cell C

if command grow cell C
and cell C is size S | Anything
and growth increment type T = Inc
and A < 20
and New = S + Inc
then myretract cell C is size S | Anything
and myassert cell C is size New | Anything
```

This makes the cell grow 1 unit in size if it has type healthy and is not yet 20 units in age or grow 2 units if it is abnormal. The move cell rules are the most complex for the simulation and due to difficulty in handling negative numbers consist of four rules to determine moves north, south, east or west and one rule to update the position.

The analysis subsystem freely forward chains on new data from the simulation and notes a history of significant events for each cell as well as keeping track of the current set of cells that fall within a given class. The event history is not recycled unlike the dynamically updated simulation. The significant events for a cell are currently: first coming into existence; first overlapping another cell; first becoming free of the overlap; and dividing which involves retraction. An example of simple analysis rules involving cell existence are:

```
if cell C is Anything
and not cyclenumber of existence of C is N1
and cyclenumber = N2
then cyclenumber of existence of C is N2
and cells existing includes C

if cyclenumber of division of C is N
then retract cells existing includes C
and retract cells overlapping includes C
and retract cells free includes C
```

These rules fire appropriately when cells are first asserted and when they have divided respectively. The division, first overlap and first free events are noted in a similar way with retraction of cell overlap when a cell becomes free etc. Note that cell number and cycle of the event are also recorded.

The query subsystem currently performs command flagged selective processing of the current database as updated by both the simulation and automatic analysis Only a fixed restricted syntax for the queries is implemented with, for example, number in class queries allowed by a simple rule:

```
if command nquery E
and number of E = N
then display number of E is N
and continue
```

The other queries allow more complex relationships to be established including the set of significant events for a particular cell; the temporal order of two events; the spatial relationship of two cells; and the ancestor relationship of two cells. For example, the simple time order rules are:

```
if command tquery E1 of C1 | E2 of C2
and cyclenumber of E1 of C1 is N1
and cyclenumber of E2 of C2 is N2
and N1 < N2
then E1 of C1 is before E2 of C2

if command tquery E1 of C1 | E2 of C2
and cyclenumber of E1 of C1 is N1
and cyclenumber of E2 of C2 is N2
and N1 > N2
then E1 of C1 is after E2 of C2

if command tquery E1 of C1 | E2 of C2
and cyclenumber of E1 of C1 is N
and cyclenumber of E2 of C2 is N
and N1 = N2
then E1 of C1 is same time as E2 of C2

if command tquery E1 of C1 | E2 of C2
and E1 of C1 is Time E2 of C2
then display E1 of C1 is Time E2 of C2
and continue
```

The ancestor rules force some further processing if a direct relationship is not found while the spatial relationships and event reporting involve straightforward pattern matching and comparison in the database.

The knowledge bases of the three subsystems could all be made more sophisticated but are sufficient to illustrate some of the main aspects of the full system as discussed in the section on Future Extensions.

Summary

The pworld directory contains knowledge bases as follows:

top level and tools

ptools - general knowledge base user commands
pworld - general cellprops commands and setup all pworld
pdbman - general pworld database status reports
pcells - initial declaration of cells specification

simulation of world

pcycle - cycles of update world and database
pagecell - ages each of the current set of cells
pgrocell - grows each current cell if appropriate
pmovcell - moves each current cell according to neighbours
pdivcell - divides each current cell if appropriate

analysis of world

pnbour - current set of cells close to each other - recycled
pexist - notes cycle of existence of cells and existing set
pdivide - notes cycle of division of cells and divided set
poverlap - notes cycle of overlap of cells and overlapping set
pfree - notes cycle of free of overlap of cells and free set

query of world

pnumber - computes and reports number of events of class E

pevent - computes and reports significant events of cell C
pspatial - computes and reports spatial relation of cells C1 C2
ptime - computes and reports temporal relation of events E1 E2
prelate - computes and reports ancestor relation of cells C1 C2

2.2. System Performance

A typical session might consist of an initial assertion of a set of four cells from which the rules for analysis would establish the full database:

```
4 | 2 are too close | 50
2 | 4 are too close | 50
3 | 1 are too close | 50
1 | 3 are too close | 50
number of cells existing = 4
cells existing includes 1
number of cells overlapping = 4
cells overlapping includes 3
cells overlapping includes 1
cell 1 is size 2 | xpos 25 | ypos 25 |
                    age 2 | type healthy
cells existing includes 3
cell 3 is size 6 | xpos 30 | ypos 30 |
                    age 4 | type abnormal
cells existing includes 2
cells overlapping includes 4
cells overlapping includes 2
cell 2 is size 4 | xpos 50 | ypos 40 |
                    age 6 | type healthy
cells existing includes 4
cell 4 is size 8 | xpos 45 | ypos 45 |
                    age 8 | type healthy
```

The cells are displayed by a SUN window graphical interface and the existence, division, overlap, position, size etc noted during the update cycles of the simulation and analysis.

The command 'go' starts the cycling and time stamping of cycles for events. After a few cycles, the database grows quite large as it contains an increasing set of current cells by division as well as the current neighbours and the full history of significant events and class membership. For example:

```
number of cells overlapping = 5
cells overlapping includes 5
cyclenumber of 5 is overlapping 6 is 2
cells overlapping includes 6
cyclenumber of 6 is overlapping 5 is 2
number of cells divided = 1
cells divided includes 3
cyclenumber of division of 3 is 2
number of cells existing = 5
cells existing includes 5
cyclenumber of existence of 5 is 2

etc.......
```

This database can be processed by particular queries. For example, the number query 'nquery cells existing' would give: 'number of cells existing is 5'. The event query 'equery 3' would give: 'significant events for cell : 3 is in existence on cycle 0 ; 3 is divided on cycle 2 ; 3 is first overlapping 1 on cycle 0 ; 3 is first free of 1 on cycle 1'. The spatial relationship query 'squery 1 | 2' would give: '1 is currently northwest of 2'. The temporal query 'tquery division of 3 | existence of 1' would give: 'division of 3 is after existence of 1'. Finally, the ancestor relation query 'rquery 3 | 5' would give: '3 is ancestor of 5' as expected by the rules and events in the simulation.

This set of examples gives a flavour of the toy system implemented in pworld. The query subsystem is the most restrictive and requires further discussion in the next subsection.

2.3. Query Based Control

As discussed in the introduction, the problem with query driven processing in PROPS is the assumption of a basic strategy of backward chaining on query. For example, 'division of 3 is before existence of 1?' would normally check the database for a matching assertion, then try to derive it from rules having a conclusion of that kind from assertions already made and finally resort to user query to reach an answer. This is inappropriate for our system since we have complete information as the system proceeds through the cycles, we just do not want to reach all possible conclusions by full forward chaining. A simple guard like asserting 'nquery' alone is also not sufficiently selective as it just delays the moment when full truth maintenance is allowed. The method adopted guards the particular set of assertions made by being specific to the cell(s) or event(s) in question. However, this seems clumsy and not easily amenable to the extension for a full query grammar where arbitrary queries like 'was the number of cells in existence 5 when cell 2 divided?' would be allowed. A possible alternative which will be explored is to keep the backward chaining strategy on query which allows the appropriate degree of specificity but block the free forward chaining by only swapping in rules from appropriate knowledge bases at the time of query. The rules can be unguarded since it is only new items in the database that trigger the forward chaining.

The ideal for the full system is a query based natural language interpreter with access to the full explicit set of axioms and semantics of a spatio-temporal logic for all levels of analysis and processing. A query will thus

set up a model for its proof at the top level and recursively backward chain through the levels until instantiated even if this involves further selective processing of the raw image sequence. An experimental system for single raw images of real cells which illustrates this kind of processing is described in the next section.

3. Analysis and Query of Single Real Images

In this system the essentials of the query based processing using the PROPS2 language have been implemented. The requirement was for control by user query and for simplicity no preprocessing was assumed. The whole system, then, with processing right down to the lowest level smoothing operations was set going on demand. This may not be optimal for the full system but some empirical testing is necessary to establish the appropriate level of preprocessing that is effective.

In the following subsections, first, the System Structure is very briefly described in terms of the knowledge bases comprising the processing and query subsystems, and second, the System Performance is described for a typical session in which the user makes queries that invoke the required processing unless this has already been completed in response to a previous query. The assertions already in the database will then enable immediate response to the current query.

3.1. System Structure

The complete system is again arranged as a modular set of knowledge bases that can be extended as appropriate. The interface on the SUN workstation consists of a menu of user commands and queries with PROPS2 input and output windows and an image canvas on which the user can move a pointer to indicate a region or cell of interest (see Figure 2). This allows the user to selectively initiate all their interactions with the system. The queries implemented were based on a survey of interesting questions for an image database and illustrate four simple generic types of query for single images: (1) What [things] have been recorded; (2) How many [objects] are [class]; (3) What [metric] is this [object]; and (4) Which [objects] are [relation] this [object]. The only object we considered here was a cell and the full query grammar was available via the menu system as 'Records', 'Cell Counts', 'Cell Metrics' and 'Cell Layouts'. Although these queries are representative, they are not the result of a systematic task analysis.

The processing subsystem is a full implementation of image smoothing, Canny edge detection (Canny, 1986), segmentation, grouping, closed region finding and feature extraction for single images (see Figure 3). These operations are available globally in a region of interest and focally as indicated by a pointer in the region. The system also incorporates basic commands to view a selected image or selected subregions and do any of the processing steps alone for testing purposes again via the menu system. Edge detection, segmentation, grouping, closed region finding and the features position, size, shape and brightness are all available at a range of scales (spatial resolutions). The parameter to control this is called 'Level' and has a default value in the system so a command is, for example, 'do grouping at Level'. The closed region and features are applied to objects found after grouping so a command is, for example, 'do position at Level | Object'. Each operation has access to an explicit statement of prerequisite operations, for example, 'fact prestep of grouping at Level includes canny at Level'. There is a task scheduler written in PROPS2 to ensure that all operations are carried out in the appropriate order. All operations assert their completion, for example, 'done canny at Level'. This enables a simple control structure for queries to test if the necessary processing is complete.

The query subsystem has available explicit rules for constructing the response to any of the queries. In the case of 'Record' queries, this consists of pattern matching in the data for records of a certain type. However, in the case of 'Cell Counts', 'Cell Metrics' and 'Cell Layouts', the response depends on establishing some prerequisite conditions by further processing unless it is already completed. For example, the rules that count cells are:

```
if count Class
and scalelevel is Level
and not done objects at Level
then do objects at Level

if count Class
and scalelevel is Level
and total objects at Level is Total
then objectnumber at Level is 1
and setall objects upto Total at Level
and doall attributes of Class at Level

if count Class
and attribute of Class includes Attribute
and Attribute of Object at Level is Value
and Attribute of Class is Value
then update Class total of Object at Level

if count Class
and Class total of Object at Level is Num
and attribute total of Class is Num
```

```
then Class includes Object at Level
```

The conditions that must be satisfied for the cells to be classified as touching, dividing, dying, elongated, or rounded are specified as facts, for example, 'fact attribute of dying cells includes brightness' and 'fact brightness of dying cells is negative'. Touching cells are more complex as they involve testing for those that merge when processed at a coarser resolution. This kind of processing is global on all objects found in the region of interest. The 'Cell Metric' and 'Cell Layout' queries require the user to indicate a cell using the pointer. This cell is processed focally using the rules in an object knowledge base which defines what it is to be a cell. The 'Cell Metric' rules also define each metric (position, size, shape, brightness, distance) available in the system. Similarly, the 'Cell Layout' rules define the spatial layouts (north of, south of, east of, west of).The essential control is given by simply testing whether processing is already done with exceptions for repeatable operations like viewing the region and selecting a focal cell.

Summary

The iprocess and iworld directories contain knowledge bases as follows:

top level

itools - general commands and iworld knowledgebase modules
iload - loading rules for iprocess knowledgebase modules
istages - list of processing and task handling modules
iclean - resets the database for a new image

task, resource and output

iexpand - uses processing prestep facts to generate tasks
ires - handles processing resource allocation
together with ialloc inewres irealloc iassign
ioutput - handles output to data from the processing
ishow - handles display of results from the processing

processing hierarchy

ismooth, icanny, isegment etc - set of processing presteps
for each of the possible processing operations

command menus

isetup - allows setup of system and parameters
ibasic - invokes image/region views and basic processing
iglobal - invokes the global processing
ifocal - invokes the focal processing

query of image database

irecord - query knowledge and database processing
to find things previously processed
ivals - object class knowledge with attributes
icount - class handling and global processing of image
to find and count cells in region of interest
icell - object knowledge and focal processing of image
to find cell at pointer in the image region
imetric - metric knowledge and focal processing of image
to find features of the focal cell in the image region
ilayout - spatial knowledge and global and focal processing
to find cells in the region related to the focal cell

3.2. System Performance

The session would typically start with the user request to view a particular image and select a region of interest. The requests that follow might be, for example, 'count cells'. This would invoke the global processing to find all objects in the region of interest since this has not yet been done. This in turn will invoke the prerequisite grouping, segmentation, edge detection and smoothing. In fact, if the image and region were not already loaded and selected, this would be invoked as well. The rules also test to check if the objects found are cells, count the objects in the cells class with the 'number of' predicate and finally, display the result for the user. The database would then contain lots of information about the objects in the region of interest:

```
number of cells = 5
cells includes object 8 at 32
cells includes object 5 at 32
etc ...
size of object 11 at 32 is 6
closed of object 11 at 32 is unclosed
etc ...
total objects at 32 is 11
done objects at 32
```

A further query to find the cells north of a selected cell would then not repeat the global object finding processing. It would, however, invoke processing to find the focal cell, test which cells are north and display the result. Queries to cell shape, position etc also invoke focal processing to find the selected cell and compute the appropriate metric.

These examples illustrate the simple queries implemented in the single image system but it should be noted that a larger range of queries are envisaged which make more use of processing at finer resolutions to establish the cell morphology. In the full system to analyse time-lapse films of moving, dividing cells, an extended space-time query grammar will be implemented. This will use recursion to allow complex queries with more than one clause.

4. Conclusions and Future Extensions

The most obvious extension for the full system is to work off image sequences. Currently work is progressing on the extraction of a description of time-lapse film data over a range of scales in space and time. This Processed Image Database then forms the basis of a set of explicit assertions in the lowest level of a Maintained Image Database which consists of simple edge assertions at each scale. The edges need to be analysed to assert possible closed cell regions together with their position and motion to give a second level MID before any reasoning about domain objects can be done. The effectiveness of object level knowledge in finding these regions also needs investigation.

Most important, however, is the development of a full spatio-temporal logic for both the qualitative and quantitative levels of analysis in the system to support full query-driven processing. A calculus that could handle events in time and space would allow the higher level knowledge of the domain to influence the lower level quantitative visual processing in a flexible way with a notion of current "focus of attention" as the query model path. Such a calculus would have application in modelling and reasoning about physical systems in general but for the particular cell analysis system here would involve formalisation of spatio-temporal knowledge about the domain.

To make substantial progress on a project of this kind is a huge undertaking and subsumes a variety of questions, in addition to those above, that have not been tackled in the experimental systems. For example, the empirical evaluation of how much automated image pre-processing is efficient in such a system; how the basic visual database and derived representations should be structured to facilitate further processing etc. In the full system it may also be appropriate to allow different paths and modes of reasoning eg we might have different axioms with the same conclusion or different algorithms with same metalevel function eg analogue style reasoning as well as proposional reasoning (Sloman 1986). It will also be necessary to look carefully at incorporating medical and clinical knowledge about the source of the cell sample in reaching any conclusion about the significance of the visual analysis.

The systems described, however, illustrate many of the features expected in the full system and clearly raise the question of reasoning in incomplete databases. The solutions offered in the section on query based control are not really adequate but a full logic of attentional processing which solved the problem of invocation of appropriate knowledge would be an amazing contribution to KBS systems in general. The basic structuring of the subsystems for analysis and query will be useful in the full system and the notion that significant events are automatically asserted by forward chaining on the database seems an effective strategy that will endure in addition to goal directed reasoning from queries.

References

Buxton BF and Buxton H 1983 "Monocular Depth Perception from Optical Flow by Space-Time Signal Processing" Proceedings of the Royal Society of London B, 218, pp27-47.

Buxton BF, Buxton H, Murray DW and Williams NS 1985 "Structure from Motion Algorithms for Computer Vision on an SIMD Architecture" Computer Physics Communications, 37, pp273-280.

Canny J 1986 "A Computational Approach to Edge Detection" IEEE PAMI, 8, pp679-698.

PROPS2 Reference Manual. Imperial Cancer Research Fund Laboratories, London.

Sloman A 1986 "What are the Purposes of Vision?" Cognitive Studies Research Papers CSRP066, University of Sussex, Falmer, Brighton.

Turner R 1984 "Logics for Artificial Intelligence" Ellis Horwood.

Acknowledgements

Many thanks to all at the ICRF who helped with this work, especially to Saki Hajnal for the graphical interfaces for CELLPROPS and IMAGEPROPS on the SUN3 workstation. Thanks also to John Fox, David Frost and Chris Rawlings for discussions on the features required for the full system of real image sequence analysis and query. Acknowledgement of financial support is made for my Alvey IKBS/060 grant which is part of the 3D Vision Research Theme.

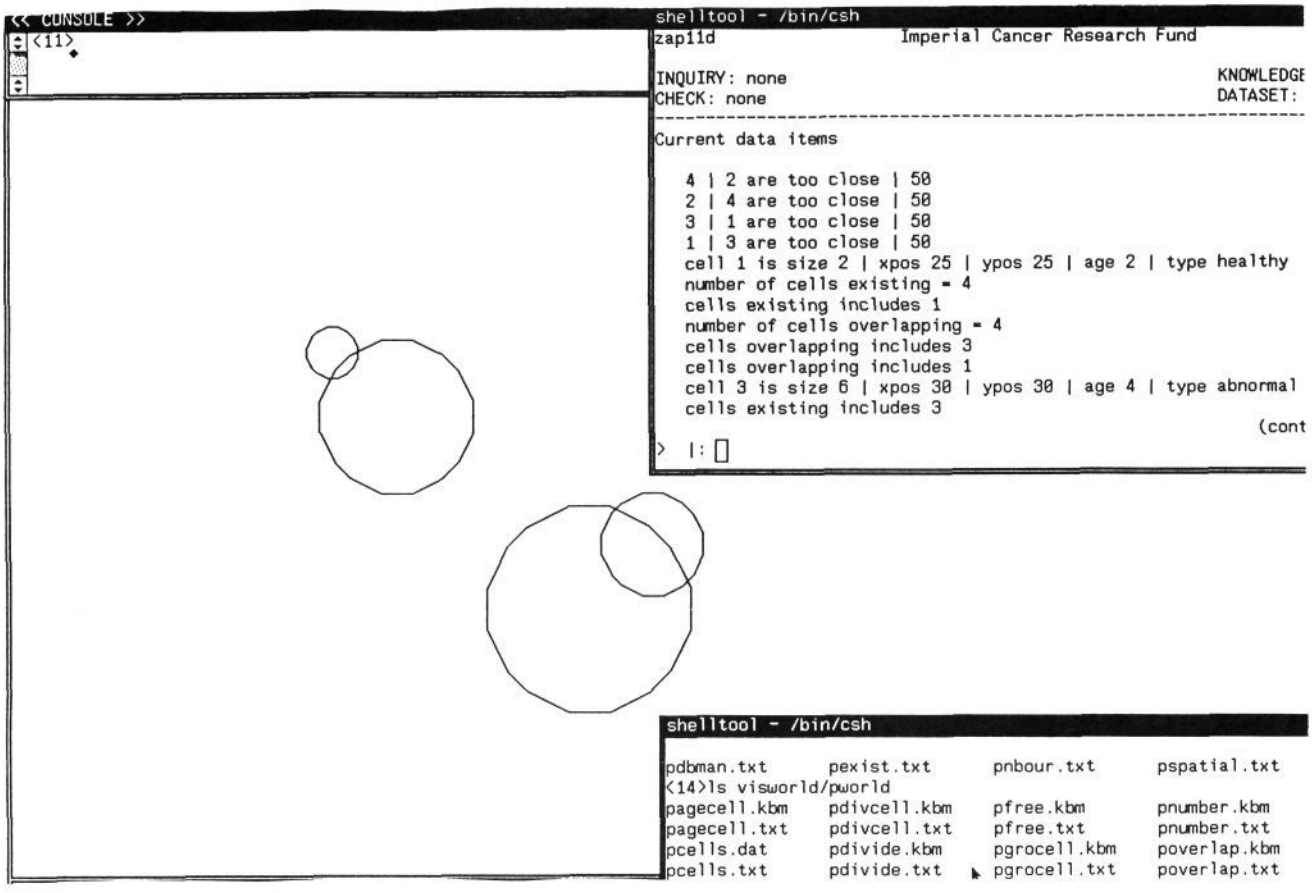

Figure 1: The CELLPROPS interface displaying the initial set of four cells used in the simulation.

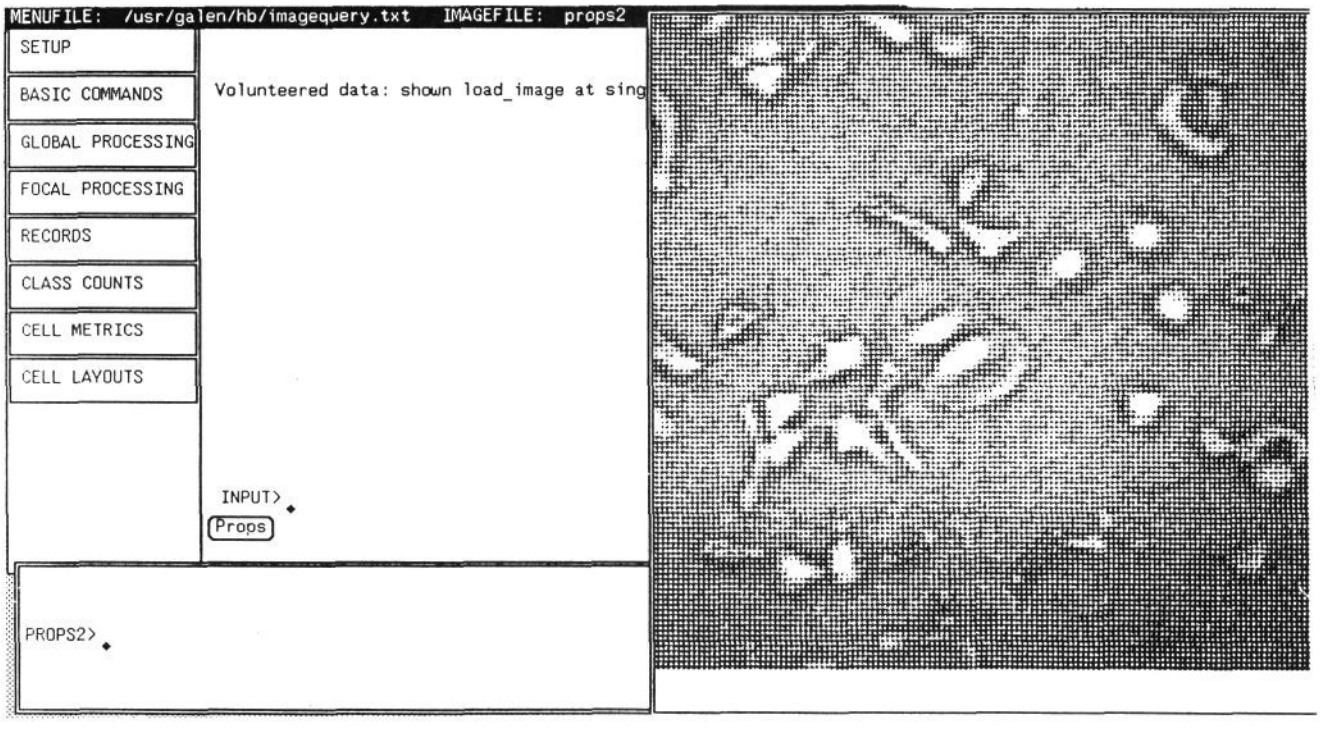

Figure 2: The IMAGEPROPS interface with the command and query menus, PROPS input and output, and image canvas.

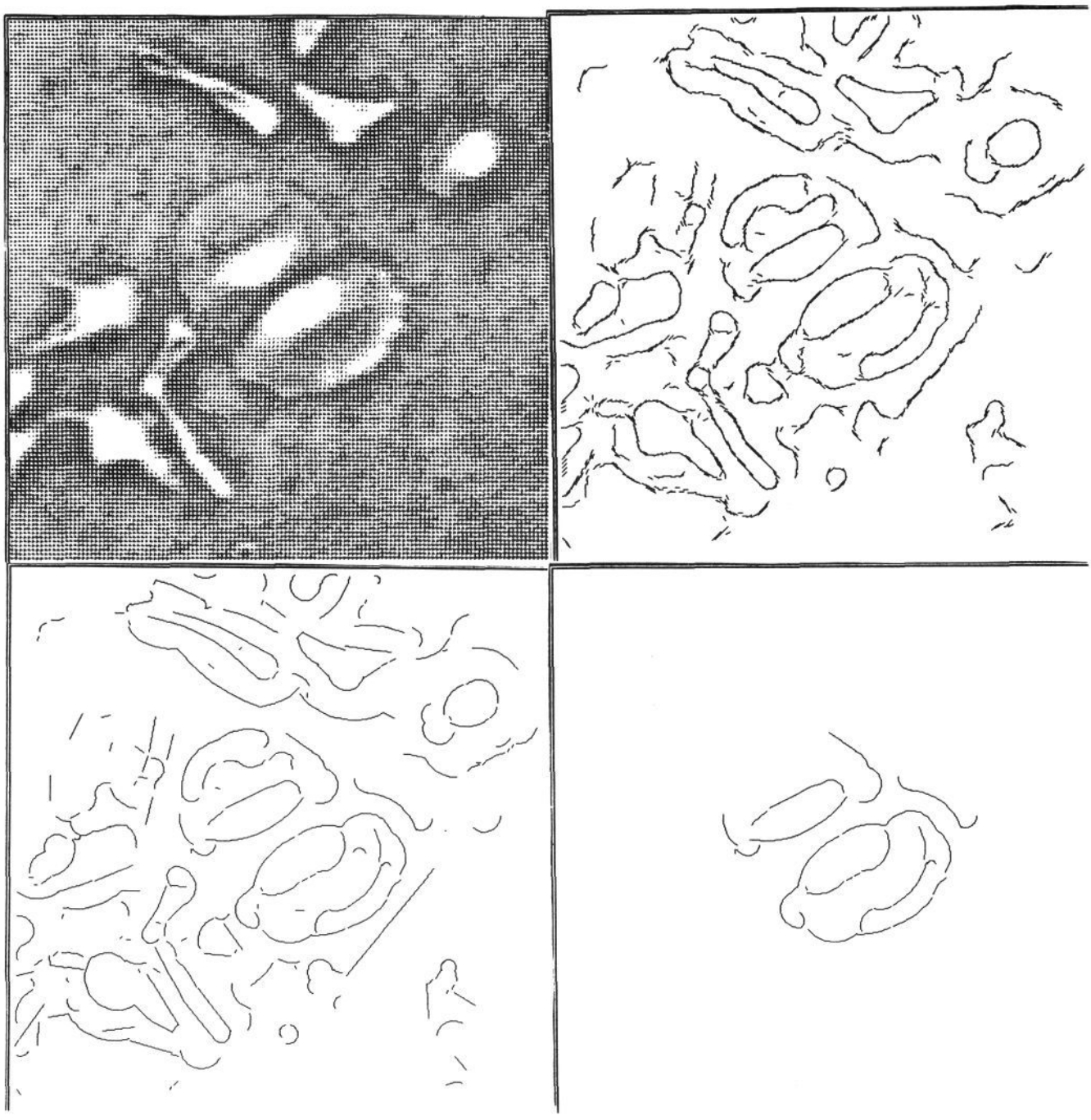

Figure 3: The raw image, edge segments, grouped primitives and dividing cell object.