HyperActive: An Automated Tool for Creating Hyperlinked Video

by

JONATHAN DAKSS
A.B., Columbia University (1997)

Submitted To The Program in Media Arts and Sciences, School of Architecture and Planning, In Partial Fulfillment Of The Requirements For The Degree Of

Master Of Science in Media Technology

At The Massachusetts Institute of Technology
September 1999

Author____________________
Program in Media Arts and Sciences
August 6, 1999

Certified By____________________
V. Michael Bove, Jr.
Principal Research Scientist
Program in Media Arts and Sciences

Accepted By____________________
Stephen A. Benton
Chair, Departmental Committee on Graduate Students
Program in Media Arts and Sciences

HyperActive: An Automated Tool for Creating Hyperlinked Video

by

Jonathan Dakss

Submitted To The Program in Media Arts and Sciences,
School of Architecture and Planning On August 6, 1999
In Partial Fulfillment Of The Requirements For The
Degree Of Master Of Science in Media Technology.

Abstract

There are currently strong motivations within the field of multimedia research to develop systems that can associate information with objects in video and allow users to access this information by selecting the objects using some form of interface. These systems could be used to create a new form of interactive video presentations, known as "hyperlinked video," in which the viewer has a direct role in influencing the content of the video. My thesis presents the schematics for a powerful, yet easy-to-use software tool for creating hyperlinked video programs. It is based upon a process for automatically tracking and segmenting video objects at a pixel-level. In addition to object tracking, this tool helps authors manage the various properties involved with a hyperlinked video production, including relevant video information, object data and "link" information, such as actions associated with each object. Unlike comparable tools, this system employs novel techniques to further automate the authoring process by recognizing the identity of objects in new sequences. The system was employed in the creation of two innovative hyperlinked video applications, one with a commercial focus and the other an educational focus.

Thesis Advisor:
V. Michael Bove, Jr.
Principal Research Scientist
Program in Media Arts and Sciences

This research was supported by the Digital Life Consortium at the MIT Media Laboratory.

HyperActive: An Automated Tool for Creating Hyperlinked Video
Thesis Committee

Reader____

Walter Bender
Senior Research Scientist
Program in Media Arts and Sciences

Reader____

Prof. Brian K. Smith
Assistant Professor
Program in Media Arts and Sciences

Table of Contents

Acknowledgements

The following people deserve my sincere thanks for their contributions to the work contained in this document:

My advisor, **Mike Bove** -- My deepest thanks to you for providing me with the opportunity to work at the MIT Media Lab. Your expert guidance, intellect, insight, resourcefulness and insistence on quality have not only structured my research here but also my entire thinking and working process as well. Plus it's not often that your boss enjoys your quirky favorites (The Residents, Ernie Kovacs, Singin' In The Rain) with more fervor than you do! I hope that I have met and will continue to meet your expectations and that we will continue to collaborate in the future.

Walter Bender -- in many ways you have been like a co-advisor to me; your comments were always encouraging, challenging and astute. Your feedback while I worked on HyperSoap was essential (I didn't forget your early comment that color could be used to distinguish "browsing" and "selecting"!) and you helped pave the way for the Julia Child demo as well. I am proud to have you on my thesis committee.

Brian Smith -- your insight on my work here, as well as your omnipresent sense of humor, were greatly appreciated.

Stefan Agamanolis, Edmond "Eddie" Chalom, Nuno Vasconcelos, and Bill "Bu-Bu" Butera -- for your assistance in my research, software development, and paper authoring. I certainly could not have been as productive as I was without your help. Extra thanks to you Stefan for co-authoring the SPIE paper with me, making the touchscreen *Isis*-friendly and for all of your hard work on *HyperSoap*.

My UROP student **Marina Zhurakhinskaya** -- for making it possible for me to deliver *An Interactive Dinner At Julia's* by the expected deadline -- or at least near it -- and for validating that spending hours scribbling on video windows does indeed make you crazy! ☺

Alex Westner, Bu-Bu, and John "Wad" Watlington -- for voluntarily reading my thesis proposal and making your astute and much-needed comments. Thank you to **Freedom Baird** for your volunteered and useful criticism on the SPIE paper. Thanks to **Pascal Chesnais** for your expert photography for the touchscreen photos.

Annie Valva and David Scutvick at WGBH and **Michael Ponder at JCPenney** -- without you the prototypes we built would not have been even remotely possible. In my opinion you are "model sponsors" and I hope our work will benefit you as much as it has benefited me. Special thanks to **William Truslow and Jeff Garmel** for their great help with *An Interactive Dinner at Julia's*, and of course to The French Chef herself, **Julia Child**, for the inspiration. Also a huge thanks to **Kevin Brooks, Paul Nemirovsky and Alex Westner** for making *HyperSoap* as professional and as fun to watch as it is.

Doug Koen, Alex Westner, and Nitin Sawhney -- for keeping me from going insane during those overnight work sessions -- or at least for going insane with me. Hey, that was fun. Care for some soup, Nitin? ☺ **Flipmode iz da Squad!!!**

Henry Holtzman, Wad, Stefan Agamanolis, and the folks at NecSys -- for helping me out many times with general hardware, software, and networking issues.

The folks in the **Garden** -- for supporting me and my endeavors and giving me a fun and helpful environment to work in ... and of course for tolerating my tastes in music (death to the Garden-Box!).

AND a very special thanks to **Marina Dadashev** for always supporting me and tolerating my prolonged stays at the Lab ... not to mention helping me develop my software and segment video for *Dinner at Julia's*!

Chapter 1

An Introduction

There are currently strong motivations within the field of multimedia research to develop systems that can associate information with objects in video and allow users to access this information by selecting the objects using some form of interface. These systems could be used to create a new form of interactive video presentations, known as "hyperlinked video," in which the viewer has a direct role in influencing the content of the video. For example, an educational program could be created in which children learn more about particular aspects of nature by selecting animals and foliage observed in "safari" footage with their remote control. Likewise, the technology could be used to create new forms of entertainment, such as a television program or movie in which selecting objects and actors influences the narrative. Playback of such material is well within the capabilities of desktop PCs and typical digital television decoders with graphical overlay capability. However, its creation has posed a challenge to authors because of the difficulty of identifying and tracking the selectable regions in every frame, by either manual or automatic methods.

I have created a tool for authoring hyperlinked video that is largely based on a process proposed by Chalom and Bove for tracking and segmenting video objects [Chalom96, Chalom98]. In this tool, the author of a hyperlinked video application uses a computer mouse to scribble roughly on each desired object in a frame of video. The system then generates full "segmentation masks" for that frame and for temporally adjacent frames in the sequence. These masks associate every pixel in every frame of the video to one of the author's defined regions. The author may then associate each region with a particular action, such as a display of information or a switch to a different video clip. I have developed a "properties manager" which simplifies the often painstaking process of linking each object in a segmented shot of video with information, such as an ID number, text string or "action." In particular, the properties manager stores information about each linked object in the video and then "guesses" the identity of pre-linked objects, making the link automatically if it guesses correctly.

This system was utilized in the creation of two very different hyperlinked video productions. The first, *HyperSoap,* is a hyperlinked television program that resembles serial dramas (known as "soap operas") in which the viewer can select props, clothing and scenery to obtain more information about the items, such as prices and retailers. There are several different ways to interact with *HyperSoap;* a viewer can click on an object and see an information text box appear instantly. He or she can click on an object and at an appropriate point in the drama the show pauses to show a catalog page, or in an entirely different mode he or she can click on an object and be shown plot details involving the object.

The second production, *An Interactive Dinner at Julia's*, is an interactive cooking show featuring Julia Child. Objects present in a dinner party video clip are starting points for exploring additional clips; for example, you can click on the napkins the guests are using and be shown a clip in which Julia explains elaborate napkin folding. At the end of each new clip, the viewer is automatically returned to the previous clip. Icons appear on the screen to indicate what the user interrupted to jump to a new clip; the user can also click on these icons and be instantly returned to the interrupted clip. Additionally, clicking on other food items and utensils cause an information text box to appear with additional object-specific details.

In the following chapter I will review the history of hyperlinked video and describe existing hyperlinked video software tools. In Chapter 3, I will identify several key issues in designing hyperlinked video systems. In Chapters 4 and 5, I will discuss the authoring tool I have created and its underlying algorithms; Chapter 4 addresses the segmentation component while Chapter 5 details the properties management component. In Chapters 6 and 7, I will describe *HyperSoap* and *An Interactive Dinner At Julia's* in more detail and discuss several key features of their functionality. In Chapter 8, I suggest how my work can be improved further, and I conclude in Chapter 9.

Chapter 2

Hyperlinked Video History

The following subsections present important milestones in the research of hypermedia, specifically hyperlinked video systems and applications. They are followed by descriptions of several commercial products based on hyperlinked video technology.

2.1 Hypertext origins

According to [Nielsen90], the history of "hypermedia" starts with Vannevar Bush, the originator of the hypertext concept. He proposed that an information resource typically read in a sequential fashion could feature a non-linear information chain in which relevant resources are connected in a manner that is visible to the reader. He described the first hypertext system, *Memex* (MEMory EXtender), which was never implemented, only invented in theory. He developed the ideas on this topic in 1932 and published the first article on this topic, "We May Think," in 1945.

Memex was proposed as a means of linking all information contained on microfilm. Bush asks the reader to envision an entire microfilm library laid out on a single table. With proper projectors, areas could be displayed in a manner similar to the X-Window system, and areas of common relevance could be linked via an associative index.

In 1967, the *Hypertext Editing System* was developed at Brown University under the leadership of Andries van Dam. This was the first working hypertext system, developed for an IBM/360 mainframe computer with 120-Kbytes of memory. It was sold and used for the documentation of the Apollo space mission. This system linked documents through pointers; the user interface was implemented through text.

2.2 *The Aspen Movie Map*

An early example of a video system that featured hyperlinks was *The Aspen Movie Map*, created at the MIT Architecture Machine Group (now the MIT Media Laboratory) in 1978 [Bender80]. With this application, a virtual drive through the city of Aspen, Colorado could be generated via a user's interaction with a touchscreen interface to computer-controlled laserdisc players. The user could "drive" forward or in reverse, turn left or right at intersections and "look" out the side "windows." To achieve this, four film cameras were installed on a truck at 90-degree angles from each other. The truck drove through all streets of Aspen, capturing frames approximately every three meters. This film was transferred to a video disk for playback.

There were two sets of hyperlinks in the Aspen system. Navigation through the streets was encoded by a hand-assembled correspondence between the layout of the town and the layout of the video on the disc. These links were accessed by touching turn signals that were overlaid on the video.

Additional hyperlinks within this video were automatically generated using a 3-D computer model of Aspen that was constructed from aerial photographs. Every building or zone in the model was "linked" – it was associated with additional slides or video clips. During playback, the user's position during their drive through Aspen could be found within the 3-D model based on knowledge about the speed of playback and the distance which lapsed before each set of video frames was recorded. This 3-D information was then mapped onto the video (which is a camera's 2-D view of an original 3-D scene), such that the user could touch buildings, see the building or zone highlight, and then access the information associated with the link. Every building had an associated link, so there was no need to provide any external cue as to the location of the links – an important issue when designing hyperlinked video which I discuss in Chapter 3.

2.3 *Video Finger*

Another early demonstration of hyperlinked video with segmented "objects" was John Watlington's *Video Finger* project at the MIT Media Lab in 1989 [Watlington89]. *Video Finger* is a software program with a purpose similar to the Unix "finger" utility, which can be used to find out the identities and activities of people who are presently using a particular computer or computer cluster. When utilizing *Video Finger*, people ascertain this information by viewing a screen showing continuous video footage of people together in a single space behaving in a manner that symbolizes their actions on the cluster. Clicking with a mouse on any part of these people as they moved shows more detailed textual information about their identities.

To create *Video Finger*, digital video clips were recorded of individual users engaged in various activities which would ultimately represent their cluster behavior, such as sleeping (remaining idle) and reading a newspaper (reading Usenet newsgroups).

These clips were then "rotoscoped" (the people were manually segmented from their backgrounds) and stored in a database. When the software runs, a static background of a computer lab is displayed. When the software detected that someone had logged in to the cluster or changed what they were doing on the cluster, it retrieved the video clip most pertinent to their behavior and composited it with the background and other clips. Since the computer composited these clips, the identity of the person to whom each displayed pixel corresponds was known and could be therefore accessed by a mouse click.

2.4 *Elastic Charles*

Another early example of a hyperlinked video system was the *Elastic Charles* project, created by the Interactive Cinema Group at the MIT Media Laboratory in 1989 [Brøndmo91]. In this project, a "hypermedia journal" was constructed which consisted of video, audio, imagery and text information concerning the Charles River. The user browses through video clips via a computer interface as he or she would browse text articles in a printed journal. Upon selecting a clip, the video plays and at certain times Micons (moving icons) appear and disappear on another screen that represent "link opportunities" to relevant video clips. Selecting these Micons, which are short digital samples of clips that loop in a small window on the screen, cause the new clip to play in place of the previous one. Not only can the duration of time that a Micon appears on the screen be made dependent on its temporal context, a Micon could also be given a signifying position within the video frame, and thus a spatial relation to objects displayed in the video's *mise en scene* is formed.

The application was created using analog laserdisc video and a set of software tools that allowed an author to create Micons and to manually select the start and end frames in which a Micon would appear. HyperCard, a commercial software tool for

Macintosh computers [Apple89], was used to create a database for information relevant to each Micon, such as key words that can be used to search for similar segments.

2.5 *HyperPlant*

Largely based on the *Aspen Movie Map* project, *HyperPlant* was developed at the Hitachi Research Laboratory in 1992 [Tami92] and featured object-based hyperlinked **live** video. In *HyperPlant*, a video camera whose panning and zooming could be controlled remotely was placed in a machine room in a power plant; the machinery can be controlled remotely as well. All of the objects in the camera view can be identified and accessed and virtual controls can also be mapped onto the camera's views of real machinery. As an example, the user can focus the camera on a fuel pipe and see a graphical "slider" positioned over it. As the user manipulates the slider, he or she controls the amount of fuel running through the pipe and can see the results of their actions in the video (such as a fuel leak). The user can also focus the camera on a real "On/Off" button in the machine room, and when he or she clicks on the button with the computer mouse, the button in the video depresses and the machine is controlled accordingly.

To achieve this, a 2-D model is created of the room and machine controls in this model are given links to actions. As the user controls the camera, the position of its 2-D view of the room can be tracked using the 2-D model. The tool for creating the model requires the user to manually draw outlines around the objects as they appear in video images (the object information is recorded with the camera's parameters). The action associated with an object is described by an event-response language that has the ability to manipulate machinery in the plant.

2.6 *The Interactive Kon-Tiki Museum*

Unlike most museums, which feature artifacts and exhibitions, The Kon-Tiki Institute for Pacific Archeology and Cultural History in Oslo, Norway consists of a large collection of film footage which document Thor Heyerdahl's many expeditions and exhibitions. The aim of the *Interactive Kon-Tiki Museum* project in 1994 was to utilize hypermedia to combine this footage with models, pictures, drawings, photographs and text to create an interactive touchscreen installation for the museum [Liestøl94].

A person using the *Interactive Kon-Tiki Museum* is shown digital video sequences in a window on a computer screen. During the sequences, graphical "buttons" appear around the video window that allow access to supplementary media. The *Interactive Kon-Tiki Museum* adopted a convention popularized by the Apple Macintosh graphical user interface to visually represent the relationship between a part of a video sequence describing a specific topic and the button that links to further information about this topic. At a point in the sequence where additional information is available, a still image is shown while narration is heard; a copy of the image then shrinks as it moves across the screen and settles into a graphical button located just outside the main video window. If the user touches one of these buttons during a sequence, and the relevant information is in the form of another video sequence, the image shown on the button will grow and travel across the screen and into the main video window, where the new sequence will begin playing.

2.7 *HyperCafe*

A recent application of hyperlinked video was *HyperCafe*, created by Sawhney, Balcom and Smith at Georgia Tech [Sawhney97]. Developed as a mechanism to explore hyperlinked video concepts, HyperCafe shows the user a virtual cafe composed of digital video clips of actors engaged in conversations. Users can follow different conversations via temporal opportunities that present alternative narratives. These opportunities are indicated in several ways: by Micons that appear throughout conversations, hyperlinked words that are overlaid on the video (such as explanatory text, contradictory subtitles, and intruding narratives) and also wireframe boxes which are centered around linked video objects. To create *HyperCafe*, Sawhney et. al. developed a scripting environment using Macromedia's Director and its Lingo programming language. However, they were required to author the object-based links manually and have indicated this as a "tedious" problem that begs for automation.

Additionally, their work includes an analysis of potential applications for hyperlinked video and presents an intriguing fictional example that they refer to as "News of the Past." In this desktop computer application, a user watches an old video clip of Walter Cronkite describing a Vietnam War battle. As Cronkite mentions the city of Hue, a Micon appears at the corner of the screen displaying frames from a documentary about Hue. When the boy moves his cursor over this Micon, Cronkite's voice fades out while the voice of the Hue documentary fades in. At this point, a white frame appears around the Micon, reminding the user that the hyperlink opportunity will soon be no longer relevant to Cronkite's subject matter and that the Micon (and the link opportunity) will therefore soon disappear. Selecting the Micon with the cursor would cause the documentary to occupy the full screen in place of Cronkite. Similar link opportunities appear and disappear throughout the rest of Cronkite's commentary, presenting a new way for people to experience news programs.

2.8 Commercial authoring tools

Outside academia, several companies have recently announced commercial products for authoring hyperlinked video. VisualSHOCK MOVIE from the New Business Development Group of Mitsubishi Electric America, Inc. concentrates on playback from the Web or a local source (e.g. CD-ROM, DVD-ROM) [Mitsubishi98]. The tool, called the Movie MapEditor, requires specifying a rectangular bounding box around the location of a desired object in the starting and ending frames of a video sequence. A proprietary tracking algorithm estimates the position of the "hot button" in intermediate frames. Manual tracking by the operator is also supported. HotVideo from IBM's Internet Media Group has a similar approach, but supports a larger set of geometrical object shapes [IBM98]. Intermediate locations of an object may be specified in multiple keyframes or located by a tracking algorithm. Additionally, Veon's V-Active is a similar tool that also incorporates an object-tracking algorithm [Veon98].

It is worth noting that the methods these products employ for associating links to objects are often too coarse for many varieties of video sequences. Particular difficulties include associating links to objects that overlap or are enclosed within larger objects, such as a window of a house and the house itself, and perhaps a large tree partially obscuring both. To enable the association of links with arbitrarily shaped regions that may be changing over time, a system that can differentiate among objects with greater precision is needed. Many such segmentation systems exist that employ an automatic process to identify objects in video or imagery with block-level or pixel-level precision. For example, the VisualSEEk system automatically identifies regions based on color and texture distribution [Smith96]. This and other automatic systems are limited by the quality of the segmentation of objects and the relevance of the identifiable objects. Precise, high-resolution segmentation masks can be an important factor in playback systems, since they could be used in graphical overlays to indicate the presence of hyperlinks or to highlight objects when selected by the user.

As stated previously, the system I have developed features a video segmentation system that has the capability of semi-automatically creating such high-resolution masks. This capability means that hyperlinked video can be made from a wide number of video sources – from home movies, to television programs, to industrial information footage, etc. It does not require the generation of graphical models of locations shown in the video, such as the ones employed by the *Aspen Movie Map* and *HyperPlant* projects, nor would it require the amount of labor needed to author the video in an object-based hyperlinked video application like *Video Finger*. Because of its high precision, it can enable the association of links with a greater number of objects per frame than supported by systems like *V-Active* and *Hot Video*.

Additionally, the two hyperlinked video applications I developed, *HyperSoap* and *An Interactive Dinner at Julia's,* explore many of the issues (addressed by the projects described in this chapter) that arise when developing a hyperlinked video system. These issues are described in more detail in the next chapter.

Chapter 3

Hyperlinked Video Design Issues

The following sections identify many of the issues that must be addressed during the design and creation of a hyperlinked video presentation. All of them needed to be resolved before the aforementioned hyperlinked video projects were developed; and needed to be confronted as well when designing *HyperSoap* and *An Interactive Dinner at Julia's.*

3.1 Venue and Interaction

One of the most critical issues in designing hyperlinked video is one of venue – the environment in which the user views and interacts with the video. The video might be presented in a window on a computer monitor, on a standard television set or large screen projection, or on a touchscreen. The user might be sitting at a desk at work, on a sofa at home or at an exhibit in a museum. Several different kinds of devices might be used to select objects and interact with the video. In some ways, using a mouse on a PC may be natural for hyperlinked video, considering that people are familiar with using a mouse to activate hyperlinks on the World Wide Web. However, the desktop is not always an ideal place to view video. Many of the genres of content suitable for hyperlinked video are those that people are accustomed to watching in their living rooms.

A television viewing situation may be more natural for interacting with certain kinds of programming, but devices enabling users to interact with the video are less developed in this environment than in other venues. Since users are comfortable using a remote control device to browse through channels of programming on their television, it makes sense that they might also use this device to browse through hyperlinks. For example, with WebTV's *WebPIP* system [WebTV98], users can press a button on their remote control when an icon appears on the screen indicating the presence of a link. They can then choose to display the web page content referenced by the link or archive the link for later viewing. However, this system allows for only one link per segment of video. One can envision several different kinds of television interaction devices that could be used to select from several links that are available simultaneously. For example, a user could cycle through available links with a button on a remote control, and then activate a link by pressing another button. Other approaches might incorporate a position touch pad or a joystick. Alternatively, embedding an inertial sensor inside a remote control would allow the user to move the remote in the air to control the position of a cursor on the screen. The laser pointer / camera interface utilized in both *HyperSoap* and *An Interactive Dinner At Julia's* (described in detail in Chapter 6) was intended to simulate this sort of interaction.

3.2 Indicating links

When displaying hyperlinked video, it is important to indicate the presence of hyperlinks to users in a manner suited to the application and to the goals of the viewing experience. In most cases, the presence of hyperlinks should be readily apparent but not overly distracting; if possible, the preferences of the user should be addressed as well. The best cases are when there is no need to artificially indicate links because the user already has an idea about where they are, as in *the Aspen Movie Map*, *Video Finger*, and also in *HyperSoap*.

Many of the existing hyperlinked video playback tools that employ a pointing device for interaction indicate the presence of hyperlinks by changing the shape of the cursor when it is positioned over a hyperlinked object. When this happens, a name or description for the link may be displayed in an area of the computer screen. For example, IBM's *HotVideo* software changes the cursor to an icon that corresponds to the media type of the information contained in the link. *HotVideo* also displays an icon at all times during playback that changes color to indicate when hyperlink opportunities exist. Similarly, the *WebPIP* system displays a small icon in a graphic overlay when a hyperlink opportunity is available.

Other approaches have been used as well. *HotVideo* is capable of displaying wireframe shapes around the hyperlinked objects in the video, although this method can cause the video to become quite cluttered if there are several hyperlinks in a single frame. It can also be confusing to the user if a linked object occupies a large portion of the video frame or surrounds other objects. A *HotVideo* author can also choose to indicate hyperlinks by changing the brightness or tint of the pixels inside the wireframe shape, or by applying other transformations. In other object-based hyperlinked video applications, such as the *Aspen Movie Map* and *HyperPlant*, a selected object featured a highlight around its edges. This is the approach utilized in *HyperSoap* and *An Interactive Dinner At Julia's*.

3.3 Initiation of hyperlink actions

Unlike traditional hypertext documents, the temporal nature of hyperlinked video raises interesting issues regarding the timing of actions associated with selected hyperlinks. In some hyperlinked video playback systems, clicking on a link causes a new information source to be loaded immediately in a separate area. However, displaying lengthy or detailed information while the video continues to run may cause the user to feel distracted from the video content. Likewise, any delay in showing the link information may blur the context of the information such that it is no longer relevant to the video presentation. Thus, it is important to consider how and when hyperlink actions should occur in order to best serve the goals of the application. It may be important to maintain the continuity of the video while still presenting the desired link information to the user in a timely and meaningful fashion.

In addition to timing, the designer of a hyperlinked video system may also need to address the issue of transitioning between multiple media types in a manner which users find comfortable. For example, in *The Interactive Kon-Tiki Museum*, some of the "links" which can be followed during a video sequence are in the form of static text, not additional video sequences. Unfortunately, many users felt uncomfortable moving immediately from the relatively "passive" mode of watching a video sequence to the more "active" mode of reading text. The designers of *The Interactive Kon-Tiki Museum* solved this problem by giving the text similar characteristics to video: text blocks were given a sequential order of appearance, thus enabling the text to change over time, and a voice simultaneously reciting the text was also added. Certain words or phrases within a block of text were also linked to additional media, including video sequences; however, users of the *Interactive Kon-Tiki Museum* did not report feeling any discontinuity when moving from static text to video.

All of these issues were contemplated during the design of both *HyperSoap* and *An Interactive Dinner at Julia's*. Before discussing these applications and how they address these issues, I will first describe the features of the software tool used to create them.

Chapter 4

HyperActive: Identifying and Tracking Video Objects

I have designed a software tool for authoring hyperlinked video programs named "HyperActive." HyperActive features two significant areas of automation. The first is a segmentation system, largely based on the method proposed by Chalom and Bove, used to classify pixels in a video sequence as members of objects or groups of objects [Chalom96, Chalom98]. The second is a novel "properties management" system which automates the process of associating information with objects. In this chapter I will explain the segmentation system in detail; the next chapter details the properties management system.

4.1 Segmentation overview

When using HyperActive, the author of the video must first digitize scenes of video that he or she wishes to hyperlink. The software then utilizes a simple method for "temporal segmentation" -- breaking each scene up into "shots," where a "shot" is used to distinguish a set of successive frames which were created by a single camera view during the shoot. The software calculates the differences between pixel values in temporally adjacent frames, sums them and compares them to a threshold value. If the threshold is exceeded, the two frames are likely to be on either side of a shot boundary. If not, they are probably both within the same shot. Once this process is repeated for all pairs of temporally adjacent frames, the software can then identify and track the objects that appear in each shot. This method accurately found all of the shot boundaries in all of the footage used in *HyperSoap* and *An Interactive Dinner At Julia's*, as well as test footage from the movies *The Godfather*, *Austin Powers: International Man of Mystery* and footage from the television programs "Mama's Family" and "Soul Train" (the "Soul Train" footage was particularly difficult to temporally segment as shots always featured the same background, and lots of flashing lights and colors).

The author must then assist the software in defining the objects in each shot by providing "training data" in the form of rough scribbles on each object in one or more frames in each shot. The system creates a statistical model for each object based on features calculated from the training data. These models are then used to classify each of the remaining pixels in the frame as a member of the object with the highest statistical similarity. By tracking the training data bi-directionally (forward and backward in time) and applying the same operations, several seconds worth of video can be segmented. The segmented objects can then be associated with actions and information in an associative database.

4.2 Feature information

Unlike many segmentation systems that use only one or two different features to help distinguish objects, this system allows the author to select any combination from a variety of different features that estimate the color, motion, and texture properties of each pixel. The row and column positions of a pixel in the frame may also be used as features. Combining the data of several features in this way yields a more robust segmentation whose quality is less affected by misleading data from a single feature.

4.2.1 Color

Color in digital video is represented by three components, and a single color can be represented in any one of many color coordinate systems. HyperActive analyzes color in three primary color spaces, RGB, YIQ and LAB. A simple matrix conversion equation is needed to calculate the YIQ values for a particular pixel in the RGB color space. Calculating the LAB values for an RGB pixel requires an intermediate transformation to the CIE-XYZ space. Although each color space presents different advantages for gleaning the color properties of an object in video, I have found that YIQ and LAB information often yields more accurate segmentation when used in this system.

4.2.2 Motion

A constant brightness constraint (or "optical flow") method is used for computing a vector which describes the motion of a pixel in a video sequence. Introduced by Horn and Schunck, optical flow estimates the apparent motion of regions between two successive frames in an image sequence by measuring the change in the brightness pattern between those frames [Horn80]. The method implemented in HyperActive, proposed by Kanade and Lucas and implemented by Krause, forms a brightness change constraint equation (BCCE) for each pixel which represents the weights of small changes in the pixel's intensity in position and time [Kanade81, Krause87]. It then solves this equation through a least squares error (LSE) calculation over a local region.

4.2.3 Texture

The system uses two different statistical methods to determine texture characteristics of objects in each frame. These methods were chosen for use in the multi-feature system because they represent texture in a concise manner and can be run relatively quickly. Although texture calculations exist which are based on interframe, intra-frame measurements are utilized for texture estimation since motion information is incorporated during the segmentation stage.

The first method is a rather simple algorithm that computes the mean and standard deviation of the luminance values of pixels within a specified window size and assigns these values to the pixel at the center of the window. This algorithm for computing the local statistics within a window can also be run with the pixel values of channels in a particular color space. HyperActive has default window size settings that can be adjusted by expert users.

The second method, known as "simultaneous autoregression" is described by Jain and Mao and uses a linear prediction method to express each pixel in a video sequence as a function of its eight neighboring pixels [Jain92]. Because of symmetry properties, it is possible to represent this function with only four coefficients and an error value. To best estimate these parameters, a least squares error (LSE) solution is employed which incorporates pixel values within a larger window size.

Before segmenting each shot, the author indicates the type of features he or she wants to use for segmentation. HyperActive's explanation of the available features is as technical as is desired; for example, a novice user might simply choose "color" and "texture" while advanced users can choose between color spaces and determine texture resolution.

4.3 Training and tracking

In the next stage, the author must select representative frames from the sequence (typically ones in which the important objects are fully visible) and highlight representative pixels inside of each object using a simple drawing tool. This information serves as the training data for the segmentation algorithm (Figure 4.1).

The system then estimates the location of the training data pixels within each of the remaining frames in the sequence by using the motion estimator. Krause's optical flow algorithm is used to calculate motion vectors for the training pixels; these vectors are then applied to the pixels to estimate their location in neighboring frames.

The original tracking algorithm estimated the position of the pixels from a single training frame (the first frame of the sequence) forward through all frames to the last frame of the sequence. However, many sequences feature objects that enter and exit frames or become alternately occluded and visible. For this reason, it was necessary to design a new tracking algorithm that allowed the author to provide training data for multiple frames in a single sequence. Using this information, the system could track training data both forwards and backwards through the sequence. Bi-directional tracking made it possible to efficiently segment objects entering frames; the author can train the system using a frame in which the object is visible and the system then tracks that object backwards until it is out of view.

To help the author develop a tracking strategy, the system uses the numbers of the trained frames to walk them through the tracking process, asking him or her questions about how he or she wanted to track. For example, if the user trained on frames 10 and 20 in a sequence with 30 frames, the user would first be asked if he or she wanted to track backwards from frame 10 to frame 0. Then the user would

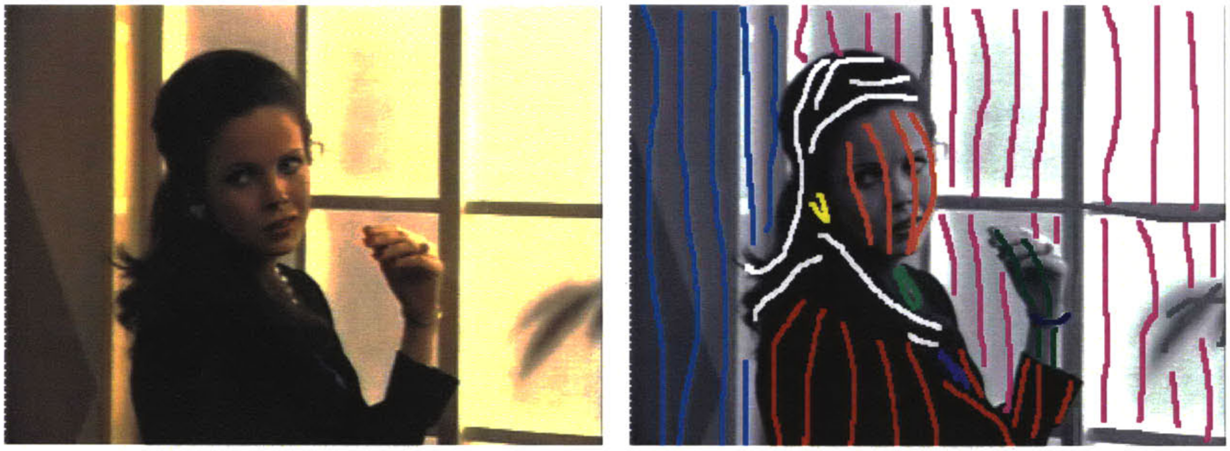


Figure 4.1: A frame from the *HyperSoap* video, and an example of "training data" scribbled by the author.

be asked if he or she wanted to track from frame 10 to frame 19, track to an intermediate frame, or skip frame 10 and instead track backwards from frame 20 to frame 11. If the user indicates that he or she wants to track forward to frame 15, the system would then ask him or her if he or she wants to track backwards from frame 20 to frame 15, etc.

Once the tracking has completed, there are pixels in each frame that have been classified to correspond to the objects defined by the author.

4.4 Classification of unlabeled pixels

After a multi-dimensional feature vector has been calculated for every pixel in the video sequence, the feature information for the tracked pixels is incorporated into a statistical model for each region. The feature information of each unclassified pixel will then be compared to these models such that the pixel will be labeled as a member of the region to which it bears the greatest statistical similarity. Basically, the feature information is used to express each region as a mathematical function such that HyperActive can input feature values of unclassified pixels and receive as output the probability that the pixel corresponds to that region.

Each feature of each region is modeled as a probability density function (PDF) where the shape of the model is assumed to be a multimodal Gaussian distribution. To model a multimodal PDF for the values of one feature for the pixels in one region, the correct number of Gaussian modes in the PDF needs to be determined, and the weight, mean and standard deviation of each mode must be computed. To do this, the Expectation Maximization (EM) algorithm [Redner84] is applied iteratively to compute these values for PDF's with different numbers of modes and then the function which most accurately represents the training data is chosen. To determine this accuracy, the change in Kullback Liebler (KL) distance of the PDF's in order of increasing number of modes is computed [Viola95], and the PDF for which there is maximal change is chosen. For a more detailed explanation of this process, please see [Chalom98].

Once there are PDF's which represent the distribution of pixel values of each feature for each region, the Likelihood Ratio Test [Therrien89] is used to classify each pixel that was not labeled by the user or by the tracking process described in Section 4.3. Essentially, a joint product is calculated of the results of inputting each value of a pixel's feature vector into their respective PDF's for a particular region. After this process has been repeated for each region, there is a product representing the probability that the pixel is a member of each region.

The pixel is classified to the region corresponding to the largest product. These labeled pixels comprise a segmentation mask, as shown in Figure 4.2. The ratio between the highest product and second-highest product is used to represent the system's "certainty" of a pixel's classification and can be incorporated during the postprocessing stage.

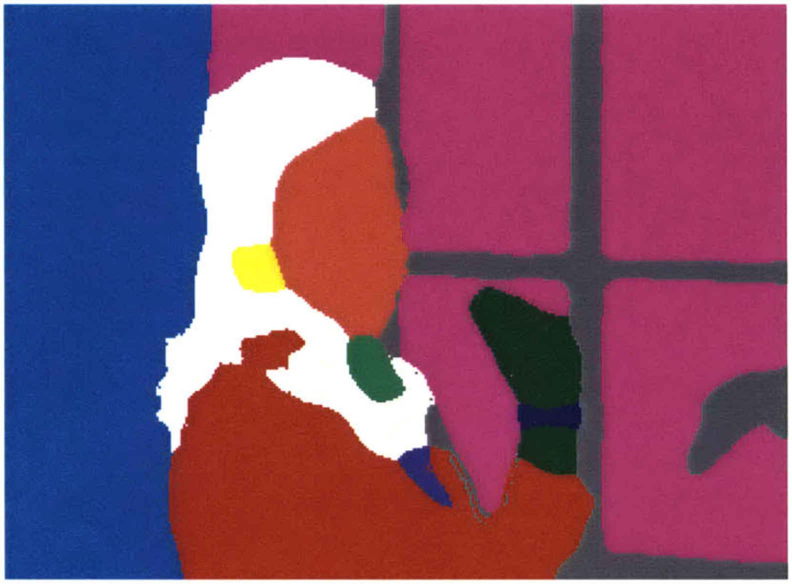


Figure 4.2: The output segmentation mask for the frame shown in Figure 4.1.

Although statistical models based on the position of pixels can be used, they do not prevent the system from classifying a pixel as a member of an object whose training pixels are wholly located on the opposite side of the frame. This is because separate models are generated for the row and column position information for an object, meaning that a pixel whose color, texture and motion characteristics are similar to pixels of an object located in a nearby row but distant column could be labeled as a member of that object. This effect is best observed in Figure 4.3.

To help eliminate this problem, I decided to use position as a stronger weighting factor of the joint products produced for each region when classifying a pixel. For each frame, a bounding box is computed for the training pixels of each region. After the joint products have been computed for each region, the standard deviation of these values is found and each value is then decreased by the product of the standard deviation and the ratio of the pixel's distance to the nearest edge of the corresponding region over the distance between one edge of the frame to the other. This method does not produce any sharp "bounding box" effects in segmentation masks and instead gradually decreases the probability of pixels belonging to a

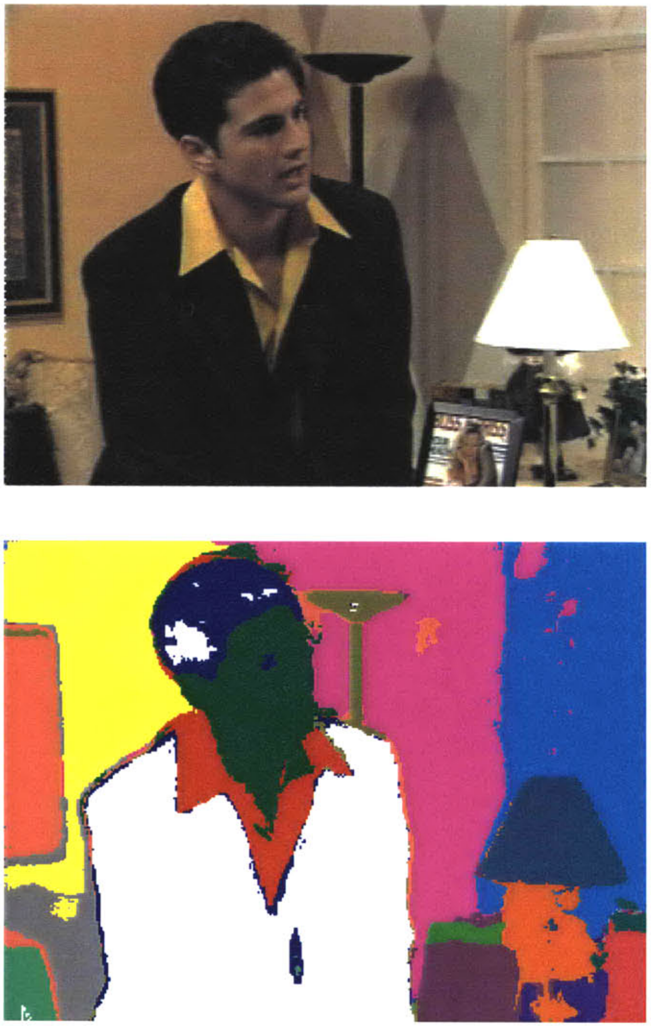


Figure 4.3: A frame from *HyperSoap*, and a mask generated without position weighting.

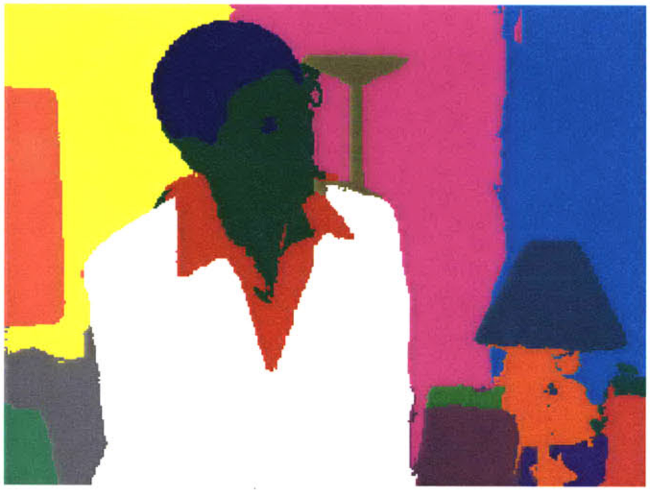


Figure 4.4: A segmentation mask for the same frame in Figure 4.3, using position weighting.

region as pixels increasingly distant from that region are examined, as shown in Figure 4.4.

4.5 Postprocessing

All of the pixels in the sequence are now associated with regions as defined by the author. Because only statistical methods have been employed, small aberrations will appear in otherwise consistent classification, such as a small group of mislabeled pixels surrounded by properly classified pixels. A postprocessing stage is employed to help rectify these anomalies.

To help remove small groups of misclassified pixels, HyperActive applies a process which "unclassifies" each pixel which was originally classified with a certainty below a set threshold. A second algorithm, known as "K-nearest neighbor" (KNN), assigns each remaining unlabeled pixel to the region which is most popular amongst its neighboring pixels.

The last step in the authoring process is to link the objects in the video with the actions that should be taken when the user selects them. Selecting an object might cause a graphic overlay to appear containing a description of the object, or cause a new video clip to be displayed. Linking objects with actions involves creating a system for associating each object with some kind of data, such as a URL, a procedure, an image, or a structure containing several separate pieces of data. In HyperActive, this is performed by the properties management component, described in full detail in the next chapter.

Chapter 5

HyperActive: Properties Management

One must manage a significant amount of data when authoring hyperlinked video. An earlier version of HyperActive that was used in the creation of *HyperSoap* (discussed in Chapter 6) required the author to keep track of all data pertaining to the hyperlinked video application themselves. This includes the video frames, the feature information for the frames, the associated segmentation masks, semantic descriptions for each of the objects present and the actions to be taken should those objects be selected. The segmentation system represented the objects segmented in each sequence by a set of "local" values. It was entirely up to the author to maintain a list of the identities of objects that appear throughout the production and to manually reassign the local values to global "object ID numbers" that could then be used to retrieve information about the object from a database. While this may not be too demanding for a 5-second video clip with only a few regions, *HyperSoap* was almost four minutes in duration, consisted of 6105 frames in 45 shots, and had a total of 50 distinctively labeled regions. Considering that many shots had more than 20 identified regions within a frame, associating information with each region in each shot of video was a painstaking process.

After using HyperActive to create *HyperSoap*, I designed a novel "properties manager" component that would assist the hyperlinked video author in both managing information and identifying regions in segmentation masks to correspond to global identities. To better manage data, the system creates an accessible list of object information, including semantic representations, which updates with the identification of new objects. Considering that objects appear in more than one shot in many video sequences, the system makes guesses about the identification of objects in a newly segmented sequence. If it guesses the object correctly, the author does not need to manually search a database of object information to make the association -- HyperActive has done the work.

5.1 Representing objects in a database

The founding idea behind the properties management system is that the probability density functions (PDF's) that HyperActive has already generated to model objects' features could be utilized not only for segmentation, but also for establishing and searching a database of objects. These PDF's are already in a compact form of no more than twelve floating-point values (mean, standard deviation and weight for a maximum of four Gaussians) for each set of feature information for each object. Additionally, their use in the classification algorithm proves that they can aid in discriminating each object effectively.

One of the first steps in creating the properties management system was to determine how to represent these descriptors in a manner that would be robust over changes between shots. Texture information is sensitive to scale, as an object appearing in a "close-up" frame would possess different texture characteristics than its appearance in a "medium shot." Similarly, motion and spatial position are useful metrics for identifying regions within temporally adjacent frames but lose their significance when comparing shots, such as ones featuring the same object at different camera angles. For these reasons, color feature information was chosen to represent objects, in particular the PDF's generated from YIQ data. This feature space was chosen above spaces like RGB so that more lenience could be given to luminance information than chrominance information during searches. This would help account for cameras' sensitivity to shadowing and changes in light.

5.2 Incorporating multiple object views

One major issue when creating the system was the need to create an effective method for consolidating an object's feature information -- gathered from multiple occurrences -- into a single descriptor that could be used for search purposes. The first instance when this issue arises is during the segmentation phase: what is the best method to represent the color information for an object that appears in many frames of a particular shot? This is crucial considering that the segmentation of a shot could be run such that YIQ PDF's are computed for each object in each frame. It seems intuitive that the best choice of frames would be ones for which training information is provided, as the pixels used to form the PDF's for all other frames could include ones that had been improperly tracked.

The next time this issue is pertinent is when determining the best descriptor to represent an object in a database such that it is always selected when appropriate during a search. This is because objects will most likely appear in multiple shots. One solution could be to simply keep in the database the PDF's generated from the first frame in the first shot in which an object appeared. Another could be keeping the PDF's generated from the frame in which the most training pixels were provided for that object.

Unfortunately, these solutions do not take into account the possible disparities between an object's appearance in more than one shot. As another example, the color information for an object could vary significantly between shots. This could be due to the effects of a camera's automatic gain control or actual changes in lighting. For example, the system would most likely fail to recognize an object if it was set in a shadow if the only PDF's that it retained for that object were computed from color information from a shot in which the object were in bright light. Fortunately, most of these changes will appear most noticeably in only the Y channel of our YIQ color information. However, an object's appearance might differ between shots in other ways, such as an object shown from different camera angles. For example, in one shot an actor's face may only appear from the side; in this case it will appear as mostly skin with a few small dark regions where his features are. But in another shot, where his face is seen directly from the front, there will be many other colored areas within his face, such as his lips, the edges of his nose and his eyes and eyebrows.

These problems might suggest another possible solution: replace the PDF's retained for an object when that object is not properly "found" during a search. This solution could be effective, but would certainly break down for instances where an object's color change is gradual, changing slightly from shot to shot. In this case, it is possible that the object will not be chosen every time a search is conducted, despite the fact that its PDF's are updated each time.

One promising solution is to keep **all** of the YIQ PDF's computed for an object in each shot and to develop a method that creates a "virtual" or composite PDF from these PDF's to use in searches. A method described by Vasconcelos and Lippman [Vasconcelos98] effectively achieve this; it utilizes the Expectation Maximization algorithm for both matching a query PDF to a set of PDF's and creating a PDF "hierarchy" for effectively representing multiple views of the same object in an object database. Note that this same approach could also be applied to create a "virtual sample" of an object in a shot, as derived from the PDF's created from multiple training frames. This would form a third layer in the hierarchy, as shown in Figure 5.1.

The hierarchy consists of 3 levels of PDF's. At the lowest level are the PDF's calculated for an object from training frames. These PDF's are utilized to calculate a "virtual PDF" which is at the second level and represents a composite of an object's appearances in a single shot. At the highest level are virtual PDF's formed by several Level 2 virtual PDF's; these PDF's are the ones which are utilized in queries.

In HyperActive, it is possible for a single object to possess more than one virtual representation in searches. This was implemented to take into account scenarios where an object's color properties vary wildly between two shots – for example, a white shirt worn in a smoky lounge with bluish lighting and that same shirt worn outside in bright daylight. Rather than form a single composite of these two incarnations of the same object, which could very well fail to represent both at once, two virtual samples could be generated where each would be a strong candidate for respective future queries.

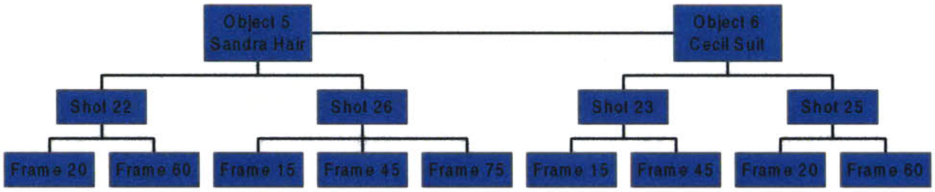


Figure 5.1: 3-Level Hierarchical representation of PDF's retained for two objects in *HyperSoap*.

5.3 Pre-searching: the need for speed

The "guesses" that HyperActive makes consider speed -- an author would most likely find typing the name of an object on a keyboard to be more efficient than having to wait while the system does an exhaustive search. One way to help limit the time spent during a search is to quickly eliminate objects whose color information differs by a significant amount. To implement this, a separate object database was created that consists of three sets of "bins," one set for each color channel. Each bin represents a single integer value of a color channel. When an object is identified for the first time, a number is computed by summing the product of each Gaussian's mean and weight for each channel's PDF. This number is an index for the channel's bin in which a marker for that object will be stored. Thus, a marker is stored in three sets of bins.

When a search is conducted for a match for a particular object, the first step consists of computing three of these weighted-mean values for that object. In the next step, HyperActive uses these three numbers to create a list of potential matches by looking into bins that are within a certain range around those three values (a range of 8 increments for luminance, and 4 increments for I and Q). It is these values which will ultimately be used in the PDF-based search. In the present system, no rules were set to eliminate those objects which did not have markers in all three color channel bins within a given search object's range. This was to allow an additional amount of flexibility to make sure only bad candidates are eliminated from the search, but all good candidates remain.

This additional "pre-search" step has the benefit of not only reducing the search space, but also expediting cases where a new object looks vastly different from all previously identified objects. If no objects have markers in each color channel's set of bins within a search range around a new object's weighted-mean values, then it is quite likely that the object is new to the database, and therefore the PDF comparison step is not necessary. Similarly, if there are only a very small number of objects whose weighted-mean values are within range of a new object's values, these objects can be displayed to the user as potential candidates without having to perform a PDF comparison. In trials conducted to gauge the accuracy of the properties manager (described in Section 5.6), the "pre-search" step often eliminated more than half of the objects in the database before applying the more resource-intensive EM query.

5.4 The object occurrence matrix

One drawback to representing objects using feature data is that many disparate objects possess similar colors and textures; for example, a woman's neck and her arm, if similarly lit, will likely yield very similar visual information. This problem could be resolved by introducing a metric for comparing objects' shapes; however, this is quite difficult considering that many objects in real video have complex shapes and constantly change their shapes throughout a sequence. Yet another problem arises when distinguishing objects which have similar feature information (including shape) but are different conceptual entities, such as two actors' hands. It is obvious that some high-level knowledge is necessary for a computer system to distinguish between such entities.

A method is needed to aid searches without causing them to depend on a pre-established knowledge base that would contain somewhat complex relationships between objects, such as "a person's eyes are most often shown above their mouth but between their ears," etc. To this end, an "object occurrence matrix" was devised. This is used to keep track of which objects appear in frames with all other objects, and in how many frames each object appears. Each time a set of objects is handed to the properties manager, entries in this database are updated accordingly. This information is used to better distinguish a large set of objects that are ranked as being similar to a search object. As the number of objects classified in a shot increases, more information can be gleaned from this "occurrence matrix" about which other objects have often appeared within shots containing these already-classified objects. This information can be used to help choose among the candidates with close likelihoods so that objects which have appeared in shots with the already-classified objects are favored. To see what this matrix looks like, please refer to the contents of the "object occurrence" matrices generated during the trials detailed in the Appendix.

The object occurrence matrix is not employed during object queries for the first two shots that are authored during a session with HyperActive. This is because it is created after the objects in the first shot are identified. If there are N objects in the first shot, the matrix is NxN and contains a "1" in every field to indicate that each object in the shot appears once with all of the other objects (its rows and columns will grow as new objects are added to the database). It does not make sense to employ the object occurrence matrix during the classification of objects in the second shot as all candidate objects have appeared once with all other candidate objects, so nothing distinguishes them in that manner.

Beginning with the third shot, the object occurrence matrix is consulted during each object query, except for the first. During all subsequent queries, the iterative EM algorithm is employed using color information to determine the likelihood that each object (referred to as a "candidate object") is the query object. Then, the entries in the object occurrence matrix are consulted for all of the objects that have already been identified in the shot (referred to as "labeled objects"). For each candidate object, a set of ratios are computed: the number of shots that the candidate object appeared in the same shot as a particular labeled object, divided by the total number of shots in which the candidate object has already appeared. For each candidate object, these ratios are computed for every labeled object; they are then summed and a mean ratio value is found. If the ratio is less than 0.5, the candidate object is eliminated from the query. As one might imagine, this process becomes more refined as the number of labeled objects increases during the classification of a shot.

5.5 A complete properties management session

During the segmentation process, the system stores each object's YIQ PDF values in a lookup table, as well as the number of the frame in which the object is largest, and the count of those training pixels. Whenever the user provides training data for objects in a particular frame, the system stores the PDF's generated for the information in that frame, and compares the number of training pixels with the number contained in the lookup table. If it has a greater square area in the new frame, the new frame number and area is stored – this is so that the frame in which the object is most visible is shown to the user at the point when he or she will have to provide a description to the properties manager. After the shot has been segmented, the Hierarchical Expectation Maximization algorithm is applied to each object's set of PDF's to create a Level 2 PDF of each object which, as previously mentioned, is a composite set of PDF's that ideally incorporates information from each of the training frames.

When an author hyperlinks the first shot in a sequence, the databases are created and each object's Level 2 PDF's are directly added. No search function is needed at this stage; the database is empty and it is assumed that the author labels duplicate objects to be a single object. A representative frame in which the object appears is shown to the user, and he or she is prompted to provide a category name and description. A weighted-mean value is then computed for each color channel PDF, and an object ID number is added into the corresponding bins. A data structure of the same ID number is created in a database of "Level 1"virtual object PDF's. In this entry, the virtual PDF's formed by all Level 3 training frame PDF's in this first shot are stored. A copy of these Level 2 PDF's is also stored in a data structure in another database that will store all Level 2 PDF's generated for each object. The Level 3 PDF's are not stored. Information is also written into a database indexed by description number, as well as a database indexed by category number. Finally, the occurrence matrix is initialized: a "1" is written into every entry.

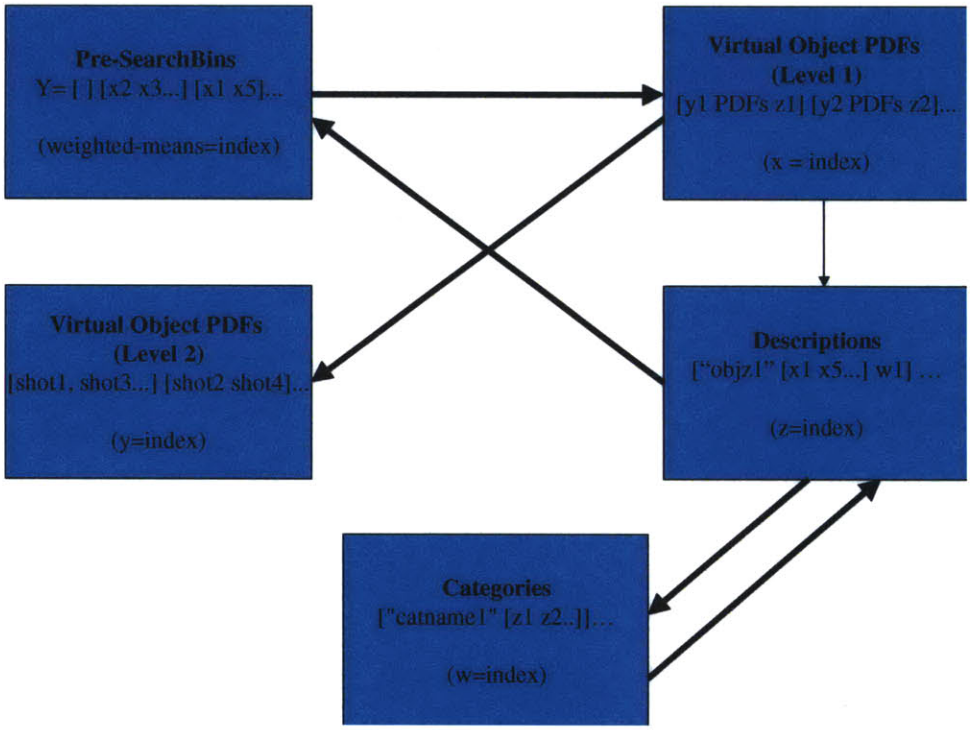


Figure 5.2: The five data structures which comprise the properties manager, and their interconnection.

After the second shot is segmented -- and YIQ PDF's have been computed for the objects in this shot -- the properties manager is summoned again. For each object in the second shot, the weighted sum of the means of the PDF's for each color channel is computed. These values are then used to index into the bins that contain pointers to objects with similar color characteristics. After this list of similar objects has been compiled, objects that have already been identified within this shot are eliminated from the list. If there are four or less objects remaining in the list, images of these objects are displayed to the user and he or she is asked if one of these match the query object. If there are more than four, the iterative EM process is employed using the query object's Level 2 PDF and the Level 1 PDF's of the candidate objects. Images of the four most likely objects are displayed to the user.

If one of these four candidates is a match, the Level 2 PDF is stored in the appropriate database, and a new Level 1 PDF is computed for the object that takes into account the new PDF. If none of them match, the user is asked to pick from the list of objects in the database (ordered by category). If the object was in the database (implying the properties manager erred), either its Level 1 PDF is updated or a new Level 1 PDF (and entry into the bins) is generated, depending on whether or not the object was found in a neighboring bin during the pre-search. If the object has not been classified in a previous shot, new entries for the object are created in all of the data structures.

For all subsequent shots, this properties management process is repeated, only with the added inclusion of information from the object occurrence matrix, as previously described. After the objects for each shot have been classified with the properties manager, the values stored in the shot's segmentation masks are adjusted based on the "global number" of each object in the database. When the author has finished adding links to all of the shots in the sequence, he or she will then have a set of segmentation mask frames, a database that has a description and category for each object, databases containing the Level 1 and Level 2 PDF's for each object, and the object occurrence matrix which shows how many times each object has appeared with every other object in the sequence.

5.6 Testing the properties manager

The properties management system was put through several trials to test its accuracy and observe its behavior. The results are described in the following chapter, and the full details of the trial are contained in the Appendix.

As a means of demonstrating the performance of HyperActive's properties management, HyperActive was utilized to hyperlink 6 consecutive shots in the HyperSoap video (keyframes shown in the Appendix). Each shot contained a number of objects that ranged from 9 to 21 with an average of 12 objects per shot. There were a total of 72 objects in the 6 shots, where 33 were an object's first appearance and 39 were repeat appearances.

The series of shots were selected for several reasons. It was chosen because objects consistently reappear among the shots. The shots contained different sets of objects, with some overlap – they show a dialogue between a man and a woman who are standing on two sides of the same room but never appear together in the same shot. Some sets of objects possess very similar color attributes. This includes the color of the two actors' faces, as well as their black clothing. Additionally, some objects reappear with different colors in the shots. For example, the dark and light wall patterns appear on both sides of the room; however, one is located in shadows while the other is near a window casting bright light. The shots also feature a wide variety of types of objects – people and their features and clothing, furniture, artwork, objects like dolls and plants, and scenery.

Of the 39 cases when an object reappeared, the properties management correctly found the object 26 times when it did not use the object occurrence matrix, and found the object 33 times when it did use it. This represented an accuracy of 67% in the former case, and 85% in the latter. In all cases, the pre-search feature eliminated more than half of the objects in the database before doing any further analysis, in some cases it quickly eliminated more than ¾ of the database.

As an example, let's examine the query for Object 42, the actor's shirt, the third object in the fourth shot. At this point, HyperActive has already labeled the actor's hair and face, which appeared with the actor's shirt in the second shot. There are 28 objects that have already been classified and are in the database.

Category 1 – Sandra
Object 1 : Hair
Object 2 : Face
Object 3 : Earring
Object 4 : Suit
Object 5 : Hands and Arms
Object 29 : Necklace
Object 31 : Blouse

Category 2 – Desk Items
Object 6 : Kleenex
Object 20 : B&P Photo
Object 21 : Desk Lamp

Category 3 – Scenery
Object 7 : Window Panes
Object 8 : Light Wall Pattern
Object 9 : Dark Wall Pattern and Window Bars
Object 14 : Orange Wall
Object 15 : Side Window

Category 4 – Cecil
Object 10 : Jacket
Object 11 : Shirt
Object 12 : Face
Object 13 : Hair
Object 27 : Hands
Object 30 : Watch

Category 5 – Furniture
Object 16 : Couch
Object 18 : Brown Pillow
Object 19 : Halogen Lamp
Object 28 : Column
Object 33 : Chair

Category 6 – Artwork
Object 17 : The Quail
Object 32 : Springtime

Category 7 – Side Table Items
Object 22 : Germany Doll
Object 23 : 5x7 Photo
Object 24 : Books
Object 25 : Plant
Object 26 : Table

Figure 5.3: The full list of objects contained in the six shots used for trials.

The Pre-Search returned 7 items, thereby eliminating ¾ of the database from the search. These items are: the actress's necklace, face, earring and arms, a table, the actor's shirt and the light pattern on the walls. Without using the information in the Object Occurrence matrix, the likelihoods based on comparing only the statistical models of the objects' color information improperly yields the actress's necklace, face and arms and the table as the best candidates. However, because the actress's objects have yet to appear in the same shot as the actor's hair and face, they are eliminated from the search via information from the object occurrence matrix. Similarly, the wall pattern has only been present in one shot with the actor's hair and face, and was in two shots without them. This yields an average ratio of 1:3, and so that object is eliminated from the search as well. Thus, using the object occurrence matrix, the two candidates shown to the user are the table and the actor's shirt – a success.

To view the results of the other queries, as well as keyframes from the six shots, please view the Appendix.

5.7 Use of the database for playback

The last step in the process of adding links to video is writing the computer program or script that will interpret data to perform desired actions when objects are selected. Associating objects with data can be accomplished in a number of ways, including information attached separately to each frame of the video or a central database that can be queried during playback. In my system, a simple text file database associates each object with important data. The playback system, also written in the *Isis* programming language [Agamanolis97], includes instructions on how to read and interpret the data in the database. When a user selects an object in the video, the value of the corresponding pixel in the segmentation mask identifies the object. The playback system retrieves the data associated with that object from the database and changes the presentation of the video accordingly.

This database could be easily erected from one or more of the data structures already maintained by the properties management system. In addition to entering a text string to describe an object during the classification stage, a user could also provide information like the location on a computer network of a media file, or more descriptive text, or an indication of another action that is facilitated by a playback script.

As a final note, HyperActive was developed using the C and Isis languages and was run on a Compaq Alpha EV56 Workstation with a 500MHz processor. It takes between 1-5 seconds to track training data, compute feature data, generate PDF's and classify mask pixels for each 320x240 video frame. The variance in time accounts for different combinations of features and quantities of training data. The time for each object query via the properties management system is approximately 1 second.

Chapter 6

HyperSoap

HyperActive was used in the creation of *HyperSoap*, a four-minute hyperlinked video drama created in April 1998 and closely patterned after daytime television soap operas. The user watches this show on a large projection screen with the knowledge that everything in the scene is for sale, including clothes, props and furnishings. Instead of using a mouse, the user selects objects with a laser pointer (Figure 6.1). Pointing the laser at the screen highlights selectable objects, and keeping the pointer fixed on one object for a short period selects it (indicated by a change in color of the highlight*).

Depending on the mode of playback and the preferences of the user, the playback system for *HyperSoap* will display information about the selected object in a number of different ways. In one mode, the system waits for an appropriate point to interrupt the video, typically when an actor has finished speaking his line. It then displays a separate screen containing a detailed still image of the selected product, along with a text box that includes the product's brand name, description, and price.

* This idea was suggested by Walter Bender after seeing an early version of *HyperSoap*.

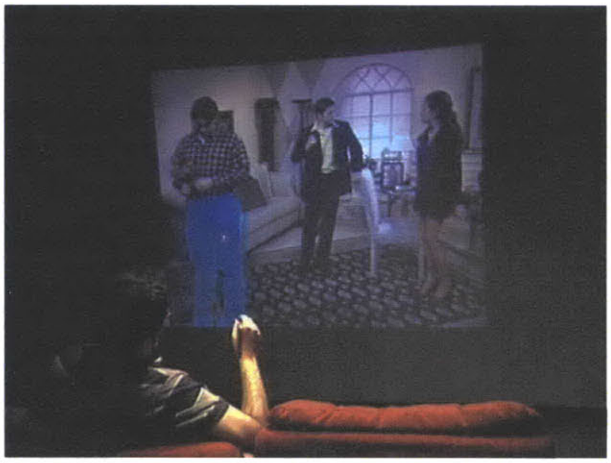


Figure 6.1: A user interacts with *HyperSoap* by aiming a laser pointer at objects in a large screen video projection.

In another mode, appropriate for a broadcast scenario or when the user desires information instantly, an abbreviated information box appears immediately, without pausing the video, and then fades away after a few seconds (Figure 6.2). If requested by the user, a summary of all the products that were selected is shown at the end of the video.

6.1 The team

Several researchers at the MIT Media Laboratory collaborated on the *HyperSoap* project. Kevin Brooks (from the Interactive Cinema group) and I wrote the script for *HyperSoap* which was meant to parody a scene from "The Young and the Restless." The details of the production were organized by myself and Michael Ponder, a liaison between the Media Laboratory and JCPenney, which provided the studio facilities for filming the content as well as the props, clothing and furnishings used on the set. Additionally, Alex Westner provided "Foley" sound effects that were later incorporated into *HyperSoap*.

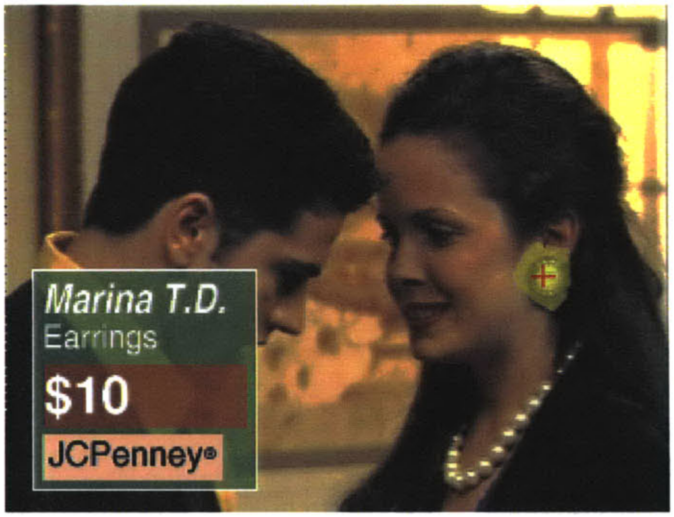


Figure 6.2: A screen grab of *HyperSoap* during playback in which an object has been selected. The crosshairs have been added to indicate where the user is aiming the laser pointer.

The person whom I collaborated with most on the project was Stefan Agamanolis, who directed the shoot at JCPenney headquarters. He also assisted me in auditioning actors, selecting props and costumes, editing the footage from the shoot, and writing the playback script in *Isis*, a language he created. Additionally, my advisor V. Michael Bove Jr. was responsible for much of the direction and concepts behind *HyperSoap*, as well as directing the funding to the project from the Media Lab's Digital Life consortium.

I also collaborated with composer and pianist Paul Nemirovsky (also from the Interactive Cinema group) to create a unique musical soundtrack for *HyperSoap* in which individual pieces, composed to match the mood of a particular part of the scene, are capable of being seamlessly looped and cross-faded. When the video is paused to display product information, the music continues to play in the background, lessening the impact of the interruption on the continuity of the video.

6.2 Technical details

HyperActive's segmentation of the 40 products appearing in *HyperSoap* was of superior enough quality to permit the use of the segmentation mask for highlighting objects. A total of 45 shots were processed through an early version of the system individually, where the number of linkable objects per shot ranged from 10 to 20. To maintain a consistently high quality of segmentation, the author needed to provide new training data to the system approximately every 30 frames, or one second of video, in order to minimize the propagation of error when estimating the location of the trained pixels in unmarked frames. However, this still provided an exceptional level of automation of the segmentation process; the author scribbled on an average of 5000 pixels in each training frame, meaning that for each 30-frame sequence the algorithm was required to classify 99.8 percent of the total pixels.

HyperSoap was in many ways a full trial of the object tracking component of HyperActive. During the authoring process, I was able to compare the results of segmenting the same shot with different combinations of feature estimation algorithms. This enabled me to determine an appropriate "default" setting of features for authoring a typical shot – spatial position, YIQ color, Krause optical flow for motion estimation, and local statistics at two resolutions (5 and 9 square pixels) in the Y channel for texture. Authoring *HyperSoap* was also the impetus to create the properties management feature; it was the result of discovering the frustration of keeping track of an object database and manually associating numbers in the masks with entries in the database.

The process of designing *HyperSoap* enabled us to explore the problems that arise when creating hyperlinked video, such as how users should interact with the video, how the presence of hyperlinks should be indicated, and how hyperlink actions should be executed. I will discuss our solutions to some of these issues in the following sections.

6.3 Venue and interaction

We felt the usual computer screen and mouse style of interaction would not be appropriate for the kind of content we were developing in *HyperSoap*. The playback system was designed with the intent of simulating a future television viewing scenario, one in which a device with significant computational ability (e.g. a digital set-top box, digital television receiver or DVD player) would be capable of mediating user interactions and modifying the presentation based on the author's design. Our prototype viewing system includes a large screen projection driven by a workstation running *Isis*, the programming language in which the playback software is written.

As discussed earlier, the user selects objects with a hand-held laser pointer. A small video camera attached to the video projector enables the playback system to sense the location of the laser dot on the projection. This is possible because the red dot generated by the laser is always brighter than the brightest possible image our projector can produce. Since the camera's image is never perfectly aligned to the projection, a coordinate transformation is applied to correct for any displacement or keystone distortion. The parameters for this transformation need to be calculated only once by a special calibration program after the position of the projector/camera is fixed relative to the screen.

6.4 Indication of hyperlinks

Typically, the method of alerting the user as to which objects are selectable in a hyperlinked video presentation is a major issue. However, in developing *HyperSoap*, we wanted every object to be selectable all the time. If the user is given this knowledge prior to watching the video, then indicating the presence of hyperlinks is not necessary. When nobody is interacting with *HyperSoap*, the video looks fairly ordinary, enabling the user to watch the presentation in a more passive manner without any distraction that might arise from indications of hyperlink opportunities. When the user selects an object with the laser pointer, the playback system utilizes the information in the segmentation mask to highlight all of the pixels that comprise the object. The highlight remains on the object for a few seconds after the laser pointer is turned off and while the corresponding product information is presented, and then fades out. We also found that subsampling the segmentation mask by a factor of two had a minimal effect on the quality of the rendered object highlights.

6.5 Initiation of hyperlink actions

From our experimentation with *HyperSoap*, we found that users have a variety of preferences regarding the timing with which product information is displayed when an object is selected. When a user selected a linked object in an early version of *HyperSoap*, the video was interrupted immediately to show the link content. After a few seconds, the link content would disappear and the video would resume from the point it was interrupted. While some users enjoyed the instant response from the system, others found it somewhat troubling that their interactions would cut off actors in the middle of delivering lines or interrupt other important moments within the program. Also, users tended to forget the context of the video content before the interruption and found themselves somewhat lost in the plot when it resumed.

Our next implementation yielded a more desirable manner of interaction: when a user selected a hyperlink, the system waited until the video reached a break in the action, typically after an actor finished speaking a line, before pausing the video and showing the link information. This yielded a more favorable reaction from users, although the delay in displaying the information caused many first-time users to think the playback mechanism was not working properly. Others who felt more interested in the link information expressed that they would have preferred instant feedback.

In the latest implementation, there are three modes from which to choose. In the first, link information is displayed at breakpoints as described above, but with the addition that a selected object remains highlighted until the breakpoint is reached in order to indicate to the user that the system hasn't "forgotten" about it. In the second mode, abbreviated product information is displayed immediately in a graphic overlay when an object is selected, without pausing the video. In the last mode, no information about selected products is displayed until the very end of the show.

6.6 Production design

HyperSoap raises several interesting issues regarding the differences between traditional television programming and hyperlinked video content. A traditional television soap opera is designed in such a way that simply adding hyperlinks and allowing interruptions would likely detract from its effectiveness because they would obstruct the flow of the story.

In designing the content for *HyperSoap*, we had to move away from a format where the story is central and any product placement or user interaction are decorations. We needed to create a new format where all of these components are equally important and dependent on each other for the survival of the presentation. The knowledge that everything would be for sale and that users would be interacting was part of the program's design from the earliest stages of planning. This entailed several changes in the way the scene was scripted, shot, and edited when compared to a traditional TV program.

For example, shots were designed to emphasize the products for sale while not diverting too much attention from the flow of the story. Certain actions are written into the script in order to provide natural opportunities for close-ups of products which would have otherwise been difficult to see or select with our interaction device. For example, an actress crying and reaching for a tissue provided the opportunity to feature a close-up of the tissue box and objects nearby. Similarly, the scene is edited so that shots are either long enough or returned to often enough for users to spot products and decide to select them. The story itself involves the products in ways that might increase their value to the consumer. Integrating the story with the products makes interruptions to show product information less jarring overall. The knowledge that everything is for sale and that the program was designed for the dual purpose of entertainment and shopping motivates the user to "stay tuned" and to interact with the products.

In summary, creating *HyperSoap* was an incredible learning experience for understanding both the technical and aesthetic issues for designing hyperlinked video systems. It not only enabled me to refine HyperActive, it alerted me to the need for an automated properties management system. It also helped me to prepare for implementing future hyperlinked video presentations.

Chapter 7

An Interactive Dinner At Julia's

After continuing to develop both HyperActive and *HyperSoap*, it was time to create another hyperlinked video application. This application would ideally focus on issues -- related to both design and content -- which were not applicable to HyperSoap. While HyperSoap was met with much praise, it did have critics who scoffed at its crass commercialism. An application built with the same tools but with an educational purpose would help showcase the merits of object-based hyperlinked video to such pundits. Additionally, it seemed a challenging task to design a hyperlinked video program which featured links to and from video clips as well as text, like those in *The Interactive Kon-Tiki Museum* and *HyperCafe*. It was for these reasons that *An Interactive Dinner At Julia's* was created in the Spring of 1999.

Produced with the help of Boston PBS-affiliate WGBH-TV, *An Interactive Dinner At Julia's* enables users to navigate a "web" of video clips featuring Julia Child. Starting with a dinner party at Julia's house, users can "click" on entrees and decorative items at the dinner table and be shown video clips in which Julia creates them. For example, when guests are shown putting their forks into a plateful of poached sa

while a new video clip showing Julia preparing the salmon fades in. Video clips of Julia preparing dishes also feature links to new clips, thereby creating the multi-level "web" of video.

A new interface was created which features an icon containing a still frame from the clip the user has left to follow a new clip (see Figure 7.1). If the user follows a video link within the new clip, another icon is added, as well as an arrow that is used to show the relationship between the two clips. If the user decides to go back to one of the previous clips, he or she can "click" on one of these icons, causing the new clip to fade out while the selected clip fades in. If the user lets the new clip play to the end, the system automatically returns the user to the previous clip. In both cases, the previous clip begins prior to the point in time when the user selected the video link. This not only helps to re-establish the context of the clip, it also enables the user to select links concurrent to the video link he or she had previously selected.

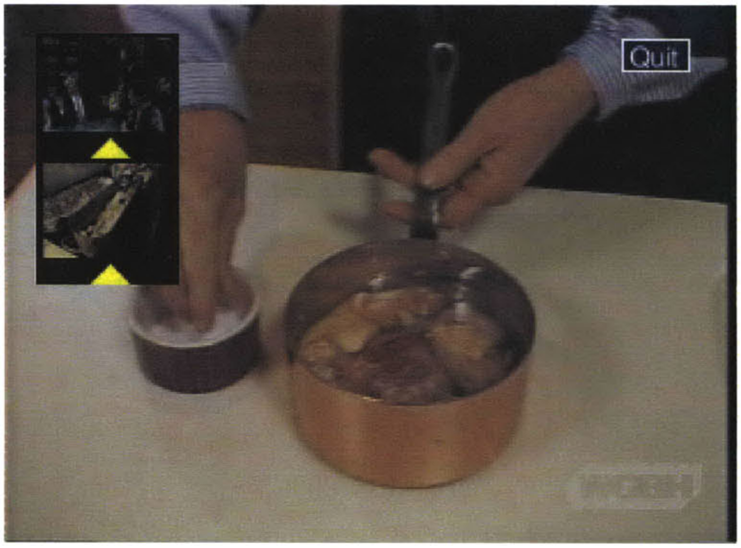
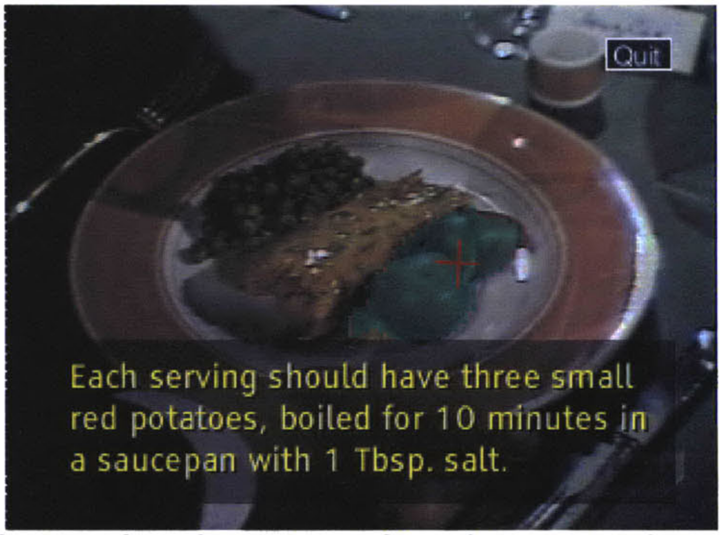


Figure 7.1: The user has descended three levels in the video web to the chicken stock clip.

Additionally, selecting certain ingredients and cooking utensils generates text boxes with relevant details, similar to the interaction featured in *HyperSoap*. This includes textboxes containing full recipes (these are updated each time Julia adds a new ingredient to a mixture) as well as useful cooking tips or trivia. For example, as a waiter presents a guest with his entree, the user can click on the potato side dish to learn how it was prepared (see Figure 7.2). Content from Julia Child's television shows and her numerous cookbooks were used as resources for text information [Child97].

7.1 The team

As will be discussed in Section 7.5, the video clips used in *An Interactive Dinner At Julia's* had already been filmed and edited for broadcast, though I was required to re-edit some of the clips to limit the duration of the demonstration. Annie Valva at WGBH-TV was instrumental in formulating the look of *An Interactive Dinner at Julia's*, including the creation of the fonts, colors and layout of the textboxes and closing credits. Annie and David Scutvick aided me in perusing Julia Child's extensive television archives, and together with Jeff Garmel facilitated the necessary blessings from WGBH-TV and Julia Child to proceed with the project.



Each serving should have three small red potatoes, boiled for 10 minutes in a saucepan with 1 Tbsp. salt.

Figure 7.2: The user selects the potatoes and an information text box appears. Yum.

Marina Zhurakhinskaya assisted me in digitizing, editing and hyperlinking the video footage, and also assisted me in the development of the playback script. Stefan Agamanolis once again volunteered his *Isis* expertise on many occasions. V. Michael Bove Jr. provided his expert guidance and assisted me in the design of the program's interface.

7.2 Technical details

Unlike *HyperSoap*, which featured a single video clip, *An Interactive Dinner At Julia's* incorporates ten video clips, each ranging from one to two minutes in duration. Because *An Interactive Dinner At Julia's* enables the user to request (and be shown) alternate video clips at various times, the duration of the program is entirely dependent on the user's level of engagement (or curiosity). Therefore, the program can be as short as two minutes or as long as twelve minutes, if the user explores all possible video links. There were a total of 5405 frames with hyperlinks.

Six of the video clips were "end nodes" in the video web, meaning that they did not feature any links to new video clips. These clips also did not feature objects whose information text boxes would pertain to the context of the program. For example, one such clip featured Julia on a boat fishing for the salmon that was cooked and served at the dinner party. While objects in this clip could have been hyperlinked -- such as Julia's raincoat and the boat's captain – the information would not have been relevant to the subject of cooking. Therefore, only four of the video clips possessed text and video links. The full web is shown in Figure 7.3.

Each hyperlinked video clip is associated with both a data file containing segmentation mask data and a lookup database file containing object information. Each entry in the database contains the name of the object and a single bit representing either "text" or "video." If the object is designated as "text," there is a corresponding entry with the object's text information (or the location of an image file with the information box pre-rendered). If the object is designated as "video," there is an entry containing a number representing the clip's index within an array. This method was chosen because there are actually several sets of data for each video file, hence a single number is used to index information in multiple arrays. The databases contained a number of entries that ranged from 8 to 18; there were a total of 57 unique linked objects in the entire presentation.

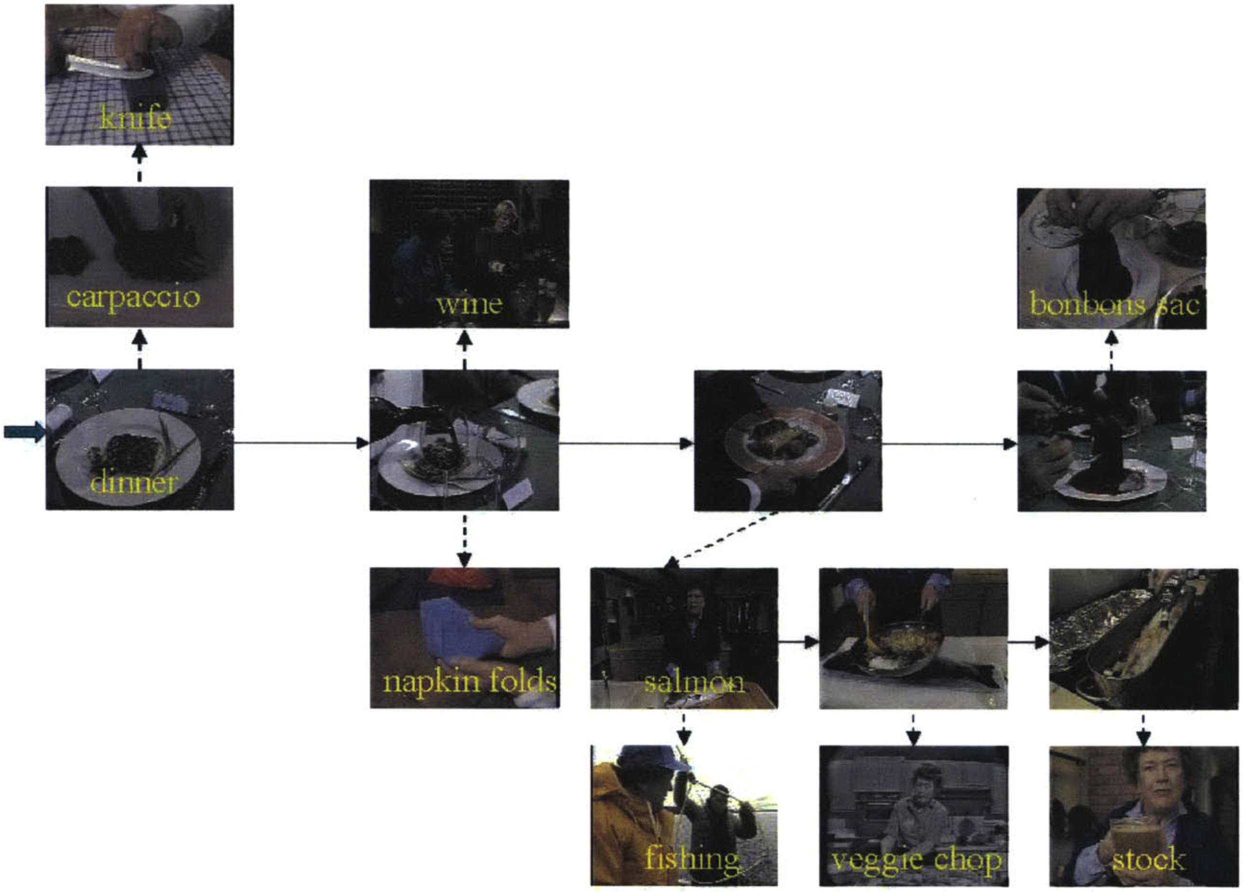


Figure 7.3: The complete "web" of video. Solid arrows indicate shots from the same clip, dashed arrows represent links. The blue arrow indicates the starting point.

7.3 Venue and interaction

The venue for interacting with *An Interactive Dinner at Julia's* is largely the same as the venue for interacting with *HyperSoap*, discussed in more detail in the previous chapter. A workstation running an *Isis* script controls the playback. Attached to the workstation are a video projector and a camera. The video is projected onto a screen, and the user aims a laser pointer at objects on the screen to select them. Before the script displays each frame of video via the projector, it grabs an image of the screen via the camera and analyzes it to find the location of a laser dot. After it has calculated its position, it first determines whether the user has selected one of the video icons. If the user has done so, the current clip fades out and the selected clip begins to play. If an icon was not selected, the script looks up the frame number and laser dot coordinates in the corresponding segmentation mask file to determine the "object number" for that pixel. This number is then used to index into the database file, and the corresponding action is executed.

This viewing scenario was meant to simulate the behavior of a person at home, watching television on their sofa and interacting with it via a remote control. The mechanism for delivering the video could be a DVD or DVD-ROM player. It could also be facilitated over a network via a broadband connection, such as xDSL or a cable modem.

Both *HyperSoap* and *An Interactive Dinner At Julia's* were also redeveloped for use with a "touch screen" where a user can tap objects with their finger to follow links (A photo of a user engaging with *An Interactive Dinner At Julia's* on the touch screen is shown in Figure 7.4). This setup would be ideal for a museum installation or, as in the case of *HyperSoap,* as a kiosk in a department store or shopping mall. This required a modification of the interaction interface, the details of which will be explained in the following section.

7.4 Indication of links

As in *HyperSoap*, the segmentation masks are utilized in *An Interactive Dinner at Julia's* to inform the user that he or she has selected a linked object. Moving the laser over a linked object causes its segmentation mask to be superimposed over the object in white. Keeping the pointer over the object for a second or more causes it to become selected, and the highlight changes to a blue color. The motivation behind this interface is to keep the video as close to "television" as possible so that the user does not become distracted by the presence of links if he or she chooses to be a passive viewer. However, if the user does wish to take an active role, he or she will be rewarded with indications of link opportunities.

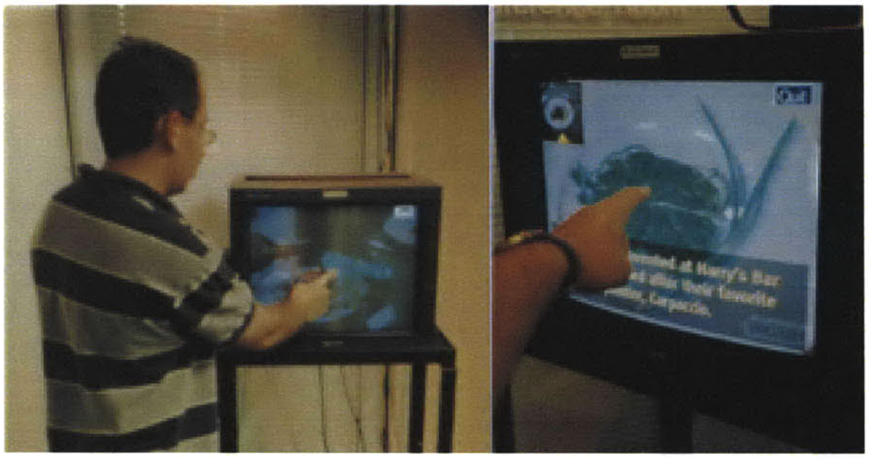


Figure 7.4: A handsome user engaging with *An Interactive Dinner At Julia's* via the touchscreen. Who is that guy and does he always wear that shirt?

One major difference between *HyperSoap* and *An Interactive Dinner At Julia's* is that the former offers only one sort of action associated with a video object (i.e. video->text) while the latter offers two types of actions (video->text and video->video). This comparison is important to note because there is a difference of "severity" between the outcomes of choosing a text link versus a video link. A video link will derail the present video clip to start a new one, while a textual link will merely display an overlay while the clip continues to play. Since there is this disparity, it would be fair to the user to somehow differentiate between these link types when he or she browses. For this reason, one version *of An Interactive Dinner At Julia's* illuminated objects with video links in yellow and text links in white, so that the user can better choose whether he or she wants to temporarily stop the video that is presently being watched in favor of a new clip. To additionally avoid unintentional derailment of the present clip, the amount of time required to lapse when a user points at an object containing a video link so that it is selected was extended by a full second. One alternative solution would be to remember all of the video links selected during a particular clip, and then display their clips to the user in sequence after the original clip has finished.

Unfortunately, the frequency and abundance of link opportunities differs significantly between *HyperSoap* and *An Interactive Dinner At Julia's*. Whereas the user can assume that all of the objects he or she sees in *HyperSoap* contain links, in *An Interactive Dinner At Julia's* this is not the case. Only objects pertaining to food and cooking utensils contain links; in some shots no such objects appear on the screen, such as when Julia greets her dinner party guests at the start of the program. This implies that when only the *HyperSoap* method of indicating links is utilized, there will be instances in a program like *An Interactive Dinner At Julia's* where the user will sweep the screen with a pointing device and find nothing. This phenomenon would likely have a negative effect on a user's desire to interact with the program, as it is essentially punishing them, instead of rewarding them, for wanting to be active instead of passive. A similar problem arose when redesigning both applications for the touchscreen interface. The method of sweeping the pointer around the screen to find links could not be applied as the touchsceen relies upon short bursts of electrical conductivity as a means of determining the location of the user's finger. Thus the users can not smear their fingers around the touchscreen to browse links -- nor would they much want to, as it is not a pleasant sensation. Thus, additional methods for indicating video links are needed for this sort of application.

One possible solution is to cause all linked objects to highlight in white whenever the user points to any non-linked object. This would save the user the time of hunting around the frame for links. However, this does not adequately address the presence of shots in which there are no link opportunities for the user to follow. To solve this problem, an additional link indicator could be used which would be similar to the icon used by the *HotVideo* playback system. For example, a subtle watermark could appear on the screen whenever there are links present. Or perhaps a "traffic light" could be located somewhere off the screen (i.e. on the user's pointing device, or in another window if the video is being played back on a computer monitor) which would turn green whenever the video contains hyperlinks. Again, this indicator is not ideal, as users could find a changing traffic light or a disappearing and appearing watermark distracting if they are focusing on the video content. The remote control indicator would be the best solution as the user could easily tune it out and could also quickly incorporate it into their field of view if he or she felt the desire to interact with the program.

7.5 Production design

When creating *HyperSoap*, we had the luxury of designing the content from scratch with the knowledge that it would be interacted with in a particular way. Since *An Interactive Dinner at Julia's* featured content which had been shot at least 15 years prior (most of which was from a single episode of the PBS series "Dinner At Julia's"), we did not have this same luxury. This meant that certain shots featured objects which moved too fast for a user to select, or are only visible for a very short period of time. One example of this was a shot in which Julia pours a measuring cup full of chicken stock into a fish poacher; the cup is so small and moves so quickly that it is very difficult to position any sort of pointer over it – even when using the touchscreen. There are also shots in which no selectable objects are present, such as shots of Julia's dinner guests conversing while there are no food items or dinner utensils visible. These are some of the drawbacks that one can encounter when repurposing normal video to become object-based hyperlinked video.

However, cooking shows offer many advantages over other genres of television programming for use in a hyperlinked video program. For example, many television shows and movies feature fast edits and cross-cutting, which give the video a very high-energy and edgy feel. Cooking programs, on the other hand, tend to have only one or two cameras during a shoot and feature lengthy shots. This format gives users an adequate opportunity to think about objects he or she might want to select and enough time to position a pointer over these objects. The bright and even lighting on cooking show sets insures that objects will not be draped in shadows and therefore difficult to see and select; this also enabled HyperActive's segmentation to function well.

Additionally, a cooking show is a program that appeals to people with varying interests, such as a person who wants to learn how to cook and a chef wanting to glean new recipes from one of the masters. *An Interactive Dinner At Julia's* was created in recognition of the fact that many educational television programs, like cooking shows or fix-it-yourself shows, are geared towards viewers with a somewhat generic level of skill. Most cooking programs are for people who are more advanced than novices, but are less skilled than a gourmet chef. *An Interactive Dinner At Julia's* marries the benefits of hypermedia, the ability to allow users to delve as deep into a subject or story as they want, with an entertaining, educational and popular television genre. It exposes some of the limitations of "clickable video" and presents solutions to problems that undoubtedly continue to plague creators of hyperlinked video applications.

Chapter 8

Future Research Directions

Although HyperActive has performed well in both trials and in creating two hyperlinked video applications, there are several features of it that could be further improved and automated. I will discuss areas of both the segmentation and properties management components that could benefit from some extra attention.

8.1 Segmentation

There are many steps during the training process that could be improved in both simplicity and efficiency. The present authoring tool I have created requires the author to browse through frames in a sequence (typically an entire shot) and make choices about which frames to use for providing training data. The author must then determine which tracking strategy would be best for the objects which appear (and disappear, and change color, etc.) in the sequence. There were many occasions while segmenting sequences from *HyperSoap* and *An Interactive Dinner At Julia's* in which I needed to rethink a tracking strategy after seeing poor results from using a previous strategy. Also, each time the author chooses a frame to provide training data, he or she needs to scribble on all of the objects, even though it is quite probable that most of the training data tracked to that frame would be adequate for many regions.

The process of providing training data and choosing the strategy for tracking the data through the video could be simplified with software that would be able to guess when objects enter and exit frames. In this way, the author would only be required to provide training data only for regions that the system feels it cannot explain, such as an object that enters the frame during the sequence. It is possible that an analysis of motion vectors near the edges of frames will aid in determining this activity. One other useful technique would be to analyze the range of probabilities for which points should be classified to regions. Assume that thresholds could be set such that there could be a group of points for which the probabilities they belong to any region are all below a certain level. The system could then assume that this is the appearance of a previously untrained region.* Because the system would be aware of the behaviors of the objects in the sequence, it would devise the best tracking scheme itself and only ask the author to occasionally provide new information.

Some of these conveniences will be challenging to implement. Although many solutions for "temporal segmentation," or shot detection, have been proposed, it will be difficult to find the solution that works best in this situation. Also, it is clear that while motion vectors will provide clues about the behavior of objects, particularly those entering and exiting frames, there are many other issues to consider, such as how to handle objects which do not leave the frame but instead become occluded by other objects.

* These approaches were the result of a conversation between John Watlington and I.

8.2 Properties management

One of the areas within the properties management system that deserves to be studied further is the object occurrence matrix and how its information affects queries. Although it improved the outcome of many queries during the trials, its construction is somewhat simple, and it is possible that it would not add value – or may even negatively affect the system's performance – after a large number of shots have been put through the system.

The data structure itself could be improved by not only recording how many times each object appears with others, but also information about the combinations in which objects have appeared in previous shots. For example, knowledge that a candidate object has appeared twice with some of the already labeled objects for a shot becomes much more valuable if it is known that in each case that set of objects appeared together. This would require an entirely new structure, one in which individual objects may point to lists of combinations in which they have appeared.

The method in which the object occurrence matrix is employed to refine searches could also be developed further. The present method will compute the same "ratio" for A) a candidate object that both appeared in all shots with one of the already labeled objects and in no shots with another and B) a candidate object that appeared in a few shots with each of the already labeled objects. It is clear that these disparate situations should be treated differently. Additionally, the use of the object occurrence matrix to eliminate candidates from queries might backfire in some situations – the most likely candidate may be incorrectly eliminated. To improve this, a system could be devised in which the ratios computed with the object occurrence matrix are used to weight the likelihoods computed during the EM query. For example, this process could improve the likelihood of a candidate object whose color information varies from the query object but has appeared in more shots with already labeled objects than any of the other candidate objects. A candidate object with very similar color characteristics to the query object but has not appeared in any shots with the already labeled objects would have its likelihood decreased.

Finally, the behavior of the properties management system would doubtlessly improve if features other than color were somehow employed. As stated previously, texture information is sensitive to scale, as an object appearing in a "close-up" frame would possess different texture characteristics than its appearance in a "medium shot." One way to address this problem might be to compute texture information at a variety of resolutions for each shot, and to also record the square area of each of the objects in the shots. When querying the objects with the database, the resolution of the texture information for each candidate object could vary depending on its size relative to the query object. For example, when comparing a large query object with a small candidate object, the respective texture information from a maximal resolution and a minimal resolution could be compared. As demonstrated in the segmentation system, multiple feature information could improve accuracy when comparing objects.

8.3 Interface

Additionally, HyperActive is largely text-based and lacks a simple and intuitive graphical interface. Until now, an author using the system has had to do more typing then drawing and clicking with a mouse. This has limited the distribution of the system to areas inside and outside of the Media Laboratory, and rightfully so. Considering that much of the existing system has been implemented in *Isis*, it would be possible to incorporate an interface based on *Isis'* support for OpenGL libraries. Once such an interface has been designed, people with a variety of technical expertise, including television producers and grade school teachers, could easily use the tool.

Chapter 9

Conclusions

The system I have developed was intended to make the process of creating object-based hyperlinked video easier. To address the time-consuming problem of outlining regions in frames, it features a video segmentation system that has the capability of automatically labeling objects at pixel-level accuracy based on scribbles in only a few frames. That this process does not require an extensive database of information about specific kinds of objects (i.e. descriptions of faces, skin, etc.) means that hyperlinked video can be made from a wide number of video sources – video of a desert or jungle or a crowded New York subway. It does not require the generation of graphical models of locations shown in the video, such as the ones employed by the *Aspen Movie Map* and *HyperPlant* projects. Because of its high precision, it can enable the association of links with a greater number of objects per frame than supported by existing hyperlinked video authoring systems like Veon's *V-Active* and IBM's *Hot Video.* Its automated properties management system addresses the time-consuming process of associating labels with objects; HyperActive is believed to be the first hyperlinked video authoring tool with this feature.

The two hyperlinked video applications I have developed, *HyperSoap* and *An Interactive Dinner at Julia's,* explore many of the issues that arise when developing a hyperlinked video system. *HyperSoap* demonstrates a highly commercial application of hyperlinked video and its "invisible" interface (clickable objects are present but are not revealed unless the user indicates an interest in interacting) is an attempt to address the omnipresent and often irritating graphics and occlusions which appear in other interactive video applications. Additionally, a unique musical soundtrack that can be seamlessly looped was developed to better marry the program and its text interruptions and make the interruptions less jarring to users. *An Interactive Dinner At Julia's* marries the benefits of hypermedia, the ability to allow users to delve as deep into a subject or story as they want, with an entertaining, educational and popular television genre. It exposes some of the limitations of "clickable video" – objects are not always easy to access and there are moments where none of the objects on the screen contain links. It also presents solutions to problems that have continued to plague creators of hyperlinked video applications. For example, its interface for allowing the user to see the clips through which he or she has navigated addresses the concern that a user may feel lost when exploring a "web" of video clips. Hopefully, HyperActive and the experiences of designing *HyperSoap and An Interactive Dinner at Julia's* will better help content creators, designers and educators to craft even more impressive hyperlinked video applications.

References

[Agamanolis97] Stefan Agamanolis and V. Michael Bove, Jr., "Multilevel Scripting for Responsive Multimedia" IEEE Multimedia, Oct. 1997: 40-49.

[Apple89] Apple Computer, Inc. "Macintosh HyperCard User's Guide," 1989.

[Bender80] Walter Bender, "Animation Via Video Disk," MS Thesis, Massachusetts Institute of Technology, 1980.

[Brøndmo91] Hans Peter Brøndmo and Glorianna Davenport, "Creating and Viewing the Elastic Charles – A Hypermedia Journal," Hypertext: State of the Art, R. Aleese and C. Green, eds., Intellect, Oxford, U.K., 1991: 43-51.

[Chalom96] Edmond Chalom and V. Michael Bove, Jr., "Segmentation of an Image Sequence Using Multidimensional Image Attributes," Proc. ICIP, September 1996.

[Chalom98] Edmond Chalom, "Statistical Image Sequence Segmentation Using Multidimensional Attributes," PhD Thesis, Massachusetts Institute of Technology, January 1998.

[Child97] Julia Child, The Way To Cook, Alfred A. Knopf, New York, 1997.

[Horn80] B.K.P. Horn and B.G. Schunck, "Determining Optical Flow," MIT Artificial Intelligence Laboratory, AI Memo #572, April 1980.

[IBM98] International Business Machines, HotVideo Web Site, http://www.software.ibm.com/net.media/solutions/hotvideo.

[Jain92] Anil K. Jain and Jianchang Mao, "Texture Classification and Segmentation Using Multiresolution Simultaneous Autoregressive Models," Pattern Recognition, Vol. 25 #2, February 1992: 173-188.

[Kanade81] Takeo Kanade and Bruce D. Lucas, "An Iterative Image Registration Technique with an Application to Stereo Vision," Proc. DARPA Image Understanding Workshop, 1981:121-130.

[Krause87] Edward A. Krause, "Motion Estimation for Frame-Rate Conversion," PhD Thesis, Massachusetts Institute of Technology, June 1987.

[Liestøl94] Gunnar Liestøl, "Aesthetic and Rhetorical Aspects of Linking Video in Hypermedia," Proc. European Conference on Hypertext, Edinburgh U.K., 1994: 217-223.

[Mitsubishi98] Mitsubishi Electric America, Inc., VisualSHOCK MOVIE Web Site, http://www.visualshock.com.

[Redner84] Richard Redner and Homer E. Walker, "Mixture Densities, Maximum Likelihood and the EM Algorithm," SIAM Review, 26(2), April 1984: 195-239.

[Sawhney97] Nitin Sawhney, David Balcom and Ian Smith, "Authoring and Navigating Video in Space and Time," IEEE Multimedia, Oct. 1997: 30-39.

[Smith96] J. R. Smith and S.-F. Chang, "Local Color and Texture Extraction in Spatial Query," IEEE Proc. Int. Conf. Image Processing, Lausanne, Switzerland, September 1996.

[Tani92] Masayuki Tani, et. al., "Object-Oriented Video: Interaction With Real-World Objects Through Live Video," Proc. CHI May 1992: 593-598.

[Therrien89] Charles W. Therrien, "Decision Estimation And Classification," John Wiley and Sons, Inc., 1989.

[Vasconcelos98] Nuno Vasconcelos and Andrew Lippman, "Learning Mixture Hierarchies," NIPS '98, Denver, Colorado, 1998.

[Veon98] Veon, V-Active Web Site, http://www.veon.com.

[Viola95] Paul A. Viola, "Alignment by Maximization of Mutual Information," TR #1548, MIT Artificial Intelligence Laboratory, June 1995.

[WebTV98] WebTV Networks, Inc., WebTV Web Site, http://www.webtv.com.

[Watlington89] John Watlington, "Video Finger," MS Thesis, Massachusetts Institute of Technology, January 1989.

APPENDIX:
Properties Management Trials Data

Summary:
Shot 1:
9 Objects
9 New Objects

Shot 2:
21 Objects
19 New Objects
0 Correct (No OOM)
2 Incorrect (No OOM)

Shot 3:
9 Objects
1 New Object
7 Correct (No OOM)
1 Incorrect (No OOM)

8 Correct (OOM)
0 Incorrect (OOM)

Shot 4:
10 Objects
1 New Object
5 Correct (without OOM)
4 Incorrect (without OOM)

7 Correct (with OOM)
2 Incorrect (with OOM)

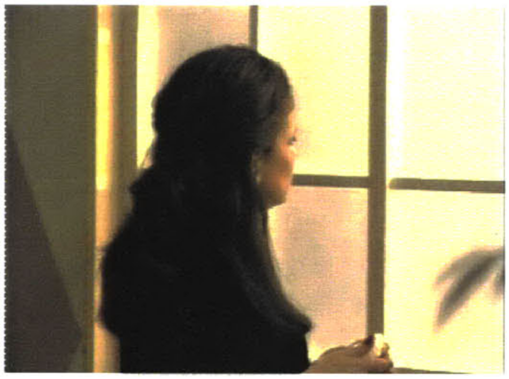
Shot 5:
10 Objects
2 New Objects
4 Correct (No OOM)
4 Incorrect (No OOM)

8 Correct (with OOM)
0 Incorrect (with OOM)

Shot 6:
13 Objects
1 New Object
10 Correct (NO OOM)
2 Incorrect (NO OOM)

10 Correct (OOM)
2 Incorrect (OOM)

SHOT 1:



1. Sandra - Hair
2. Sandra - Face
3. Sandra - Earring
4. Sandra - Suit
5. Sandra - Hands and Arms
6. Desk Items - Kleenex
7. Scenery - Window Panes
8. Scenery - Light Wall Pattern
9. Scenery - Dark Wall Pattern and Window Bars

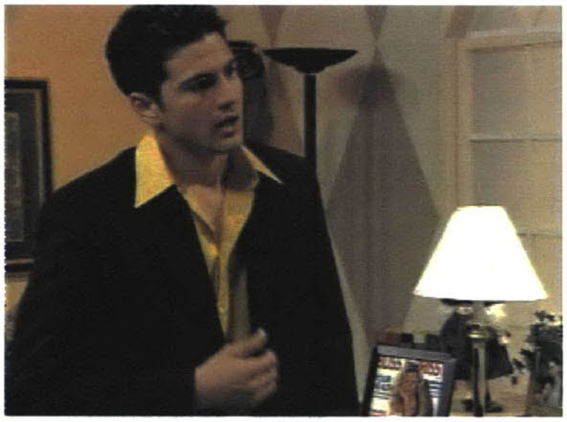
The database now has 9 objects.

Object Occurrence Matrix:

```
[[ 1 1 1 1 1 1 1 1 1 ]    Sandra - Hair
 [ 1 1 1 1 1 1 1 1 1 ]    Sandra - Face
 [ 1 1 1 1 1 1 1 1 1 ]    Sandra - Earring
 [ 1 1 1 1 1 1 1 1 1 ]    Sandra - Suit
 [ 1 1 1 1 1 1 1 1 1 ]    Sandra – Hands and Arms
 [ 1 1 1 1 1 1 1 1 1 ]    Desk Items - Kleenex
 [ 1 1 1 1 1 1 1 1 1 ]    Scenery – Window Panes
 [ 1 1 1 1 1 1 1 1 1 ]    Scenery – Light Wall Pattern
 [ 1 1 1 1 1 1 1 1 1 ]]   Scenery – Dark Wall Pattern and Window Bars
```

SHOT 2:

(Object Occurrence Matrix not employed)



10: Cecil - Jacket

Guess 1 : Sandra - Suit

OUTCOME: NEW OBJECT

The database now has 10 objects.

11: Cecil - Shirt

Guess 1 : Sandra - Face
Guess 2 : Sandra - Hands and Arms
Guess 3 : Scenery - Dark Wall Pattern and Window Bars

OUTCOME: NEW OBJECT

The database now has 11 objects.

12: Cecil - Face

Guess 1 : Sandra - Hair
Guess 2 : Sandra - Suit
Guess 3 : Desk Items - Kleenex

OUTCOME: NEW OBJECT

The database now has 12 objects.

13: Cecil - Hair

Guess 1 : Sandra - Hair
Guess 2 : Sandra - Suit

OUTCOME: NEW OBJECT

The database now has 13 objects.

14: Scenery - Orange Wall

Guess 1 : Sandra - Face
Guess 2 : Desk Items - Kleenex
Guess 3 : Scenery - Window Panes
Guess 4 : Scenery - Dark Wall Pattern and Window Bars

OUTCOME: NEW OBJECT

The database now has 14 objects.

15: Scenery - Light Wall Pattern

Pre-Search returned 6 out of 14 objects.

Guess 1 : Sandra – Face	5.573079
Guess 2 : Sandra - Hands and Arms	5.628081
Guess 3 : Scenery - Dark Wall Pattern and Window Bars	6.382852
Guess 4 : Sandra – Earring	7.209526

OUTCOME: INCORRECT

16: Scenery - Dark Wall Pattern

Pre-search returned 5 out of 14 objects.

Guess 1 : Sandra - Hands and Arms	5.511391
Guess 2 : Sandra – Face	5.948317
Guess 3 : Sandra – Earring	7.534066
Guess 4 : Desk Items – Kleenex	8.036083

OUTCOME: INCORRECT

17: Scenery - Side Window

Pre-search returned 5 out of 14 objects.

Guess 1 : Sandra – Face	5.775443
Guess 2 : Sandra – Earring	6.286242
Guess 3 : Desk Items – Kleenex	6.365490
Guess 4 : Sandra - Hands and Arms	6.724208

OUTCOME: NEW OBJECT

The database now has 15 objects.

18: Furniture - Couch

Guess 1 : Sandra - Face
Guess 2 : Sandra - Earring
Guess 3 : Sandra - Hands and Arms
Guess 4 : Scenery - Window Panes

OUTCOME: NEW OBJECT

The database now has 16 objects.

19: Artwork - "The Quail"

Pre-search returned 5 out of 16 objects.

Guess 1 : Sandra - Hands and Arms	7.442303
Guess 2 : Sandra – Face	7.584197
Guess 3 : Sandra – Earring	10.217957
Guess 4 : Desk Items – Kleenex	12.812216

OUTCOME: NEW OBJECT

The database now has 17 objects.

20: Furniture - Brown Pillow

Guess 1 : Sandra - Hair
Guess 2 : Sandra - Suit
Guess 3 : Desk Items - Kleenex

OUTCOME: NEW OBJECT

The database now has 18 objects.

21: Furniture - Halogen Lamp

Guess 1 : Sandra - Hair
Guess 2 : Sandra - Suit
Guess 3 : Desk Items - Kleenex

OUTCOME: NEW OBJECT

The database now has 19 objects.

22: Desk Items - B&P photo

Guess 1 : Sandra - Hair
Guess 2 : Sandra - Suit
Guess 3 : Sandra - Hands and Arms
Guess 4 : Scenery - Window Panes

OUTCOME: NEW OBJECT

The database now has 20 objects.

23: Desk Items - Desk Lamp

Guess 1 : Sandra - Hair
Guess 2 : Sandra - Suit
Guess 3 : Desk Items - Kleenex

OUTCOME: NEW OBJECT

The database now has 21 objects.

24: Side Table Items - Germany doll

Pre-search returned 5 out of 21 objects.

Guess 1 : Sandra – Face
Guess 2 : Sandra - Hands and Arms
Guess 3 : Sandra - Earring
Guess 4 : Desk Items - Kleenex

OUTCOME: NEW OBJECT

The database now has 22 objects.

25: Side Table Items - 5x7 Photo

Guess 1 : Sandra - Hair
Guess 2 : Scenery - Window Panes

OUTCOME: NEW OBJECT

The database now has 23 objects.

26: Side Table Items - Books

Pre-Search returned 5 out of 23 objects.

Guess 1 : Sandra - Hands and Arms
Guess 2 : Sandra - Face
Guess 3 : Sandra - Earring
Guess 4 : Desk Items - Kleenex

OUTCOME: NEW OBJECT

The database now has 24 objects.

27: Side Table Items - Plant

Guess 1 : Sandra - Hair
Guess 2 : Scenery - Window Panes

OUTCOME: NEW OBJECT

The database now has 25 objects.

28: Side Table Items - Table

Pre-search returned 5 out of 25 objects.

Guess 1 : Sandra – Face	7.428690
Guess 2 : Sandra - Hands and Arms	7.860703
Guess 3 : Sandra – Earring	9.530826
Guess 4 : Desk Items – Kleenex	11.525938

OUTCOME: NEW OBJECT

The database now has 26 objects.

29: Cecil - Hands

Guess 1 : Sandra - Suit
Guess 2 : Sandra - Hands and Arms

OUTCOME: NEW OBJECT

The database now has 27 objects.

30: Furniture - Column

Guess 1 : Sandra - Hair

OUTCOME: NEW OBJECT

The database now has 28 objects.

Summary:

19 New Objects

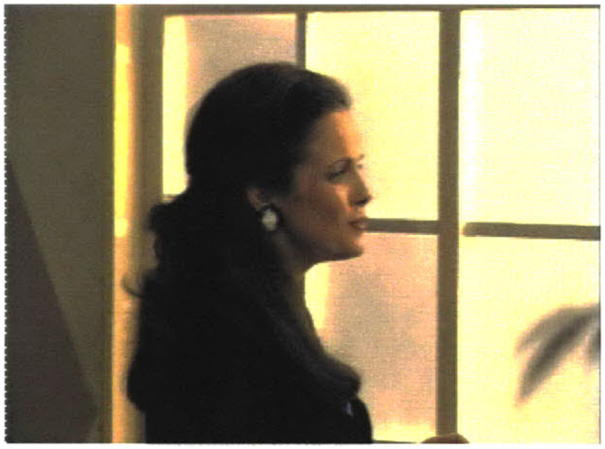
2 Incorrect

0 Correct

Object Occurrance Matrix:

```
[[1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] Sandra - Hair
 [1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] Sandra - Face
 [1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] Sandra - Earring
 [1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] Sandra - Suit
 [1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] Sandra – Hands and Arms
 [1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] Desk Items - Kleenex
 [1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] Scenery – Window Panes
 [1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Scenery – Light Wall Pattern
 [1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Scenery – Dark Wall Pattern
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Cecil - Jacket
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Cecil - Shirt
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Cecil - Face
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Cecil - Hair
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Scenery – Orange Wall
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Scenery – Side Window
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Furniture - Couch
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Artwork – The Quail
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Furniture – Brown Pillow
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Furniture – Halogen Lamp
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Desk Items – B&P Photo
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Desk Items – Desk Lamp
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Side Table Items – Germany Doll
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Side Table Items – 5x7 Photo
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Side Table Items - Books
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Side Table Items - Plant
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Side Table Items - Table
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] Cecil - Hands
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]] Furniture - Column
```

SHOT 3



31. Sandra - Hair

Pre-search returned 14 out of 28 objects.
First object in shot - can't use OO matrix.

Guess 1 : Sandra – Hair	5.616593
Guess 2 : Side Table Items – Plant	6.299464
Guess 3 : Side Table Items – Germany Doll	6.565022
Guess 4 : Furniture - Halogen Lamp	6.613008

OUTCOME: CORRECT!!!

32. Sandra - Face

Pre-search returned 12 out of 28 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Sandra – Face	6.495827	Sandra - Face:	6.495827
Guess 2 : Furniture – Couch	6.560541	Sandra - Hands and Arms:	6.613662
Guess 3 : Sandra - Hands and Arms	6.613662	Sandra - Earring:	8.127041
Guess 4 : Artwork - The Quail	6.680591	Desk Items - Kleenex:	9.335214
OUTCOME: CORRECT		OUTCOME: CORRECT	

33. Sandra - Earring

Pre-search returned 12 out of 28 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Side Table Items – Plant	5.904599	Sandra - Hands and Arms:	6.945960
Guess 2 : Artwork - The Quail	6.183932	Sandra - Earring:	8.927382
Guess 3 : Desk Items - B&P Photo	6.481832	Scenery - Window Panes:	13.692898
Guess 4 : Side Table Items – Germany Doll	6.607393	Scenery - Dark Wall Pattern and Window Bars:	14.478390

OUTCOME: INCORRECT	OUTCOME: CORRECT

34. Sandra - Suit

Pre-search returned 8 out of 28 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Sandra – Suit	3.747533	Sandra - Suit:	3.747533
Guess 2 : Cecil – Hair	3.788423	Desk Items - Kleenex:	12.590549
Guess 3 : Furniture - Brown Pillow	4.250740		
Guess 4 : Cecil – Jacket	4.284844		

OUTCOME: CORRECT	OUTCOME: CORRECT

35. Sandra - Hands and Arms

Pre-search returned 13 out of 28 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Cecil – Hands	5.425071	Sandra - Hands and Arms:	5.537581
Guess 2 : Sandra - Hands and Arms	5.537581	Desk Items - Kleenex:	7.419966
Guess 3 : Furniture – Couch	5.552218	Scenery - Dark Wall Pattern and Window Bars:	8.253524
Guess 4 : Desk Items - B&P Photo	6.128764		

OUTCOME: CORRECT	OUTCOME:CORRECT

36. Scenery - Window Panes

Pre-search returned 9 out of 28 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Desk Items - B&P Photo	7.437192	Scenery - Window Panes:	7.856150
Guess 2 : Scenery - Window Panes	7.856150	Scenery - Dark Wall Pattern and Window Bars:	21.218636
Guess 3 : Side Table Items – Table	8.575925		
Guess 4 : Furniture – Couch	9.820878		

OUTCOME: CORRECT	OUTCOME: CORRECT

37. Scenery - Light Wall Pattern

Without OO Matrix:	With OO Matrix:
Guess 1 : Scenery - Light Wall Pattern	Scenery - Dark Wall Pattern and Window Bars:
Guess 2 : Furniture - Couch	
Guess 3 : Side Table Items - Plant	
Guess 4 : Side Table Items - Table	
OUTCOME: CORRECT	OUTCOME: CORRECT

38. Scenery - Dark Wall Pattern and Window Bars

Pre-search returned 8 out of 28 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Cecil – Hands	5.486306	Scenery - Dark Wall Pattern and Window Bars:	5.746959
Guess 2 : Scenery - Dark Wall Pattern and Window Bars	5.746959		
Guess 3 : Desk Items - B&P Photo	6.159233	Desk Items - Kleenex:	6.729066
Guess 4 : Side Table Items - 5x7 Photo	6.192598		
OUTCOME: CORRECT		OUTCOME: CORRECT	

39. Sandra - Necklace

Pre-search returned 5 out of 28 objects.

Without OO Matrix:		With OO Matrix:
Guess 1 : Furniture – Couch	5.822998	
Guess 2 : Side Table Items – Table	6.374471	
Guess 3 : Side Table Items - 5x7 Photo	7.113588	
Guess 4 : Cecil – Shirt	8.428144	

OUTCOME: NEW OBJECT (OO eliminated all)

There are now 29 objects in the database.

Summary:

1 NEW OBJECT

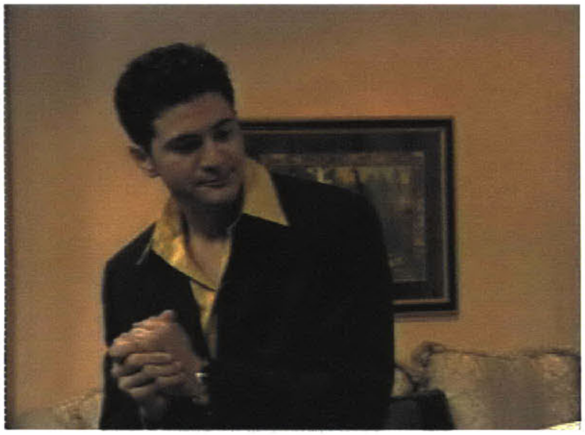
7 CORRECT (No OO) **8 CORRECT (OO)**

1 INCORRECT (No OO) **0 INCORRECT (OO)**

Object Occurrence Matrix:

```
[[2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1]     Sandra - Hair
 [2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1]     Sandra - Face
 [2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1]     Sandra - Earring
 [2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1]     Sandra - Suit
 [2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1]     Sandra – Hands and Arms
 [1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]     Desk Items - Kleenex
 [2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1]     Scenery – Window Panes
 [2 2 2 2 2 1 2 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]     Scenery – Light Wall Pattern
 [2 2 2 2 2 1 2 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]     Scenery – Dark Wall Pattern
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Cecil - Jacket
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Cecil - Shirt
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Cecil - Face
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Cecil - Hair
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Scenery – Orange Wall
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Scenery – Side Window
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Furniture - Couch
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Artwork – The Quail
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Furniture – Brown Pillow
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Furniture – Halogen Lamp
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Desk Items – B&P Photo
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Desk Items – Desk Lamp
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Side Table Items – Germany Doll
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Side Table Items – 5x7 Photo
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Side Table Items - Books
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Side Table Items - Plant
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Side Table Items - Table
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Cecil - Hands
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]     Furniture - Column
 [1 1 1 1 1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1]]    Sandra - Necklace
```

SHOT 4:



40. Cecil - Hair

Pre-search returned 10 out of 29 objects.

First object in shot - can't use OO matrix.

Guess 1 : Cecil – Hair	4.570631
Guess 2 : Cecil – Jacket	5.026062
Guess 3 : Furniture - Brown Pillow	5.457394
Guess 4 : Sandra – Suit	5.477111

OUTCOME: CORRECT

41. Cecil - Face

Pre-search returned 10 out of 29 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Cecil – Hands	5.782030	Cecil - Hands:	5.782030
Guess 2 : Cecil – Face	5.897450	Cecil - Face:	5.897450
Guess 3 : Desk Items - B&P Photo	6.300311	Desk Items - B&P Photo:	6.300311
Guess 4 : Desk Items - Desk Lamp	7.251763	Desk Items - Desk Lamp:	7.251763
OUTCOME: CORRECT		OUTCOME: CORRECT	

42. Cecil - Shirt

Pre-Search returned 7 out of 29 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Sandra – Necklace	6.512932	Side Table Items - Table:	6.732697
Guess 2 : Side Table Items – Table	6.732697	Cecil - Shirt:	9.409222
Guess 3 : Sandra – Face	7.147389		
Guess 4 : Sandra - Hands and Arms	7.912048		
OUTCOME: INCORRECT		OUTCOME: CORRECT	

43. Cecil - Jacket

Pre-search returned 8 out of 29 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Cecil – Jacket	5.070073	Cecil - Jacket:	5.070073
Guess 2 : Furniture - Halogen Lamp	5.978194	Furniture - Halogen Lamp:	5.978194
Guess 3 : Sandra – Suit	6.249255	Desk Items - B&P Photo:	7.019581
Guess 4 : Sandra – Hair	6.449439	Desk Items - Desk Lamp:	7.351408
OUTCOME: CORRECT		OUTCOME: CORRECT	

44. Cecil - Hands

Pre-search returned 10 out of 29 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Furniture – Couch	6.390820	Furniture - Couch:	6.390820
Guess 2 : Sandra – Necklace	6.761527	Desk Items - Desk Lamp:	7.379698
Guess 3 : Desk Items - Desk Lamp	7.379698	Furniture - Halogen Lamp:	10.541412
Guess 4 : Sandra – Earring	8.005239	Furniture - Brown Pillow:	174.978760
OUTCOME: INCORRECT		OUTCOME: INCORRECT	

45. Cecil - Watch

Pre-search returned 15 out of 29 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Side Table Items – Plant	5.993326	Side Table Items - Plant:	5.993326
Guess 2 : Artwork - The Quail	6.365924	Artwork - The Quail:	6.365924
Guess 3 : Desk Items - B&P Photo	6.484156	Desk Items - B&P Photo:	6.484156
Guess 4 : Furniture – Couch	6.930558	Furniture - Couch:	6.930558

OUTCOME: NEW OBJECT

There are now 30 objects in the database.

46. Furniture - Couch

Pre-search returned 11 out of 30 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Furniture – Couch	5.832693	Furniture - Couch:	5.832693
Guess 2 : Sandra – Necklace	5.888259	Side Table Items - Table:	6.322322
Guess 3 : Side Table Items – Table	6.322322	Artwork - The Quail:	7.476311
Guess 4 : Sandra – Face	6.766833	Scenery - Orange Wall:	8.110137
OUTCOME: CORRECT		OUTCOME: CORRECT	

47. Furniture - Brown Pillow

Pre-search returned 14 out of 30 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Artwork - The Quail	5.369579	Artwork - The Quail:	5.369579
Guess 2 : Furniture – Column	5.497327	Furniture - Column:	5.497327
Guess 3 : Side Table Items - 5x7 Photo	5.825308	Side Table Items - 5x7 Photo:	5.825308
Guess 4 : Sandra – Hair	5.847046	Side Table Items - Plant:	6.093400
OUTCOME: INCORRECT		OUTCOME: INCORRECT	

48. Artwork - The Quail

Pre-search returned 10 out of 30 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Artwork - The Quail	5.190749	Artwork - The Quail:	5.190749
Guess 2 : Sandra – Face	7.202230	Side Table Items - Table:	7.681213
Guess 3 : Side Table Items – Table	7.681213	Scenery - Orange Wall:	18.817886
Guess 4 : Sandra – Necklace	7.831002		
OUTCOME: CORRECT		OUTCOME: CORRECT	

49. Scenery - Orange Wall

Pre-search returned 8 out of 30 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Sandra – Necklace	6.083580	Side Table Items - Table:	6.437154
Guess 2 : Side Table Items – Table	6.437154	Scenery - Orange Wall:	8.226588
Guess 3 : Sandra – Face	6.827585		
Guess 4 : Sandra – Earring	7.556805		
OUTCOME: INCORRECT		OUTCOME: CORRECT	

Summary:

1 NEW OBJECT

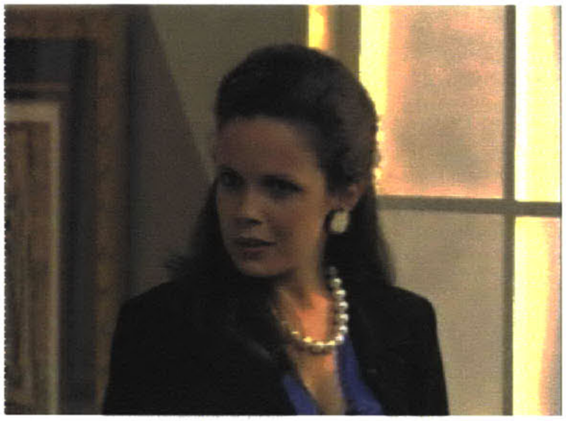
5 CORRECT (without OO) **7 CORRECT (with OO)**

4 INCORRECT (without OO) **2 INCORRECT (with OO)**

Object Occurrence Matrix:

```
[2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]    Sandra - Hair
[2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]    Sandra - Face
[2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]    Sandra - Earring
[2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]    Sandra - Suit
[2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]    Sandra – Hands and Arms
[1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]    Desk Items - Kleenex
[2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]    Scenery – Window Panes
[2 2 2 2 2 1 2 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]    Scenery – Light Wall Pattern
[2 2 2 2 2 1 2 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]    Scenery – Dark Wall Pattern
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1]    Cecil - Jacket
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1]    Cecil - Shirt
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1]    Cecil - Face
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1]    Cecil - Hair
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1]    Scenery – Orange Wall
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0]    Scenery – Side Window
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1]    Furniture - Couch
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1]    Artwork – The Quail
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1]    Furniture – Brown Pillow
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0]    Furniture – Halogen Lamp
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0]    Desk Items – B&P Photo
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0]    Desk Items – Desk Lamp
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0]    Side Table Items – Germany Doll
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0]    Side Table Items – 5x7 Photo
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0]    Side Table Items – Books
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0]    Side Table Items – Plant
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0]    Side Table Items – Table
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1]    Cecil - Hands
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0]    Furniture - Column
[1 1 1 1 1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]    Sandra - Necklace
[0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 1 1 1 0 0 0 0 0 0 0 0 1 0 0 1]    Cecil - Watch
```

SHOT 5



50. Sandra - Hair

Pre-search returned 15 out of 30 objects.

First object in shot - can't use OO matrix.

Guess 1 : Sandra – Hair	5.918703
Guess 2 : Cecil – Jacket	6.404220
Guess 3 : Furniture - Halogen Lamp	6.662089
Guess 4 : Side Table Items – Germany Doll	6.774389

OUTCOME: CORRECT

51. Sandra - Face

Pre-search returned 17 out of 30 objects.

Without OO Matrix:

Guess 1 : Cecil – Watch	5.762370
Guess 2 : Cecil – Hands	5.790921
Guess 3 : Desk Items - B&P Photo	6.281323
Guess 4 : Side Table Items – Table	6.595700

OUTCOME: INCORRECT

With OO Matrix:

Sandra - Face:	7.076553
Sandra - Hands and Arms:	7.507453
Sandra - Earring:	7.733479
Desk Items - Kleenex:	7.913020

OUTCOME: CORRECT

52. Sandra - Earring

Pre-search returned 9 out of 30 objects.

Without OO Matrix:	
Guess 1 : Side Table Items – Table	7.531633
Guess 2 : Sandra – Necklace	7.742431
Guess 3 : Scenery - Light Wall Pattern	8.904819
Guess 4 : Furniture – Couch	8.907988

OUTCOME: INCORRECT

With OO Matrix:	
Sandra - Necklace:	7.742431
Scenery - Light Wall Pattern:	8.904819
Sandra - Earring:	9.001574
Scenery - Dark Wall Pattern and Window Bars:	14.690640

OUTCOME: CORRECT

53. Sandra - Suit

Pre-search returned 14 out of 30 objects.

Without OO Matrix:	
Guess 1 : Cecil – Jacket	5.068393
Guess 2 : Furniture - Halogen Lamp	5.418674
Guess 3 : Cecil – Hair	5.651020
Guess 4 : Sandra – Suit	5.752752

OUTCOME: INCORRECT

With OO Matrix:	
Sandra - Suit:	5.752752
Scenery - Dark Wall Pattern and Window Bars:	9.820546
Desk Items - Kleenex:	11.923499

OUTCOME: CORRECT

54. Scenery - Window Panes

Pre-search returned 13 out of 30 objects.

Without OO Matrix:	
Guess 1 : Desk Items - B&P Photo	7.384476
Guess 2 : Cecil – Watch	7.924389
Guess 3 : Side Table Items – Table	8.406830
Guess 4 : Sandra – Necklace	9.036329

OUTCOME: INCORRECT

With OO Matrix:	
Sandra - Necklace:	9.036329
Scenery - Window Panes:	9.762552
Sandra - Hands and Arms:	11.596829
Scenery - Dark Wall Pattern and Window Bars:	19.067238

OUTCOME: CORRECT

55. Scenery - Light Wall Pattern

Pre-search returned 9 out of 30 objects.

Without OO Matrix:	
Guess 1 : Sandra – Necklace	5.970243
Guess 2 : Side Table Items – Plant	6.033715
Guess 3 : Side Table Items – Table	6.387764
Guess 4 : Scenery - Light Wall Pattern	6.970492

OUTCOME: CORRECT

With OO Matrix:	
Sandra - Necklace:	5.970243
Scenery - Light Wall Pattern:	6.970492
Desk Items - Kleenex:	7.084949
Scenery - Dark Wall Pattern and Window Bars:	9.921776

OUTCOME: CORRECT

56. Scenery - Dark Wall Pattern and Window Bars

Pre-Search returned 10 out of 30 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Cecil – Watch	5.437913	Sandra - Necklace:	6.023977
Guess 2 : Side Table Items - 5x7 Photo	5.644818	Scenery - Dark Wall Pattern: and Window Bars	8.492279
Guess 3 : Sandra - Necklace	6.023977		
Guess 4 : Side Table Items – Table	6.456033		
OUTCOME: INCORRECT		OUTCOME: CORRECT	

57. Sandra – Necklace

Pre-Search returned 15 out of 30 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Cecil – Watch	6.062335	Sandra - Necklace:	7.177580
Guess 2 : Cecil – Hands	6.326106	Sandra - Hands and Arms:	8.437216
Guess 3 : Desk Items - B&P Photo	6.511305	Desk Items - Kleenex:	9.047560
Guess 4 : Sandra – Necklace	7.177580		
OUTCOME: CORRECT		OUTCOME: CORRECT	

58. Sandra – Blouse

Pre-Search returned 2 out of 30 objects.

Without OO Matrix:	With OO Matrix:
Guess 1 : Cecil – Face	
Guess 2 : Cecil – Watch	

OUTCOME: NEW OBJECT

There are now 31 objects in the database.

59. Artwork - Springtime

Pre-Search returned 5 out of 31 objects.

Without OO Matrix:		With OO Matrix:
Guess 1 : Side Table Items - 5x7 Photo	5.253156	
Guess 2 : Side Table Items – Plant	5.316410	
Guess 3 : Cecil – Hands	5.797669	
Guess 4 : Cecil – Face	6.799435	

OUTCOME: NEW OBJECT

There are now 32 objects in the database.

Summary:

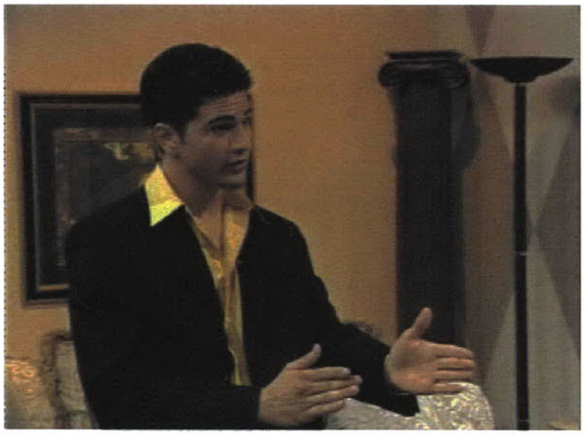
2 NEW OBJECTS

4 CORRECT (No OO)	8 CORRECT (with OO)
4 INCORRECT (No OO)	0 INCORRECT (with OO)

Object Occurrence Matrix:

```
[3 3 3 3 2 1 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 1 1] Sandra - Hair
[3 3 3 3 2 1 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 1 1] Sandra - Face
[3 3 3 3 2 1 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 1 1] Sandra - Earring
[3 3 3 3 2 1 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 1 1] Sandra - Suit
[2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0] Sandra – Hands and Arms
[1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] Desk Items - Kleenex
[3 3 3 3 2 1 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 1 1] Scenery – Window Panes
[3 3 3 3 2 1 3 4 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 0 1 1] Scenery – Light Wall Pattern
[3 3 3 3 2 1 3 4 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 0 1 1] Scenery – Dark Wall Pattern
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1 0 0] Cecil - Jacket
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1 0 0] Cecil - Shirt
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1 0 0] Cecil - Face
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1 0 0] Cecil - Hair
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1 0 0] Scenery – Orange Wall
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0] Scenery – Side Window
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1 0 0] Furniture - Couch
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1 0 0] Artwork – The Quail
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1 0 0] Furniture – Brown Pillow
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0] Furniture – Halogen lamp
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0] Desk Items – B&P Photo
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0] Desk Items – Desk Lamp
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0] Side Table Items – Germ. Doll
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0] Side Table Items – 5x7 Photo
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0] Side Table Items - Books
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0] Side Table Items - Plant
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0] Side Table Items - Table
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1 0 0] Cecil - Hands
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0] Furniture - Column
[2 2 2 2 1 0 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 1 1] Sandra - Necklace
[0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 1 1 1 0 0 0 0 0 0 0 0 1 0 0 1 0 0] Cecil - Watch
[1 1 1 1 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1] Sandra - Blouse
[1 1 1 1 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1] Artwork - Springtime
```

SHOT 6



60. Cecil - Hair

Pre-search returned 10 out of 32 objects.

First object in shot - can't use OO matrix.
I believe Object 1 is one of:

Guess 1 : Cecil – Jacket	5.341644
Guess 2 : Sandra – Suit	5.814922
Guess 3 : Cecil – Hair	5.909383
Guess 4 : Furniture - Halogen Lamp	6.545926

OUTCOME: CORRECT

61. Cecil - Face

Pre-search returned 11 out of 32 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Cecil – Hands	5.529164	Cecil - Hands:	5.529164
Guess 2 : Artwork – Springtime	6.648735	Cecil - Face:	6.851098
Guess 3 : Cecil – Face	6.851098	Desk Items - Desk Lamp:	7.190682
Guess 4 : Desk Items - Desk Lamp	7.190682	Furniture - Halogen Lamp:	9.831314
OUTCOME: CORRECT		OUTCOME: CORRECT	

62. Cecil - Shirt

Pre-search returned 5 out of 32 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Sandra – Earring	7.390963	Cecil - Shirt:	7.889021
Guess 2 : Cecil – Shirt	7.889021	Scenery - Orange Wall:	8.566061
Guess 3 : Sandra – Face	8.062239		
Guess 4 : Scenery - Light Wall Pattern	8.079296		
OUTCOME: CORRECT		OUTCOME: CORRECT	

63. Cecil - Jacket

Pre-search returned 9 out of 32 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Cecil – Jacket	5.153050	Cecil - Jacket:	5.153050
Guess 2 : Sandra – Suit	5.593382	Furniture - Halogen Lamp:	6.370705
Guess 3 : Furniture - Halogen Lamp	6.370705	Cecil - Hands:	7.911875
Guess 4 : Sandra – Hair	6.577944	Desk Items - B&P Photo:	7.290842
OUTCOME: CORRECT		OUTCOME: CORRECT	

64. Furniture - Column

Pre-search returned 5 out of 32 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Furniture – Column	4.798909	Furniture - Column:	4.798909
Guess 2 : Sandra – Hair	6.40957	Side Table Items - Table:	9.478822
Guess 3 : Side Table Items – Table	9.478822	Furniture - Brown Pillow:	10.076010
Guess 4 : Furniture - Brown Pillow	10.076010		
OUTCOME: CORRECT		OUTCOME: CORRECT	

65. Furniture – Halogen Lamp

Pre-search returned 8 out of 32 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Sandra – Hair	5.869368	Furniture - Halogen Lamp:	6.622506
Guess 2 : Sandra – Suit	5.908371	Cecil - Hands:	7.121848
Guess 3 : Furniture - Halogen Lamp	6.622506	Desk Items - B&P Photo:	7.344704
Guess 4 : Cecil – Hands	7.121848	Desk Items - Desk Lamp:	7.901878
OUTCOME: CORRECT		OUTCOME: CORRECT	

66. Artwork - The Quail

Pre-search returned 15 out of 32 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Cecil – Watch	5.695131	Cecil - Watch:	5.695131
Guess 2 : Artwork - The Quail	6.523623	Artwork - The Quail:	6.523623
Guess 3 : Side Table Items – Germany Doll	6.815670	Side Table Items – Germ. Doll:	6.815670
Guess 4 : Side Table Items – Books	7.261171	Side Table Items - Books:	7.261171
OUTCOME: CORRECT		OUTCOME:CORRECT	

67. Scenery - Orange Wall

Pre-search returned 8 out of 32 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Side Table Items – Table	6.204501	Side Table Items - Table:	6.204501
Guess 2 : Sandra – Necklace	6.550072	Scenery - Orange Wall:	7.313588
Guess 3 : Scenery - Orange Wall	7.313588	Side Table Items - 5x7 Photo:	7.906590
Guess 4 : Sandra – Face	7.433901		
OUTCOME: CORRECT		OUTCOME: CORRECT	

68. Scenery - Light Wall

Pre-search returned 14 out of 32 items.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Cecil – Watch	5.603948	Cecil - Watch:	5.603948
Guess 2 : Side Table Items – Table	6.111399	Side Table Items - Table:	6.111399
Guess 3 : Sandra – Necklace	6.375080	Furniture - Couch:	6.703860
Guess 4 : Desk Items – Kleenex	6.483084	Side Table Items – Germ. Doll:	6.774905
OUTCOME: INCORRECT		OUTCOME: INCORRECT	

69. Scenery - Dark Wall Pattern

Pre-search returned 15 out of 32 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Side Table Items – Plant	5.204290	Side Table Items - Plant:	5.204290
Guess 2 : Cecil – Watch	5.382334	Cecil - Watch:	5.382334
Guess 3 : Scenery - Dark Wall Pattern and Window Bars	5.998507	Side Table Items - Table:	6.515735
Guess 4 : Side Table Items – Table	6.515735	Side Table Items - Books:	6.649657
OUTCOME: CORRECT		OUTCOME: CORRECT	

70. Cecil – Hands

Pre-search returned 9 out of 32 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Cecil – Hands	5.867694	Cecil - Hands:	5.867694
Guess 2 : Desk Items - B&P Photo	6.320011	Desk Items - B&P Photo:	6.320011
Guess 3 : Sandra – Earring	7.226053	Desk Items - Desk Lamp:	7.658471
Guess 4 : Desk Items - Desk Lamp	7.658471	Furniture - Brown Pillow:	174.978760

OUTCOME: CORRECT

OUTCOME: CORRECT

71. Furniture - Couch

Pre-search returned 12 out of 32 objects.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Cecil – Watch	5.686023	Cecil - Watch:	5.686023
Guess 2 : Side Table Items – Table	6.333865	Side Table Items - Table:	6.333865
Guess 3 : Sandra – Necklace	6.711828	Side Table Items – Germ. Doll:	6.853434
Guess 4 : Side Table Items – Germany Doll	6.853434	Side Table Items - Books:	7.055503

OUTCOME: INCORRECT

OUTCOME: INCORRECT

72. Furniture - Chair

Pre-search returned 12 out of 32 items.

Without OO Matrix:		With OO Matrix:	
Guess 1 : Cecil – Watch	6.551734	Cecil - Watch:	6.551734
Guess 2 : Desk Items - B&P Photo	6.598215	Desk Items - B&P Photo:	6.598215
Guess 3 : Desk Items – Kleenex	6.828788	Side Table Items – Germ. Doll:	6.874578
Guess 4 : Side Table Items – Germany Doll	6.874578	Side Table Items - Table:	6.915253

OUTCOME: NEW OBJECT

Summary:

1 NEW OBJECT

10 CORRECT (NO OO)   10 CORRECT (OO)

2 INCORRECT (NO OO)   2 INCORRECT (OO)

Object Occurance Matrix:

```
[3 3 3 3 2 1 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 1 1 0]    Sandra - Hair
[3 3 3 3 2 1 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 1 1 0]    Sandra - Face
[3 3 3 3 2 1 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 1 1 0]    Sandra - Earring
[3 3 3 3 2 1 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 1 1 0]    Sandra - Suit
[2 2 2 2 2 1 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]    Sandra – Hands and Arms
[1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]    Desk Items - Kleenex
[3 3 3 3 2 1 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 1 1 0]    Scenery – Window Panes
[3 3 3 3 2 1 3 5 5 2 2 2 2 2 1 2 2 1 2 1 1 1 1 1 1 1 2 2 2 0 1 1 1]    Scenery – Light Wall Pattern
[3 3 3 3 2 1 3 5 5 2 2 2 2 2 1 2 2 1 2 1 1 1 1 1 1 1 2 2 2 0 1 1 1]    Scenery – Dark Wall Pattern
[0 0 0 0 0 0 0 2 2 3 3 3 3 3 1 3 3 2 2 1 1 1 1 1 1 1 3 2 0 1 0 0 1]    Cecil - Jacket
[0 0 0 0 0 0 0 2 2 3 3 3 3 3 1 3 3 2 2 1 1 1 1 1 1 1 3 2 0 1 0 0 1]    Cecil - Shirt
[0 0 0 0 0 0 0 2 2 3 3 3 3 3 1 3 3 2 2 1 1 1 1 1 1 1 3 2 0 1 0 0 1]    Cecil - Face
[0 0 0 0 0 0 0 2 2 3 3 3 3 3 1 3 3 2 2 1 1 1 1 1 1 1 3 2 0 1 0 0 1]    Cecil - Hair
[0 0 0 0 0 0 0 2 2 3 3 3 3 3 1 3 3 2 2 1 1 1 1 1 1 1 3 2 0 1 0 0 1]    Scenery – Orange Wall
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0]    Scenery – Side Window
[0 0 0 0 0 0 0 2 2 3 3 3 3 3 1 3 3 2 2 1 1 1 1 1 1 1 3 2 0 1 0 0 1]    Furniture - Couch
[0 0 0 0 0 0 0 2 2 3 3 3 3 3 1 3 3 2 2 1 1 1 1 1 1 1 3 2 0 1 0 0 1]    Artwork – The Quail
[0 0 0 0 0 0 0 1 1 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 2 1 0 1 0 0 0]    Furniture – Brown Pillow
[0 0 0 0 0 0 0 2 2 2 2 2 2 2 1 2 2 1 2 1 1 1 1 1 1 1 2 2 0 0 0 0 1]    Furniture – Halogen Lamp
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0]    Desk Items – B&P Photo
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0]    Desk Items – Desk Lamp
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0]    Side Table Items – Germ. Doll
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0]    Side Table Items – 5x7 Photo
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0]    Side Table Items – Books
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0]    Side Table Items – Plant
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0]    Side Table Items – Table
[0 0 0 0 0 0 0 2 2 3 3 3 3 3 1 3 3 2 2 1 1 1 1 1 1 1 3 2 0 1 0 0 1]    Cecil - Hands
[0 0 0 0 0 0 0 2 2 2 2 2 2 2 1 2 2 1 2 1 1 1 1 1 1 1 2 2 0 0 0 0 1]    Furniture - Column
[2 2 2 2 1 0 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 1 1 0]    Sandra - Necklace
[0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 1 1 1 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0]    Cecil - Watch
[1 1 1 1 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0]    Sandra - Blouse
[1 1 1 1 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0]    Artwork - Springtime
[0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 1 1 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 1]    Furniture - Chair
```