Automated Generation of Intent-Based 3D Illustrations

Dorée Duncan Seligmann
Steven Feiner

Department of Computer Science
Columbia University
New York, New York 10027

Abstract

This paper describes an automated intent-based approach to illustration. An *illustration* is a picture that is designed to fulfill a *communicative intent* such as showing the location of an object or showing how an object is manipulated. An illustration is generated by implementing a set of stylistic decisions, ranging from determining the way in which an individual object is lit, to deciding the general composition of the illustration. The design of an illustration is treated as a goal-driven process within a system of constraints. The goal is to achieve communicative intent; the constraints are the illustrative techniques an illustrator can apply.

We have developed IBIS (Intent-Based Illustration System), a system that puts these ideas into practice. IBIS designs illustrations using a generate-and-test approach, relying upon a rule-based system of methods and evaluators. Methods are rules that specify how to accomplish visual effects, while evaluators are rules that specify how to determine how well a visual effect is accomplished in an illustration. Examples of illustrations designed by IBIS are included.

CR Categories and Subject Descriptors: I.3.3[**Computer Graphics**]: Picture/Image Generation–display algorithms, viewing algorithms; I.3.4[**Computer Graphics**]: Graphics Utilities–Picture description languages; I.3.7[**Computer Graphics**]: Three-dimensional graphics and realism; I.2.1[**Artificial Intelligence**]: Applications and Expert Systems.

Additional Keywords and Phrases: illustrations, automated picture generation, knowledge-based graphics, non-photorealistic rendering.

Introduction

The development over the last few centuries of printing and photographic technologies, and more recently of electronic mass media, has revolutionized communication by making the *exact same presentation* accessible to larger and larger groups of people. Nevertheless, communication involves both intent and interpretation. The same presentation, viewed by several people, may be interpreted to mean different things, while different presentations may be interpreted to mean the same thing. To further complicate matters, none of these interpretations may be the one intended by the presenter. With recent advances in computer technology, we may now embark upon a new phase of communication. By formalizing the intent of a communication, the language or medium to be used, the audience and context of the communication, and the way in which the language is used to achieve intent, we may create systems that generate presentations, each designed to satisfy the same communicative intent for a particular audience, thus making the *exact same meaning* accessible to many different people.

This paper describes the first steps in developing such a system for illustration. An *illustration* is a picture that has been designed to fulfill a communicative *intent*. For example, the intent of an illustration may be to show an object's material, size, or orientation. The intent might be more complex. It may, for example, be more important to show how to turn a dial, and less important to show where the dial is located.

Human illustrators plan and replan an illustration, considering at all times how the final illustration will look. An illustrator may try something on paper and then, after evaluating it, erase it and adopt another plan. Or, the illustrator may be so skilled that it is enough for her to simply imagine the consequences of a stylistic choice.

This characterization of illustration serves as the foundation for intent-based illustration. An *intent-based* illustration system designs illustrations to fulfill a high-level description of the communicative intent. The illustration process can be formalized as a goal-driven process: the goal is to achieve a specified communicative intent within a complex of stylistic choices. In order to use a generate-and-test approach, such a system must represent style in two ways. First, each stylistic choice represents a method for achieving a particular goal. For example, in order to highlight an object, it may be brightened or it may be colored in a special way. Second, each stylistic choice is associated with a set of criteria used to judge how well it has been accomplished.

This paper describes IBIS (Intent-Based Illustration System), concentrating on its rule base, architecture, and design process. It explains how IBIS both achieves and evaluates the highlighting, recognizability and visibility of objects using several examples.

IBIS

Overview

IBIS utilizes a generate-and-test approach to illustration design. Starting with a description of the communicative intent and a knowledge base representing the world to be depicted, IBIS begins to design an illustration. The communicative intent is specified using a language of *communicative goals*. For example, a communicative goal may be to show how an object has been moved or to show its color. For each communicative goal there exists at least one design rule in IBIS's rule base. A *design rule* specifies a prioritized set of style strategies. A *style strategy* specifies a visual effect, such as highlighting and is achieved by a set of style rules. A *style rule* determines some part of the traditional computer graphics specification of an image: a viewing specification, a lighting specification, the objects to be depicted, and rendering instructions. A style rule calls upon procedures that directly access and manipulate illustrations. Illustrators select style strategies by selecting design rules to accomplish communicative goals. Drafters select illustration methods by selecting style rules to accomplish style strategies.

The following subsections describe all the components of the system and their interaction.

Input: Communicative Goals

IBIS currently supports communicative goals that have been designed to satisfy the needs imposed by COMET, a knowledge-based multi-media explanation generation system for which IBIS generates graphics [Elhadad et al. 89, Feiner and McKeown 90a, Feiner and McKeown 90b]. COMET designs explanations for equipment maintenance and repair that include pictures and text. Its current domain is the army radio shown in the figures in this paper. The communicative goals that IBIS can satisfy are:

- *location*: show the location of an object in a context (either explicitly specified or derived by the system)
- *relative location*: show the relative location of two or more objects in terms of a specified or derived context
- *property*: show one of the following physical properties of an object: material, color, size, shape
- *state*: show an object's state
- *change*: show the difference between a set of states

Both the goals *state* and *change* may be further qualified by concepts that refer to how the object is manipulated or has changed. For example, the state of a dial can be shown in terms of an agent turning it. IBIS currently supports three dozen concepts useful to our maintenance and repair domain, among them, *pushing*, *pulling*, *loosening*, *lifting*, *inserting* and *blinking*.

In response to a user request for information, COMET's *content planner* generates a description of the communicative intent for an explanation that is sent to COMET's *media-coordinator*. The media-coordinator annotates the intent specification to indicate which generators should communicate which information and passes the same intent specification to COMET's *media generators*. All generators work from the same annotated intent specification [Elhadad et al. 89]. IBIS translates the intent description into a prioritized list of *communicative goals*. This translation is more or less direct; concepts such as *location* and *turn* are identified in the intent specification. IBIS associates with each goal an indication of its importance, which in turn is used to calculate an acceptable degree of success when the goals are evaluated.

Knowledge Base

IBIS has a knowledge base of the physical objects to be illustrated that includes not only geometric and material information, but also information about the object's features, physical properties, and abstract properties. Information about the features and abstract properties of physical objects is a superset of the information traditionally passed to a graphics system. It is, however, necessary to an intent-based system that designs its own pictures. For example, it may be important to represent how an object moves or its limits of articulation. IBIS currently utilizes a very simple model for object states. For example, the dials on the radio are represented as having discrete or continuous ranges with associated orientations; the latches have two states: snapped and unsnapped.

Design Rules: Mapping Intent to Stylistic Choice

Design rules describe on a high level how illustrations should be put together. A design rule consists of a communicative goal and a set of style strategies. There are two types of design rules: design methods and design evaluators. *Design methods* specify how to accomplish communicative goals; *design evaluators* determine how well communicative goals have been accomplished. A design method specifies what *style strategies* must be achieved, in addition to how well each should be achieved in order to accomplish a communicative goal. A design evaluator determines how well a communicative goal is achieved based on the achievement ratings of a collection of *style strategies*. Each communicative goal formalized in IBIS's intent-specification language [Seligmann 91] must have one or more design rule to accomplish and evaluate it.

Showing Location

Figure 1 lists two design rules for satisfying the communicative goal *location*. Figures 2 and 3 are illustrations that IBIS generated using design rules 1 and 2. In both illustrations, the location of the function dial is shown in context of the parent object, the radio. (How design rules are activated is described later.)

Design Rule 1 specifies that to show the location of an object (?object) in a specific context (?context-object), the following style strategies must be accomplished:

- The object must be *included* in the illustration. The achievement threshold "highest" indicates that this style strategy must be fully satisfied.
- The object must be recognizable.
- The context object must be included.
- The object must be visible.
- The object must be highlighted.
- The context object must also be recognizable, but with a lower threshold.
- The context object must also be visible, but with a lower threshold.

Design Rule 2 requires that a landmark object of the context object be visible and recognizable. A *landmark* is an object that serves as a key for identification, position, and/or location [Feiner 85]. IBIS uses a simplistic approach for identifying landmarks. It

Design Rule #1:

```
(method
(location ?object ?context-object highest)
=>
(include        ?object           highest)
(recognizable   ?object           high)
(include        ?context-object   highest)
(visible        ?object           high)
(highlight      ?object           high)
(recognizable   ?context-object   high)
(visible        ?context-object   medium-high))

(evaluator
(include        ?object           highest)
(recognizable   ?object           high)
(include        ?context-object   highest)
(visible        ?object           high)
(highlight      ?object           high)
(recognizable   ?context-object   high)
(visible        ?context-object   medium-high)
=>
(location ?object ?context-object highest))
```

Design Rule #2

```
(method
(location ?object ?context-object highest)
=>
(include        ?object           highest)
(recognizable   ?object           high)
(include        ?context-object   highest)
(visible        ?object           high)
(highlight      ?object           high)
?y <- (Landmark ?context-object)
(recognizable   ?y                high)
(visible        ?y                high))

(evaluator
(include        ?object           highest)
(recognizable   ?object           high)
(include        ?context-object   highest)
(visible        ?object           high)
(highlight      ?object           high)
?y <- (Landmark ?context-object)
(recognizable   ?y                high)
(visible        ?y                high)
=>
(location ?object ?context-object highest))
```

Figure 1. Two design rules for showing location

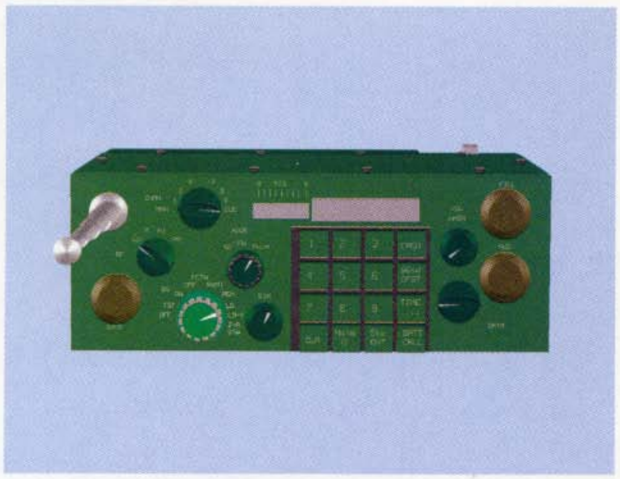

Figure 2. Showing location using Design Rule 1

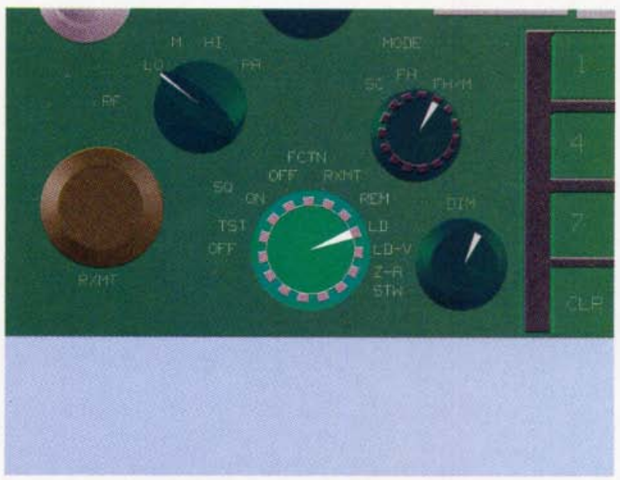

Figure 3. Showing location using Design Rule 2

considers any object with a unique property (such as shape, material, or color) to be a landmark.

Style Rules: Mapping Stylistic Choice to Visual Effects

There are two types of *style rules*. *Style methods* accomplish visual effects specified by style strategies and *style evaluators* determine the success of style strategies in a given illustration. Style methods specify *illustration methods*, which are procedures to accomplish specific visual effects. Style evaluators match *illustration evaluators* which examine a representation of the planned illustration to determine how well a visual effect is accomplished. Illustration methods and illustration evaluators are the only components of IBIS that directly access the illustration.

A Style Strategy: Highlighting

Highlighting is a style strategy that can be accomplished by applying one of several style rules. The purpose of highlighting an object is to emphasize it and draw attention to it. This can be accomplished by rendering it in a manner that distinguishes it from all the surrounding objects. Consider the following two style methods and two style evaluators.

Style Methods: Highlight Object x

1. Brighten x: Increase the intensity of the lights shone on x.
2. Subdue other objects: Decrease the intensity of the lights shone on other objects.

Style Evaluators: Highlight Object x

1. For every object with modified lighting: Evaluate the contrast between the object before and after the modification.
2. Evaluate the contrast between x and other objects.

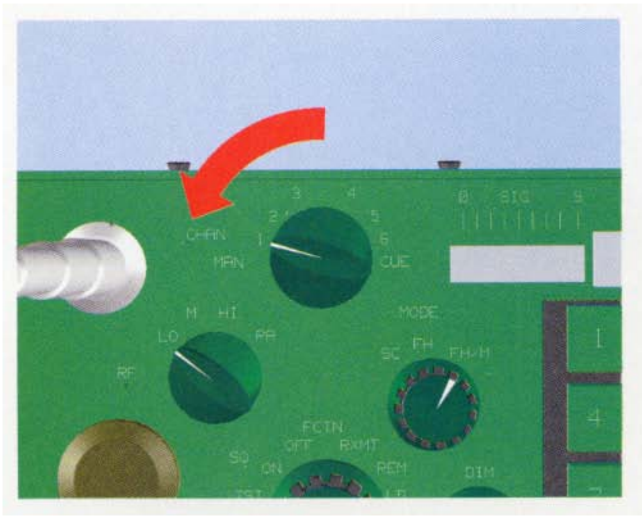
Figure 4. State of channel dial with no highlighting

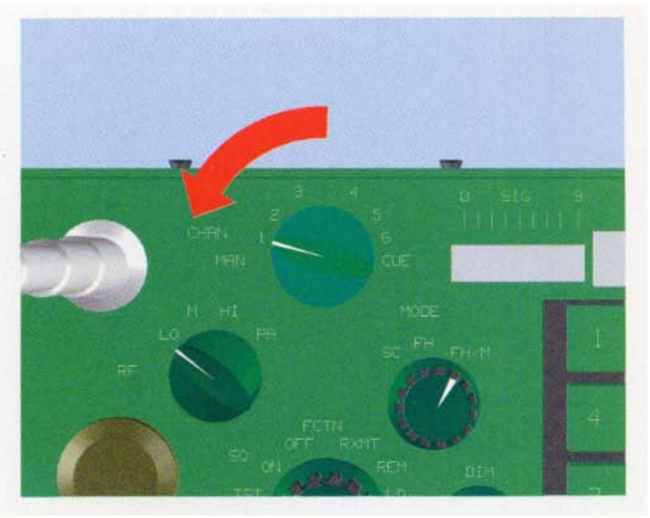
Figure 5. Highlighting by Style Method 1: Brighten object

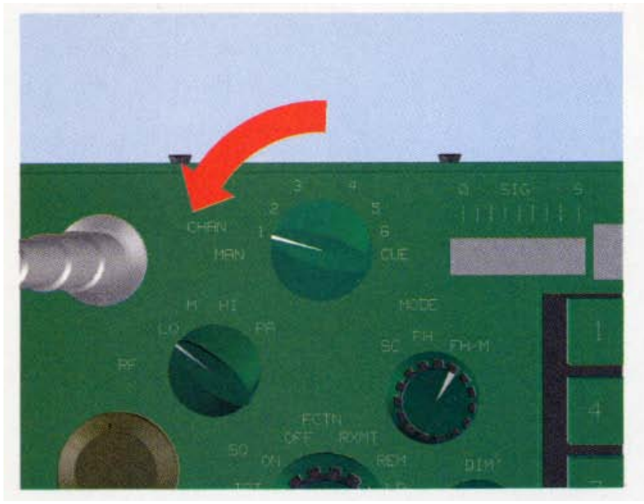
Figure 6. Highlighting by Style Method 2: Subdue other objects

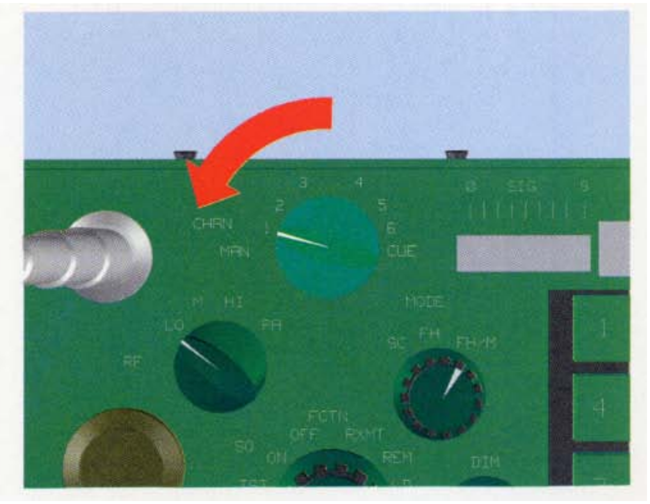
Figure 7. Highlighting: Combine Style Methods 1 and 2

Let us examine how IBIS uses these rules to highlight an object. At this point, a design rule has already been activated that asserts that the channel dial should be shown in addition to the other parts of the radio. This design rule also specifies that the channel dial should be highlighted.

Figure 4 shows the illustration IBIS would generate if highlighting were not specified. (Figures 4–6, which show intermediate states of the illustration during the illustration design process, were generated by requesting that IBIS render its intermediate results. They would not normally be rendered during the illustration design process.) The style rules are prioritized so that Style Method 1 is tried first, which executes illustration methods whose results are shown in Figure 5. Both style evaluators return unsatisfactory ratings. Style Evaluator 1 fails because its illustration evaluators detect that the dial's markings are brightest white, so increasing their lighting does not change their appearance. Style Evaluator 2 fails because its illustration evaluators detect that the other objects also have markings that are white and therefore do not contrast sufficiently with the channel dial's markings. Therefore, IBIS backtracks and returns the illustration to the state shown in Figure 4.

IBIS next tries Style Method 2, resulting in Figure 6. Style Evaluator 1 is successful, since the other markings are now darkened. Style Evaluator 2, however, returns a poor rating since the contrast between the channel dial and other objects is insufficient. Once again, IBIS backtracks and returns the illustration to the state shown in Figure 4.

IBIS now tries both style methods in combination, as specified by its search control strategy, with the results shown in Figure 7. Both style evaluators now return success. Therefore, IBIS asserts that the channel dial has been successfully highlighted.

Style Method 2 attempts to mute the objects at varying percentages, stopping at a prescribed threshold. In Figure 6, the global lighting is dimmed by 40%, the maximum allowed. In

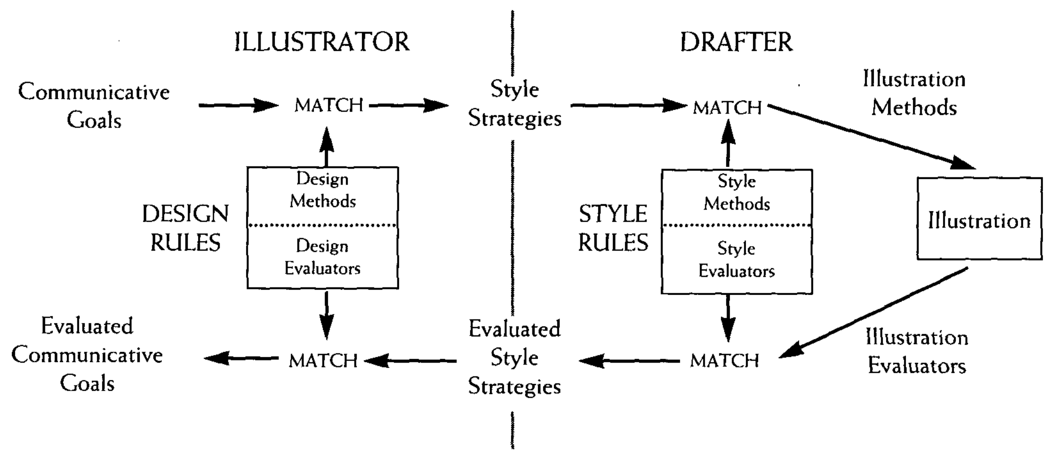

Figure 8. IBIS's illustration process

Figure 7, in which the global lighting is dimmed by 25%, IBIS decides that the brightened channel dial is sufficiently contrasted with other objects and that no additional muting is necessary.

Architecture

Illustrators

An IBIS illustration is designed by a component called an *illustrator*. An illustrator is assigned a set of communicative goals to fulfill. After trying the techniques at hand, an illustrator may detect that it cannot fulfill the complete set of communicative goals in just one illustration. For example, the communicative intent may be to show the opposite faces of the same object, or to show parts of an object in great detail, but also in context of a much larger object that must also be legible. No one view can satisfy these constraints. IBIS's rules allow it to create a *composite illustration*, which is defined as a set of related illustrations that in concert fulfill the communicative intent [Seligmann and Feiner 89]. Composite illustrations are made up of several sub-illustrations, each of which may be inside, overlapping, or next to others. The illustrator creates subordinate illustrators to which it contracts sets of communicative goals. One subordinate illustrator is responsible for the work already completed; the rest are assigned the remaining communicative goals. The original illustrator, which we call the *master illustrator*, is responsible for the work of the subordinates and the placement and sizing of their sub-illustrations. Although illustrations may have arbitrarily deep recursive hierarchies in theory, in practice the hierarchy is usually not very deep or broad.

While illustrators map communicative goals to style strategies with design methods and evaluate the success of communicative goals with design evaluators, they assign to *drafters* the task of accomplishing and evaluating style strategies.

Drafters

Drafters do not know about communicative intent. They are the unheralded workers who translate the illustrators' plans into reality. Drafters are tied to the hardware they utilize. For example, it is the drafters who apply the procedures that examine the contents of the framebuffer. Drafters share a body of style rules. Each style rule specifies illustration methods or evaluators to call in order to achieve or evaluate visual effects. Drafters report back to the illustrators with the achievement rating of the various style strategies they implement. Once an illustration has been approved by the master illustrator, it is the drafters who render the illustration.

Illustration Objects, Physical Objects, and their Relations

An illustration contains a set of *illustration objects*, each of which is created for that illustration. The drafter generates illustration objects when achieving style strategies. IBIS selects the objects to depict based on the communicative goals and design rules activated. Each illustration object usually depicts one or more corresponding physical objects in the knowledge base. Some illustration objects, however, may not correspond to any physical object, such as the arrow appearing in Figure 7. Such objects are called *meta-objects* [Feiner 85]. They are generated by the system to serve as visual annotations that illustrate those concepts that do not directly correspond to physical objects in the world being illustrated, such as the concept of turning in Figure 7.

An illustration includes a set of *object relations* that specifies the relationship between each illustration object and zero or more corresponding physical objects. Some physical objects have no corresponding illustration objects. These are the objects IBIS selects not to depict. In contrast, a physical object may correspond to several illustration objects. For example, two or more illustration objects can depict the same object in different states.

Generate and Test Approach

Figure 8 summarizes IBIS's illustration design process. Communicative goals match with a design method in the illustrator's design rule base. The design method asserts a set of style strategies. Style strategies match with style methods in the drafter's style rule base. This, in turn, activates a set of illustration methods that access the illustration object directly. Corresponding style evaluators activate a set of illustration evaluators that also access the illustration. The illustration evaluators match with style evaluators to assert the success ratings for style strategies. The evaluated style strategies match with design evaluators and assert the success ratings for communicative goals.

All illustrators share a set of design rules. All drafters share a set of style rules. Each illustrator or drafter, however, can be specified with a different illustrative style. An *illustrative style* is represented by ordering the rules so that preferred methods are always attempted first. When a preferred method fails, the illustrative style is overridden. The illustrations IBIS generates can combine different illustrative styles. The illustrations shown in the figures were generated using the illustrative style *realistic*. The *realistic* illustrative style favors methods that do not alter physical properties of objects. For example, highlighting methods that use lighting are preferred to methods that change the color of an object.

Evaluation is based on a system of ratings and thresholds. *Thresholds* are assigned by the illustrators and are inherited by the drafters for each style strategy. A threshold represents a minimum degree of success required for a method to be considered acceptable. Evaluators can directly assert that either a style strategy or communicative goal has been achieved.

Visibility and Recognizability

Style rules for evaluating and achieving highlighting were illustrated earlier. We now describe visibility and recognizability. First, we discuss the evaluators that the drafter uses to determine if objects are visible and recognizable in the current illustration. Then we discuss the methods the drafter uses to achieve visibility and recognizability.

Evaluating Visibility

Every IBIS illustration depicts at least one object that must be visible. We call an object that must be visible an *unoccludable* object. IBIS stores its objects in a parts hierarchy. An unoccludable object may be any node in the hierarchy. An object is considered completely visible if it resides entirely within the view volume and no other objects obscure it. Thus, an object is partially visible if it is obscured by other objects or if it is not completely within the view volume.

We use several approaches to determine visibility quickly, described in [Feiner and Seligmann 90]. One approach uses z-buffer picking [Foley et al. 90] to detect occlusion. The hardware z-buffer is loaded with an unoccludable object and the remainder of the z-buffer in the unoccludable object's bounding box is set to the closest possible z-value. For each remaining object, the system will determine if any part of the object is visible relative to the z-buffer, which occurs only if the object (partially) obscures the unoccludable object. This returns a binary occluding/non-occluding status. Another approach using shadow volumes [Chin and Feiner 89] returns a partially occluding/completely-occluding classification.

Evaluating Recognizability

An object is *recognizable* if its distinguishing features are shown. Some features depend upon the view; others depend on certain characteristics or attributes. We do not address the very difficult problem of determining or generating automatically *characteristic views* [Chakravarty 82, Kamada and Kawai 88] that ensure that an object's distinguishing characteristics are apparent. Instead each object is stored with an a priori characteristic view.

We represent a characteristic view as a union of volumes and a set of constraints. Each volume is specified by a ray originating from a point on the object. This volume represents a set of legal viewpoints that may be further restricted by the characteristic view's constraints. The constraints include a minimum screen size

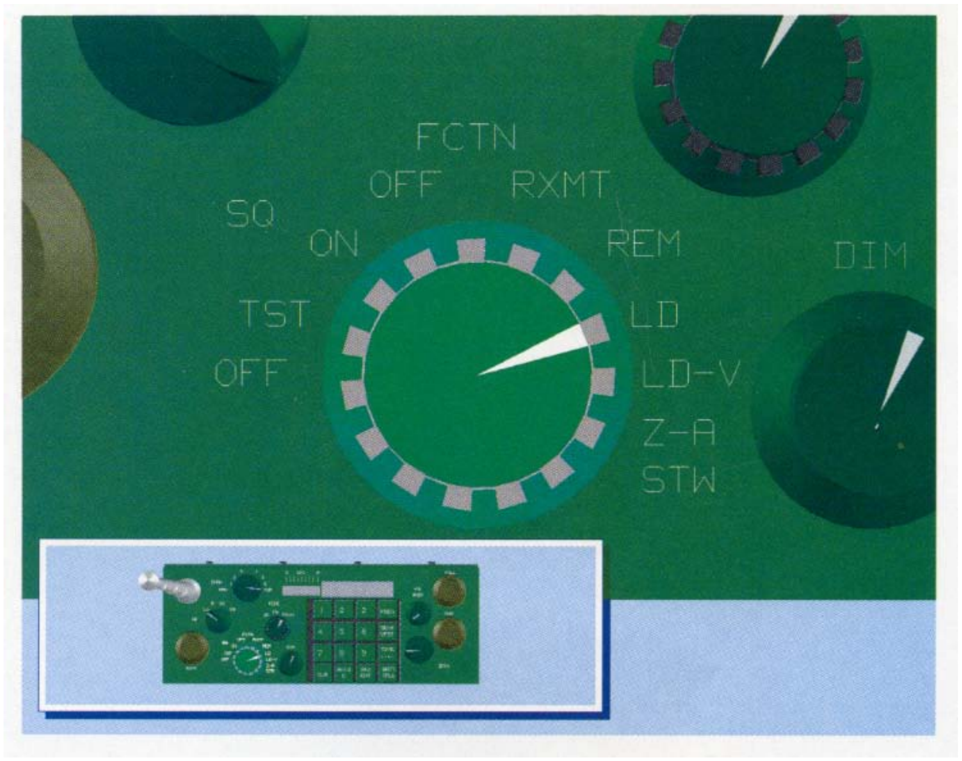

Figure 9. Automatically generated inset to show location during user navigation

the object must occupy and a list of properties that must be depicted.

Style Methods for Visibility and Recognizability

The drafter maintains a set of possible view specifications for every object that must be recognizable. A view specification satisfies the recognizability goals associated with these objects if the viewpoint lies within the intersection of the characteristic views' volumes and if the additional constraints are satisfied.

The visibility of each unoccludable object is maintained, if possible, by selecting a view in which unoccludable objects are not obscured. IBIS has several different methods for realizing visibility constraints when an unoccludable object is obscured by another object (that is not itself unoccludable). The first and simplest method is to remove from an illustration an object that obscures an unoccludable object. An object can be made visible by removing from the illustration all the objects that obscure it. This solution is problematic. In some cases, it would be misleading to remove objects from the scene. In other cases, it would be ideal.

A variety of illustrative styles have been developed by technical illustrators to depict obscured objects more clearly without completely eliminating those that obscure them [Giesecke et al. 36, Thomas 68, Martin 89]. These techniques include cutaways, transparency, and ghosting. We have developed several approaches for efficiently applying simple versions of these techniques interactively using z-buffer–based graphics systems [Seligmann and Feiner 89, Feiner and McKeown 90a, Feiner and Seligmann 91].

Interactive Illustrations

So far, we have treated IBIS's illustrations as static presentations. However, the same mechanisms that enable IBIS to design illustrations are utilized to maintain illustrations in their interactive state. An *interactive illustration* may be manipulated by a user. Currently, IBIS supports user-controlled view specification. In traditional user-controlled navigation, when the user specifies a new view, the same set of illustration objects is rendered from that view. In contrast, navigation in an illustrated 3D environment is more complex. The illustration is bound to the communicative goals with which it is specified. The illustration system's task is to satisfy continuously these communicative goals while the user changes the view specification. For example, consider an illustration in which the illustrator has determined that certain objects are unoccludable. As the user alters the view, these unoccludable objects may be obscured by other objects. The appearance of these otherwise occluding objects must be modified dynamically to maintain the unoccludable objects' visibility. (In [Feiner and Seligmann 91] we describe techniques for automatically maintaining visibility during an interactive session.)

Alternatively, different design rules may be activated to satisfy a communicative goal as the view specification changes. Consider an interactive session beginning with Figure 2, in which the communicative goal is to show the location of the function dial. Figure 2's view specification is generated by IBIS. As the user zooms in, using IBIS's interactive interface, Design Rule 1's evaluator is no longer satisfied: the context object is no longer completely recognizable and visible. However, Design Rule 2's evaluator is activated because the current view includes the keypad buttons, which are unique objects on the radio and considered landmarks of the radio. The communicative goal to show location is maintained and IBIS does not have to redesign the illustration. The user continues to zoom. Now, only the function dial is visible and recognizable. If design rules 1 and 2 are the only rules for showing location, then the communicative goal has been violated, since no design rule is satisfied. IBIS opts to generate a composite illustration, and designs and positions an inset illustration (using Design Rule 1), which pops up during the interactive session. The resulting illustration is shown in Figure 9.

Composite Illustrations

Here we describe some of the top-level decisions IBIS made when designing the illustration shown in Figure 10. The illustration is intended to show the user how to snap the latches of the primary battery box, as well as to indicate, with lesser importance, where another battery (the holding battery) is located. The master illustrator is assigned the following communicative goals:

(state latch1 snapped highest)
(state latch2 snapped highest)
(state latch3 snapped highest)
(state latch4 snapped highest)
(location holding-battery radio medium-low)

These communicative goals activate the following design rules that specify the following style strategies.

For each latch:
(include latch highest)
(context latch medium)
(recognizable latch high)
(visible latch high)
(highlight latch high)
(change latch snapped highest)
(meta-object latch snapped highest)

For the battery:
(include holding-battery highest)
(visible holding-battery medium-low)
(recognizable holding-battery low)
(context holding-battery medium-low)
(highlight holding-battery medium-low)

The illustrator's drafter tries to satisfy the highest priority style strategies first and begins by generating illustration objects for the latches, holding battery, and the rest of the radio. The recognizability constraints are set up for each object. The drafter fails when trying to make the fourth latch recognizable. Since all goal cannot be satisfied, IBIS decides that a composite illustration is needed. The master illustrator contracts two subordinate illustrators to handle the following communicative goals:

Illustrator One:
(state latch1 snapped highest)
(state latch2 snapped highest)
(state latch3 snapped highest)
(location holding-battery radio medium-low)

Illustrator Two:
(state latch4 snapped highest)

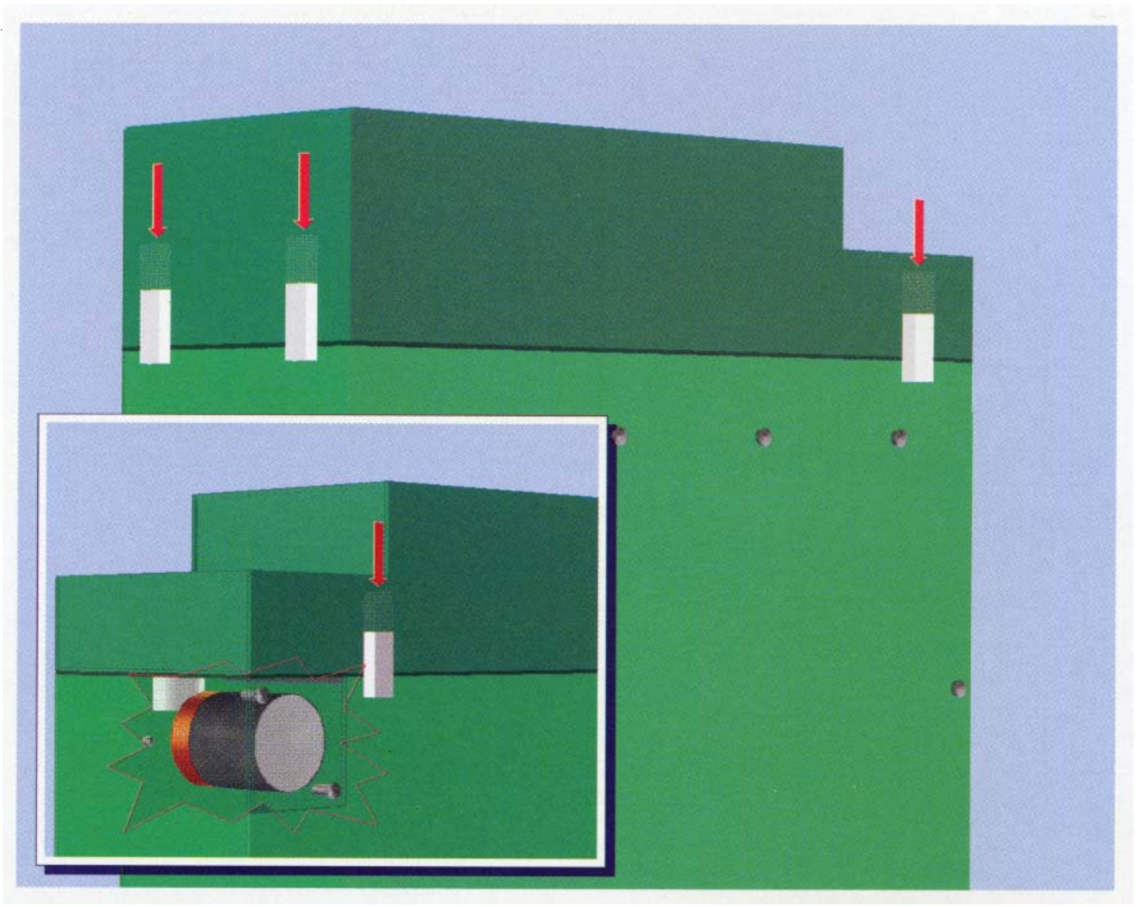

Figure 10. Showing snapping of latches and location of holding battery with a cutaway view

The latches are highlighted by increasing the intensity of their lighting. The drafter for Illustrator One reports failure for showing the battery—an unoccludable latch obscures it. Because the visibility goal for the holding battery is of low priority, the view specification that currently satisfies the high-priority recognizability and visibility goals associated with the latches is not altered. The master illustrator therefore assigns the holding battery's location goal to Illustrator Two:

Illustrator One:
(state latch1 snapped highest)
(state latch2 snapped highest)
(state latch3 snapped highest)

Illustrator Two:
(state latch4 snapped highest)
(location holding-battery radio medium-low)

Illustrator Two's drafter determines that the holding battery's recognizability and visibility goals can be achieved using the current view specification. The drafter then specifies a cutaway view for the holding battery. In the rule base for this illustration, a style method specifies that occluding objects be drawn using a wireframe style, and that the cutaway itself be semi-transparent. The concept of *snapping* is shown in the following way. The style rule specifies how to shape, position and orient an arrow meta-object based on the geometric information of the latch in the two states as well as the final view specification. The arrow begins at the previous state and points to the next state. The communicative goal to show the change of state is activated by a design rule that handles state and snapping. It activates a style strategy to show the object in both states. A style method specifies that a "ghost image" [Martin 89] be used to show the previous state of each latch. Illustration objects representing each latch in its previous state are generated. These ghost objects inherit the material and lighting from the illustration objects that are related to the same physical object, but their material is set to be partially transparent. The following constraints are added for each arrow and ghost object:

(visible ?object high)
(recognizable ?object high)

The master illustrator is notified that both illustrators have achieved the communicative goals they have been assigned. The master illustrator must now size and position the two illustrations. An inset style is selected for the illustration generated by Illustrator Two. The illustration must be sized so that the constraints are not violated (such as recognizability) and it must be positioned so that it does not obscure the unoccludable objects in Illustrator One's illustration. The resulting illustration is shown in Figure 10.

Related Work

Several researchers have addressed the problem of automatic picture generation. Simmons's CLOWNS [Simmons 75] generates simple line drawings of a 2D clown. Neiman's GAK [Neiman 82] generates animated pictures for a CAD system help facility. Both these systems, however, rely on predesigned vector objects. Friedell [Friedell 84] has generated synthesized 3D graphic environments using evaluators and backtracking, but this work emphasized modeling environments, rather than designing pictures. Feiner's APEX [Feiner 85] system designs pictures that depict actions performed in a 3D world, but without backtracking, self-evaluation, style combination, or visibility checks. Mackinlay's APT system [Mackinlay 86] designs 2D presentation graphics for quantitative data using a system of evaluation and backtracking, which enables the system to combine styles. Strothotte's chemistry explanation system [Strothotte 89] generates pictorial explanations automatically, but relies on handmade bitmapped images.

Other researchers have addressed rendering problems related to the illustration of objects. Kamada and Kawai [Kamada and Kawai 87] have developed techniques for generating line drawings that show the internal structure of complex objects. Saito and Takahashi [Saito and Takahashi 90] and Dooley and Cohen [Dooley and Cohen 90a, Dooley and Cohen 90b] have also developed non real–time techniques using transparency, cross-hatching, and different line styles to generate high-quality images that convey shape and construction.

The work described here differs from previous work in a number of ways emphasized in this paper. Our approach to automated illustration of 3D worlds is intent-based and depends upon a system of methods and evaluators that enables multi-level backtracking based on evaluations of a partially generated illustration. Illustration objects are generated based on both the representation of the physical object as well as the communicative intent. IBIS's evaluation process attempts to approximate the relationship between the visual appearance of an object in the real world (limited by the models used) and its appearance in the illustration. Finally, IBIS introduces an approach for generating composite illustrations, as well as semantically bound interactive illustrations.

Implementation

IBIS is written in C++ and the CLIPS production system language [Culbert 88]. It runs under UNIX on an HP 9000 375 TurboSRX workstation, which provides hardware support for realtime 3D shaded graphics. Drafters currently use the HP Starbase 3D graphics package, while the user interface is written in X.

The radio featured in the illustrations consists of over 8000 polygons rendered at 1280 x 1024 resolution. IBIS took .8 seconds to design Figure 7 and 7 seconds for IBIS to design Figure 10. It takes approximately .3 seconds to render either illustration.

Summary and Future Work

IBIS demonstrates an automated intent-based approach to illustration. Illustrations are designed by first considering a specified communicative intent and the world depicted. IBIS treats illustration as a goal-driven process using a generate–and–test approach and relies upon a rule base to make stylistic and design choices. These rules are represented as both methods for accomplishing visual effects and evaluators for determining how well visual effects have been accomplished in an illustration. Any choice may negatively affect the success of others; IBIS backtracks to find alternative solutions.

Our current efforts concentrate on the development of a visual language for 3D worlds [Seligmann 91] that will incorporate formalisms for communicative intent, style, design, viewer model, and session model. Communicative intent will be extended to include goals to represent the *purpose* of the communication, such as warnings and reminders. Style rules are being arranged into a hierarchy of constraints, ranging from those that identify conformant classes of illustration elements (e.g. colors and lines) to those that identify unaesthetic choices. We are also developing meta-rules to select methods based on the overall problem (rather than searching for the first adequate solution). For example, while IBIS currently generates composite illustrations only as a last resort, a meta-rule could allow them to be created as a regular design option. Finally, IBIS is being enhanced to allow for user control on all levels of specification, including the choice of design rules and style strategies.

Acknowledgments

This work is supported in part by the Defense Advanced Research Projects Agency under Contract N00039-84-C-0165 and the Hewlett-Packard Company under its AI University Grants Program. Esther Woo, John Edmark, Garry Johnson and Alan Waxman implemented portions of the system. Norman Chin developed the efficient procedures that we use to manipulate shadow volumes. Michael Elhadad is a a fellow comrade in arms in the COMET project. Conversations with Tom Ellman, J.R. Ensor, Allen Ginsberg, Jacques Robin, Frank Smadja have been more than helpful. Much appreciation is due to Suzanne Oboler and Cynthia King for their critical reading of this paper. Many thanks to David Kurlander and Rick Beach for help with the color separations.

References

Chakravarty, I., and Freeman, H. Characteristic Views as a Basis for Three-Dimensional Object Recognition. In *Proc. Society for Photo-Optical Instrumentation Engineers Conf. on Robot Vision*, Bellingham, WA, SPIE, vol. 336, 1982. 37–54.

Chin, N. and Feiner S. Near Real-Time Shadow Generation using BSP Trees. In *Proc. ACM SIGGRAPH 89 (Computer Graphics*, 23(3), July 1989), Boston, MA, July 31–August 4, 1989, 99–106.

Culbert, C. *CLIPS Reference Manual.* NASA/Johnson Space Center, TX, 1988.

Dooley, D. and Cohen, M. Automatic Illustration of 3D Geometric Models: Lines. In *Proc. 1990 Symp. on Interactive 3D Graphics* (*Computer Graphics* 24(2), March 1990), Snowbird, UT, March 25–28, 1990, 77–82.

Dooley, D. and Cohen, M. Automatic Illustration of 3D Geometric Models: Surfaces. In *Proc. Visualization '90*, San Francisco, CA, October 23–26, 1990, 307–314.

Elhadad, M., Seligmann, D.D., Feiner, S., and McKeown, K. A Common Intention Description Language for Interactive Multi-

Media Systems. *IJCAI-89 Workshop on Intelligent Interfaces*, Detroit, MI, August 22, 1989, 46–52.

Feiner, Steven K. APEX: An Experiment in the Automated Creation of Pictoral Explanations. *IEEE Computer Graphics and Applications* 5(11), November 1985, 29–38.

Feiner, S. and McKeown, K. Generating Coordinated Multimedia Explanations. In *Proc. CAIA90 (6th IEEE Conf. on Artificial Intelligence Applications)*, Santa Barbara, CA, March 5–9, 1900, 290–296.

Feiner, S. and McKeown, K. Coordinating text and graphics in explanation generation. In *Proc. AAAI-90*, Boston, MA, July 29–August 3,1990. 442–449.

Feiner, S. and Seligmann, D.D. Dynamic 3D Illustrations with Visibility Constraints. In *Proc. Computer Graphics International 91*, Cambridge, MA, June 24–28, 1991.

Foley, J., van Dam, A., Feiner, S., and Hughes, J. *Computer Graphics: Principles and Practice 2nd Edition*. Addison-Wesley, Reading, MA, 1990.

Friedell, M. Automatic Synthesis of Graphical Object Descriptions. *Computer Graphics* 18(3), July 1984, 53–62.

Giesecke, F., Mitchell, A., and Spencer, H. *Technical Drawing*. New York, The Macmillan Co., 1936.

Kamada, T. and Kawai, S. An Enhanced Treatment of Hidden Lines. *ACM Trans. on Graphics* 6(4), October, 1987, 308–323.

Kamada, T. and Kawai, S. A Simple Method for Computing General Position in Displaying Three-Dimensional Objects. *Computer Vision, Graphics and Image Processing* 41(1), January, 1988, 43–56.

Mackinlay, J. Automating the Design of Graphical Presentations of Relational Information. *ACM Trans. on Graphics* 5(2), April 1986, 110–141.

Martin, J. *High Tech Illustration*. Cincinnati, OH, North Light Books, 1989.

Neiman, D. Graphical Animation from Knowledge. In *Proc. AAAI '82*, Pittsburgh, PA, August 18–20, 1982, 373–376.

Saito, T. and Takahashi, T. Comprehensible Rendering of 3-D Shapes. In *Proc. ACM SIGGRAPH '90 (Computer Graphics, 24(4)*, August 1990). Dallas, TX, August 6-10, 1990, 197–206.

Seligmann, D. D. Intent-Based Illustration: A Visual Language for 3D Worlds. Thesis Proposal. Department of Computer Science, Columbia University. New York, January 1991.

Seligmann, D. D., and Feiner, S. Specifying Composite Illustrations with Communicative Goals. In *Proc. UIST '89*. Williamsburg, VA, November 13–15, 1989, 1–9.

Simmons, R. F. The Clowns Microworld. In *Proc. TINLAP '75*, 17–19.

Strothotte, T. Pictures in Advice-Giving Dialog Systems: From Knowledge Representation to the User Interface. In *Proc. Graphics Interface '89*, London Ontario, June 19–23, 1989, 94–99.

Thomas, T.A. *Technical Illustration, 2nd. Edition*. McGraw-Hill, New York, NY. 1968.