The Image Understanding Environment Program

J. Mundy*
GE Corporate Research and Development
Schenectady NY 94403

T. Binford
Stanford University
Stanford CA 94305

T. Boult
Columbia Univ.
NYC NY 10027

A. Hanson and R. Beveridge
Univ. of Mass.
Amherst MA 01003

R. Haralick and V. Ramesh
University of Washington
Seattle WA 98196

C. Kohl
Amerinex
Artificial Intelligence, Inc.
Amherst MA 01002

D. Lawton
Georgia Institute of Technology
Atlanta GA 30332

D. Morgan
Advanced Decision Systems
Mountain View CA 94043

K. Price
University of Southern California
Los Angeles CA 90089

T. Strat
SRI International
Menlo Park CA 94025

Abstract
The Image Understanding Environment (IUE) project is a five year program, sponsored by DARPA, to develop a common software environment for the development of algorithms and application systems. This paper reviews the history of the project and provides an overview of some data structures which are currently evolving as a specification for the IUE. The ultimate goal of the project is to provide the basic data structures and algorithms which are required to carry out state of the art research in image understanding.

1 Introduction
1.1 Scope and Mission
The primary purpose of the Image Understanding Environment (IUE) is to facilitate exchange of research results within the IU community. The IUE will provide a platform for various demonstrations and tools for DARPA applications. These demonstrations and tools will become a primary channel for IU technology transfer. The IUE will also serve as a conceptual standard for IU data models and algorithms. The availability of standard implementations for basic IU algorithms will facilitate performance evaluation of new techniques and to track progress in algorithm improvements. The IUE is designed to support significant evolution of IU approaches and an effective programming environment for rapid prototyping.

The IUE is not intended to be a real time system although tools will be provided for the simulation of real time applications such as navigation. The IUE will not support special hardware accelerators but a standard image processing interface will be provided. There is no intention to generate a design suitable for embedding in larger systems, although object class components can certainly be used in the construction of new systems.

The goal of the IUE is to provide the following benefits.

Research Productivity

Many IU labs are constantly reimplementing code, such as the Canny edge detector, as students leave or as new labs are formed. The IUE will provide a standard interface for the development and sharing of such code. In addition, standard libraries of mature IU code will be distributed with the IUE so that the code can be easily shared and incorporated in new research projects.

*The committee effort has been funded by various DARPA grants.
Technology Transfer

The IUE will provide a platform for demonstrating the benefits of IU algorithms in the context of various application models. The user models to be supported by the IUE are summarized in the next section. It should be emphasized that it is not expected that the IUE will provide the full performance required for such applications, but the IUE will provide the necessary capability to support feasibility demonstrations in a realistic application context. Particular attention is given to support for the types of algorithms involved in photo-interpretation, smart weapons, cartography, visual navigation and industrial vision.

Education/Development

The IUE data structures and operations will provide a basis for a formal description of IU algorithms. The IUE will provide consistent and standardized format for encoding IU algorithms which is suitable for teaching and as a base for further development of IU technology. It is envisioned that, eventually, new IU papers will include an IUE description of the algorithm which can be directly implemented in terms of the basic IUE data structures and functions.

Computational Model

It is also expected that the IUE will serve as a computational model to the development of IU hardware architectures. The IUE specification will provide the basic data structures and associated operations and any IU architecture must support. From this standpoint, the IUE may lead to a useful language interface for IU architectures.

1.2 The Nature of IU Research Software

Image understanding research is typically carried out by individual contributors who develop a specialized software environment for implementing and evaluating new approaches and concepts. In most cases, the software environment is thrown away as new approaches are considered or the researcher moves to a new research problem or new hardware platforms are introduced. A clear example of the latter case is the decline of the Symbolics Lisp Machine as a platform for rapid prototyping of IU algorithms.

This volatility of research software is not particular to image understanding research, but is perhaps characteristic of software systems in general. These systems evolve rapidly with time as new languages, programming techniques and hardware platforms emerge. However, this volatility and diversity places a heavy toll on the efficiency of the image understanding research community. It is difficult to share and evaluate new research ideas because they can be demonstrated only in the specialized environments generated by the originator of the idea. A steep learning curve is encountered in any attempt to acquire and operate these specialized and often fragile environments.

Another major source of inefficiency is that most IU experiments require an extensive set of rather standard algorithms and data structures to reach the level of processing and feature extraction upon which the new ideas can be tried. Typical examples are image smoothing, edge detection, curve fitting, feature grouping and camera calibration. Since each environment is somewhat different in its design of the basic data structures, it is necessary for the algorithm developer to reimplement the basic IU infrastructure in order to evaluate a complex algorithm. This process is repeated over and over at dozens of IU research laboratories each year as new research projects start.

Finally, it is now widely recognized that significant applications of IU research can only be realized in the context of large systems. Examples of such applications are the UGV (Unmanned Guided Vehicle) project which will exploit image understanding for navigation and surveillance tasks, and the RADIUS (Research and Development for Image Understanding) project which is focused on the application of Image Understanding to photointerpretation. These extensive application projects cannot be realized without a common software environment to provide the integration of diverse system components, developed at different research institutions.

1.3 Initiation of the IUE

In late 1989, Rand Waltzman of DARPA, then manager for Image Understanding programs conceived and developed a new program called IUES. The decoding of this acronym is Intelligent Integrated Interactive Image Understanding. The name has since been shortened to IUE, for Image Understanding Environment.

The IUE program was announced at a meeting for DARPA Principal Investigators in Scottsdale, Arizona at the end of February, 1990. The project goal, as announced by Rand, was a five year program to design and implement a common software environment for the development and demonstration of image understanding algorithms and techniques.

At the time of the meeting, there were three environments which had reached a reasonable maturity and had attracted a sufficient number of users to demonstrate that the idea of an IUE was feasible. These three systems and general characteristics are summarized below.

- The Cartographic Modeling Environment (CME) has been developed by Lynn Quam of SRI International over the past decade [1]. The focus of CME is the efficient handling of large images and the representation of configurations of 3D object models on the earth’s surface under perspective viewing, in support of site modeling for cartography and photoreconnaissance. CME is implemented in Lisp on the Symbolics Lisp Machine. CME is currently playing a central role in the RADIUS program as a proposed development environment for RADIUS experiments. This development environment is called RCDE, or RADIUS Common Development Environment.
CME is currently being ported to the SUN-UNIX platform under X-Windows and Common Lisp.

- KB Vision (Knowledge-Based Vision) is a product of Amerinex Artificial Intelligence Inc., and is loosely based on the VISIONS system developed at the University of Massachusetts at Amherst over the last fifteen years [3]. KB Vision combines Lisp and C components. C is used primarily for numerical and image feature processing tasks while Lisp is for high level reasoning about image content. KB Vision is widely distributed among users of image understanding technology and provides an effective interface for developing image feature segmentation and feature grouping algorithms. KB vision is currently being extended to provide the programming environment for the Image Understanding Architecture or IUA, a highly parallel, multi-granularity design.

- Power Vision was developed by Advanced Decision Systems (ADS), now a division of Booz, Allen and Hamilton, starting about 1986 [2]. Power Vision provides an object-oriented programming environment in Symbolics Flavors for a wide range of Image Understanding data structures. An effective user interface has been developed for displaying graphical interpretations of relationships between image features. Power Vision was used by ADS in many of their application studies, including their work on the Autonomous Land Vehicle Project (ALV).

These systems provided the conceptual basis for the IUE, but the scope of the IUE project is much broader than any of these existing systems. The goal is to provide an environment which can cover the full spectrum of IU research and support both C and Lisp application development.

To initiate the project, Rand Waltzman convened and chaired three meetings during the 1990-1991 period to develop an consensus in the IU community about the requirements for the IUE. A number of teams were established to suggest specific application scenarios and propose skeleton architectures for the IUE. In April 1991, these team reports were reviewed, and the IUE committee was formed from representatives of each team. Since April 1991, the IUE committee has met 7 times and has produced a requirements and design specification for the IUE. The design consists of over a hundred classes at the time of this writing and over 400 pages are required to document the classes! The extensive nature of the design illustrates one facet of the complex nature of the Image Understanding problem.

2 The IU Problem Domain

The general Image Understanding problem domain is illustrated in Figure 1. The figure shows the major aspects of current research in Image Understanding. The interaction of light with surfaces is illustrated by the variation in intensity and shadows associated with the cube. Numerous investigations are being carried out to develop theories of the relationship between intensity and surface geometry and surface reflectance properties, as well as the projection shadows.

The figure also shows the possibility of relative motion between the sensor and the scene objects. Again, there is considerable interest in the IU community in the problem of determining object structure and motion from image time sequences.

Finally, the figure shows the geometry of perspective viewing where a 3D object is projected onto a 2D image plane by a central projection. This geometry accurately models the image formation process by a pinhole camera and is a reasonable model for standard cameras.

Much of the research in IU has centered on developing mathematical models for these processes and then testing the applicability of such models on image data. The major issue for the IUE is to provide adequate structure and function to support research in these areas. IU research to date has involved quite sophisticated and diverse mathematical techniques which often require complex data structures and algorithms. It is quite challenging to efficiently accommodate such a diverse spectrum of concepts.

3 The Abstract Basis for the IUE

A major insight achieved by the IUE committee is the concept of the spatial object. In experimenting with different class hierarchies in the IUE, it was observed that certain spatial attributes and operations that occur over and over again. The concept of the spatial object is an ongoing effort to abstract these properties into a compact set of generic classes.

For example, expressing neighborhood operations over a discrete curve should be similar to expressing neighborhood operations with respect to an image (the basic difference is dimension). Similarly for operations on representations such as multi-resolution pyramids, spatial indexing, and generalizations of grouping operations. The key idea is abstraction to increase ease of use and learning and also to reflect powerful generalizations.

Spatial objects intuitively correspond to familiar geometric concepts such as Point, Curve, Surface, Volume. In addition the result of segmentation operations on an image produces geometric features which may also be interpreted as a spatial object. Indeed, it can be argued that much of the machinery of Image Understanding algorithms can be cast in terms of the interaction of spatial object with image data. In addition, it is often reasonable to think of an image as a type of surface and, as such, it inherits many geometrically oriented image processing methods from a general 2 dimensional spatial object.

The following sections review some of the concepts currently in the IUE design.
Figure 1: The major elements of Image Understanding research.

Figure 2: Several of the image data types supported by the IUE.
4 Image Structures

The IUE image object class supports many forms of image data, from intensity images to color images to complex composites such as pyramids. Some of the most important image structures are shown in Figure 2. IUE images fall into one of two subclasses: simple or composite. A simple IUE image is a specialization of an IUE shared-array. The semantics of simple-images are fairly well defined. In particular simple images have two primary dimensions, x and y, with possible additional dimensions such as color and/or time. In contrast, the class of composite-images is very broad and the semantics vary considerably between specializations. Composite-images provide the flexibility required to develop objects such as image pyramids and mosaics.

A composite-image, as the name implies, is composed of other images. In addition to whatever additional structure might be imposed through specialization, a composite-image may be manipulated as an ordered set of images. Often the constituent images will all be simple-images, but this need not be the case. The composite-image provides a general mechanism for composing both simple-images and composite-images. Hence multiple levels of composition are possible.

Stereo Image

An image produced by two black-and-white cameras calibrated for stereo.

Image Pyramid

An Image-Pyramid is an ordered set of images, each a power of two reduction of the predecessor. This makes several restrictions evident. The first image must be square, the dimension being a power of 2, i.e., $2^n$. The depth is of the pyramid, or number of images, is simply $n + 1$. The term ‘level’ applied to a pyramid means the following, the image at level n is $2^n$ pixels on a side.

Image Sequence

An image sequence can be considered as a three dimensional image where the third dimension is time. A more general and flexible representation is provided by the composite-sequence. A composite-sequence is intended for sets of images that fall into a natural sequence. The composite-sequence has both advantages and disadvantages when compared to the simple-image-sequence. One advantage is that images may be inserted and removed from a composite-sequence. Hence a composite-sequence might be the more logical object with which to maintain a set of the k most recently acquired images. A disadvantage is that because the images are not stored together in a single multidimensional array it is not possible to define slices across images in the sequence.

RGB Color Image

A color image with distinct red, green and blue color data. Can be represented as a vector image where each pixel is a three-element vector or as an image sequence where there is a separate image for red, green and blue.

Mosaic

A Mosaic-Image is a patchwork of images, some overlapping, and partially covering an extended 2D area. The proper metaphor is a stack of photographs on a table. Hence, the get-pixel method returns the pixel value for the first image in the set which is defined at the specified point.

Range Image

Specifically for a depth image acquired with a laser triangulation or time-of-flight range finder.

5 The Spatial Object

The spatial object is a central concept in the IUE. Essentially, a spatial object is a point set in n-dimensions. The point set has an associated coordinate frame so that projections and transformations can be applied to the point set. It is not essential that the point set be a simple flat set, but can be represented as a hierarchical group or other relationship among groups of point sets. A polyhedral object represents a complex spatial object which represents a set of points in 3D space.

5.1 Definition

The attributes associated with a spatial object are:

1. The dimension of the point set, e.g., 3D points.
2. A domain coordinate system
3. A bounding box or sphere
4. Centroid
5. Attribute/relationship property list (values of texture measures)

5.2 Operations

Some of the operations associated with a spatial object are illustrated in Figure 3. The general types of methods associated with a spatial object are:

- **Boolean Operations**
  - Intersect
  - Union
  - Difference
  - Inside
  - OnBoundary
- **Distance to Point**
- **Surface Normal at Point**
- **Transform to Coordinate Frame**
- **Sample**
  - Fit Analytic Form
  - Embed
  - Compose
Figure 3. Typical operations associated with a spatial object.

The method *Embed* simply means the association of a particular coordinate space with a spatial object. For example, a planar curve can be embedded in 3D space by extending the coordinate space of the domain of the curve.

The last method, *Compose*, is very powerful and would provide considerable flexibility in manipulating spatial objects. The concept applies to spatial objects which are defined parametrically. For example, consider a parametric curve such as a planar cubic curve defined by

\[ x = a_xt^3 + b_xt^2 + c_xt + d_x \]
\[ y = a_yt^3 + b_yt^2 + c_yt + d_y \]

The curve can be represented in vector notation as \( \mathbf{x}(t) \), where \( \mathbf{x} = (x, y) \). Next consider a surface in 3D space, also defined parametrically, \( \mathbf{X}(u, v) \), where \( \mathbf{X} = (x, y, z) \). The two spatial objects can be composed \( \mathbf{X}(u(t), v(t)) \) so that the planar cubic curve becomes a 3D space curve, lying on the surface. A particularly common case of composition in image understanding is to use a curve to sample points in an image to return to the image along the curve, such as intensity or image gradient.

3 **Transformations and Coordinate Systems**

The geometric relationship between sensors and scenes, among physical objects, and between pixels and the world has been a core component of the science of image understanding since its inception. Nearly every IU system makes use of coordinate systems and transforms either implicitly or explicitly. The multitude of representations that have been devised, some of which incorporate arbitrary conventions, has been a key obstacle precluding the transfer and sharing of code and results. We begin with definitions of our terminology:

**Coordinate System:** A coordinate space, in the mathematical sense. It is represented in the IU environment by an instance of a coordinate system class.

**Coordinate:** The coordinate(s) of a point are represented by a series of numbers, and are implicitly associated with a coordinate system.

**Coordinate Transform:** A specification of a mapping between two coordinate spaces. It is represented in the IU environment by an instance of a coordinate transform class.

Conceptually, the relation between coordinate systems and coordinate transforms can be expressed by a directed graph in which a coordinate system is represented by a node and a coordinate transform is represented by a directed arc between two nodes. Both coordinate systems and coordinate transforms are represented by instances of object classes. The classes of transforms that can relate two coordinate systems is governed by the classes of those coordinate systems.

A coordinate transform specifies the mapping between two coordinate systems, and is represented in
the IUE by an instance of a coordinate transform class. There is an implied directionality to each transform, and any individual transform may or may not be invertible.

The relationships among coordinate systems and transforms form a graph. Coordinates can be mapped between any two coordinate systems by finding a sequence of transforms that connect them in the graph. There is a potential problem when more than one path exists between two coordinate systems. Ideally, all such paths specify the same transform, but the existence of numerical errors, and the need for high-performance approximations preclude the treatment of all such paths as equivalent. In IUE, the first path established between two coordinate systems is treated specially – all others are considered to be derived transforms. Whenever a transform is requested from the coordinate transform graph, the non-derived transform is retrieved. This policy ensures that coordinate transforms are performed consistently and without introduction of excessive numerical error (unless specifically directed otherwise). It is anticipated that coordinate systems and transforms will be used to relate all geometric objects within the IUE.

5.4 Segmentation

Figure 5 illustrates the extraction of geometric features from image data. The discrete nature of the image pixel tessellation introduces the concept of discrete spatial objects. We still consider that there is an underlying continuous space and each discrete sample is associated with a neighborhood (e.g., square pixel region) so that topological properties can be represented and constructed. In this case the spatial object is initially defined by a discrete set of points. An example would be a mapping from an ordered set of integers onto discrete image positions which corresponds to a curve extracted from an image. Discrete objects can be formed from analytical ones by sampling methods, such as forming a circular region by determining a regular grid of points which are contained in it. Similarly, continuous spatial objects can be formed from discrete representations by curve and surface fitting procedures. An example of obtaining line segments from image pixel chains is also illustrated in Figure 5.

Digital spatial objects can be indexed or unordered. In an indexed digital spatial object the object is stored as an array with dimensional indices for accessing values. A bitmap is an example of an indexed spatial object which is defined by spatial occupancy in a minimum-bounding, hyper-rectangular prism (in 1D, 2D, 3D, nD arrays). A discrete unordered spatial object contains a list of the discrete positions in the object. An example would be a volume described by the points which are contained in it.

5.5 The Spatial Index

Any spatial object can be used to perform a spatial query into another spatial object. The query is performed by embedding one spatial object in another spatial object which provides pointers to objects and then checking the attributes of these objects. Several examples are illustrated in Figure 6. A widely used spatial index is the Hough transform where the orientation and position of line fragments are sorted into a Hough array where individual cells correspond to distinct line equations. In this way, line fragments belonging to the same line will be grouped together.

A standard n-D array may also be considered as a spatial index where points occupying similar positions in space are collected into a single cell. The octree is a typical example.

More sophisticated data structures, such as the R-tree, provide compact and efficient retrieval of spatial adjacency.

5.6 The Perceptual Group

Perceptual Groups are described as spatial objects and their processing involves general operations supported by the IUE: spatial indexing, dynamic association of attributes. The shape description associated with a group is often an existing spatial object, such as a circle or a line segment. It is significant that we don’t need to have new objects for describing groups. Several examples of perceptual groups are shown in Figure 7. Groups can also be represented as composite spatial objects based upon different types of relational networks. Composite spatial objects are a set of spatial object related by a different types of networks. Some of the general networks used with spatial objects are:

- Topological: The description in terms of point, edge, chain, loop, face.
- Coordinate System: Transform Network between spatial objects. Developed in the section on Coordinate Systems.
- Adjacency Specialization: Non-topological information associated with adjacency relations between spatial networks. Junctions between curve segments which may include such information as angle and blending criteria; similarly for relations between parametric parametric surface patches.
- Processing Relationship Network. This probably comes from the active object database. It may involve extracting some portion of this network into a composite spatial object.
- Temporal Relation Networks.
- Image Projection Network. This describes the relation between image features to world objects from which an image was formed.
- Networks describing matches between spatial objects.
- Spatial Relations (parallel, aligned, perpendicular, within-distance) [user specifies the type of relations and objects and the network is instantiated].
Figure 4: Segmentation - the extraction of spatial objects from image data.

Figure 5: Several examples of spatial indexing.
In the IUE we take the strategy that networks are implemented through class inheritance. That is, a class becomes part of a network by additionally inheriting from a network node class. By inheriting more network node types, an entity becomes part of more networks. By this strategy, if an entity is not part of a network, there is no overhead incurred.

6 Sensors

The concept of sensor involves two quite different descriptions. The sensing device and the data produced by the device. The spatial object plays a central role in the description of both aspects of a sensor. In addition, new concepts are needed to provide the description of signals, point spread functions and probability densities.

We have proposed an implementation that has two main classes, sensor and sensor-datum. In addition, there are a number of supporting classes including energy-transfer-system (e.g. lenses), energy-thing and filter (which includes digitizers). Generally a sensor is composed of a number of simpler sensor devices, for example in the case of a color camera or a range sensor. The need for a general notion of sensor data is easily seen in the case of stereo cameras. It is reasonable to assume that a stereo camera could return as complex as a polyhedral model or at least a hierarchical group of 3D line segments.

Also included in the sensor area is the idea of producing synthetic scenes. Given a description of the world as a configuration of spatial objects, it is often desirable to be able to render the scene and produce the set of sensor data objects that would be ideally produced by the sensor. This capability is essential for testing and evaluation of IUE algorithms.

Examples of sensors under design in the IUE are illustrated in Figure 8.

7 IUE Architecture

It is planned to implement the IUE in a number of stages according to a rapid prototyping strategy. Figure 9 shows the three layers of the IUE software design. The innermost circle represents the basic IUE classes which support primitive operations of the IUE such as user interface interactions, object persistence and specialized I/O, such as image buffering. The next layer provides the core IUE which is the standard representation for the range of IUE concepts just outlined. It is envisioned that this layer will provide a standard for data exchange within the IUE community. Finally, the outer layer which we call libraries constitute the standard repertoire of IUE algorithms. These algorithms normally make use of many objects classes from the core IUE.

It is hoped that the libraries will be largely implemented by the IUE community itself and that a critical mass will be achieved to support the long term maintenance and enhancement of the IUE as a shared resource.
Figure 7: Various sensors of interest to the IU community.

References


Figure 8: The major partitions of the IUE.