N89-19878

A Representational Framework and User-Interface for an Image Understanding Workstation

Joyce D. Schroeder
Speech and Image Understanding Laboratory
Computer Science Center
Texas Instruments
P.O. Box 655474, M.S. 238, Dallas, TX 75265

Abstract

Problems in image understanding involve a wide variety of data (e.g., image arrays, edge maps, 3-D shape models) and processes or algorithms (e.g., convolution, feature extraction, rendering). This paper describes the user-interface and underlying structure of an Image Understanding Workstation designed to support multiple levels and types of representations for both data and processes.

The Image Understanding Workstation consists of two parts: the Image Understanding (IU) Framework, and the user-interface. The IU Framework is the set of data and process representations. It includes multiple levels of representation for data such as images (2-D), sketches (2-D), surfaces (2-1/2 D), and models (3-D). The representation scheme for processes characterizes their inputs, outputs, and parameters. Data and processes may reside on different classes of machines.

The user-interface to the IU Workstation gives the user convenient access for creating, manipulating, transforming, and displaying image data. The user-interface follows the structure of the IU Framework and gives the user control over multiple types of data and processes. Both the IU Framework and user-interface are implemented on a LISP machine.

By developing a fundamental set of representations and a consistent user-interface, the IU Workstation provides a rich environment for algorithm developers and system engineers to implement and study applications in image understanding.

INTRODUCTION

Problems in image understanding involve a wide variety of data (e.g., image arrays, edge maps, 3-D shape models) and processes or algorithms (e.g., convolution, feature extraction, rendering). A critical issue in image understanding (IU) is the representation of data. Data in IU work includes initial sensor-derived numeric arrays and more abstract, symbolic representations such as lists of edges. Appropriate representations must exist in an IU system to accomodate these different levels of information. The characteristic computations in IU work are also diverse, ranging from numerically intensive to symbolic. The goal of the Image Understanding Workstation described in this paper is to provide an integrated and highly interactive software environment to support such diverse IU activity.

In general, IU software environments are evolving into integrated systems of considerable computational and representational power [4]. Some recent systems illustrating these trends are: Visions [2], Powervision [5], and SuperSketch [6]. As described in the review article by McConnell and Lawton [4], the basic components of general IU environments are: representations, programming constructs, system-specific databases, and user-interfaces.

Several current IU environments are built on top of object-oriented programming environments that readily support data and process abstraction. Key attributes of object-oriented environments are: abstract type definition, combination and inheritance mechanisms, and access to diverse objects through a uniform set of messages [4].

The Image Understanding Workstation described in this paper consists of two parts: the Image Understanding (IU) Framework, and the user-interface.

IU FRAMEWORK

The IU Framework consists of the fundamental set of data and process representations in the IU Workstation. As shown in Figure 1, the IU Framework includes representations for IU data and a core set of IU algorithms. The IU Framework is intended to be a foundation for other program modules such as control strategies and application-specific environments.

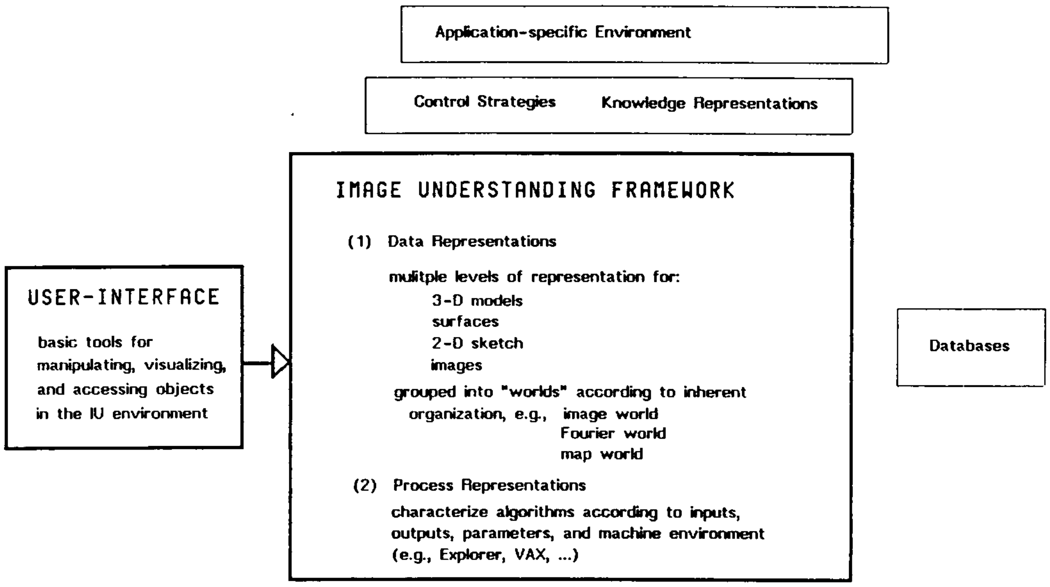

Application-specific Environment
Control Strategies
Knowledge Representations
IMAGE UNDERSTANDING FRAMEWORK
(1) Data Representations
mulitple levels of representation for:
3-D models
surfaces
2-D sketch
images
grouped into "worlds" according to inherent organization, e.g., image world
Fourier world
map world
(2) Process Representations
characterize algorithms according to inputs, outputs, parameters, and machine environment (e.g., Explorer, VAX, ...)
USER-INTERFACE
basic tools for manipulating, visualizing, and accessing objects in the IU environment
Databases

Figure 1.

(i) Data Representations

Problems in image understanding involve data at different levels of abstaction, ranging from numeric arrays created by a sensor, to extracted feature objects such as curve lists. The goal is to provide a range of data representations appropriate for each of these types of information.

Data objects share several kinds of information such as: data type, data, name, domain and range dimensionality, and axis information which completely describe the underlying data. In addition, data objects have an associated display-options attribute which specifies the default way to display the object in the IU Workstation.

Figure 2 illustrates two different types of data objects in the IU Workstation: the bridge image in the left window is a 2D-ARRAY, and the set of extracted contours in the right window is a 2D-INSTANCE-LIST. The underlying data for the extracted contours is a list of contour line objects (not a one-bit array). Each contour line object in the list is a data object characterized by: number, type, length, elevation, list of points, thickness, and color. Further data abstraction to lines of a minimum length is illustrated in Figure 3.

Data objects in the IU Workstation can be created using data from either LISP arrays, VAX files, or Odyssey [1] format files. Appropriate defaults are provided for axis information and display options.

Data objects in the IU Workstation are assigned to "worlds" according to the inherent organization of the data in the object. For example, an Image World has objects whose data is addressed by rows and columns in an array. Map World data is accessed by latitude, longitude, and elevation; Hough World data is accessed by radius and angle. The use of "worlds" classifies data objects by type, allowing processes and methods selective access to particular groups of data objects.

The relationship between data objects and process objects is shown in Figure 4.

(ii) Process Representations

Processes or algorithms are also implemented as objects in the IU Workstation. A process is characterized by its input data-object(s) and world, output data object(s) and world, parameters, and class of machine. Processes are invoked via an evaluate method that automatically saves a copy of the information about the process being executed: its inputs, outputs, and arguments.

Processes in the IU Workstation are not limited to the LISP environment. A Remote Processing Object has been implemented to forward commands and monitor responses from remote hosts. Currently, the Remote Processing Object uses the Telnet window protocol to communicate with VAX systems. This integrated approach allows numerically intensive operations to be sent to an appropriate machine for computation. The IU Workstation provides the entire interface for the remote process, collecting pathnames and arguments, then sending the instruction to the remote host. Remote computations can be monitored from the IU Workstation and output files

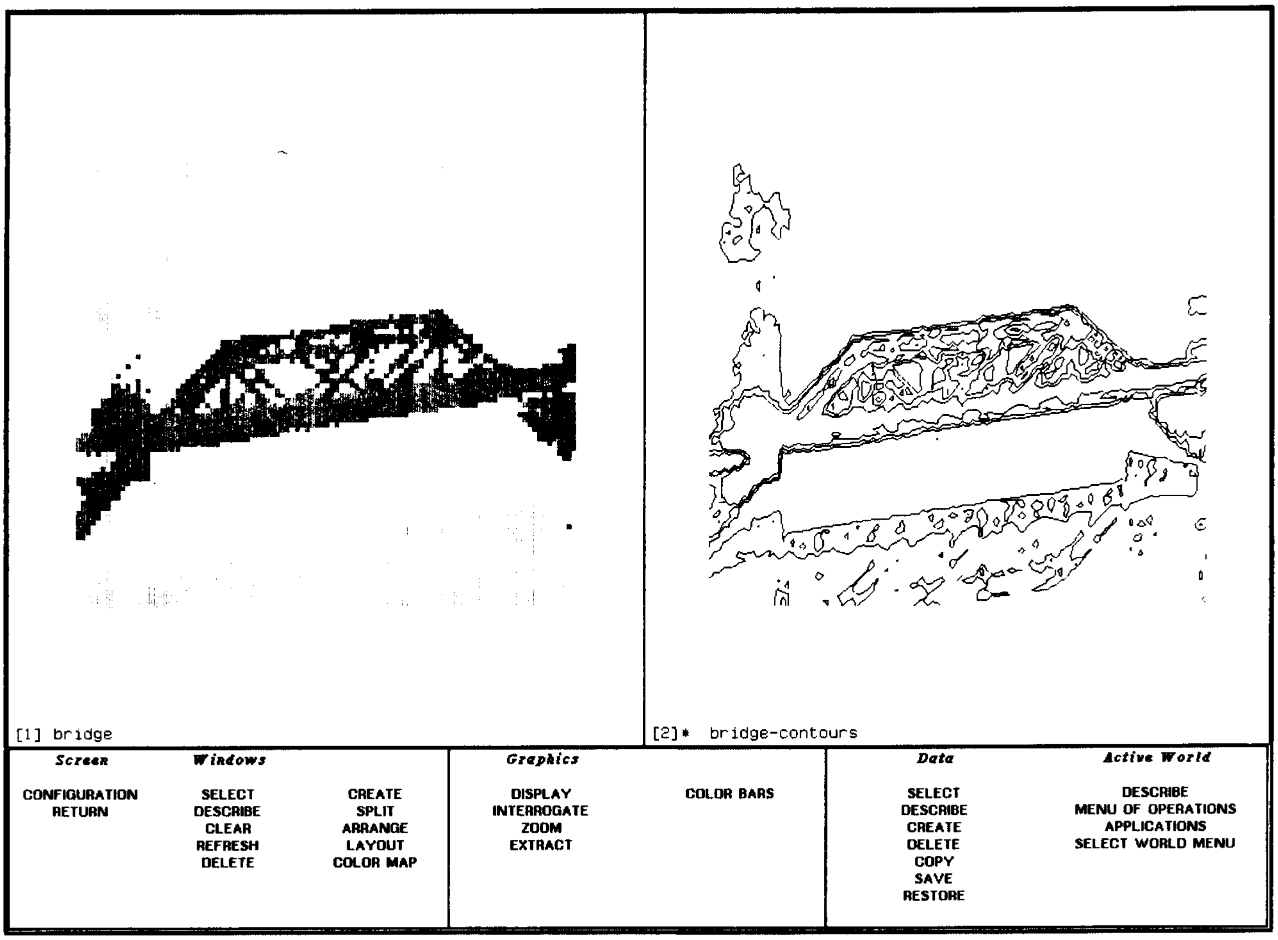


Figure 2.

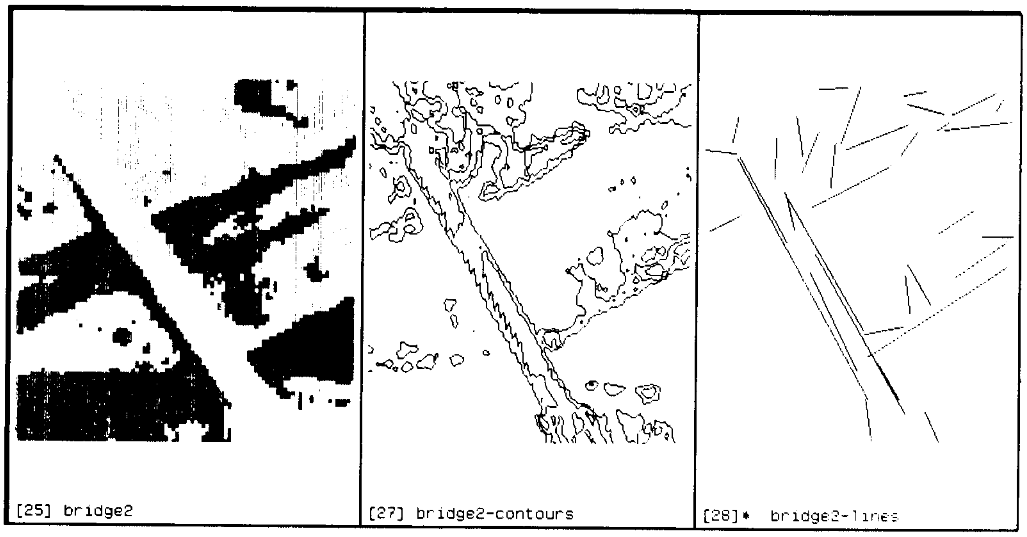


Figure 3.

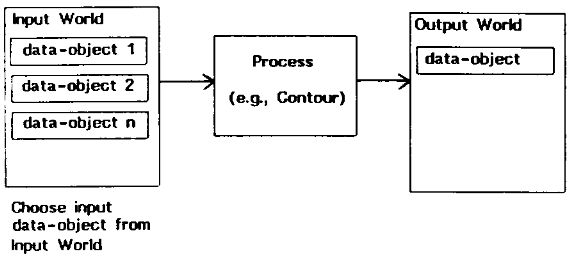


Figure 4.

can be automatically set up as data objects in the IU Workstation.

Figure 5 shows the results of an algorithm executed through the Remote Processing Object. The input image is shown in the upper left window. A VAX-based FORTRAN scene segmentation program was used to create the output segmented images shown in the upper right and lower left windows. A pop-up menu showing the parameters used in the Scene Segmentation process is also displayed.

Note that processes are also implemented as objects in the IU Workstation. As such, they can be treated as "data" in the sense that higher level control strategies or knowledge representations can be constructed to manipulate or reason about process objects. (See Figure 1.)

USER-INTERFACE

The IU Workstation user-interface is the set of tools for accessing, manipulating, and visualizing objects in the IU environment. The user-interface is window-oriented and gives the user a consistent set of commands for managing a complex set of image data and processes which reside either on the LISP machine or a remote machine.

The goal of the IU Workstation user-interface is to give the user simple and rapid communication with the IU envrionment through mouse-button selectable commands, menus with appropriate default settings, and pop-up description windows for all objects in the environment: data objects, process objects, and windows and associated displays.

The screen of the IU Workstation is comprised of several types of windows: command menus, display windows, and a Lisp typein window. The screen can be reconfigured with fewer command menus in order to maximize display space. The three bottom-level menus shown in Figure 5 contain the basic commands for window reconfiguration, graphics, and data manipulation.

IU Workstation commands are selectable by mouse, assigned keystroke, or typein, and extensive help and documentation are a part of each command definition. In general, commands in the IU Workstation are mouse-button specific. That is, left mouse clicks on a command give default behavior; middle or right mouse clicks give menus. Default behaviors (e.g., default parameters for displays, or default input and output worlds for processes) allow the user to rapidly specify simple tasks.

Pop-up command menus are used in the IU Workstation to hold less frequently used commands. Screen space is saved by grouping these commands into a pop-up menu that is accessed by an assigned keystroke. For example, commands related to the Remote-Processing-Object are handled in this manner.

The commands in the IU Workstation provide a common interface (or set of messages) to the different data objects in the IU environment. For example, the DISPLAY command uses appropriate display code according to whether or not the selected data object is a 2D-array, 2D-instance-list, or other data type. Similarly, a DESCRIBE command is applicable to all objects in the IU environment.

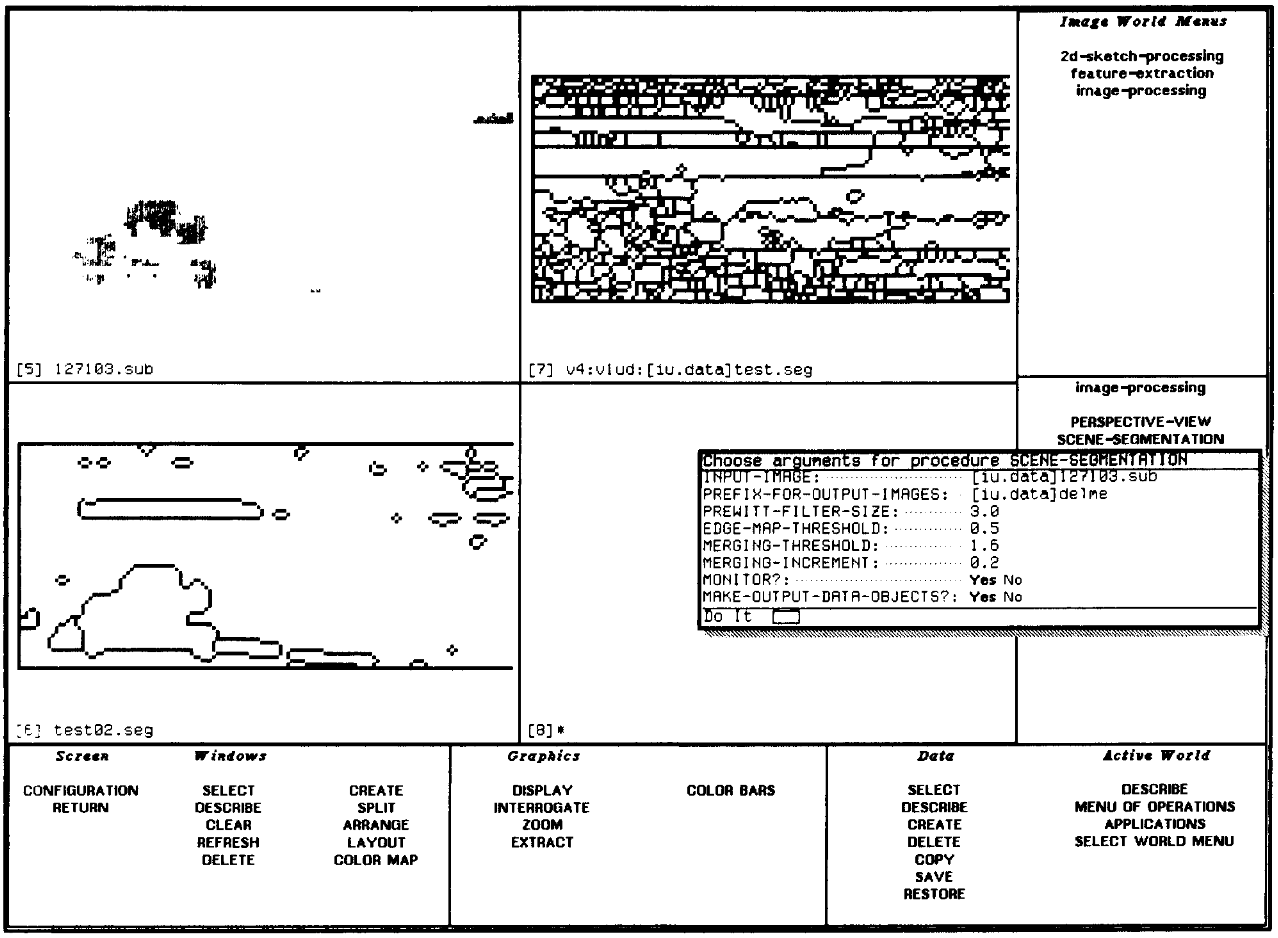
Another key feature of the IU Workstation user interface is the functionality included in the display windows. Display windows have an associated virtual window which describes the mapping between displayed data values and window coordinates. Thus, display windows support operations for zooming, copying, interrogating, and extracting portions of the displayed data using the mouse. The IU Workstation display windows maintain information about the data being displayed, and also how it is displayed (i.e., magnification, color map, annotation, etc.) Display capability exists for both color and black and white monitors. The IU Workstation is implemented on the Texas Instruments Explorer II Lisp Machine.

SUMMARY

The Image Understanding Workstation described in this paper is implemented in an object-oriented programming environment and includes multiple levels of representations for IU data. Processes or algorithms are also characterized and computations on different classes of machines are supported.

The user-interface to the IU Workstation is composed of basic tools for visualizing, manipulating, and accessing objects in the IU environment. The user-interface is highly interactive, window-oriented, and gives the user a consistent set of commands for managing the complexity inherent in IU envrionments.

ORIGINAL PAGE IS
OF POOR QUALITY



```
Choose arguments for procedure SCENE-SEGMENTATION
INPUT-IMAGE: ........................ [iu.data]127103.sub
PREFIX-FOR-OUTPUT-IMAGES:  [iu.data]delme
PREWITT-FILTER-SIZE: ........ 3.0
EDGE-MAP-THRESHOLD: ........... 0.5
MERGING-THRESHOLD: ............ 1.6
MERGING-INCREMENT: ............ 0.2
MONITOR?: ............................ [✓ Yes] No
MAKE-OUTPUT-DATA-OBJECTS?: [✓ Yes] No
```

Figure 5.

By developing a fundamental set of representations and a consistent user-interface, the IU Workstation provides a rich environment for algorithm developers and system engineers to implement and study applications in image understanding.

ACKNOWLEDGMENTS

The Image Understanding Workstation has been jointly developed by Bruce Flinchbaugh, Joyce Schroeder, Marion Lineberry, and Bob Gove. All are members of the Speech and Image Understanding Laboratory, in the Computer Science Center, Texas Instruments.

REFERENCES

1. Gove, R.J., "Integration of Symbolic and Multiple Digital Signal Processors with the Explorer/Odyssey for Image Processing and Understanding Applications", *Proceedings of the IEEE International Symposium on Circuits and Systems*, Philadelphia, Pennsylvania, May, 1987, pp.968-971.

2. Hanson, A. and Riseman, E., The VISIONS Image Understanding System- 1986, In Chris Brown, editor, *Advances In Computer Vision*, Erlbaum Press, 1987.

3. Kopec, G.E., "The Integrated Signal Processing System ISP", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-32, No. 4, August, 1984, pp. 842-851.

4. McConnell, C.C., and Lawton, D.T., "IU Software Environments", *Proceedings: Image Understanding Workshop*, Vol. II, Cambridge, Massachusetts, April, 1988, pp.666-677.

5. McConnell, C.C., Nelson, P.C., and Lawton, D.T., "Constructs for Cooperative Image Understanding Environments", *Proceedings: Image Understanding Workshop*, Vol. II, Los Angeles, California, February, 1987, pp. 497-506.

6. Pentland, A.P., "Perceptual Organization and the Representation of Natural Form", *Artificial Intelligence 28*, 1986, pp. 293-331.