ActiveStories

Infusing author's intention with content to tell a computationally expressive story

Phillip Rodrigo Tiongson
S.B. Electrical Engineering and Computer Science
S.B. Humanities
Massachusetts Institute of Technology
June 1996

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 Arts and Sciences
at the
Massachusetts Institute of Technology

August 1998

© Massachusetts Institute of Technology, 1998.
All rights reserved.

author
Phillip Rodrigo Tiongson
Program in Media Arts and Sciences
August 7, 1998

certified by
Glorianna Davenport
Principal Research Associate, MIT Media Laboratory
Thesis Supervisor

accepted by
Stephen A. Benton
Chair, Departmental Committee on Graduate Studies
Program in Media Arts and Sciences

ActiveStories

Infusing author's intention with content to tell a computationally expressive story

Phillip Rodrigo Tiongson
S.B. Electrical Engineering and Computer Science
S.B. Humanities
Massachusetts Institute of Technology
June 1996

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 Arts and Sciences
at the
Massachusetts Institute of Technology

August 1998

0.0 Abstract

Current digital tools for cinema have been cast in the same mold as their analog equivalents. The computational medium promises more: to expand expression beyond the physical edge of the film frame.

In traditional film, the camera constrains how light falls onto the film, but not how you tell a story. In the computer, if authors rely on tools built by others, the tool builder determines the vocabulary that authors use to tell their stories. However, if authors build their own tools, they access the entire vocabulary of expression that the computational medium provides.

This thesis presents ActiveStories interactive stories that express an author's intention in computational forms. The stories illustrate novel, dynamic representations of narrative and author's intention. ActiveStories put viewers into the story, giving them dynamic, interactive control over the content, while the author's intention is expressed in the process of watching.

Thesis Supervisor: Glorianna Davenport
Title: Principal Research Associate

This work was supported in part by the Eastman Kodak Company.

ActiveStories

Infusing author's intention with content to tell a computationally expressive story

Phillip Rodrigo Tiongson
S.B. Electrical Engineering and Computer Science
S.B. Humanities
Massachusetts Institute of Technology
June 1996

The following people served as readers for this thesis:

reader
John Maeda
Assistant Professor of Design and Computation
Program in Media Arts and Sciences

reader
Errol Morris
Director
Fourth Floor Productions

reader
Brian Smith
Assistant Professor
Program in Media Arts and Sciences

Credits

<voice-over>

Of all the pages in my thesis, this one took the longest. I have been writing this page for the past twenty four years. Ever since I was little, I dreamed of coming to MIT and finishing something. This morning, I woke up, as usual, got in the shower, as usual, and started to think about all the steps it took to get here. I remembered my Mom and Dad telling me I could do anything. I remembered Bink, taking me every day to the lake to teach me to swim, and even though I hated the water, I loved being with him, and so I learned. Whether the stress finally got to me or whether it was the intensity of remembering all the images of my friends and family supporting me throughout these twenty four years, I started crying, first a little, then as I touched each memory in my mind, more and more tears fell, in what literally became a shower of tears. It wasn't sad. In fact, it was incredible to remember so clearly into my childhood and feel the emotions of fear of failure and doubt while being safe, knowing that I am a mere twenty four hours from finishing a Master's Thesis. I thought of the past six years at MIT and of my friends who make my life worth living. I still remember the first time I met Peter and showed him my first poster. I was so happy just to share it with him. On the day before, Phoebe Jaysue Lam had walked right into my life and my arms, ever so briefly. It was as if I were seeing a trailer of my life, and I cried like never before. I remember clutching Caraway's hand - ever so cautiously - to say goodbye, just after seeing a play with her - a play that I had seen before, but went again anyway. In my original MIT entrance essay, I wrote a script to show them that I was a clever nerd, and now I am writing the credits to that six year epic adventure. Son! is to my (extended) family most of all that I dedicate this work: Caraway, Peter & Phoebe, Mom & Dad, Gigi & Rhona, Joe, and most of all, Bink and the new "boy," Miles Thompson Burnette.

<roll credits>

To...

Mom, for never telling me no.
Dad, for always testing me.
Gigi, for cleaning my ears.
Rhona, for taking me seriously.
Bink, for teaching me to keep my head above water.
Joe, for making Rhona happy.
Miles, for just being.

Phoebe, for recognizing me, "You like Woody Allen!"
Peter, for my first design critique, "I usually don't use so many fonts."
Caraway, for giving me my two weeks, "That's a definite possibility."

Teresa, for talking me down.
Deirdre, for talking to me all these years.
Jason and Doug, for all the stupid trivial arguments.

Arjan, for telling me to value myself.
Grenby, for sharing with me the Jax of life.
Larry Tom White, for THE EXISTENTIAL YUK ENTHUSIAST, whatever [illegible]

Cherry, for her perfect pitch.
Colleen, for her clever pitch.

Mark, for my first film.
Pat, for my next.

Kram, for the kiss.
Chloe and Tomas, for the longest day.
daniel, for the approving eye.
Fernanda, for the hugs.

jseo, for bringing sushi to the Seattle.
mumaugh, for late night LaVerde's and Star Trek: TNG.
dikung, for accosting me.
robin, for driving around with me.
the ghosts of macondo, remember the spirits of jh, beb, ppk, paulh, baird, nchiash, davet, brooks, & Big Daddy Cool.

adoqand vrio, for the family.

Finally...

to Brian Smith, for the last minute dutch thesis saving diving catch.
to Glorianna Davenport, for every moment of conversation, a nurturing eye, and her unending vision.

and to John Maeda, for always, always, always, demanding more.

a White Frames of a Dead Cow production
Made possible by Cary Bolger and your friends at Eastman Kodak

Contents

1.0 Introduction

1.1 My personal interactive cinema

[1] E.T., 1982, 115m, dir Stephen Spielberg.

Ever since I can remember, I would put myself *into* the movies. When I was eight, I remember crying in my bed after seeing *E.T.*,[1] because I could see E.T. waving goodbye to *me*. My Mom says I cried for *three* days. I was so excited when I bought E.T., the Extra Terrestrial—The Game for my Atari 2600. You were E.T. That was the first disappointment. I had always imagined myself as the boy, not the alien.

You were alone and a clock was ticking. If you didn't find all the pieces of the phone to "Phone Home," then the "Evil Scientist" would find you and take you back to the lab. If the Scientist found you three times, then you saw yourself put to sleep, forever trapped inside a box. I remember being scared when the Scientist would take me away. (Is it any wonder that I wound up at MIT and became an engineer instead?) But the thing that I remember most about the game was that it was *boring*. You were pixellated beyond all recognition except for your distinctive big head. You walked around these empty sets *alone*. That was the other thing I remember, it was so *lonely* playing the game. The movie was about how Elliot made friends with E.T. and how they helped him go find his parents. The game was about wandering around empty sets by yourself and getting captured by the Scientist.

That was my first experience with the difference between cinema and interactive cinema.

I have been participating in interactive cinema my whole life by watching movies. In my mind, I have blown up the Death Star in *Star Wars*[1] and worked at the Quick Stop in *Clerks*.[2] I played the video games from the original Combat for the Atari to the fully-disorienting virtual reality PacMan 3D, searching for a way to recreate the interactive experience in my mind. So far, like my experience with E.T.—The Game, I have been disappointed.

[1] *Star Wars*, 1977, 121m, dir George Lucas.
[2] *Clerks*, 1994, 90m, dir Kevin Smith.

1.2 The computational medium

This thesis proposes that the interactive cinema that I want to experience can exist in a computational medium. The experience can be more emotional and personal by extending the author's vocabulary of expression. The medium evolved from machines that were designed to compute partial differential equations into machines with interactive graphical displays. By drawing numbers, text, and images, at 60 frames per second [fps], the medium can imitate traditional media like paper or film. Unlike the physical media, the computational medium can react and change its behavior when a "user" approaches it.

However, the computational machines were fundamentally designed to process data with procedures. The original process of programming was more concerned with optimizing memory usage than emotional expression. No new process has yet emerged for the artist in the computational medium to create an expressive language. Using programming languages designed to process missile trajectory data, we will craft the computational procedures into a new set of tools. Programs and procedures become the sculptor's chisel, and the digitized data is the granite.

Commercial digital tools have modeled themselves after traditional editing tools. Avid's Media Composer mimics the behavior of an editor's splicer and trim bins.[3] This provides both the user and the toolmaker with expectations of what tools are needed and how they should work. Every digital non-linear editing program implements the same basic functionality: to cut and paste clips into a visual timeline, just as you would on a Steenbeck editing table.

[3] A "trim bin" is a cart of hooks on which hang the pieces of film trimmed before they are edited into a scene. Avid Media Composer's digital equivalent is called a "bin" and is functionally equivalent to a directory on a file system. Even though it is logical on the computer to want to make a hierarchy of bins, the "bin" metaphor does not support the idea of a making a "bin" inside another "bin," and so Avid's tool does not either.

These tools are modeled after the traditional process in a traditional medium. Even a "multimedia" tool like Macromedia's Director[1] uses the metaphor of frame-based animation cels. Director is the tool of choice for most authors in the computational medium, and yet, its functionality is still deeply rooted in the assumptions of physical media.

[1] Macromedia Director is a commercially available software program used by the majority of multimedia designers.

The computational medium is fundamentally different from physical media:

> A tool is something that operates on or processes some medium. In the digital domain, tool and medium coexist in the same entity. An analogy would be to imagine a screw welded to the tip of a screwdriver or a typewriter that can only type on a single piece of paper, which is glued to the roller. If these examples seem absurd to you, it is because they are. Our medium and tools are always separated physically because otherwise the medium cannot be realized as a deliverable "object." However, on the computer, this distinction does not apply because the digital domain does not need to obey the laws of the physical world. [Maeda 1997]

Because the tool and the medium exist in the same form, as digital data, the tool becomes part of the medium. Authors can express their stories not only as sequential images, but also in the process that displays each image. Director and Apple's HyperCard provide templates for the multimedia

author, but also constrain authors to think in a certain way and to modify their process of storytelling to fit within those templates.

1.3 The story is the tool is the process.

When you depend on other people's tools to express yourself, then you also depend on them to expand your language. For instance, if I want to make an animation in Macromedia Director, I place each of the frames into the time line and hit "play." The animation plays back at 4 frames a second. If I consider that too slow, what options do I have to increase the performance, besides buying a faster machine? I could reduce the number of frames, or tweak the preferences of Director looking for a setting that happens to increase the frame rate, or send email to Macromedia and request that they make my animation run faster.

On the other hand, if I write a program which displays an animation, I can decide exactly what I want to trade off for better performance. I could decide not to double buffer the animation, I could prefetch all the images into RAM, I could vectorize the drawing, I could draw every ninth pixel, etc. The important point is that the author has full control of the process which is creating and displaying the content. As a result, I could decide to draw only a part of the animation that is currently underneath the user's mouse because I think that is probably the most important information to the user.[2]

[2] In ActiveXmas, I use this process of revealing what is under the mouse to focus the user on what I think is important in a video frame.

Author's intention and vocabulary. I create a process which conforms to my intention: a computational procedure for manipulating the data of the story. A "procedure for manipulating data" is usually called a tool, and yet, this tool is useful only in telling my story. The computational story becomes a merging of traditional content with the author's

process of telling. This ability to infuse content with an author's intentions enhances the expressive power of the computational medium.

Macromedia Director provides the user with a fixed vocabulary of processes with which an author composes to build a story. Manipulating the computational medium at the level of a programming language enables the author to develop unique processes from a richer vocabulary. This may serve to explain why "all Director movies look alike." All multimedia authors who use Director are constrained to use the same vocabulary of actions. That vocabulary is only extended when Director releases a new upgrade or plug-in.

ActiveStories attempt to move beyond the current interactive vocabulary of buttons, scrollbars, and check boxes. I use the computational medium to store a *process* of presenting a story, affecting how each frame is drawn to the screen at 15 frames per second. The process reflects my intention as author, encoded as custom Java programs.

1.4 Contributions

This thesis contributes to the study of the field in the following ways:

A brief history of the evolution of interactive cinema. In Section 2, I trace the development of filmmaking and interactive computing up to the creation of the Interactive Cinema Group at the Media Lab. I outline the causes and effects of certain enabling technologies which made narrative innovation in each medium possible. [see "The evolution of interactive cinema" on page 18]

My experiments in computational narrative. I highlight some of my past experiments which eventually led to the development of this thesis, the ActiveStories. I describe in detail the ActiveStories—ActivePoem and ActiveXmas—their motivation and mechanics. [see "Experiments in ActiveStories" on page 92]

Figure 1. "Building a mystery" is the theme song of this thesis.

1.5 Why is this thesis at the Media Lab?

This thesis is an exploration of the intersection of technology and storytelling. The Media Lab's unique blend of resources and insatiable curiosity makes it an environment of nervous and creative energy. Being a part of the Interactive Cinema and Aesthetic and Computation Groups has surrounded me with extraordinary people with an incredible depth of design, computational, and storytelling skills. It has been this mixed environment of intensely curious people and *bleeding-edge* technology which enabled me to pose the questions and answers embodied in this thesis.

1.6 Final note

[1] Sony/Loews Theme Song

I wish I could sing, "Sit back and relax. Enjoy the show!"[1] Unfortunately, this thesis is printed on paper and can not be musical or interactive. The projects described in this thesis should be experienced, not flipped through. So I ask you to imagine I am singing and that you can see the programs running, and I will describe what is going on.

In Section 2, I present a brief history of interactive cinema technology. In Section 3, I describe the influences on my design from the Interactive Cinema and Aesthetics and Computation Groups. In Section 4, I highlight my past work that led to Section 5, the ActiveStories.

For the five minute reader. Look at the drawings at the beginning of each section, flip through the pictures, and skip to the Conclusion.

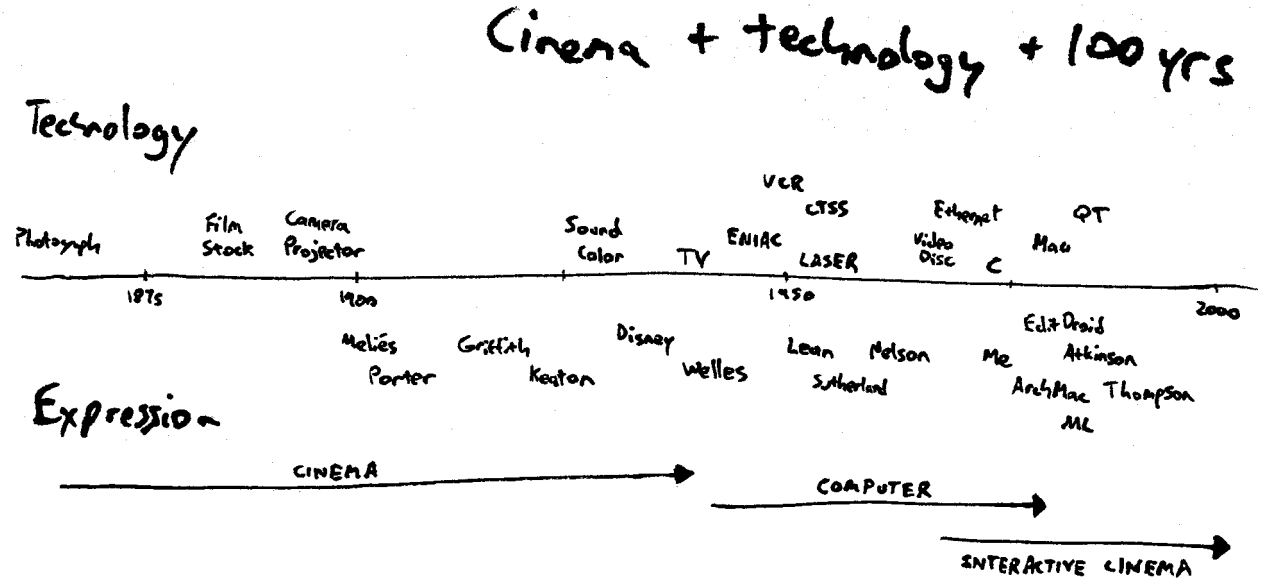
For the twenty minute reader. Browse the history section, read about ActiveArticle [see "ActiveArticle" on page 83] and look at the table [see "Projects and Lessons Learned" on page 63], and then the Conclusion.

For the enterprising UROP. Browse the influences section [see "Reflecting on the influences" on page 36], then peruse the implementation appendix for clues about java and Quicktime. [see "Appendix" on page 120] Ask lots of questions. Don't be intimidated. Make really cool hacks. Watch lots of movies. Don't stress out too much.

2.0 The evolution of interactive cinema

Interactive Cinema has at its root *cinema*. Cinema—the art of visual storytelling—is inexorably connected to the technology of images and sound. Interactive cinema adds the technology of computation into the mix, inheriting from two traditions, cinema and computers. Glorianna Davenport coined the term "Interactive Cinema" in 1988 as a way to describe the academic endeavor of her research group. Her academic search was to satisfy the "longing of cinema to become something new, something more complex, and something more personal, as if in conversation with an audience."

Figure 2. Timeline of 100 years of cinema and computer. The timeline illustrates the timing of technological innovation with individual expression. Abbreviations: Computer Time-Sharing System [CTSS], QuickTime [QT], Media Lab [ML]



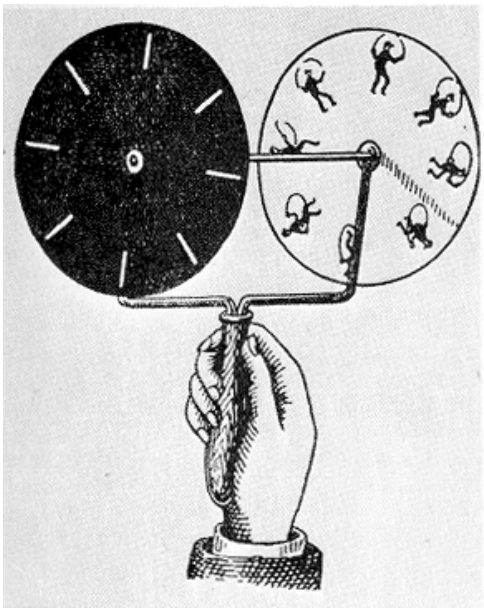
In the 100 years before the group was founded, the two machines which enable interactive cinema was invented, namely the camera and the computer, but they had yet to be combined into a single medium. Cinema was born with the invention of Edison's Kinetoscope in 1891. Filmmakers turned it into a means of expression.

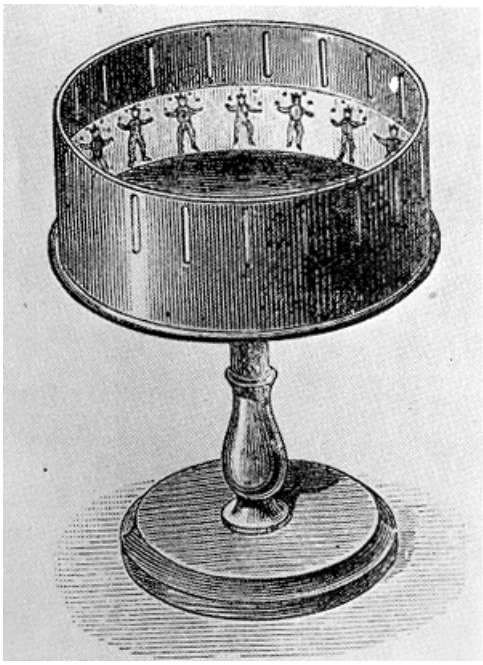
In the 25 years after the camera was invented, Meliés, Edwin S. Porter, D.W. Griffith, Sergei Eisenstein, and Buster Keaton each pushed the expressive use of the film medium to tell their own stories. After the invention of synchronized sound, Orson Welles made his masterpiece *Citizen Kane*—a prime example of using the limits of technology to tell an epic story in black and white. Technicolor enabled Walt Disney to release the first two color cartoons. After technology enables a new form of expression, an artist who understands the conventions of the past and has a vision of the future will tell new kind of story.

In the 25 years after ENIAC—the first electronic computer—Ivan Sutherland expressed his vision of the power of computation by creating *SKETCHPAD*, the first interactive graphical computer interface. As non-linear access to information became possible, Ted Nelson began to implement *Xanadu*, his vision of the ultimate hypertext system that sought to record and cross-reference all human literature.

In this section, I will chronicle the three periods. The evolution of the cinematic medium begins with the invention of the photograph [1850] and ends with Citizen Kane [1941]. The development of interactive computing begins with ENIAC [1945] and ends with the Macintosh [1984]. The last period is the birth of interactive cinema, which begins with the invention of the optical videodisc [1976] and continues until today. In each period, I describe the enabling technologies of the day and some of the people who expressed themselves in the new medium.

Figure 3. Phenakistoscope and Zoetrope, early motion picture machines





2.1 Cinema and its machines, 1850–1941

> Cinema was born as an independent medium only *after* the cinema machines had been evolved for purposes other than the establishment of such a medium. That is, the invention of the machines preceded any serious consideration of their documentary or aesthetic potential; and this relationship has remained constant throughout the history of film because the cinema at its material base is a technological form—one in which technological innovation precedes the aesthetic impulse (i.e. no artist can express him- or herself in cinema in ways which would *exceed* the technological capabilities of the machines). [Cook 1990]

The film medium's first incarnation was as a light-sensitive emulsion smeared onto a glass plate. For the budding photographer like George Eastman, going outside to take a picture meant bringing a light-tight tent full of glass bottles with you. Eastman pioneered the use of celluloid as a flexible, portable, and highly flammable film base. This advance made possible amateur photography, the moving picture, and many small fires.

The "moving picture" was born of two technologies, mechanical and photographic. After you have film with sprocket holes, then you need a camera to move the film at sixteen frames per second, starting and stopping the film behind a rotating shutter—a difficult mechanical task. Early film cameras had difficulty controlling the inertia of large film reels, which would spin too fast and snap the film passing through the camera. Early movies were shorts that could only last from thirty seconds to two minutes. [Cook 1990]

Figure 4. Muybridge's galloping horse

Thomas Edison's Kinetoscope did not even use a film reel, instead stretching the entire film out on rollers. The first nickelodeon shows were primarily novelty acts that could be brought into a studio and performed in front of a fixed camera.

Figure 6. A shot from a typical nickelodeon film *Parisian Dance*

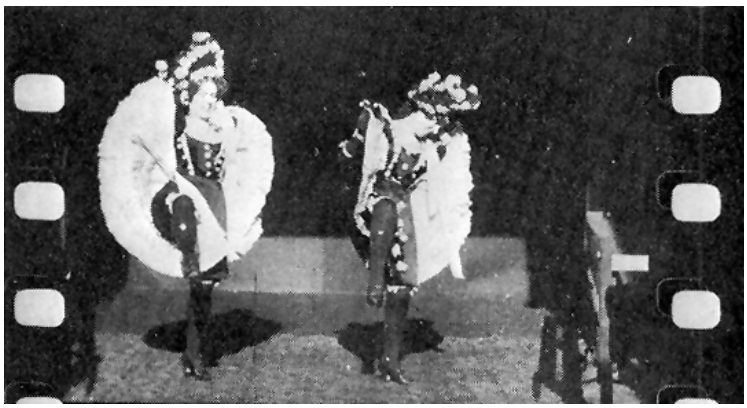
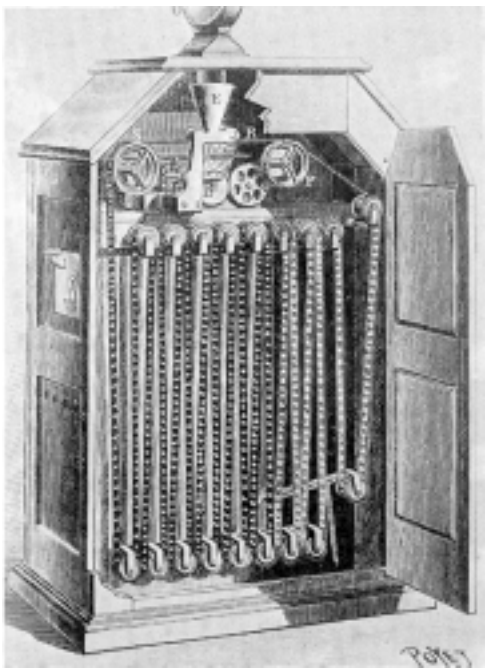


Figure 5. Edison's first projection machine, the Kinetoscope



In 1895, The Lumiére Brothers create the Cinématographe, a single machine which was camera, film printer, and projector, all in one. The hand-cranked camera weighed only 16 pounds making it quite portable. The brothers freed the camera from being fixed inside a studio and began to capture scenes from the world around them.

Figure 8. The first Lumiére film: *La Sorite des ouvriers de l'usine Lumiére* (*Workers Leaving the Lumiére Factory*)

Figure 7. The Cinématographe. The wondrous machine did everything a filmmaker needed as camera, film printer, and projector. Its hand-cranked drive mechanism made it much lighter than Edison's battery-driven camera.

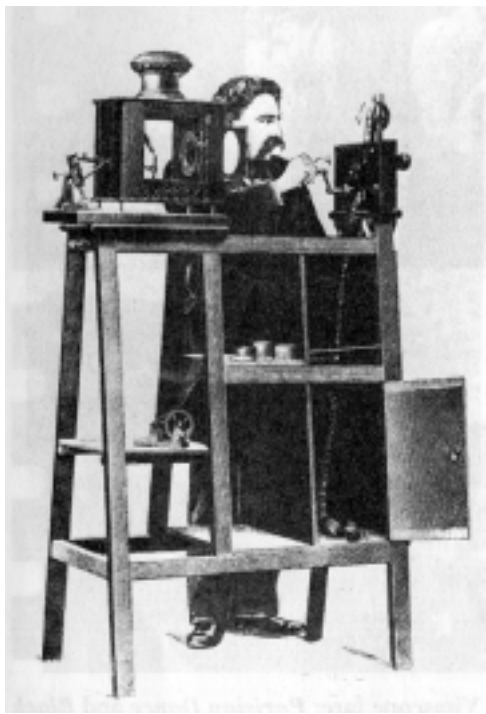
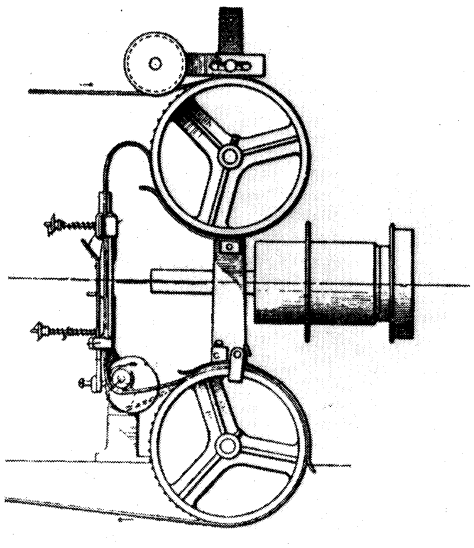


Figure 9. The Latham Loop



In 1901, the Latham Family (Gray, Otway, and father Woodville) discovered that placing a small loop above and below the shutter reduced the stress placed on film passing through projectors and cameras. Cameras and projectors using the Loop could reliably hold 20 minutes of film instead of 120 seconds. This incremental advance in technology made it possible for filmmakers to record and project much longer sequences. Filmmakers began to experiment with longer shots and editing different shots together. The advance in technology allowed filmmakers to extend their narrative language.

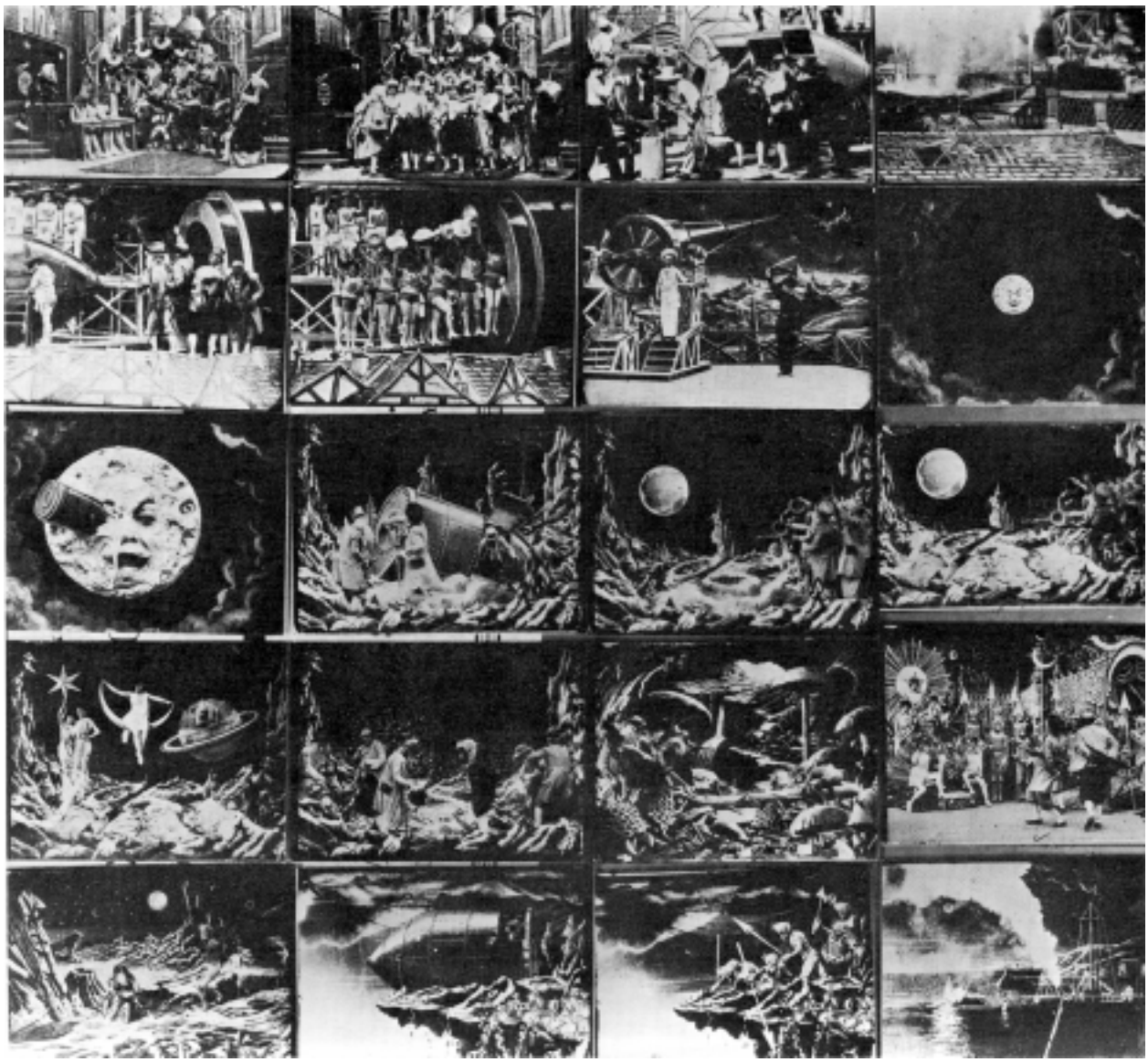
Meliés. The thirty second nickelodeon show *Parisian Dance* soon evolved into George Meliés's ten minute *A Trip to the Moon*. Meliés began to use the technology in a different way, to tell a story. He adapted much of his material from books or plays, such as Jules Verne's *A Trip to the Moon*.

Meliés composed one-shot scenes which he called *tableaux*, e.g. "The flight of the projectile through space" or "The astronomers' escape." The actors, as if on a stage, would begin and finish *tableaux* in continuous shots while Meliés filmed the action as if he were sitting in a theatre house. Meliés spliced each complete scene together into a narrative. Voilá, Cinema! [Cook 1990]

Meliés's films exhibit mastery of the techniques of photography. His films contain camera-trick illusions, clouds of hand-tinted color, and visual transitions like lap dissolves[1] to edit smoothly from scene to scene. Meliés understood how the camera worked mechanically—he had built it from scratch—but not once in 500 films did his camera move during a shot. He fixed the camera in the best seat of his theatre and left it there, to record his stage plays—the prosceniu m arch[2] just outside the frame. He extended the language of visual storytelling using narrative and editing, but he was entrenched in the traditions of stage and theatre.

[1] Lap Dissolves are an in-camera transition from scene to scene. The cameraperson closes the iris at the end of a scene, rewinds the film a few seconds, then open the iris slowly while the next scene begins. Today these effects are usually done in post-production either using an optical printer or digitally on a computer.
[2] The proscenium arch is the physical frame which surrounds the theatrical stage. It is the boundary between the real world and the drama on stage. In early films, this arch was often included in the frame to signify that was a play.

Figure 10. *Le Voyage dans d lune* (*A Trip to the Moon*), dir George Meliés, 825 ft. ~16 minutes



Continuity editing. The film medium would not sit still, and just one year later, Edwin S. Porter would extend the language of both cinematography and editing by releasing *The Great Train Robbery.* Up to this point, film editing was performed as a way of to connect whole actions. If it took the actor two minutes to get dressed, the action was recorded in a two minute scene.

Figure 11. The compression of time

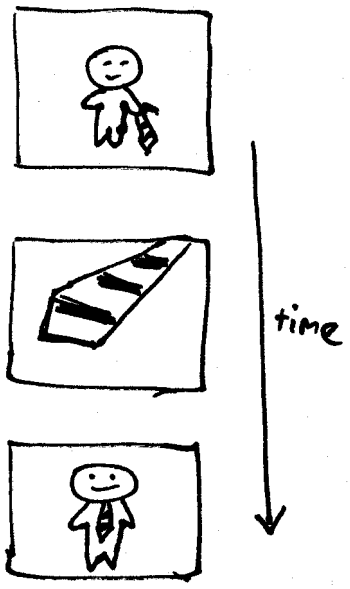
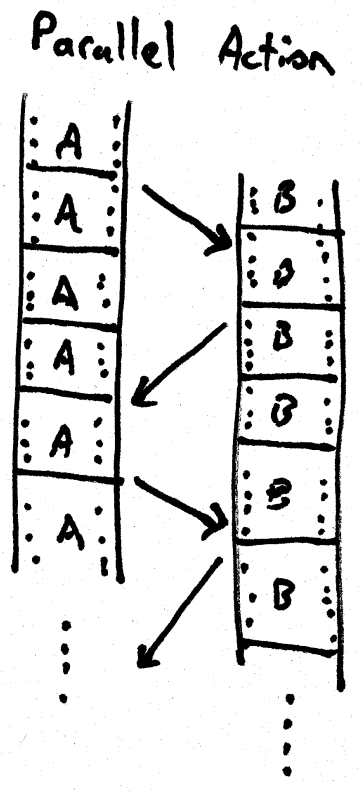


Figure 12. Editing for parallel action



Porter pioneered the editing conventions that today are known as the rules of continuity editing. By cutting scenes before the end of an action, you create an ellipsis in time. An actor can get dressed in two shots, rather than two minutes. Porter's basic element for telling a story was not a *scene* but a shot. He sequenced multiple shots together to compose scene, breaking the one shot-one scene convention of Meliés. [Cook 1990]

The basic underlying assumption of continuity editing is that time is always progressing forward, even if the camera is not present. Porter could portray two actions happening in parallel, by intercutting between them. Editing became part of the process of telling the story.

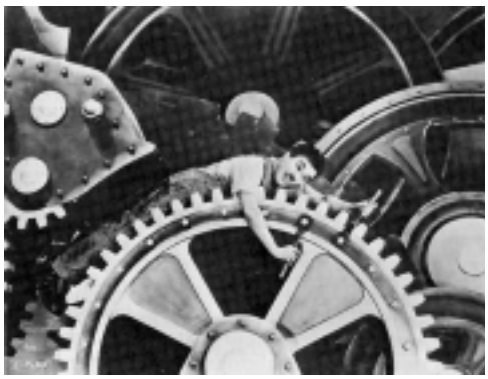
Because complete actions no longer had to happen in a single shot, Porter experimented with moving the camera closer to actors. He staged his scenes using the depth of the frame, so action would proceed from the background to the foreground. In the Great Train Robbery, he panned the camera to follow the action in the scene, moving the camera during a shot! In later films, he even mounted his camera onto cars and trains. [Cook 1990]

Porter's films had a strong effect on everyone who saw them, especially other film makers. Film makers delighted by the novelty built on what they had learned. Audiences learned a new visual language. Soon everyone imitated Porter's techniques, which were codified slowly into the conventions we use today. His films destroyed the proscenium arch that constrained films to the staged action of the theatre.

Figure 13. *The Great Train Robbery*, dir Edwin S. Porter, 740 ft. ~12 minutes

Porter had faster film stocks, reliable cameras, and was part of a generation of filmmakers evolving the visual language of cinema. The filmmakers before him saw movies as a way to make photographs come alive, or to record a stage play with special effects, but Porter and filmmakers after him showed the world what was possible.

Figure 14. The Tramp in the machine: *Modern Times*, 1936, 87 minutes, dir Charles Chaplin



The silent masters. The filmmakers who followed—D.W. Griffith, Charlie Chaplin, Buster Keaton—would evolve these narrative techniques to tell their unique bands of stories. D.W. Griffith's controversial epic *Birth of a Nation* [1915] was a mind-numbing 185 minutes with 1154 shots. Charlie Chaplin's Tramp was loved throughout the world.

Twenty years after Porter, Buster Keaton made *Sherlock. Jr*. [1936], his greatest and most self-reflexive film. He plays the part of a poor projectionist who falls asleep during his movies. As part of his dreams, he finds himself stepping into and out of other movies, revealing to the real audience the techniques of the filmmaking, within a movie inside another movie. He was *very* meta.

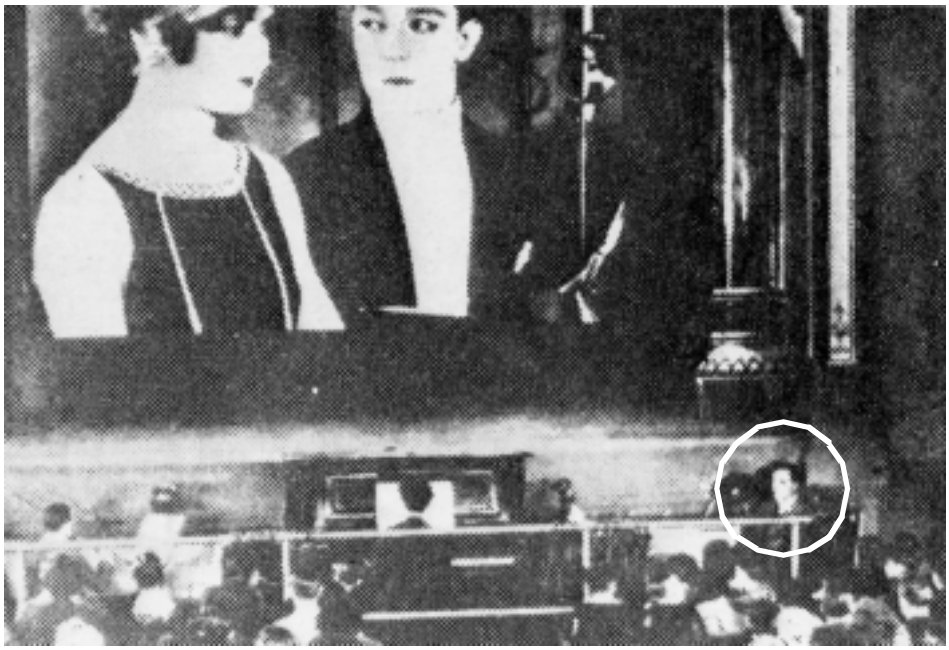


Figure 15. *Sherlock Jr*., 1945, 45 minutes, dir/editor Buster Keaton. (Keaton is in the circle.) In this scene, he sees his beloved on the screen and jumps *into the movie* to save her from the evil boyfriend.

Sergei Eisenstein defined the language of montage editing in the earth-shattering *Battleship Potemkin* (*Ten Days That Shook the World*) revealing to the world the power of Soviet filmmaking.

Figure 16. The promotion poster for *Battleship Potemkin*, 1925, 75 minutes, writer/dir Sergei Eisenstein. Below, a still from the oft imitated Odessa Steps sequence. I love the design of this poster. Soviet design and filmmaking influenced each other in their search for the best way to motivate the common man to action.

Figure 17. *The Jazz Singer*, 1927, 89 minutes. Al Jolson in black face talks to the audience.

Sound and color. The technologies that followed would enhance the filmmaker's pallet, but would not change it fundamentally. The techniques of continuity editing, fluid camera movement, and montage would all continue to apply to sound and color films. The coming of sound was denounced by studios as a passing fad, hailed by inventors as a way for small towns without orchestras to have movie music, and unappreciated by everyone. *The Jazz Singer* [1927] surprised everyone, even its producers. The star Al Jolson ad-libbed dialogue during the musical numbers, but for the most part, the film was treated like a silent picture. During the sound scenes, audiences were surprised that they were "overhearing" real people having a conversation. They loved it so much, that the worst sound picture would beat the finest silent film at the box office. Within two years, silent movies were dead. [Cook 1990]

In 1932, three MIT graduates perfected the expensive three-strip **Tech**nicolor process. Walt Disney used it first in his color cartoon, *Three Little Pigs*. Color would be adopted slowly because of its prohibitive expense.

Figure 18. Whoa dude, it's coming right at me!

In response to black and white television, Hollywood adapted color in the 50s and widescreen CinemaScope in the 60s. Hollywood turned to any technology to stem the tide of viewers flocking to TV, from making 3D "Depthies" to engaging our keen sense of smell with Smell-o-Vision. The first "Smellie" was *The Scent of Mystery*, 1959, 125 minutes. Its advertising slogan did not convince audiences of the narrative value of smell, "First they moved (1895)! Now they smell!" [Halliday 1993]

[1] *Citizen Kane*, 1941, 119 minutes, writer/dir Orson Welles, dir of photography Gregg Toland, ASC

By this time, audiences had become accustomed to the Hollywood conventions of narrative filmmaking, and the culmination of this style is exemplified *and defied* in Orson Welles's radical first film *Citizen Kane*.

Kane, Charles Foster Kane.

> Right away I want to make a distinction between "commandment" and "convention." Photographically speaking, I understand a commandment to be a rule, axiom, or principle, an incontrovertible fact of photographic procedure which is unchangeable for physical and chemical reasons. On the other hand, a convention, to me, is a usage which has become acceptable through repetition. It is a tradition rather than a rule. With time the convention becomes a commandment, through force of habit. I feel that the limiting effect is both obvious and unfortunate.... Orson Welles was insistent that the story be told most effectively, letting the Hollywood conventions go hang if need be. [Toland 1941]

In *Citizen Kane*, Orson Welles with his Director of Photography Gregg Toland used the latest technology in film stock and super-wide angle lenses to invent a new visual style called deep-focus cinematography. Welles told Toland "the technique of filming should never be evident to the audience" [Toland 1941]. By creating a frame with virtually infinite depth of field, the film most closely approximates how the eye sees. Welles staged his scenes in depth and eliminated much of the need for hard cuts, thus allowing the eye to ignore the film mechanism. Welles and Toland used the tools of the day and consciously discarded the common assumptions about how you visually construct a story. In collaboration, they evolved the language of cinema to a new level. The result is a combination of cinematic technology and narrative vision in which technology is used uniquely in service to the narrative. I strive for this same unity of vision and technology in my interactive work.

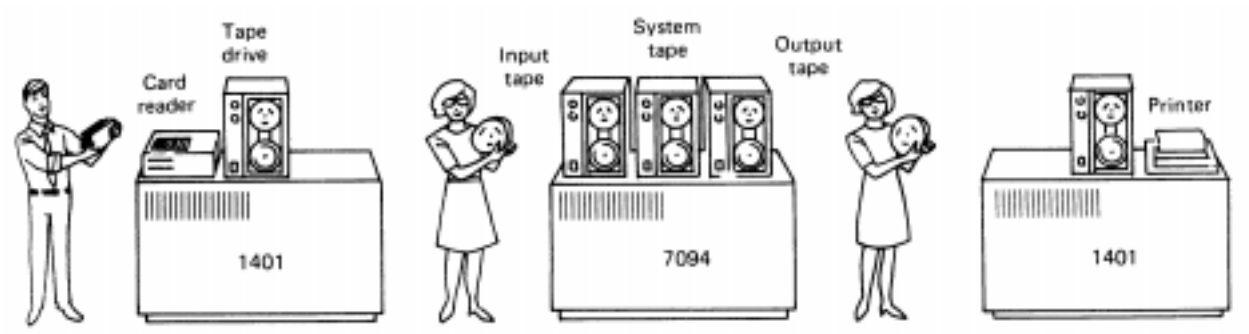
Figure 19. The grandiose Kane (Orson Welles) just before his fall.

2.2 The digital calculating machine

In 1945, fours years after *Citizen Kane*—perhaps the culmination of narrative film—ENIAC the first electronic computer was born. Unreliable, filling several rooms with vacuum tubes, and programmed with banks of switches, its performance rivaled the reliability, memory footprint, and usability of today's Microsoft operating systems.

The first computer systems, like ENIAC, were large, expensive, and completely non-interactive. Programmers wrote their first programs by flipping switches. Later they wrote machine language onto punch cards. These programs were run through the computer as a batch, to minimize idle time between jobs on the expensive computers. However, because programmers did not receive feedback until the batch was completed, a misplaced comma could ruin a whole day's work. [Tanenbaum 1992]

Figure 20. The typical process to program a computer. Man writes card. Woman with glasses carries tapes back and forth between computers. Obviously, some things have changed and, unfortunately, some have not. [Tanenbaum 1992] (!)



The programmer was the only user, and the only input was text. Usually, the only output was text as well...

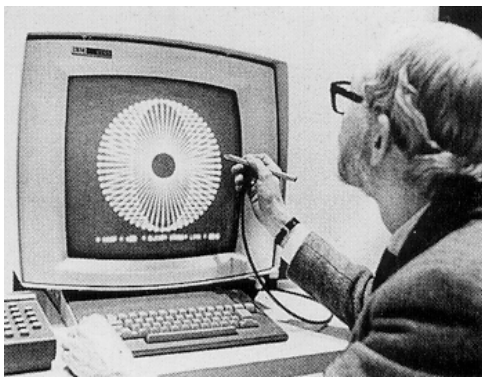
> But one day, the icy clamorous cardprinter room was turned into a whimsical cabaret: a clever young hacker had created a set of punch cards that worked like a piano roll and caused the card reader to chug out a recognizable version of the Marine Corps Hymn: bam-bam-THUMP bam-THUMP bam-THUMP-THUMP-THUMP. All day long, programmers sneaked away from their work to hear the mesmerizing concert. [Murray 1997]

2.3 Interactive computing

Obviously, programmers wanted to have a closer relationship with their machines. In order to provide faster feedback to programmers, MIT built the prototype Computer Time Sharing System [CTSS] in 1962. The system was invented to speed the task of debugging programs. Multiple people could access CTSS at the same time to run and compile programs interactively. Interactive access to a computer was considered wasteful, because a computer is idle for most of the time it takes for a person to type in a program. Making the system available to multiple users reduces the chance that the computer is not being used. CTSS became the model for all modern operating systems, especially UN*X. CTSS was the first step to interactive computing, but the interface to most computers was a teletype machine with a typewritten paper display, not a graphical one. [Tanenbaum 1992]

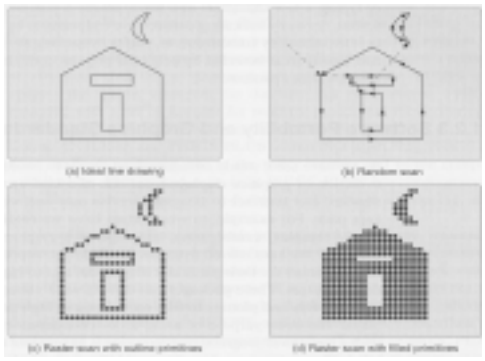
A year later, Ivan Sutherland published his landmark paper, *SKETCHPAD: A Man-Machine Graphical Communication System*. SKETCHPAD is the first graphical user interface to a computer. Completely built out of custom hardware, Sutherland uses a light pen and a vector graphics display to sketch graphics interactively. The vector display uses a "random scan" pattern to trace arbitrary line drawings, however the display is expensive to produce. [Foley 1994]

Figure 21. Sutherland using SKETCHPAD.



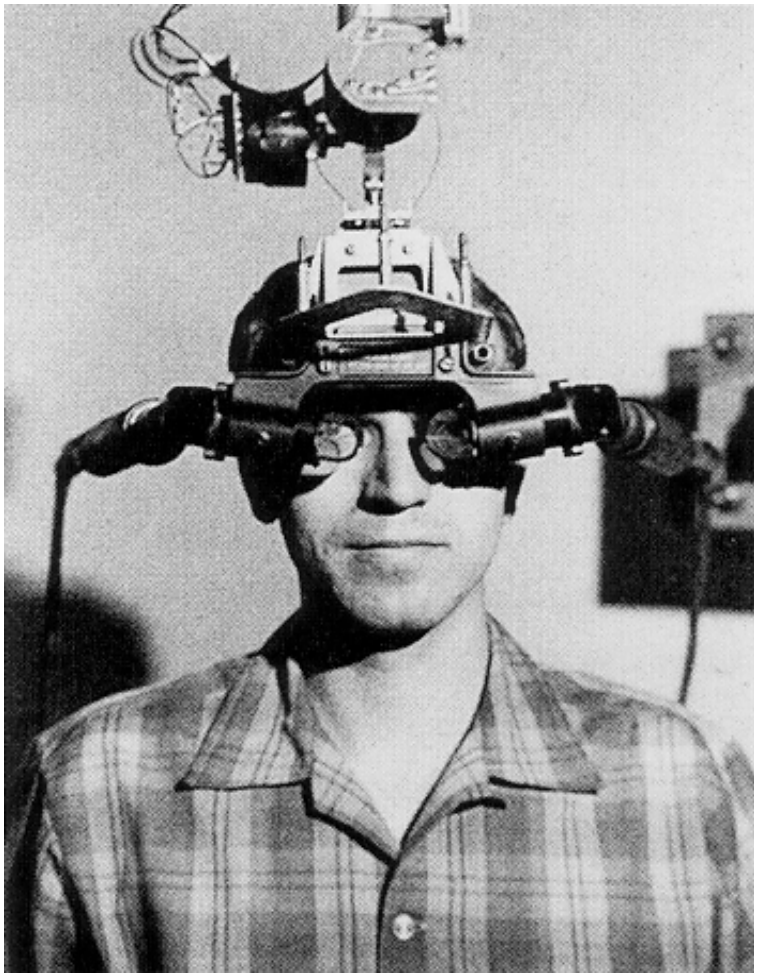
Sutherland's work inspired Xerox PARC and Apple Computer to integrate graphical user interfaces into their operating systems. They would use a much cheaper form of rendering called raster graphics, which divided the screen up into an addressable array of pixels. This "fixed scan" display could be used on televisions, making displays cheap and easy to produce.

Figure 22. The two types of graphic displays



Sutherland was an interface genius, grasping that the power of the computational medium could be used for visual communication. He invented the first 3D headset with a real time visual display. Although few people had access to his work, those who did invented the next generation of computers.

Figure 23. Ivan Sutherland, father of the graphical user interface, had intense vision.



With the invention of hard disks and access to interactive computing, the capacity to access text in a non-linear manner enables Ted Nelson at Brown to begin work on *Xanadu* [1965] during which he coined the term *hypertext*. Nelson's vision was to digitize and store all human knowledge into a single storage medium. Nelson was inspired by Vannevar Bush's original conception of hypertext in his article, "As We May Think," which Bush wrote in 1945! Nelson's vision has never been realized, but he showed a human need for interactive computing.

Figure 24. Bill Atkinson, father of HyperCard.

Hypertext was not available to anyone except programmers until Bill Atkinson at Apple shipped HyperCard [1987]. Its built-in scripting language, HyperTalk enabled anyone to write simple programs to access graphics, text, and serial devices, like videodisc players, and it was shipped with *every* Macintosh.

2.4 The birth of interactive cinema

Two things needed to happen for interactive cinema to be born. First, the computer needed a graphical interface and non-linear access to narrative images, like the optical videodisc. Second, someone had to ask the question, "What do computers have to do with video?"

Non-linear editors. In 1982, Montage released their first editing system, and in 1984 LucasFilm released a hybrid videotape/videodisc, non-linear computer editing system called EditDroid. It was a tool for the production of film and video, which used the computer as an intermediate editing medium for the eventual compilation of a film. EditDroid would foreshadow later tools like Avid's Media Composer and the Media 100.

Nicholas Negroponte forecasted a convergence of broadcast, print, and computation into one medium. [Brand 1987]

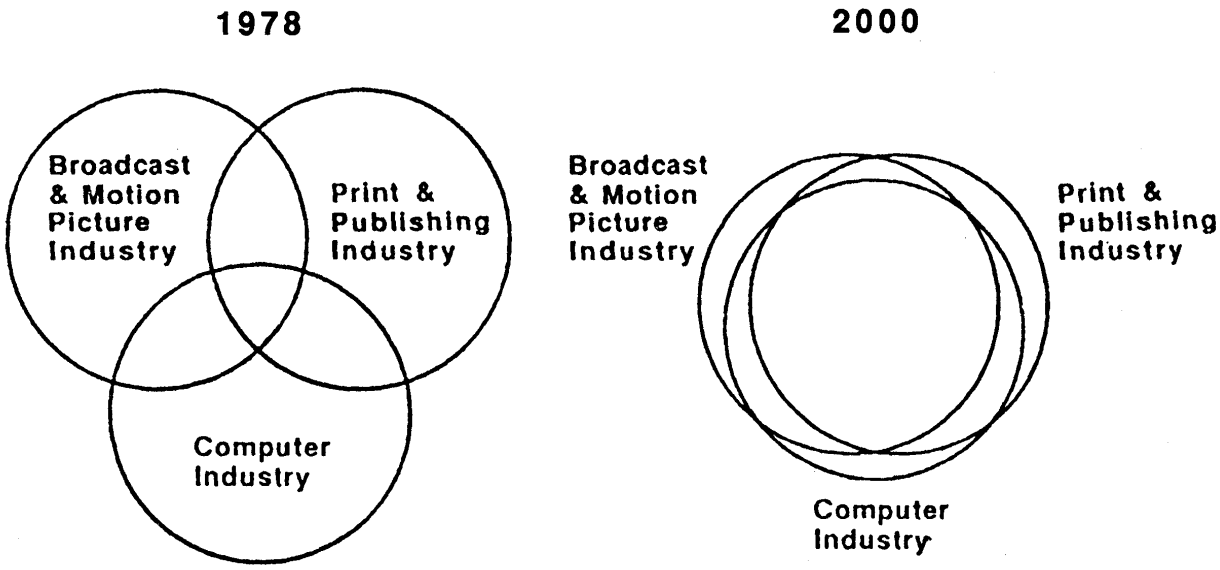


Figure 25. Negroponte's famous circles

At MIT, Glorianna Davenport thought about how she could use the medium to tell more interesting stories, and how would it affect the most lucrative medium of television. Andy Lippman asked, "What is the Television of Tomorrow?"

Figure 26. The Aspen Movie Map

Aspen. MIT's fledgling Architecture Machine Group, led by Nicholas Negroponte, began to answer those questions by building the *Aspen Movie Map.* It illustrated one application of non-linear access to photographic images. The system used optical videodiscs because of their ability to access video non-linearly. Unlike video cassette recorders, the first LaserDisc players also shipped with RS-232 serial ports enabling computer control. *Aspen* was one of the first virtual reality environments. By interacting with a touch screen, viewers could drive and turn onto any street of Aspen, Colorado, in different seasons and different times of day. In places like the Court House, users could go in and experience meeting the Police Chief through short narrative movies. [Brand 1987]

With HyperCard, the Macintosh, and optical videodiscs, anyone could create interactive movies using off the shelf hardware and software. The narrative work shot by Prof. Ricky Leacock and edited by Glorianna Davenport provided the first participation in a new interactive form for these filmmakers. Finally, the computational medium had all the technology it needed to begin telling visual stories, and storytellers searching to extend the language. After Leacock left the Media Lab, Davenport would found the Interactive Cinema Group [1988]. Thus interactive cinema was born.

Figure 27. Nicholas Negroponte pondering the future.

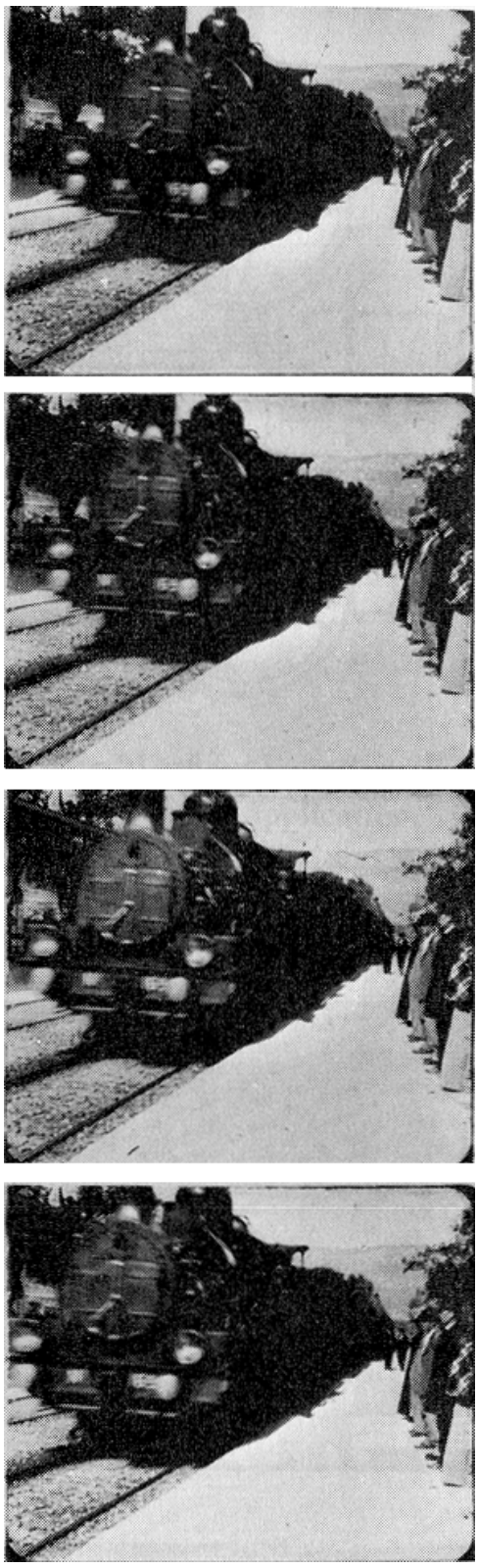
2.5 Developing a new expressive medium

> Individually it's difficult for people to imagine a thing that hasn't been invented yet. If you had asked people in 1955 what new kind of toy they wanted, nobody would have said, "a plastic hoop that I can rotate around my hips." And extrapolating from the current interactive entertainment landscape is risky too, because it may give us a future that unconsciously incorporates the limitations of the past [Laurel 1989].

What conclusions should one draw from all this history? The development of a new expressive medium happens in three stages. First, the medium is invented because of an advance in technology: moveable type created the print medium, motion picture cameras enabled cinema, and computers enabled interactive media. The next stage of development is defines an expressive language. Through experimentation, luck, and observation, the practitioners in the new medium begin to learn new techniques of expression. The first forms of expression will inevitably be unsatisfying translations of older media into the new medium. The first films were essentially stage plays acted in front of a camera. Refinements in the technology and the economics of the new medium enable whole new forms of expression. Finally, as the new medium matures, the tools for creating the medium become stable and cheap (relative to the cost at the invention of the medium). At that point, new practitioners of the medium will step in and discard the assumptions of the past and refine the new conventions for the medium.

The computational medium is on the brink of entering into the final period of development. The machines have become stable and accessible. A new generation of computational designers and storytellers are beginning to cast off the assumptions of the old and discover the new.

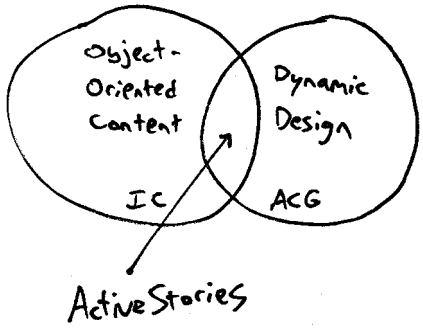
Figure 28. The future rushing to meet us.



3.0 Reflecting on the influences

My first experience with the Interactive Cinema Group [IC] was when my friend James Seo took me to his UROP in Macondo E15-441—the home of the group. The lights were dimmed, the Macintosh computer screens were surrounded by little movie boxes hanging from the ceiling, and I was accosted by David (Baby Dave) Kung MS95, who asked "Who are you?" and "Seen any good movies?"

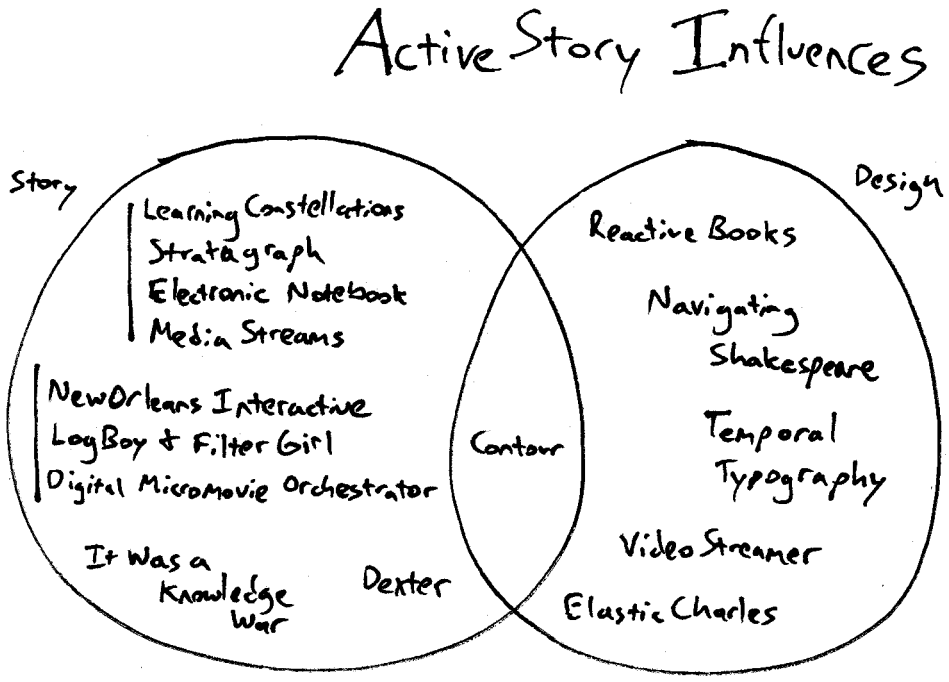
Figure 29. ActiveStories fit right in the middle.



3.1 Object-oriented story and dynamic design

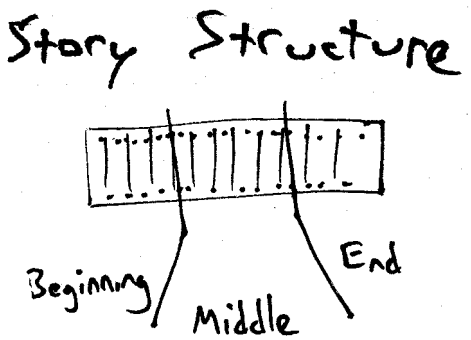
All the people in the Interactive Cinema Group asked questions about the construction of stories, "Did you notice how Hitchcock edited this sequence to imply that she will betray him?" or "Have you looked at the formulaic structure of all soap operas?" This environment encouraged me to look at the techniques that filmmakers use to build sequences of shots into sequences and build sequences into conflicts and resolutions.

Figure 30. In the left circle are the content-centric projects, and on the right are design-centric projects.



The idea that influenced me the most was the concept of object-oriented content. By determining the structure of the story you want to tell, you can to break the content into semantic chunks, i.e. story granules. Then, you label each granule so that you can retrieve it for later use. This object-oriented approach would be a starting point for my own exploration of story.

Figure 31. Deconstruct the story into component parts

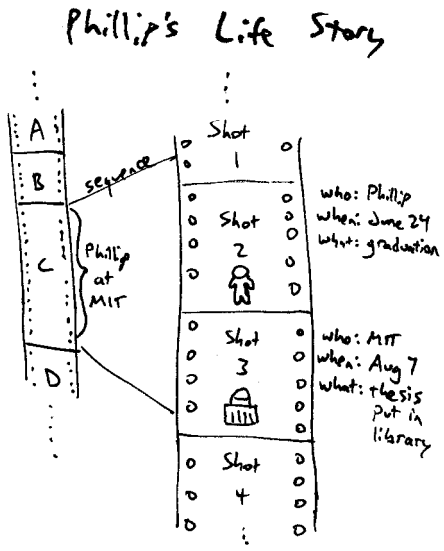


Simultaneously, I heard whispers of "cool stuff" happening on the other side of the building in the Visible Language Workshop [VLW]. VLW graduate student Robin Kullberg MS95 gave me my first glimpse at the temporal typography and dynamic design being developed there. The VLW pioneered the concept of an information landscape where typography lives in a 3D-rendered world. Information is longer fixed on a page, but instead is organized in an explorational space. Continuing design research in the Media Lab, John Maeda came to start a new group to explore the intimate connection between expression and computation—the Aesthetics and Computation Group [ACG]. His approach applies to all kinds of information, not only typography, and I began to explore with him the connection between dynamic design and visual storytelling.

Object-oriented story and dynamic design are the two threads of research I will describe in this section. They form the example base on which I built my assumptions about what makes a good interactive story.

3.2 (De)constructing a story

Figure 32 A hypothetical film about my life



Before we jump into the examples, I need to define the basic process of creating object-oriented content.

> A narrative is *a chain of events in cause-effect relationship occurring in time and space*. [Bordwell 1990]

All narratives have a structure. By studying its structure, you can determine how a narrative builds meaning from moment to moment. In a film, the smallest chunk of meaning is usually a shot. Sequencing shots makes a scene. Sequencing scenes makes a movie. Choosing the level of detail at which you want to manipulate meaning is also called choosing the granularity. A system can work at the granularity of a shot, scene, sequence, or even an entire movie depending on the needs of the creator.

Once you choose the granularity, then you must decide what variables you are interested on operating on. Usually the most salient information is contained in the 5 Ws—Who?, What?, When?, Where?, and Why? All the systems described in this section provide some access to this information.

When that information is linked to a list of in and out points on a video cassette or the frame numbers of a videodisc, then you have a process to operate on the story granules. This basic system of a slot (Who?) and value (Phillip) is a representation of the content of the clip to the computer—a knowledge representation. With this representation, if you ask "Give me the video clips which contain Phillip?", the computer does not need to process the video image to look for a pattern of pixels which resembles me, instead it looks through its list of frames for the ones marked "Who? = Phillip." This places the burden of recognition on the person annotating the movies, instead of the computer. This pro-

cess mimics how film editors' brains work, mentally connecting important information to the sequences of film frames hanging in their bins.

What follows are four examples of annotation systems, Learning Constellations, Stratagraph, The Electronic Scrapbook, and Media Streams. They are tools for the search, retrieval, and organization of video information, but each has different users in mind, the Video Ethnographer, the home consumer, and the Garage Cinema filmmaker.

Constellations of annotation. Ricki Goldman Segall is a professional ethnographer who uses video as a tool for her research. In her process of observation, she wanted to annotate her video data to help her communicate contextual information to her audience:

> *Thick descriptions* are descriptions that are layered enough for readers/viewers to draw conclusions and uncover the intentions of a given act, event, or process. [Segall 1990]

Using a videodisc player connected to a computer which served as her notebook, she created a hypermedia system which links her journal entries to video clips. The system allowed users viewing her journal to add their own observations creating constellations of information.

Figure 33. The interface to Learning Constellations, Segall's hypermedia system

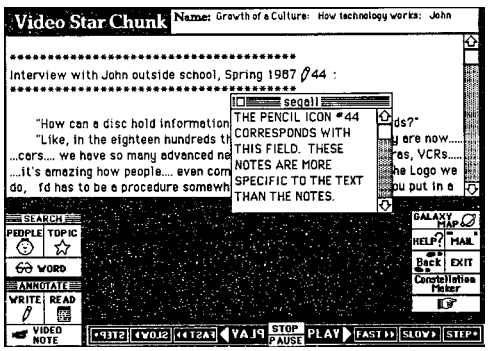
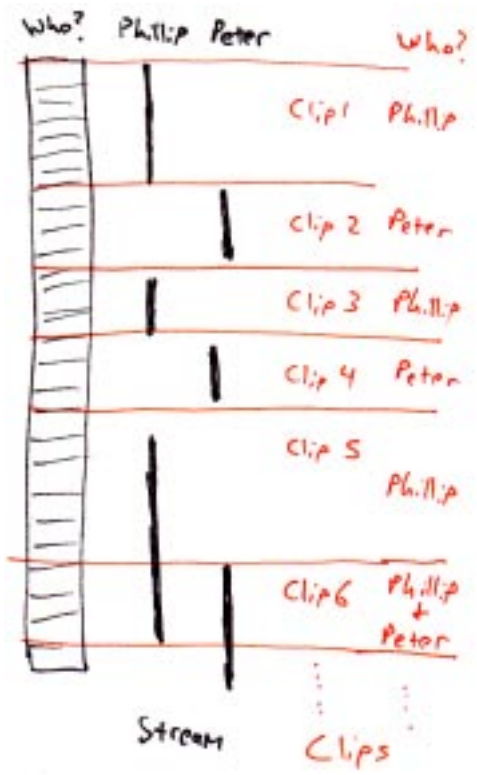
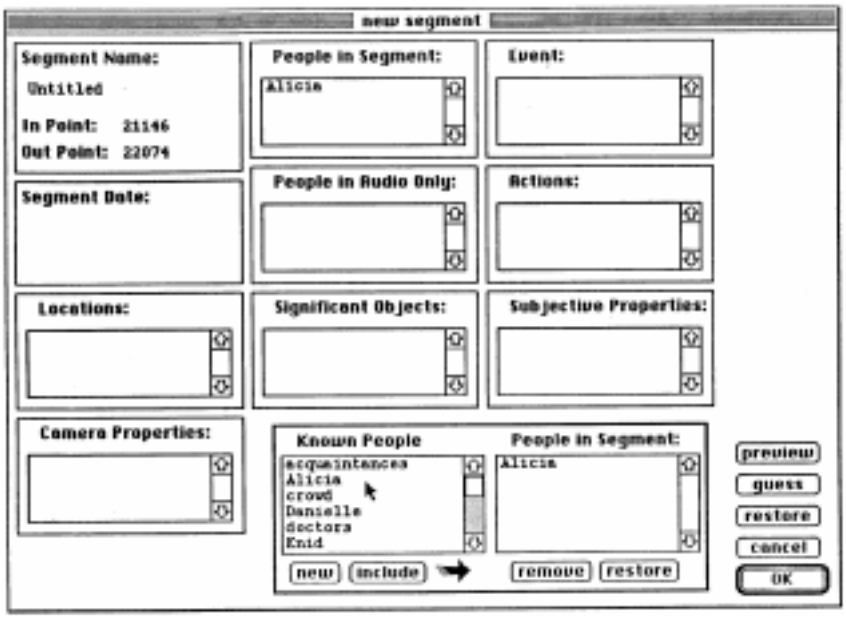


Figure 34. A movie contains a conversation between me and Peter Cho MS 99. With a stream-based annotation scheme, you can annotate the conversation as a single clip, annotating the changes in Who?. On the other hand, a clip-based annotation scheme would have to segment the conversation into six separate clips. The system would also have to handle the possibility that both Peter and I are in the video clip with multiple values in the Who? slot.



This concept of content description evolved in Thomas G. Aguierre Smith's Stratagraph in which he describes a search and retrieval system based on stream-based descriptions of video. Smith created his system to annotate his own anthropologist's video notebook. While annotating his content, he ran across a segmentation problem. In a clip-based annotation system, a value (Who? = Phillip) is assumed to be true for the entire video clip. If one annotation slot (Who?) changes over time, then either you have to segment your video into smaller pieces or label the changes over time. Segmenting video into smaller and smaller pieces increases the overhead of annotation and can make it difficult to make meaningful video clips. A stream-based annotation system annotates the changing attributes of the video, essentially labeling every frame with its content. [Smith 1992]

Figure 35. The HyperCard interface to The Electronic Scrapbook



Also faced with an abundance of home video, Amy Bruckman developed the Electronic Scrapbook, a system for archiving, saving, and indexing home video. Her system used a scrapbook metaphor allowing users to place video clips onto a HyperCard page, as if they were photographs on a page of a scrapbook. The Electronic Scrapbook could compile a page of "Important Firsts" or "Alicia's Birthdays" using a clip-based annotation system. The user toiled in front of a HyperCard stack connected to a videodisc player.

The video content would appear on a second NTSC monitor, while the user typed in keyword annotations. Both Stratagraph and Scrapbook required a lot of manual labor in front of an interface that was not much fun to operate. The high annotation overhead and the cumbersome interfaces made few people use these systems. [Bruckman 1991]

Icon-based streams. The most ambitious project in annotation has to be Marc Davis's Media Streams.[1] His goal is to provide a generic language for categorization of all video everywhere. This system would be able to locate found footage[2] from a large unstructured database to edit into sequences in new ways. Davis imagines a global database of clips annotated in his iconic language. He insists that a description language built on keywords has five inherent problems:

1. Keywords do not describe the complex *temporal* structure of video and audio information.
2. Keywords are not a *semantic* representation. They do not support inheritance, similarity, or inference between descriptors. Looking for shots of "dogs" will not retrieve shots indexed as "German shepherds" and vice versa.
3. Keywords do not describe *relations* between descriptions. A search using the keywords "man," "dog," and "bite" man retrieve "dog bites man" as well as "man bites dog" videos.…
4. Keywords do not *converge*. Since they are laden with linguistic associations and are not a structured, designed language, keywords as a representation mechanism for video content suffer from the "vocabulary problem."
5. Keywords do not *scale*. As the number of keywords grows, the possibility of matching a query to the annotation diminishes. As the size of the keyword vocabulary increases, the precision and recall of searches decrease. [Davis 1995]

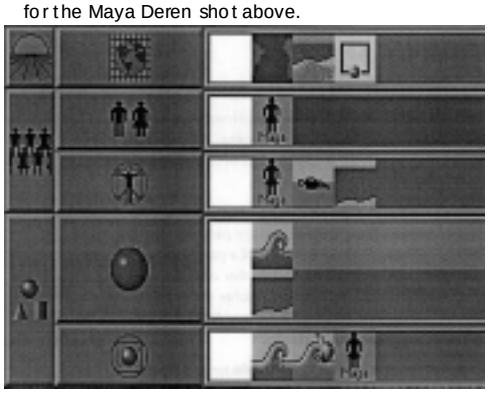
As a result, Davis designed a stream-based iconic language to replace keywords. He believes that video when properly annotated in his language is maximized for retrieval and repurposing into new compositions.

[1] Davis continues his research at Interval Research corporation, looking to build a commercial product.

[2] Found footage is random scraps of film not shot for a particular production, i.e. the bits of film left on the editing room floor.

Figure 36. A shot of Maya Deren, experimental filmmaker

Figure 37. Davis's stream-based iconic representation for the Maya Deren shot above.



[1] Garage Cinema is the movement of amateurs to reedit found footage to make new episodes of TV shows and wacky videos.

He envisions that large on-line databases of found content will grow into libraries for Garage Cinema[1] filmmakers to find material from. In a way, this vision mirrors Ted Nelson's hope to create a global hypertext repository to store all human literature, except this time for video.

The annotation problem. All of these content annotation systems arose from the needs of non-moviemakers users to be able to find material from large databases of video: videos from ethnographic studies or the unwieldy shoebox of home movies. Documentary filmmakers tend to shoot first and ask questions later and record hours of footage. Documentarians follow developing processes and may not know exactly what is "in the can" until they review the all the footage in editing. Providing annotation tools gives these users an interface to add contextual information to their content. The knowledge representation makes two things possible: the machine can find clips for you from your database, and it might make connections that you may have overlooked.

Fiction filmmakers tend not to have large unstructured databases because the fiction filmmaking process keeps track of every shot from inception, to storyboard, to film, to lab, to editing room, and finally to the theatre. Therefore, fiction editors probably (but not always) have an idea of the material on a tape before they edit it. Editors usually receive a log of every shot from the set including "circle takes" of the best performances. As a result, even today's professional editing tools have meager facilities for annotating video content.

The biggest problem with these annotation tools is the labor cost of annotation. None of them have automatic ways to aid the process of labeling, segmenting, and storing your clips. Users with large pre-existing databases of content would have to spend significant amounts of time to

organize their video. After all that work, these systems aid users in retrieval of content, but do not automatically edit sequences for viewing.

Reduce the overhead, reduce the scope. My efforts at annotation point toward real-time interfaces for annotation. My tools become part of the process of watching, not separated into a different interface. Rather than using a generic keyword or icon-based language to annotate my video, I develop several rapid processes of shallow annotation, each designed to communicate the content of the video to *some other computational process*, not for a person. [see "Representation without taxation" on page 104] Rather than asking, Who? or How?, I want to know exactly when someone is speaking a word. The annotation is not necessarily useful in every domain, but for the expressive display of the words of a poem, it is very useful.

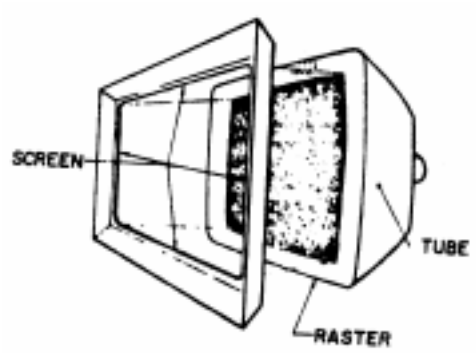
The methods of annotation described in these projects are primarily designed to help a computer find content for a *person*. However, the next step after deconstructing a story into granules, is to reconstruct new stories from the granules. The following systems strive to become "editors in software."

3.3 Narrative engines or the editor in software

Three projects constitute the first stage of the development of the editor in software. New Orleans Interactive was the first system which could compile a sequence of video clips and play them out for you. The process evolved when Ryan Evans created the graphical annotation tool called LogBoy and a constraints-based selection algorithm called FilterGirl. Evans worked hand-in-hand with Mark Halliday who fed Evans's system with compelling fictional content and a narrative structure. The process began with Glorianna Davenport's idea for an "evolving documentary."

The evolving documentary. Glorianna Davenport has often told me, "It's always more fun to make a documentary than to watch one." When you make a documentary, you go into a situation not necessarily knowing what is going to happen. You record the events that happen and discover different perspectives on the story that evolves in front of the camera. When you have to stop shooting, you have a rich understanding of the different issues and perspectives on the subject you started out to document.

Figure 38. The boob tube



If television is your distribution medium, commercial considerations force you to take the story that took you months or years to understand and capture and then cut it into a thirty-minute "After School Special." You have to assume the lowest common denominator of knowledge of the subject by the audience. Because of time/cost constraints, you must either provide between in-depth converge on a single aspect of the story or shallow coverage of the whole story. Even though you, as the documentarian, understand the different biases of your subjects, you can not share all of that knowledge with a large audience. Part of Davenport's greater ethical goal for the evolving documentary is to use computation to give viewers access to a broad base of content which contains multiple viewpoints, letting the viewer shape what is most important to them. By covering a multiplicity of opinions, users can make their own decisions about the biases of the speakers.

New Orleans. *New Orleans: A City in Transition* is the first project in this category, a project in which Glorianna Davenport spent three years documenting the evolution of a city. The 50 hours of footage was compiled, annotated, and edited into 3 hours of edited sequences. The footage turned into two different forms, an edited film and a videodisc system designed for "augmented delivery." *New Orleans Interactive* contained all of the supplementary content surrounding the project: filmmakers' notes, journal entries,

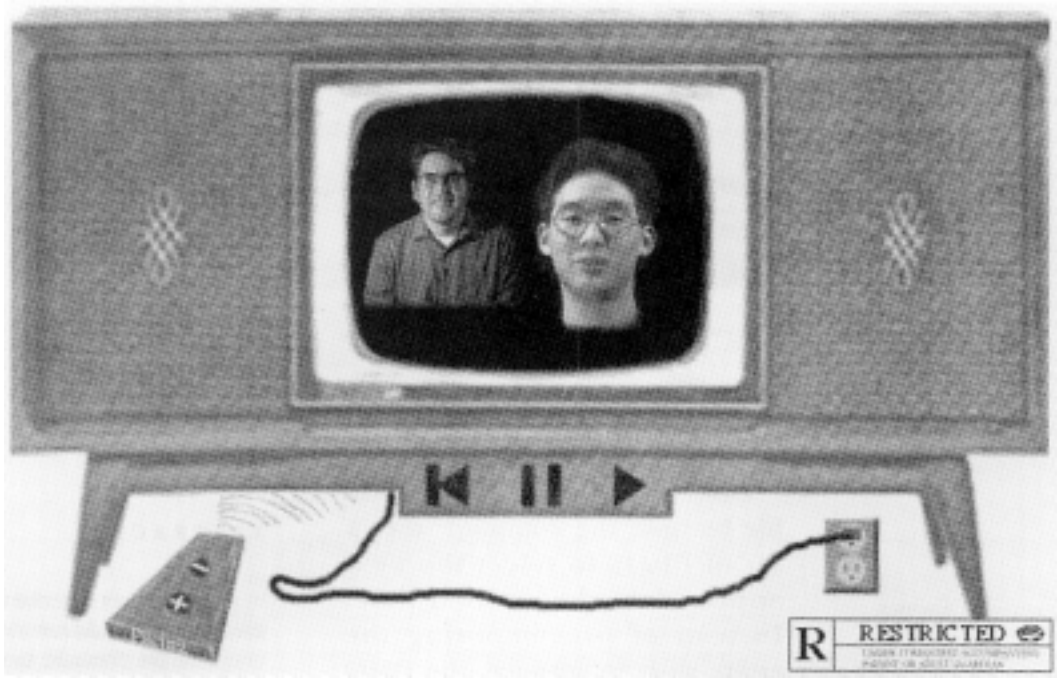
transcripts, legal documents, student papers, etc. The system combined static text display with real-time access to video and ran on a Dec MicroVax II, using MIT's fledgling X-Window system. The system provided architecture students access to the database allowing them to get contextual information about the clip currently being played or to query the system on a topic and receive a mini-movie based on the query. The clips were annotated in a strict way which guaranteed that clips would appear in conflict-resolution pairs. The story continued to evolve, and three years later Davenport went back to New Orleans to shoot follow-up interviews with the people involved. These clips became part of the database and grew the story.

Davenport describes the interactive story as "not a branched structure but a sort of free-form, associative information resource." It was available to architecture students as a way to encourage learning through exploration and interaction. [Davenport 1987]

A fundamental question for the Interactive Cinema Group became "how can a computer know enough about the content and the process of telling to tell a coherent story?" [Davenport 1989]

DMO, LogBoy, and FilterGirl. Mark Halliday MS93 worked closely with Ryan Evans MS94 to create the Digital Micromovie Orchestrator [DMO] a constraint-based editor in software to playout multivariant movies. Halliday and Evans would create several stories together: *IC Portrait*, *An Endless Conversation, Just One Catch* and *Train of Thought*. Evans's MIT background made him particularly interested in building a storytelling system. Halliday was more concerned with pushing the boundaries of interactive narrative. *An Endless Conversation* was a movie designed to run forever. *Conversation* contained a large database of questions and answers regarding the Interactive Cinema Group.

Figure 39. The Endless Conversation interface starring Thomas Smith MS92 and David Lyn Kung MS95 (right).



Users turned the TV on, and the conversation began until the pulled the plug. The user had only a few ways to interact with the movie: selecting R or PG, choosing a fast or slow pace, and the ability to rewind the conversation. The DMO system selected content in real time based on the current user preferences. The system provided a pretty funny conversation which played out passively, requiring no interaction. However, that was also the problem, the user had few ways to affect the playout or realize that they had an effect. Nor did they have any narrative reason to interact. Ryan Evans MS94 decided to approach the orchestration of content at the lowest level. He built a system to edit sequences from component shots. Evans evolved the DMO into a new pair of tools: LogBoy and FilterGirl. They were a pair of tools for interactive movie makers to annotate and create a set of rule-based algorithms for story. Ideally, this system could assemble raw footage into a movie according to rules of continuity and narrative.

LogBoy was a WYSIWYG annotation tool which allowed the user to drag and drop video clips into different bins and log the clips with different keywords. LogBoy's counterpart was FilterGirl—a constraint-based rule editor. The storyteller used FilterGirl to create semantic filters to define how a story could be played out.

Filters were composed of rules that defined the narrative progression of the story, guaranteed spatial continuity, and read in user preferences. At run-time, the playout system applied the author's filters to the database and found all the clips that satisfied those filters. If the system could find what it needed, it could play a complete movie. Evans called the new form a "multivariant movie," but warns users:

> LogBoy and FilterGirl do not make up a system that understands stories. Instead they provide a way for moviemakers to encode knowledge about the story they are trying to tell and the ways it can be changed for multivariant playout. [Evans 1994]

Figure 40. An example of how multiple filters constrained the selection of movie clips

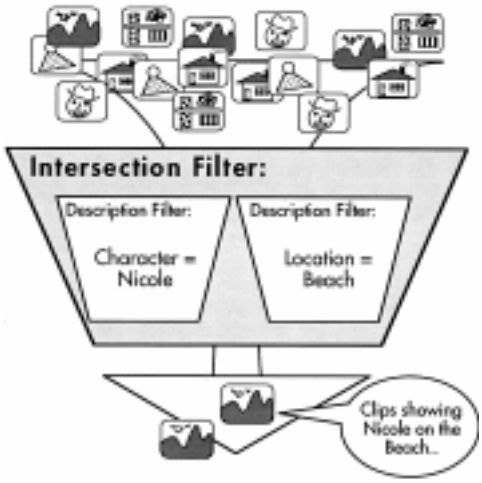
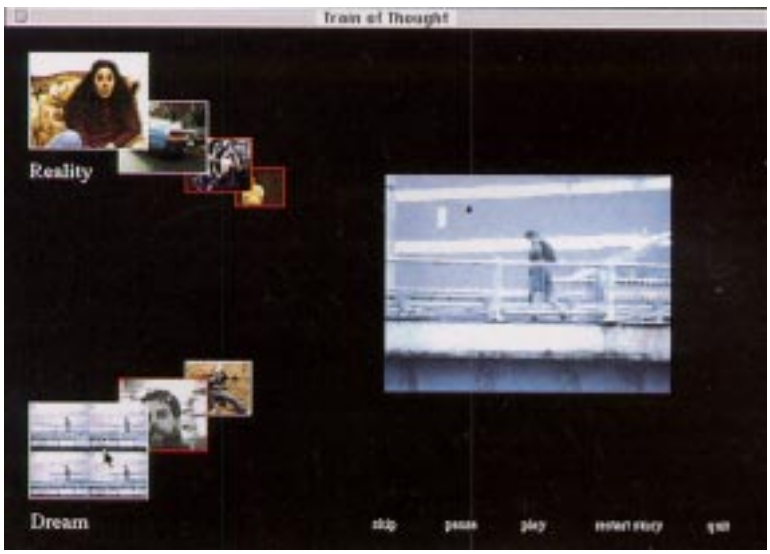


Figure 41. The interactive interface to *Train of Thought*. The interface still centers around a central playout window and linear presentation.



The coherence of the narrative depended solely on the author's ability to properly define filters and shoot enough footage to satisfy all the possible combinations of filters. It proved difficult for authors to shoot the density of clips needed for a multivariant story.

Train of Thought used positional editing—placing video clips in 2 1/2D space—to represent temporal and narrative relationships. The story used visual design to guide user interaction, but the visual display was mostly static. Most importantly, users had little meaningful interactive control.

Train of Thought began to bring together elements of visual design with procedural storytelling, but the techniques of dynamic design had not yet reached the group. The interface to *Train of Thought* gave the user no meaningful ways to affect the story. Users could choose to follow a branch of the story or to view more information at a node, but the user primarily sat and watched. [Evans 1994] [Halliday 1993]

Gilberte Houbart MS94's thesis *It was a Knowledge War* was also a multivariant movie system. It was an attempt to make a "smart VCR": her system represents in software the opinions of the speakers, so that users can specify different viewpoints. A single grey knob and a "Story" button were its interactive interface. Both *Knowledge War* and *Train of Thought* systems could assemble sequences from a database, but viewers could communicate little more than a few preferences to these systems. [Houbart 1994]

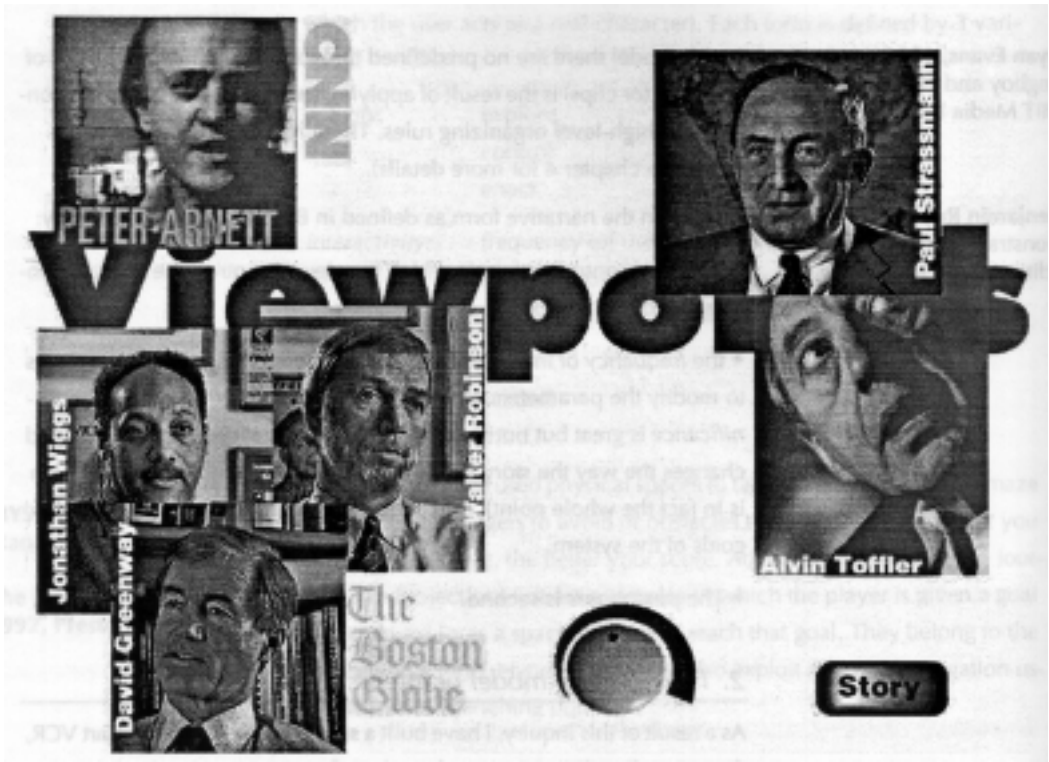


Figure 42. The interface to *Viewpoints on Demand:* the knob and button are the user's interface to the story.

Michael Murtaugh MS94 worked with Houbart on *Knowledge War*, and wrote a tool to help make it—ConArtist, a graphical interface built in Macintosh Common Lisp [MCL] to Ken Hasse's Framer knowledge representation system.

Contour and Dexter. Murtaugh's experience with *Knowledge War* would enable him to make a breakthrough in the fundamental way a "narrative engine" chooses clips from a database. The previous systems use rule-based algorithms and a complicated annotation system which turned out to be very cumbersome for authors to use. It was difficult to write a story, specify the correct story filters, and shoot the required content to make a coherent and compelling narrative. Murtaugh took a different approach.

Figure 43. The initial screens for *Boston: Renewed Vistas* and *Jerome B. Wiesner: A Random Walk*

Granules of story. Rather than trying to orchestrate the assembly of clips at a shot-by-shot level as Evans had done in LogBoy and FilterGirl [see "DMO, LogBoy, and FilterGirl" on page 45], Murtaugh fixes his granularity at short, edited, atomic clips. Each clip contains a self-contained story and stand on its own. Murtaugh system concentrates on creating a continuity engine, a clip selection algorithm based on *thematic* continuity. Thus, he simplified both the logging task and the clip selection task. Each clip could be annotated with an *arbitrary* number of keywords, e.g. Who?, What?, When?, Where?

This representation required no initial categorization of the keywords. There were no required slots to fill in. You annotated the clip with the keywords you felt were most appropriate. If you annotated the entire database with a relatively small number of keywords, then by necessity some clips would have keywords in common. Density was again a big obstacle because the database had to have many annotated clips for the system to find a coherent path through the content.

Figure 44. This sequence of four successive screenshots illustrates a Contour sequence. The largest clip in the center is currently active and its keywords are activated along the edges of the frame. Future clips to be played are ranked by relevance and sorted visually by size.

However, this annotation scheme made it easy to find the thematic relationships between clips: the higher the number of common keywords, the greater the thematic continuity. It was a very simple algorithm with one simple rule: more is better. Murtaugh combined this technique with a selection algorithm called a spreading activation network. This system is based on a semantic network of agents. Each piece of content monitors the activities of the others. When a clip is activated by the user, the other clips check to see if they are related to it. If they are, then they activate themselves increasing the likelihood that they will be played in the future.

The spreading activation network is simple and clean to implement and has several positive emergent properties: a user's history of browsing influenced the future sequencing of clips, the system had a sliding scale of interactivity, the system would recover gracefully from interruption, and lent itself to a dynamic visual representation.

Elastic Boston. *Boston: Renewed Vistas* was the first evolving documentary to be built with this system. Its visual interface is striking, a field of all the video clips in the system arranged in 2 1/2 D space. Along the edges of the screen are the keywords which are contained in the database. In the previous multilinear systems, users had a very limited model of interaction, two or three knobs which could be adjusted within a range. In *Renewed Vistas*, the user has access to all the clips in the database all the time. The user can click on a video clip to play it or on a keyword to activate that theme. The field of content reacts by reorganizing itself according to the each clip's relevance to that keyword.

Unlike any of the previous movie "orchestrators," Murtaugh's system allows the user to exercise a great deal of control over the system. Interactivity was not restricted to clicking the pause button or turning a knob, instead Murtaugh gave the user access to the entire database of narrative content. Moreover, because of the nature of the spreading activation network algorithm, users could learn how to specify what kinds of stories they wanted to hear.

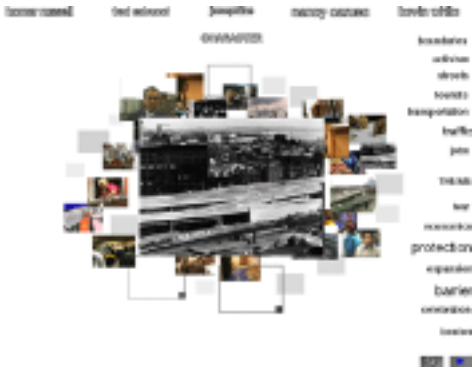
User queries. The spreading activation network is based on a system of additive weights and a finite total amount of energy. The more positive weight a keyword receives, the more likely it is that a clip containing that keyword will be played. The energy of each keyword was expressed graphically in its size and color. However, it is just as easy to negatively weight keywords to suppress content containing the associated topics as well. Clicking a keyword gave that topic its maximum weight, shift-clicking gave it a negative value. Users could ask the system a complex query of A AND B AND C NOT D by clicking and shift-clicking the keywords, and the database interactively responded.

Figure 45. This screenshot specifies a user query of "Future" AND "Homer Russell."

Moreover, when a clip starts playing, the keywords that annotate that clip are also positively weighted, activating thematically-related clips. The act of playing a clip has prepared the database to choose a new clip for you, and an Autoplay feature does just that. The Autoplay allows the user to sit back and receive a passive presentation until they want to push the movie in a direction. Then they can click on a character's name or theme, and "lock" that theme in. The system does not interrupt the current playout, but instead guides the *future* of the presentation to satisfy your new preferences. When clips are finished playing, they turn in to empty rectangles and visually represent a user's history of use.

Contour provided the user a "sliding scale" of interaction from no interaction to complete interactive control. The system could Autoplay a whole documentary for you, and you could take control at any time to see more content on a character or a theme.

Figure 46. This shot illustrates the weighting "Character" super-keyword positively and the "Time" and "Place" super-keywords negatively.



User-specified stories. The spreading activation network provided a way to specify a hierarchy of keywords. Keywords could inherit weight from super-keywords that defined categories. For example, the keywords "Nancy Caruso," "Kevin White," "Homer Russell," "Frank Salvucci," and "Josephine" were all grouped under the super-keyword, "People." By manipulating the People keyword, you could positively weight all the characters at once, effectively asking for a character story from the system. Negatively weighting the People category was the equivalent of asking the system for a breadth-first search, "show me something about each character." The user could then specify a specific *kind* of story they wanted to hear by appropriately weighting keywords. For instance, "I want to hear about the North End from Nancy Caruso's perspective" can be expressed by clicking on the keywords "Nancy Caruso" and "North End." "I want to hear everyone's perspective on economics" can be expressed by negatively weighting the "People" keyword and clicking on "Economics."

Breakthrough. The system is an jump forward in both expressiveness and visual design. The database's state is transparent and visually available to the user. The user had a sense of what they had seen, what is coming, and how much was left. The interface is fluid and reactive to the user, helping them to build intuition about how to specify what they wanted. To add content to the story, you annotate new clips with keywords. The system generates a new semantic network at run-time so no additional manual work had to be performed to add new material to the database.

The system finally lives up to the term "evolving documentary," in that the story can evolve over time, as new content is added, and the story evolves during each viewing for each user.

Figure 47. A sequence of material using the Dexter interface. Unlike Contour, Dexter had no Autoplay mechanism, because at the time the Web did not support push technology. The system depended on the user to navigate the content using the concept map on the left, viewing content on the right. Content could take any HTML form: text, photograph or movie.

Dexter. Murtaugh adapted the algorithms to create a Web-based version of Contour, first using a script to generate properly linked HTML pages. It exhibited none of the dynamic properties of the original, so he implemented a java-based version of the Contour algorithm called Dexter. He applied Dexter to a new evolving documentary project, *JBW: A Random Walk Through the Twentieth Century*. Dexter uses a similar spreading activation net and annotation scheme as used in Contour.

Dexter's visual interface is completely different from Contour's, using a static "concept map" with four discrete activation states as a way to communicate how clips are related to each other. Each keyword in the system is represented by an icon on the map, and would light up according to four different states: active, decision, suggestion, and inactive. Dexter has no AutoPlay feature because of the inability of the Java implementation at the time to be notified when a video clip has finished downloading and playing. On the other hand, Dexter does have a dynamic text index of content, which would visually show the state of the database, in a similar way to the field of video clips in Contour.

All together, Dexter is harder to use than Contour, mostly because of its static map and discrete visual representation of states. Still, compared to other systems for browsing content on the Web, it is more effective because it presents a consistent visual interface, while content is displayed in a persistent content frame.

Semi-coherent databases of content. Both in Contour and Dexter, Murtaugh used databases of limited scope. By using the granularity of stories instead of shots and a database of related content, Murtaugh made it much easier for the system to successfully discover thematic links. The database could be well described using a small number of keywords, and yet, new content could be added at any time. The problem of having the machine assemble a story from random shots proved to be too difficult, both for the computer, and the filmmaker trying to shoot for it. However, if artistic expression is the ultimate goal, is it necessary to build a generic all-purpose storytelling system?

Figure 48. A closer look at Dexter's dynamic materials listing

Dexter definitely inspired me to continue programming in Java, because I could see that the language had a lot of potential. Dexter contained a dynamic text index of the materials in the Dexter database. It was my first exposure to dynamic design implemented in java. Contour and *Boston: Renewed Vistas* were only available to people who could physically travel to the Media Lab and see it. Dexter and *JBW* were immediately available to the entire java-enabled world, a very compelling audience, even for family movies.

Contour's well-designed visual interface together with a parameterized authoring system made me want to experiment with my own content. I annotated over 100 of pictures from my bike trip through Portugal, and I ran into scalability problems both with the unorganized visual space and keywords. Most importantly, it was impossible for me to actually define a story that I wanted to tell with my pictures. The system afforded me no way as an author to leave my intention in the system except in the keywords. [Murtaugh 1996]

Dexter and Contour challenged me to work on my own systems and provided solid models to start from. They raised the bar of storytelling and design, creating a dynamic visual

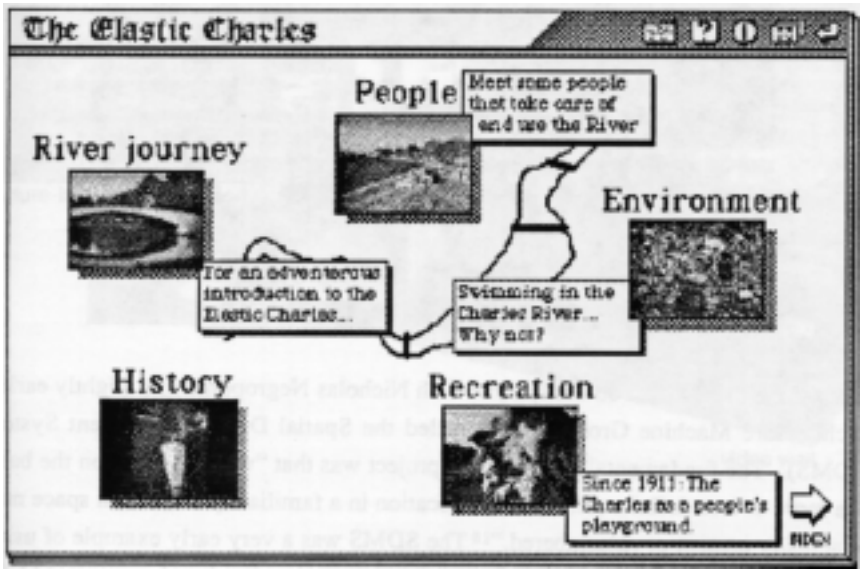
display that reflected the state of the narrative database. I turned to the VLW and ACG to learn more about dynamic design.

3.4 The design task

Four projects touch on the design questions which I explore in this thesis. The Elastic Charles was the first hypermedia journal to combine text, graphic, movie, and photograph. The VideoStreamer was a visual representation of video from which I drew much inspiration in the simplicity of its form. Navigating Shakespeare considers the problem of representing large bodies of text on the computer screen, and its elegant interface also makes intuitive sense of navigation. I end the section with a discussion of John Maeda's Reactive Books. These interactive works are quiet, contemplative, and the expression of a new computational language in which expression is encoded into a computational process.

Text and movies. The Elastic Charles project asked the question "What if *Time Magazine* could had live video inside?" After the release of Apple's HyperCard, the Interactive Cinema group had a new tool to create hypermedia. A users's story was created by playful exploration of the spaces depicted on a map of the Charles River. The river served as the central image around which all the stories revolved.

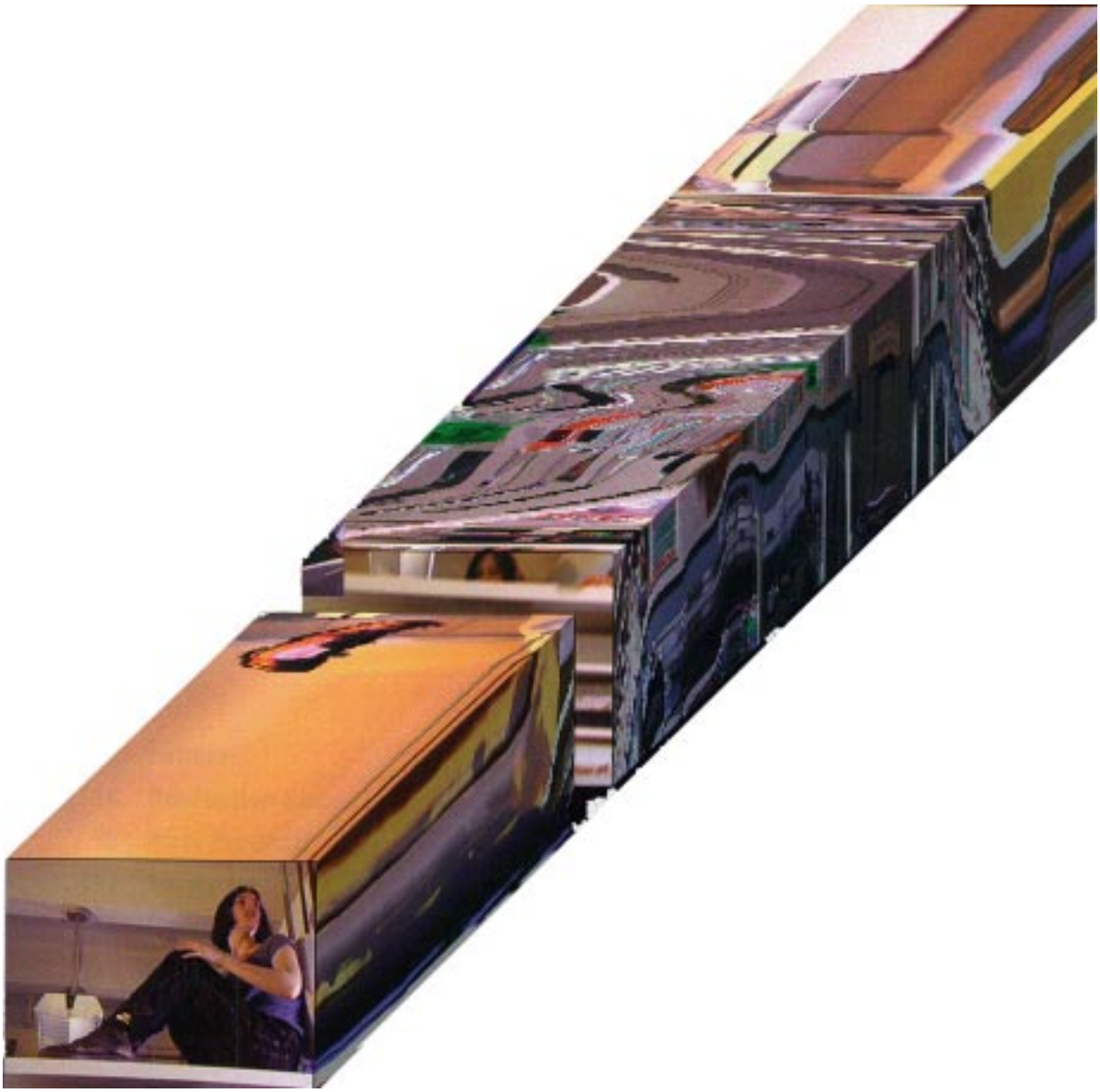
Figure 49. The Elastic Charles Interface.The River journey was told completely in time lapse photography. On an NTSC monitor the main movies would play. The second monitor could overlay multiple movies as part of the story interface.



The system was the first to design with movie icons, or "micons." In the days before Quicktime made desktop video accessible to everyone, video required dedicated hardware and normally played on a separate screen from the computer interface. The micon was a sixty second movie that could be placed like a picture into the midst of the computer interface. Elastic Charles was the first step to integrate text and movie into a single interactive form. In ActivePoem [see "ActivePoem" on page 96], I also bring text and movies together into a single interface. [Davenport 1989]

VideoStreamer: video in x, y, and t. Eddie Elliot's MS93 VideoStreamer is a spatial representation of video in 2 and 1/2 D space. Each frame of a live video stream is overlaid onto the next with a slight offset. The resulting effect is to see the video "stream" onto the canvas of the screen. The edges of the video visually reveal information about the history and content of the video clip, presented in space. Cuts in the stream are immediately apparent, as is camera movement and subject movement. Characteristic visual patterns can be matched to tilts, pans, and zooms. Elliott created a selection mode, in which users clicked and scanned quickly thorough the video to a frame, playing samples of the audio simultaneously. By dragging a selection of frames out, the system created a new clip that could be saved. The program was built as a tool for videographers to quickly scan through video and easily find salient points. The VideoStreamer was used with a collage space where people could drag and drop VideoStreamer clips and annotate them with drawings, pictures and text.

Figure 50. The Video Streamer



This program had a lasting effect to me, because it is such an elegant solution to the problem of visually representing the content of a video clip. Using Apple's MoviePlayer, every Quicktime Movie clip looks identical. A three-second clip with no edits and two-hour copy of *Brazil* have the same visual representation, a single poster frame with a scrollbar beneath the frame, and a play button. The content of the clip is not revealed at all. However, Eddie's solution represents the content of the movie itself using the pixels

from the edge of each frame. The user at a glance knows where the edit points are, how long the clip is relative to others, and can choose to instantly access any point in the movie. Because the visual representation is built in front of your eyes while live video streams onto the screen, you can see where the data comes from and its significance. The interface is built solely from the movie's data, with no pixels wasted on buttons or scrollbars. VideoStreamer was a beautiful; video volume, which could also be turned into a physical object by printing the VideoStreamer object onto a cardboard box. Macondo [E15-434] used to be filled with these boxes of video—each illustrating different camera moves. [Elliot 1993]

No one applied the VideoStreamer to a narrative application, and after Eliott left the Interactive Cinema Group the streamer boxes slowly fell apart and disappeared. The representation gave the user a lot of control over scanning through the content of a linear movie, but no other ways to interact with the content. I use a VideoStreamer-like effect in ActivePoem to give the user a sense of the history of a video clip, while telling a story to the user. [see "The visual interface" on page 98]

Figure 51. Four different views of the same text

3.5 Dynamic typography: Small & Wong

While walking around the lab, I was bound to walk into the Visible Language Workshop. David Small PhD98 had hung outside the door a single panel with frames from every scene of the Wizard of Oz. His goal was to visualize the entire movie in a single representation. His *Navigating Shakespeare* project took the unabridged works of William Shakespeare placed it into a single 3D space. He created a dynamic visual space in which the user had absolute control over the position of the camera. The user could see the entire works of Shakespeare all at once or zoom into a single word.

Figure 52. The system had a novel physical interface, with a LEGO stage and figures marked "Oberon" and "Titania." When you placed "Titania" onto the stage, the system highlighted all of her lines in *A Midsummer Night's Dream* (left). All information is positioned in 3D space, footnotes and commentary can be stored out of view, unless you zoom and rotate the camera to see it (right).

His work was fun, and the dynamic visual organization of the content was easy to understand and navigate. I would value and try to recreate those attributes in future interactive systems. [Small 1996]

Temporal Typography. Yin-Yin Wong MS95 also worked with type, but taking the opposite approach to Small's. Rather than presenting an entire body of text on the computer screen at once, she chose to only show a single word at a time. The Rapid Serial Visual Presentation [RSVP] of words has been shown to be an effective way to communicate text. Words are flashed quickly in the same space, eliminating the need for large displays. Because the user is focused on one word at a time, the designer can encode that word with expression by changing its shape over time. [Wong 1995] Yin-Yin's temporal typography was the model that I drew from for ActivePoem's text display. [see "Reading the words" on page 101]

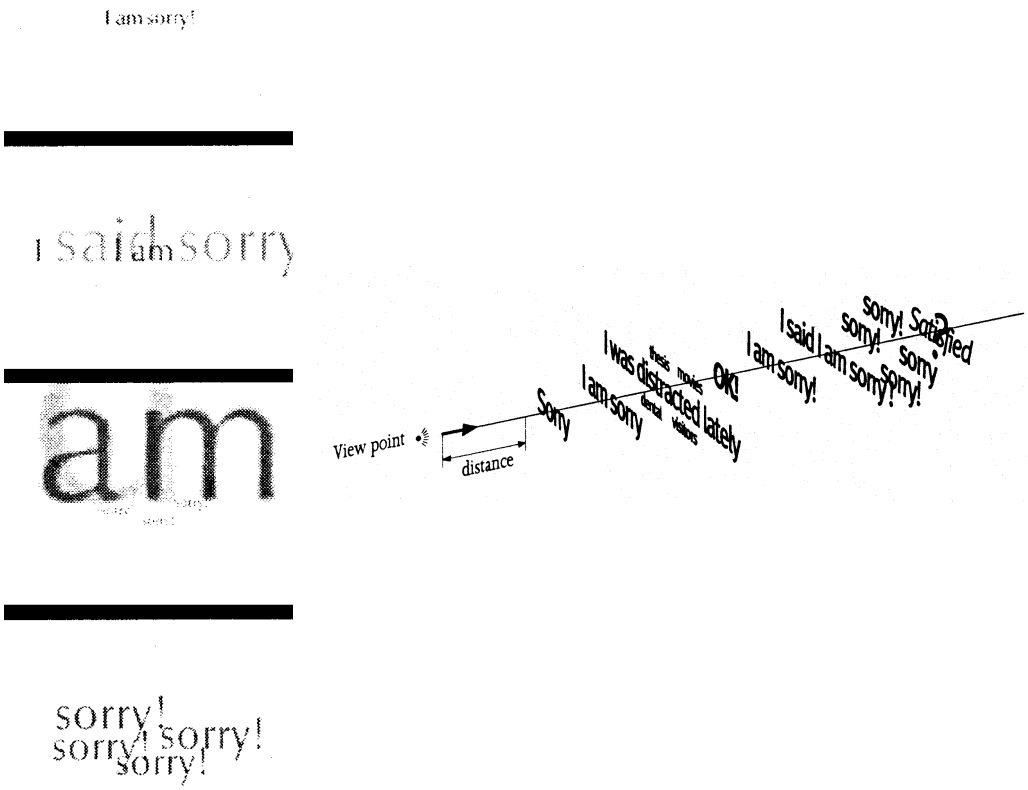


Figure 53. Frames from the animation (left) and the spatial layout of an expressive email message

3.6 Maeda: Reactive Books

John Maeda's Reactive Books represent examples of pure digital expression. *The Reactive Square*, *12 o'clocks*, *Flying Letters*, and *Tap, Type, Write* each look at a single thought, and explore that thought using the computational medium. *The Reactive Square* begins these explorations with the expression of a single pixel in ten unique ways.

When I first saw these projects, I realized that there was something fundamentally different from what I was trying to do and what these did. I strove to change my own process and to learn how a lonely square was able to touch me and react to my presence.

4.0 Discovering the principles

The ActiveStories are built around four basic principles:

1. The interface is out of the story content, without traditional graphical user interface elements, like buttons or scrollbars.
2. The story content is fixed and compelling.
3. The user is given a high level of control.
4. The presentation is visual and dynamic.

These four principles grew out of my experience with building interactive narrative systems. In the previous section, I describe a family of projects from the Interactive Cinema Group, most of which I experienced personally. In this section, I illustrate the lessons I learned when I stopped trying other peoples' systems and started discovering my own language of techniques.

For quick reference, let me chart out the projects, the tools[1] I used to build them, and what I learned. The Rules column specifies which of the four ActiveStory principles that a specific project influenced. If the Rule number is listed in parentheses, it means that the project broke that rule at the time, but I did not know it. For instance, in Hindsight, I had a funny scripted story that followed Rule 2, but I had a confusing interface built out of buttons and timelines. The project broke Rule 1, but made me realize that interfaces make more sense to the user when built out of the story content. [see Table 1 on page 63]

[1] In the chart, I make a distinction between using just the java language and the java Abstract Windowing Toolkit [AWT]. Using the AWT means that the project was primarily visual.

Table 1: Projects and Lessons Learned

Name	Date	Tools	What it was.	What I learned.	Rule
Panorama	1996	java AWT	Geographical location editor	Tools are hard to build for other people.	(1)
Shipwreck	1995	html	First non-linear text story	Text is hard to read on-line.	(2)
Knight in NY	1995	16mm	Twelve minute film	Film is a powerful way to deliver a story.	2
Search & Norman Conquests	Feb 1997	perl html	Web search engine that asks users for preferences and computes a personalized sequence of video clips	Segmenting movies and recombining them create incoherent sequences. Asking users for preferences is not satisfying interaction for them.	(2) (3)
Hindsight	May 1997	java audio	Scripted multi-linear story in audio	Authors in control of a script can tell a good story. People like to be entertained.	(1) 2
Patient	Mar 1997	perl	Eliza-like conversational text program	My first "generative" character. It talked back to you. It was fun and simple to interact with.	3
WebEndless	June 1997	java audio	Audio player which generates a conversation from pre-recorded clips	If you limit the domain, you can create a program which generates coherent conversation, but users can't interact very much.	(3) (4)
Kite	Mar 1997	java AWT	Applet which flies a kite from the kite's point of view	Dynamic interactive visual displays are fun and can contain some narrative.	1 3 4
Two 3D Streamers	Jan 1998	c++ openGL Inventor	Interpretations of Eddie Elliot's VideoStreamer in true 3D space	3D space is difficult to manipulate. VideoStreamer is cool, but it is difficult to apply it as an interface to narrative.	1 (3) 4
Fluid Transition	June 1997	c++ openGL	Real-time multi-linear music video	I was thinking of interactive cinema as something compiled into a single visual stream.	1 (2) 3 4
ActiveArticle	June 1997	java AWT	Dynamic display of a news article	Authors want to control the content, users want to control the presentation.	1 2 3 4

The table is broken into five sections. Panorama was the first and only tool that I built for other people, and I began to think about what people want from a tool (Rule 1). Shipwreck, Knight in New York, Search & Norman Conquests, and Hindsight all taught me about how different ways of handling content and what makes a good story (Rule 2). Patient and WebEndless were two different approaches to a "generative" conversation algorithm and comparing them made me rethink the role of the audience in an interactive story (Rule 3). The last three projects, Kite, Fluid Transition, and 3D Streamers, center around the visual display of information, and they trace the development of my visual language (Rule 4). And last, there is ActiveArticle.

As you can read from the table, I touched on all the rules in various experiments but did not put them all together until ActiveArticle, in June 1997. ActiveArticle changed my approach to interactive stories, leading me to the research described in this thesis, and so I will explain it in more detail at the end of this section.

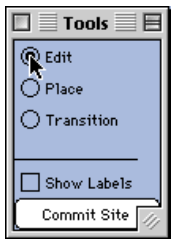
4.1 Rule 1: Content should be interface.

Figure 54. The Panorama window with a geographical organization of three nodes, the World, the United States, and Phillip

Panorama, a tool for the World Wide Movie Map. Panorama is a graphing tool, which allows a user to create conceptual maps of geographical places and annotate them with identifying information, names, dates, and keywords. It was to be used as an interface to the World Wide Movie (Story) Map [WWMM]. WWMM was an ambitious project to create a global geographical story network, to which anyone could easily attach their own town or street. My tool was to enable users to make story "sites" to which they could attach movies, pictures, and media, then upload that package to a central server.

This is the first tool that I programmed to other people's specifications. This is the usual process for a computer programmer. You receive specifications for a widget that does X, and you are expected to deliver that widget exactly "to spec." I designed a simple layout program which stored, edited, and retrieved site information. I used palettes and radio buttons, basing my tool design on Adobe Illustrator.

Figure 55. The Tools palette



I wanted to build an intuitive tool for Joe User and learned how difficult it is to build an all-purpose tool. No matter what features you provide, somebody wants something slightly different. Someone was used to Adobe Illustrator, someone else was used to Macromedia Freehand, and each person had different expectations of my tool. I realized that there is not one generic interface that *anyone* can intuitively use, because the intuition is built up from a user's experi-

ence with other tools. If your tool looks like Illustrator, a user expects that it must behave the same way. I began to think about other ways to make interfaces not built from buttons (Remember Rule 1?). That search would be extended through my explorations in John Maeda's class MAS934: Principles of Interface Design. I will talk about that class more when I discuss the Kite Applet. [see "Kite" on page 78]

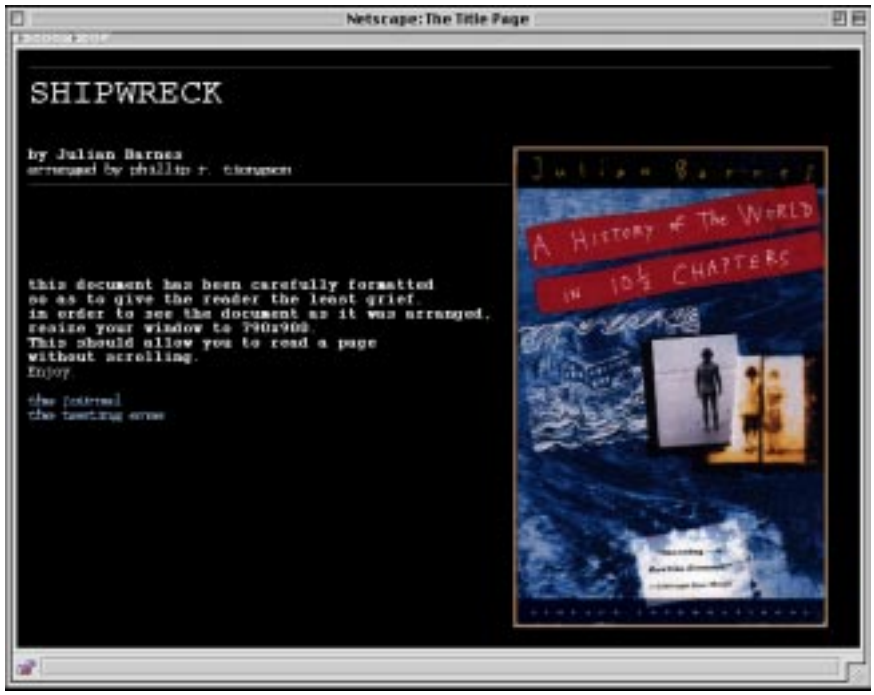
Panorama was a "tool" for other story "builders." I wanted to tell my own stories, and I began by writing a Web site.

4.2 Rule 2: The story content is fixed.

Shipwreck. In my first taste of non-linear, interactive narrative, for Glorianna Davenport's and Ken Haase's class MAS134: Story, I designed a web-site based on a story by Julian Barnes, in his book *A History of the World in Ten-and-a-Half Chapters.*

The chapter I chose chronicled the sinking of a ship called the *Medusa*. The story was in two parts, a first-hand journal of the events that befell the crew of the Medusa, followed by an analysis of all the events that led up to the painting *The Sinking of the Medusa*. Because the first half is a journal, the entries can be read in a non-linear order, in theory at least.

Figure 56. The title page from *Shipwreck*. http://ic.media.mit.edu/~phillip/shipwreck



I designed the interface to the story around the painting, because it was a non-linear telling of the story and it provided a unifying visual element. Different parts of the painting hyperlink to entries of the journal. When users read the last journal entry, they jump into the second half of the story, a discussion of events surrounding the actual painting.

Figure 57. The fragmented painting served as the interface to the text of *Shipwreck*.

In trying to design my first "interactive" story, I hit upon issues that I would consider again and again. Although I could create a logical segmentation of the content, simply providing non-linear access to the content of the story did not make the story experience more pleasurable. In fact, the story made more sense when read in order. I discovered the difficulty of using content designed for a different medium and "repurposing" it in the name of adding interactivity.[1]

I believed that you could use the Web to reveal more context than you could in a book. The intent of my interface was to link the tiles of the painting directly to the author's words, but the static interface and random reordering of the story worked *against* the narrative.

[1] A couple funny things happened to me as a result of this experiment. Six years later, one of the discoverers of the remains of the actual ship *Medusa* was visiting MIT and called to talk about the actual archaeological find. He was eager to speak to me having seen my website. Then one day Davenport told me about a demo she had given in the morning. Random House was a sponsor of the lab at the time, and Glorianna was giving a demo to a group of designers. She showed my website, which begins with an image of the book's cover. One of the designer's said, "Hey that's my book!" He had designed the cover, and of course I had just scanned everything in, the text of the story, and the pictures, without regard to copyright. At the end of the demo, the designer apparently liked what I had done and apparently overlooked the copyright infringement.

A Knight in New York. I tried writing my own stories and shooting my own movies at New York University in the Spring of 1995. Using the most non-computational tools available, I experimented with the art of linear film making. I had complete control of the film from script, to editing, to projection. At the end of the experience, I enjoyed having complete control of the story. However after returning to MIT, I again willingly gave away this control in exchange for interactivity.

The Search and The Norman Conquests. Janet Murray teaches a writing workshop in non-linear narrative, 21W765J. In that class, she encouraged us to develop a way to tell stories procedurally. As exercises, she gave us content from movies and stories that were non-linear in nature and asked us to compose a process to retell the stories.

She gave us content from two plays that had both been filmed and made into movies: *The Search for Intelligent Life* and *The Norman Conquests*. I built systems which programmatically sequenced video clips from the movies into a personalized narrative. My formula was as follows:

1. Segment the movie into short video clips.
2. Annotate the clips with some narrative information.
3. Write an interface to get input from users about the story they want to see.
4. Write a program which takes the users' input and uses the annotations to create a sequence our of the video clips.
5. Done! Reconfigurable movies, sit back and enjoy.

This approach was inspired by the segmentation of content in the Contour and Dexter projects. [see "Contour and Dexter" on page 49] Search used content from *The Search for Intelligent Life*, a one-woman play by Jane Wagner and Lily Tomlin. Arjan Schütte MS98, a fellow classmate and Interactive Cinema Group member, and I worked on an interactive interface to the play.

Figure 58. The Search interface was a series of Web pages which asked the user to click on topics of interest.

The play's structure is built around short vignettes from around twenty different characters. The film rapidly intercuts between these characters seemingly at random times and in a random order. The natural segmentation was to cut each vignette into a video clip. To each clip we attached keywords like "hippie," "cancer," and "sigh," which corresponded to the themes in the clips. Then I programmed a CGI web interface to the database of clips. The interface consisted of a series of web pages which made the user choose 5 keywords. My CGI script would generate a web page with video clips matching each of the keywords you clicked on, yielding a linear sequence.

The film used disjointed visual transitions to mark the beginning and end of each vignette. I digitized those and placed them into the sequence. A perl script generates a Web page (shown at left) with all the clips sequenced down the page.

Even though the content was inherently non-linear, and the user did get to "choose" their content, the interface and sequences seemed only to make sense to us, the people who annotated the content and had watched the movie.

Classmates did not feel that the selection of keywords was meaningful way to interact with the story, because my system did not guarantee any narrative structure on the sequence page. The experience was like choosing words out of a hat, and expecting them to write a sentence.

We tried this approach again with *The Norman Conquests*, a series of three plays set in three different rooms of the same house on the same day with the same characters. Every exit from one play is an entrance into one of the others. This play is a multithreaded story, instead of a non-linear one, and begged for us to make an interactive interface. Again, we segmented the clips, but used a slightly more complicated annotation system responding to class criticism that the Search sequences were too confusing.

We created an annotation that would theoretically allow the user to specify how confusing the compiled sequence would be. We organized clips according to which play it belonged to, its physical length, and its position in the play, and a sense of its importance in the overall narrative. We created subtitles for each clip which added our own witty commentary on the unfolding action. The text subtitles helped to provide narrative context for the clip.

This time the interface was a set of radio buttons, allowing the user to specify the length of their program, from one minute to longer than ten minutes, and the "chaos level." If the chaos was low, the system would maintain linear continuity in the sequence. If the chaos was high, then it would make a non-linear sequence, after all the original plays did not appear in order. The system would generate either a trailer of the three plays depending on your preferences.

Our algorithms were based on very simple assumptions about narrative structure and a random number generator for variety (which hurts me now). After watching the sequences generated by these programs, a viewer could think very hard and *try* to make sense of the collection, but it was far easier to assume that the clips were still purely random, even given the "chaos" knob we provided.

Figure 59. The Norman Conquest interface and a one minute sequence.

Although it is true that the clips are chosen according to the viewer's preferences, and the content is assembled at run-time, the result is not a coherent, compelling story. Plays are not meant to be cut into tiny pieces and rearranged at random. The content was not designed to be read in this manner, and we tried to shoehorn interactivity into a very tiny slipper. Although the content seemed easy to adapt for an interactive story, I realized I could not create an algorithm that could tell the story better than the original. After all, that should be the metric of success of the system. If my interactive system could deliver a "more pleasurable" sequence than a strictly linear presentation, the system was successful. At this point, I considered how to design content to be inherently interactive.

Hindsight. The last class project and, I think, the most narratively successful is Hindsight, a multi-linear story I wrote for interactive presentation.

Figure 60. The destroyed Rashomon gate where all the stories are retold in the movie

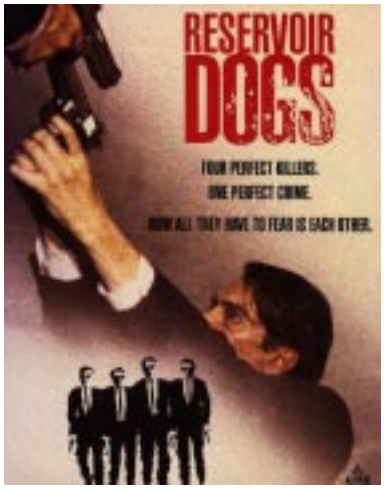
The plot is based on the idea that our own actions look different to us when we look back on them, in "20/20 hindsight." When we relate events in our lives as stories, depending on whom we are telling and how we have changed, we tell different stories. We emphasize different details; we exaggerate events; we retell what we think is important to retell.

Two movies influenced the narrative structure of Hindsight the most, Akira Kurosawa's *Rashomon*, and Quentin Tarantino's *Reservoir Dogs*. *Rashomon* is a brilliant multi-linear narrative where four people—a bandit, a woman, her dead husband (via a medium), and a woodcutter—tell versions of the same murder. Each person's version of the story is colored by their own self-interest.[1] Kurosawa's plot is based on the stories of Ryunosuke Akutagawa: "Akutagawa's point was the simple one that all truth is relative, with the corollary that there is thus no truth at all." [Richie 1984]

[1] There is a terrific discussion of the film in the book *The Films of Akira Kurosawa* in which Donald Ritchie explicates all the subtle differences of the four versions, and the production of the film.

The narrative of Hindsight is based on *Rashomon*, but the structure of the tellings is based on the non-linear structure of *Reservoir Dogs*. Tarantino's film proceeds in jumps and starts, cuts backward and forward in time, from character to character, continually switching the narrative context. The action revolves around a bank robbery, and what went wrong. The characters retell the events trying to determine the "truth" of what happened. Ending with a bloodbath of bullets, Tarantino creates a riveting story where the audience is dragged through different narrative threads until the conclusion, when all the threads come together.

Figure 61. Reservoir Dogs poster



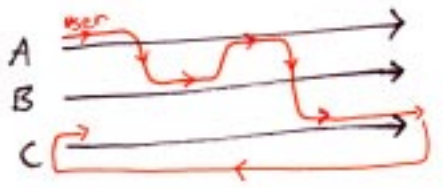
The interface. Hindsight's visual interface consists of a display screen on top, where still pictures from the story appear. At the bottom of the screen, a set of horizontal bars represents the different narrative threads. Clicking on one of the threads starts the story in motion. The first audio clip plays, and turns red, as it is played. On the display screen, a photograph relating to the current audio clip appears in one of three slots. If the user does not interact, the system plays the next clip in the current thread, which triggers a new photograph to appear. If the user clicks on a new thread, then the system begins playing a clip from the new thread at approximately the same elapsed time. After a clip is played, it turns grey creating a visual record of your progress through the story.

Figure 62. The progression of a user through the story of Hindsight: the initial display (top), after the user starts listening to the story (middle), after the user changed narrative threads (bottom).

The narrative always progresses forward, skipping from thread to thread as the user desires, until a path all the way to the right hand of the screen is made. Then the system restarts from the left hand side of the screen playing clips that have not been heard. The first time you switch to a thread, a title card appears like "ten years later, with her husband," to let you know what the new thread is about, and how it fits narratively with the other threads.

I wanted to create an interactive structure that would allow the viewer to be in control of switching the narrative context. In my system, the user can switch to any of the first three versions of the story at any time. Compared to the limited radio button interface in Norman Conquest, Hindsight gives the user much more control while the story is playing out.

Figure 63. The eavesdropping metaphor means the user must always travel forward in time. However when reaching the end of the thread, the presentation starts over.



As if eavesdropping on four different conversations, the user is constrained by the passing of time. By choosing to listen to stream B, you have to miss a little of stream A, and so on. Glorianna Davenport told me that if you are participating in an interactive movie about running, and you are not tired by the end, something is missing. I built the story around the eavesdropping metaphor so that viewers would understand the scope of their interaction, and have the responsibility to make a decision about who to listen to, at the consequence of missing something else. However, the user actually risks nothing by interacting and can hear all the versions the narrative, because the system goes back to let you hear what you missed.

My narrative goal was to let the audience make assumptions about the real "truth" of what happened on the street. All audience members have the opportunity to construct a unique truth, because their initial paths through the narrative threads are determined by their actions. However, when the narrative starts over again, the listener will "discover" a second version of the "truth," from a different perspective, and they can resolve for themselves what is the real "truth."

To me, the most satisfying part of watching *Rashomon* and *Reservoir Dogs* was the sense of discovering the "real" truth. You watch the film and judge who was lying and who was honest. Both movies begin with a mystery, "what hap-

pened in the bank?" or "what happened in the forest?" I beg the user to ask, "Who does Amanda eventually marry?"

At the time, I referred to this as a "probabilistic approach"[1] to non-linear story telling: scatter enough common story references in each thread and the user is guaranteed to hear some of them and begin to make assumptions about the story. Users construct stories about each character in their heads and the order the story is discovered does not matter. As a result the writer does not have to trace exhaustively though each possible path to check if each path is a reasonable story.

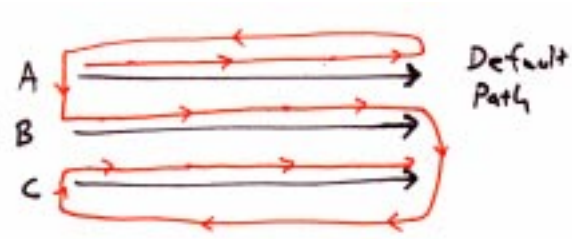
[1] *The Dictionary of the Khazars* is a "lexicon novel" which also uses a similar "shotgun" approach to non-linear narrative. It served as another source of inspiration for Hindsight.

I produced Hindsight[2] and encouraged many people to try the system out. To my surprise, I discovered that almost no one *wanted* to interact with the system. In fact, unless prodded by me, most people were interested enough in the story to listen to all four threads in sequence, without interacting at all. After listening to the whole story, most users who had the time would try it again and experiment with interaction the second time. This was even true for "expert" users who were familiar with interactive narrative systems, such as my teacher Janet Murray and Intel visiting affiliate, John D. Miller. Although I was happy that people enjoyed the story, this was not what I expected.

[2] I had a lot of fun writing and shooting this story. I rapidly prototyped the system. I set up a recording studio in my office and recorded each narrative track in the studio, then shot stills of the couple on the street miming the actions described in the dialogue. As a result we were able to wrap production in twenty-four hours, from start to finish. I spent the next day segmenting the audio and choosing the best takes, digitizing the photographs, and designing the system. I had the initial version of the story ready by the next day, which after feedback from class and Janet Murray, over the next month was refined to become Hindsight 2.0.

Even though I had designed the story and the interface to be interacted with, most users were *afraid* to interact for fear of missing something. Moreover, my system did not *require* the user to interact and would present each thread in order, if never interacted with. As a result of this experiment, I began to ask different questions of myself as a designer. Rather than asking how can I design an interactive narrative, I began asking why should a reader want to interact at all? This question still haunts me as I continue design my interactive stories.

Figure 64. The default path through the content required no interaction and played all the streams in linear order.



There are other reasons which could explain participants' unwillingness to interact with the story. Perhaps there are stories that are more natural for viewers to interrupt or guide. The interface was static and clunky and did not necessarily make users feel like that were in control.

Figure 65. Hindsight's exciting progress bar interface.

The interface for Hindsight was not part of the cinematic experience, it was built out of a time line and progress bars. The only dynamic element on the screen was the red progress bar at the bottom of the screen which indicated where you were. I even found myself concentrating on that bar, rather than the story.

From that experiment and my experience with Contour [see "Contour and Dexter" on page 49], I realized that there is a conflict between immersive cinematic experiences and user interaction. The more a user is immersed in the cinematic experience of receiving a story, the less inclined they are to do anything to interrupt it. However, if you force the user to interact at certain points in the narrative, then you are likely to pull the user out of the immersive experience and break the narrative illusion. I believed that this was the biggest conflict between interaction and cinematic experience. I ventured that if the cost of user interaction was made minimal by using narrative and cinematic transitions to maintain the immersive quality of the story, there would be a way to bridge the conflict between immersion and interaction.

This experiment illustrated that when I carefully constrained how the story content is sequenced, I could tell a good interactive story (Back to Rule 2). It clarified to me relationship between the graphical interface and how users feel they can interact with the content of the story (Back to Rule 1).

4.3 Rule 3: Give the user control.

Patient. My first completely interactive program was called Patient. It is a program that you can type any phrase into, and it talks back. The first "chatterbot" was Eliza, programmed by Joseph Weizenbaum as an experiment in Artificial Intelligence. He was astounded how people actually thought that Eliza was alive and a real psychiatrist, even though she was based on a simple conversational algorithm. My program is an homage to Eliza, but rather than being a doctor, it is a patient. In fact, my program acts like a person who is treating your input as if *you* were Eliza.

Although the system was text based, it was pretty compelling to sit and chat with the Patient and people had fun *even if they knew how it worked*. As a class contest, the goal of the interactor was to have as many possible coherent exchanges with the chatterbot as possible, trying not to break the system.

This system was different from any system I had made before. All the systems so far had used video or audio stories, which I cut into pieces and tried to make interactive. This system was actually "generative." It could process input from the user and always generate a conversational response. Users interact using natural language, without buttons or any constraints. In this system, the user has far more control than any system I had built so far. Furthermore, the user enjoyed having full control. Compare the user's experience to the experience in WebEndless conversation on the next page.

Figure 66. An excerpt from a Patient session, remember that the patient is the program, and the doctor is the Arjan Schütte MS98.

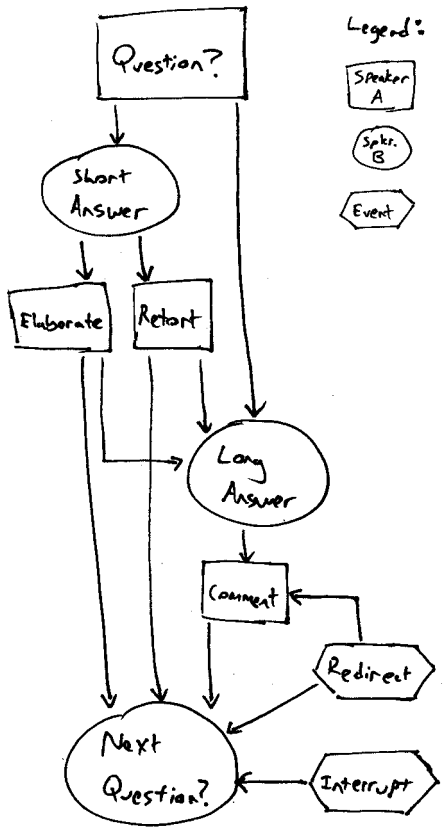
```
patient >> doctor, i am very depressed. i don't
          know what's wrong with me. i just feel
          listless and tired all the time.
doctor >> why do you feel listless all the time?
patient >> i feel like i'm in an endless cycle.
doctor >> tell me more about that?
patient >> i can't keep a girlfriend, my job is
          a mess.
doctor >> tell me about your girlfriend
patient >> leave her out of this. she is fine.
doctor >> earlier you mentioned your job
patient >> you really don't want to know about
          my job.
doctor >> i would like to know about your job
patient >> fair, yes, i admit it, i am an adjunct
          professor at mit. the bureau. it makes
          me depressed.
doctor >> so you're a professor
patient >> i am not really a professor.
doctor >> are you a student
patient >> act me again.
doctor >> are you a student?
patient >> |i think it needs more keywords.| are
          you a programmer?
doctor >> no i'm not
patient >> i don't know why i pay you. at least
          i have a real job.
doctor >> what's your job?
patient >> uh, i am not really a professor. i
          wouldn't want to be one anyway.
doctor >> what do you do?
patient >> professors don't make any real money.
doctor >> who makes real money?
patient >> my job is to use computers.
doctor >> do you like computers?
patient >> i am an ai programmer!
doctor >> what kind of programs do you make?
patient >> |this program is pretty good. look at
          this.|
doctor >> what?
patient >> you are an alice program written to
          pass the turing test.
doctor >> no. you are
patient >> i am so tired of you acting my
          questions over and over.
doctor >> you are part of the turing test. not
          me.
patient >> it acted about the turing test!
doctor >> what do you mean, "it"?
patient >> if you can fool me into thinking you
          are human, you pass. you are doing
          pretty good so far!
doctor >> i am a human. you are the computer.
patient >> a program computer to give life-like
          human responses. have you ever heard of
          alice?
doctor >> sure
patient >> stop trying to put words into my
          mouth.
doctor >> i'm not. tell me about alice i'm not.
          tell me about alice
patient >> i am alive, you are a program. a
          chatterbot.
doctor >> no. you are the chatterbot
patient >> you respond to the inputs i give you.
          |this is funny.| and algorithmically
          produce responses.
doctor >> you are wrong
patient >> don't you want to know about my sex
          life too?
doctor >> tell me about your sex life. if you
          have to talk about it?
patient >> my life sucks.
doctor >> why does your life suck?
patient >> i am alive. you are just a computer
          program.
doctor >> would you rather be a computer program?
patient >> i can't believe this program thinks
          it is alive.
doctor >> i am alive.
patient >> you are not alive. i wonder how you
          do pattern matching?
doctor >> what pattern matching?
patient >> |i wish this program were smarter.| i
          am depressed.
doctor >> come on. don't be depressed. i'm real
patient >> |it is sort of getting repetitious
          now.| i still hate my job.
```

WebEndless Conversation. Based on Mark Halliday's An Endless Conversation [see "DMO, LogBoy, and FilterGirl" on page 45], the idea for WebEndless was to create a conversation system which programmatically generates a spontaneous, witty conversation from prerecorded clips.[1]

[1] Arjan Schütte MS98 and I wrote a new set of questions, and he recorded interviews with Kevin Brooks PhD98, Freedom Baird MS97, and Richard Lachman MS97. The content was patterned directly after the funny stories in Halliday's original project.

I devised a new model for the underlying conversation which allowed for a more complex and natural conversation than the original system. The conversation model is based on a spreading activation network, like Murtaugh's in Contour. However, my spreading activation network had a the property of forward motion, in addition to Contour's property of narrative continuity. Murtaugh's model does not guarantee any structure to the sequence it plays. On the other hand, my system had to sound like a coherent conversation generated from audio fragments.

Figure 67. The stages of a conversation: the conversation proceeds between states, and speakers must alternate. External events trigger the system to react. The chart ends with a new question, starting the process over.



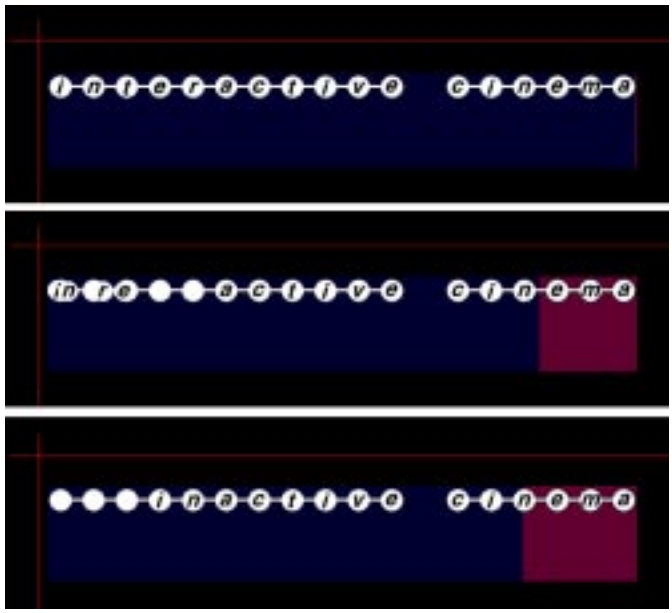
I had to build into the model a sense of stages of the conversation, such as Questions, Answers, and Retorts, and the system had to move forward between the stages in a conversational way. The system could recover gracefully when it ran out of clips about one topic by switching to a new topic. The new system was successful in creating coherent conversations from the large database of audio clips we had gathered. I redesigned Halliday's original conversation model for an arbitrary number of speakers and for longer possible discourse about a particular topic than the original. It was implemented in java and so was able to play over the Web (using audio only).

The vision. Although this was never implemented on the IC Web site, what was most interesting to me was the possibility for the applet's conversation to be affected by a user's activity on a Web site. Because the system used a spreading activation network, new topics could be selected and talked about, in real time. So a hyperlink on a Web page not only requests a new page, but also informs WebEndless

about a new topic of interest. WebEndless could run in the background of a web page in its own frame, its conversation being continually updated by the effects of the user's Web browsing. The result would be a personalized radio program being run "live" by the members of Interactive Cinema. The program would always be tuned in to the user's topics of interest. To me this was an interesting mode of interaction, basically the user does what they are accustomed to doing anyway, browsing web pages, and the application listens for those events and shifts its content accordingly and transparently.

However, compared to the Patient, the user only indirectly interacts with the conversation. The user has very little control over the direction of the conversation, except to interrupt it, by clicking on it. When a user did interrupt the conversation, someone would exclaim "Hey I'm talking' here!" and the banner would change from "interactive" cinema to "inactive" cinema. The conversation worked so well that I had to explain to many people that the conversation was not scripted, but was actually being assembled on the fly.

Figure 68. The WebEndless interface: clicking pauses the conversation and the applet's background is a progress bar that shows how much content is left.



Users had no direct way to push the conversation in a direction, no useful interface to the available topics, and, as a result, very little feeling of control. WebEndless was interesting to listen to, but did not feel interactive, because of the lack of user feedback.

So by contrasting Patient and Hindsight, I realized that people have fun when they have more control and understand their role as participants in the story, thus Rule 3.

4.4 Rule 4: Visual & dynamic presentation

[1] From http://acg.media.mit.edu/mas964/ps1

Kite. In the Spring of 1997, John Maeda taught his first class:[1]

MAS 964

PRINCIPLES OF VISUAL INTERFACE DESIGN

press the ok button. select the text. drag the slider to scroll. we know these activities very well from our daily interactions with the modern graphical user interface. our successful interaction with computers relies heavily upon over two decades of research devoted to making computers easier to use. however there has been little emphasis on core issues in visual communication design in the interface due to the common assumption that a 3D-ized interface equates to an ample sensitivity to issues in design. This course focuses upon spotlighting expressive properties that are unique to the computing medium as a means to introduce a set of liberating extensions to the interface designer's vocabulary.

Over ten weeks, each student programmed ten java applets. The class approached design on the computer from first principles: drawing circles, squares, and lines. I felt I lacked a sufficient design background, so I strove to take narrative approaches to solve the design problems. Although the narratives were implemented abstractly, the class gave me a new approach to think about the visual representation of a narrative. Let me illustrate with one example. The task was to build a computational kite.

For PS4 the assignment was:

1. Figure 3-7 has a kite. Make a kite that you can fly with your mouse. Limit your solution to monochrome (no grays). Be sure that the interface for flying the kite is self-evident.

While thinking about the design of a kite, I imagined the person flying the kite. Wouldn't it be interesting to see the reactions of the person flying the kite, or the kite's point-of-view? I designed my kite to be a fixed part of the frame, while the user could see the landscape and the person flying the kite (theoretically, themselves). It turned out to be a reversal of point-of-view from all the other kite applets, which rendered the kite itself. In my applet, the user is in control of the flight of the kite, but when the kite gets too close to the man, he throws up his hands, both scared that the kite might fall to the ground, and to pull on the string to give it more lift. It is a very simple story, rendered in a few lines, but it is a story that is alive. It is like a sketch that breathes.

The class approached the space of the computer screen completely differently than I had ever experienced before. I had taken 6.170 Software Engineering and 6.837 Computer Graphics and written my first graphical tool, Panorama, and I had made the same assumptions in all of them: Graphical User Interfaces consist of radio buttons and menu bars. The computer is a tool, and with it you build tools for other people to use. You write code which is generic, documented, reusable, and with well-defined interfaces. That makes you a good software engineer.

This class was the first time that I looked at the computer as a medium of expression. The computer screen is medium that can display other media, but is inherently different because it can *operate* on other media. What I considered the primitives of the Graphical User Interface, the button, the scroll bar, the text box, were in fact themselves made up of rectangles, circles and lines. In this class, I began to explore and understand the use of a single line.

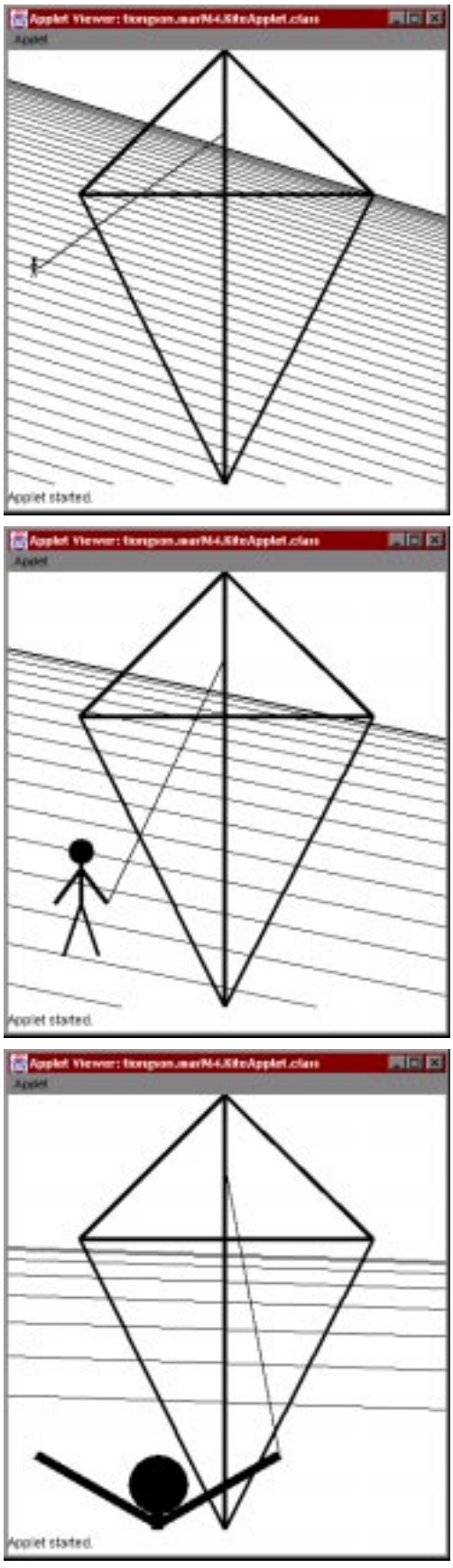
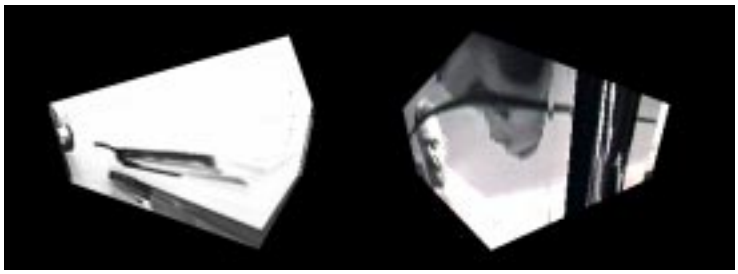
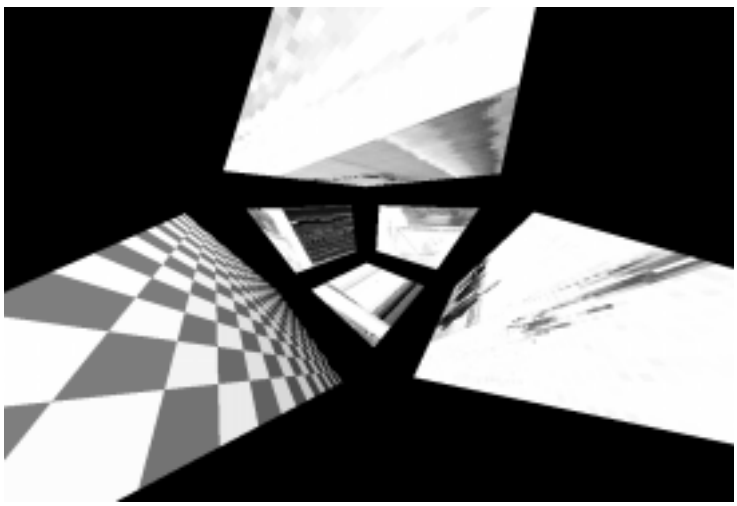


Figure 69. The kite in action.

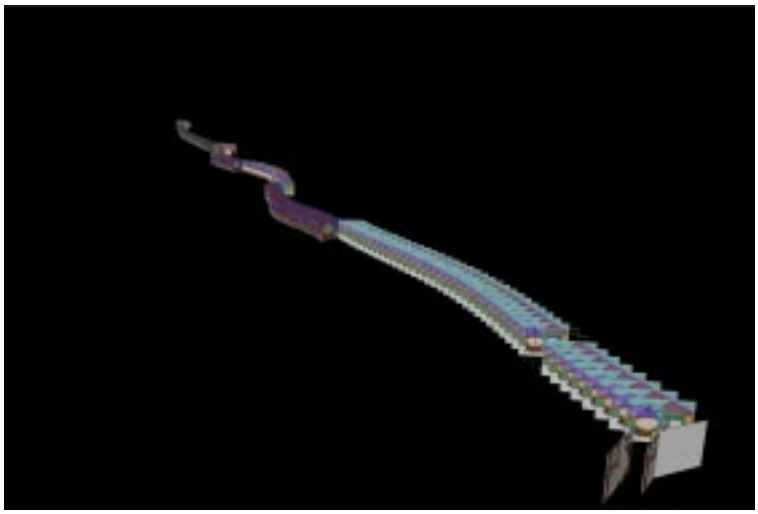
3D Streamers. I began to explore different visual representations of video, looking for a natural way to manipulate it. I felt that a 3D representation might be helpful in organizing an interactive narrative in space, though I did not have a specific application in mind. I had always been interested in Eddie Elliot's Video Streamer. He plotted the frames of a video clip on the X,Y, and T (time) axes creating a spatial form out of the video in 2 1/2D space. [see "VideoStreamer: video in x, y, and t" on page 56]





I was so compelled by the VideoStreamer that I built two different versions of it in true 3D space on an SGI. Like the original Video Streamer, the first system reads a live stream from the video-in port of the SGI. The system reads the movie data into a buffer and texture-map the pixels onto the font face of the cube.I copy and map the edge pixels to the corresponding sides of the cube, creating a Video Streamer effect. Because it is implemented in 3D, users can explode the faces of the cube and travel inside it to see the data of the movie flowing around them. Being able to navigate inside the streaming cube was visually interesting, but I had no idea how to use it as an interface to a narrative.

The second version of the 3D Streamer texture-maps each frame onto a plane in space, then places those frames into the Z-axis. As the data streams in, I calculate the delta between the pixels of each frame to determine where cuts in the video clip are.



I identify each cut in space, by placing the frames on a slight arc. I programmed odd cuts to arc to the right, and even cuts to arc to the left. The resulting form was a true 3D Video Streamer, with every frame of the movie plotted in space, creating a video volume. The volume's shape was determined by the length of individual shots. To "play" the movie, I put the camera in motion in the Z-axis, revealing successive frames by passing through them. The user had no control of the story, and I could not figure out a way to use the 3D environment to tell a story with video.

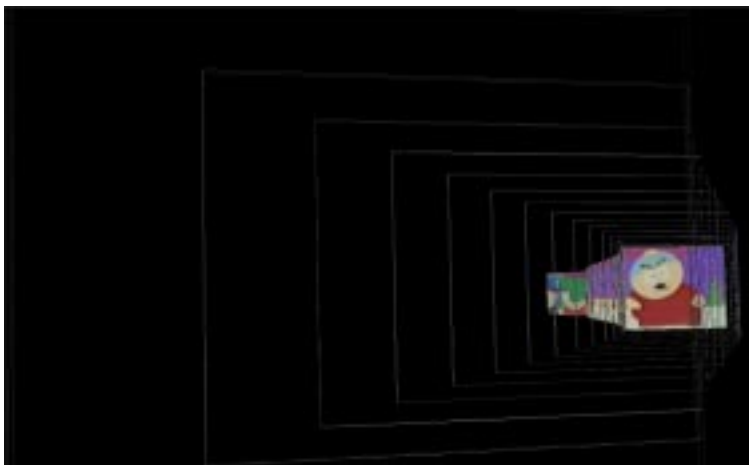
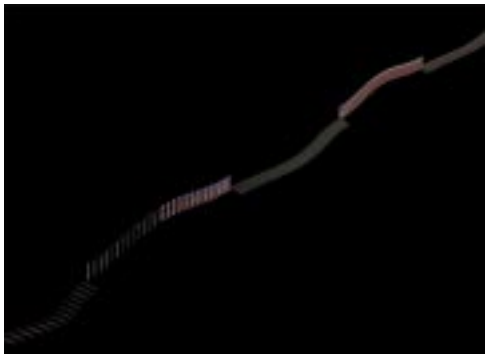
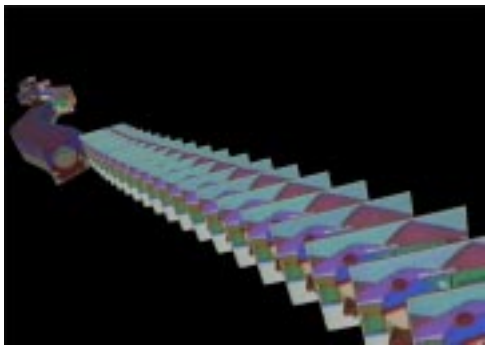


Figure 70. On the left hand page, scenes from *A Knight in New York* and *The Wrong Trousers* illustrate the first 3D streamer. Notice how the sides of the cube can be exploded to see inside the streamer.

Below, various views of the original South Park video, streamed onto 3D planes. The arcs in the video denote cuts.





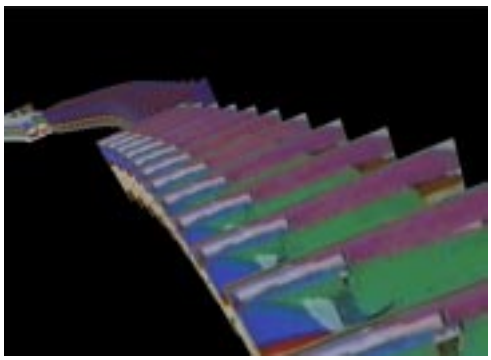


Figure 71. By dragging the thumbnail video clips around the screen, users could choose which stream of video they wanted to see and hear. The selected stream of video would dissolve into the center screen, while the other two streams faded out.

Fluid Transition. Fluid Transition was a short-lived project I worked on during the summer of 1997. The goal of this project was to create a system with real-time access to the language of cinematic transitions: dissolves, wipes, and fades. My theory was that a system that could manipulate multiple streams of video in real time would give the user more freedom to interact with a narrative without losing their immersion in the narrative process. I designed a system on the SGI which reads Quicktime files and applies arbitrary audio and visual transitions to each media stream. The system could composite all the streams into a continuous strip of playing video or could dissolve from one stream to the next in real time.

I chose content that could be manipulated visually, namely uncut footage from a music video. Music videos do not have to conform to any formal cinematic rules, and so a system which applied arbitrary cinematic transitions to the video could still make sense to the user. In fact music videos pride themselves on breaking all the traditional rules of continuity editing, borrowing heavily from techniques used in experimental films and Eisenstein's montage (the horror). I digitized my next multi-linear "narrative"—four independent views of the band and its members playing and singing the title song "My Star."

The fundamental flaw in this project was that it was tied to the idea of combining multiple movies into a single more "interactive" movie. Although I was building my own tools to craft the content, I had gotten stuck into thinking about interactive cinema as a fixed rectangular frame, governed by the same rules as the film medium. I had made a primitive real-time editing system, but there was not a compelling narrative reason to interact in real time. The system lacked any representation of the content of the different streams so the system could not operate on the data either. I realized that I needed a good story to tell and started over.

4.5 ActiveArticle

ActiveArticle was actually made before the 3D Streamers and Fluid Transition. I wanted to explore the experience of 3D, hoping to find a use for it. By approaching the technology without a story, I ended up with very pretty projects with no story. So I went back to the experience of ActiveArticle, where I felt I had all the elements of a compelling interactive story: good interface, compelling content, users felt in control, and the presentation was visual and dynamic. These elements eventually became the four principles on which I would base the ActiveStories.

The Goal. ActiveArticle grew out of a collaboration between the Media Lab and the Chicago Tribune.[1] Going into the project, our goal was to study the process of news gathering, editing, and publishing and see how we would apply the Media Lab's unique understanding of technology to that process. The Tribune assembled a team of their editors, writers, photographers, videographers, and web designers to work together with us on a new story. The "hook" of the story centered around Wrigley Field and the possible tension between the residents of Wrigleyville and the nightly crowds that attend the games.

[1] The Tribune invited Glorianna Davenport, Ron MacNeil, Chloe Chao MS98, Matt Grenby MS98, Arjan Schütte MS98, and myself to participate in an experimental story project. The Tribune Company owns not only the *Chicago Tribune* newspaper, but also local radio stations, TV stations, and an internet site. They are interested in "converging" all of their independent media sources into a more cooperative collective.

[2] For instance, they had to decide whether they were going to hold the presses until 12:05am or 12:10am for the results from the Academy Awards. If they held the presses, the difference of five minutes was the difference of 8000 papers. 8000 people would be getting their paper late in the morning, and might cancel their subscriptions. They decided to wait, and notice in the paper below that the Academy Awards are included on the front page.

The first day, we attended the daily Front Page meeting, where the Section editors (Front Page Editor, Metro Section Editor, etc.) meet with the Managing editor and identified their "best." The Managing editor discusses what should run on the next day's front page and always makes the final decision.[2] It was an intense, hierarchical process, streamlined for the daily generation of news.

Figure 72. The front page we witnessed being made.

Figure 73. Chris Walker, Tribune Photographer, shooting pictures in the MIT Au Bon Pain 2000

Collecting the content. That day we also met with the journalists working on our story, two videographers, one photographer, one newspaper writer, one internet writer, and two Website designers. We had an organizational meeting and decided to pair one MIT student with one Tribune journalist, and go out to Wrigley field that night with Press passes. However, at the meeting we did not come up with the spine of the story, a central theme around which everything would fit.

Already we had broken with the traditional process of a story. Never before had journalists from such different disciplines met in the same room to work on the same story at the *Chicago Tribune*. Journalists are also usually sent on assignment with more direction as to the story they would write. I accompanied Chris Walker, the photographer, for the first half of the night, and Louise Kiernan, the staff writer for the last half. It was an educational experience.

These people were professional storytellers. I watched Walker shoot hundreds of photographs and edit them to forty pictures. Each photo told a different story even though it was a single frame. I watched him as he talked to people and collected contextual information about where they were from, why they were at the field, and who they were. He also told me that a picture without a caption is meaningless because captions tell you *why* you should care about these people.

Then I walked with Kiernan around Wrigley as she interviewed the rent-a-cop at McDonald's, the owner of a sports bar—the Cubby Bear—and people enjoying a post-game drink. Her job was not only to collect these people's stories and publish them, but to reveal the stories' biases and subjectivity. She had knowledge about the neighborhood and the local politics which she would bring to bear when finally writing her story.

That night the group collected hundreds of photographs, several hours of video footage, notes for several text stories, and live audio from inside the park. We brought all the material together into the editing room to try to author a single coherent story.

Showing them Contour. The MIT group met with the Tribune editors and journalists to show them Dexter and Contour, Mike Murtaugh's evolving documentary system. [see "Contour and Dexter" on page 49] Though interested, they were critical of three things, its lack of context, voice, and structure. To be fair, Contour was designed as an "Automatist Storytelling System":

> Automatism is a word used to describe a branch of the surrealist movement; it represents the process of creating art based on a kind of "automatic" or unconscious free association. The intention is a "truer" experience as meaning emerges from the interactions of individual expressions rather than from a structure imposed from an "exterior consciousness." [Murtaugh 1996]

Figure 74. Contour showing *Boston: Renewed Vistas* in its initial state.

In the business of news, there is little room for "unconscious free association." The Tribune wanted a system which delivered the *Chicago Tribune*'s News Story. Every story they published under the Tribune banner carried the Voice of the newspaper. The Voice reverberated in every Tribune journalists' ear and that Voice was an asset to both the authors and the readers who purchased it. They wanted a system that allowed them to do their jobs—tell the story—but still took advantage of computational medium.

After returning to Boston, I took these criticisms to heart, after all I had just spent five days in an entire organization filled with people who devote their hearts and souls to publishing the "truth." It amazed me that not only the editors and journalists were concerned about the integrity of the newspaper, but so were people running the presses. While

[1] While in Chicago, Chloe Chao MS98 and Matt Grenby MS98 of the Aesthetics and Computation Group created two demonstrations, Cluster and Glom. Their applets were computational tools to streamline the editing process. These applets are available at http://ic.media.mit.edu/tribune.

the huge five-story high presses churn out 8000 papers a minute, people pull copies of the paper out of the assembly line and read the stories, looking for typos and misprints.

Figure 75. Two different layouts which express different emphasis with the same photographs. The picture of the girl tells the human story, the picture of the park shows the grand picture

Macro and micro. I decided to focus on one single story and a single concept: context. I remember a discussion with the *Chicago Tribune*'s Photography Editor about why it is important to run two pictures of big events on the front page. One picture captures an overview of the whole event: the entire town flooded by heavy rains, or the thousands of people watching fireworks. The second picture shows the human side and brings the event to a personal level: the mother crying in front of her flooded house or the ten-year-old gaping at the sky. The two pictures allow the reader to understand both the macroscopic event and the human detail and how they relate to each other. Those pictures draw the reader into the text that describes the event in further detail. All the pieces work in harmony for the *Chicago Tribune* to give the reader an accurate and full idea of what happened on July 4th, or on the plains of the Midwest, or even on Wrigley Field. Each photograph and paragraph adds context to the whole.

I worked to deliver a computational system which could reveal context to the reader in a way that a newspaper could not. The result is ActiveArticle.

ActiveLayout sketch. ActiveArticle is based on the ideas of traditional newspaper layout. In December 1996 as a demonstration to the News in the Future Consortium, I made a sketch in Java of a proposed dynamic newspaper layout. It consisted of three photographs and three headlines taken from a front page of the *Chicago Tribune*. The page was devoted to the new Bishop who was coming to the Chicago area, and the three headlines and photographs were devoted to him.

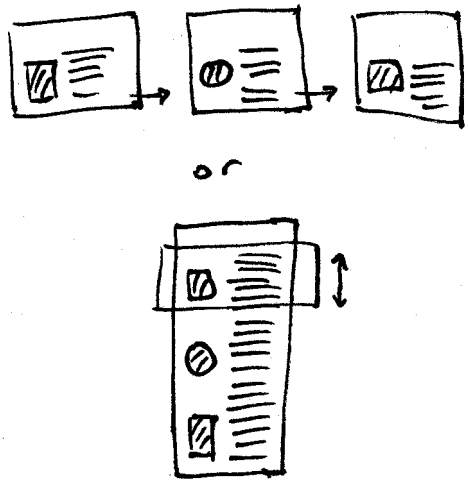
I scanned in the front page of the paper and used the original layout of the stories and headlines as a starting point.

Then I laid the page out three times, once for each headline. Each headline talked about a different aspect of the bishop, and so, I arranged the pictures differently, trying to optimize the effect of the pictures with the headline, using the same process a Layout editor would at a newspaper. Then I connected the three layouts together and created animated transitions between them.

In this simple sketch, the user clicks to change the headline story. This action triggers the pictures to animate to their corresponding news layout. This sketch did not manipulate the text of the stories, nor could the user do anything but click. However, the concept of a dynamic layout that responds to a shift of the reader's context, in this case changing headlines, forms the basis for ActiveArticle.

Figure 76. Java sketch of ActiveLayout. The layout of the photographs and headlines change as you pick different headlines from the original newspaper.

Figure 77. On-line text is either broken into many pages or put on a single page that you have to scroll up and down.



The problem of reading on-line. ActiveArticle builds on this concept by integrating the active layout with the act of reading. Newspaper articles can be between 500 and 1000 words long—more words than can legibly fit on a 640x480 computer screen. On-line newspapers have two solutions for this problem, either scroll the text off the bottom of the page, or break the text of the article up into screen-sized chunks of text and hyperlink between them. Neither approach makes on-line reading enjoyable. I wrestled with this issue when designing Shipwreck. [see "Shipwreck" on page 65]

When a text is broken into screen-sized chunks, it is difficult for the user to browse the article, to navigate around, or to know how long the story is. However, it is relatively simple to specify what the user will see, by designing one screen at a time. Designing one large page forces the user to scroll around the text. The "thumb" of the scrollbar gives the user an idea of their position in the story, a rough idea how long it is, and the ability to move quickly through the whole text. However, scrolling through an article makes it difficult to read because it is easy to lose your place in the text while scrolling it up. The user's machine can also override common text attributes like typeface and type size wreaking havoc on any layout.

Figure 78. Initial ActiveArticle screen

In ActiveArticle, I combine the best of both these approaches into a dynamic presentation of text. On the left-hand side of the screen, I fit the entire text of the news article by scaling its size down to fit the screen. The text is "greeked out" but serves as an indicator to the user how long the article is exactly. I overlay the article with a series of red rectangles representing the size of each line of text in the center of the screen. I will refer to this part of the interface as the TextBar. The rectangles form a "bubble" which initially bulges out at the top of the article.

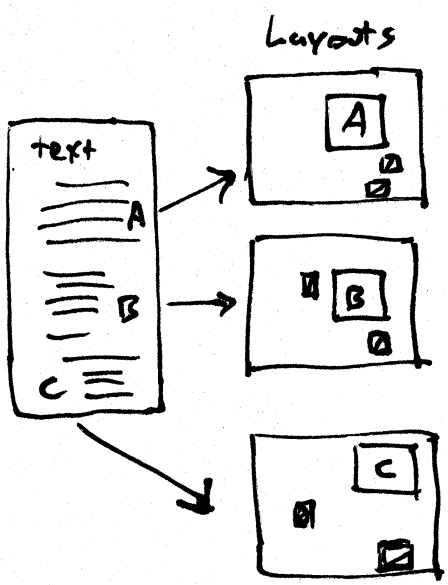
Figure 79. Active Article after scrolling down to different parts of the story

In the center of the screen, I draw the text of the article again, and scale up only the lines of text indicated by the TextBar. When the user drags their cursor along the TextBar, the "bubble" follows underneath the mouse. This causes the text in the center of the screen to "roll" to the position indicated by the user. The TextBar acts like a scrollbar for the reader to change their position in the story. The "bubble" of text underneath the TextBar on the left is scaled up to a legible size in the center as if underneath a magnifying glass. Using this interface it is possible to jump to a specific paragraph, scroll quickly or slowly, and constantly know the reader's absolute position in the story. Furthermore, because the text is rendered inside a java applet, the designer has complete control over the layout of text and photographs.

Though the TextBar interface to reading is an improvement over the traditional Web navigation interfaces, it barely matches the functionality of a real newspaper. With paper, the user has instant access to the entire text, a high resolution display, and a tactile interface. Combining the TextBar with the dynamic layout component from my first sketch tells a story in a way a newspaper can never do, at least until electronic paper and ink.

ActiveArticle content. I chose three photographs by Walker and a story written by Kiernan, the two journalists I worked with. Together they tell the story of a young mother and her daughter who live just across the street from Wrigley Field. I met them and was there while Kiernan interviewed them. I was with Walker when he shot his photographs. I had an experiential, emotional relationship to the story I was designing. I understood their intention as authors and looked for a way to reveal that intention in a computational story. So I used the ActiveLayout idea for the three photographs, but instead of changing the headline, I synchronized the changing layouts with the changing themes of the story that Kiernan had written.

Figure 80. As the themes in the text changed, I designed a graphical layout to reflect the shift in theme.



The first line of the story is "To Stefanie Bogdanovska, who is not quite six, the event transpiring across the street from the sidewalk where she Rollerbladed was a mystery." Obviously, Kiernan wants you to see the young girl at this point, so I placed the picture of Stephanie in a dominant position and size, in the upper right corner of the screen, the photograph of Wrigley Field below it, and the picture of the Partiers below that (Layout A). As the user reads the story, they drag the cursor along the TextBar to scroll through the story.

The story ends with the lines "'When I go back to the old neighborhood, it's sad, people have a lot of problems,' she said. 'When you come here, it's really fun.'" and I change the layout one last time to bring a picture of partying baseball fans to the front and push Wrigley Field back to the bottom—not out of sight—simply de-emphasized. The graphic layout reflects the state of the story, the last image is of people having fun around Wrigley Field (Layout C).

Clicking a picture. When a user clicks on a photograph, they are showing interest in the photograph itself, instead of the text. Clicking a picture in ActiveArticle moves it *in front of* the text, scales it up to full resolution, and shows its caption. The user can return to reading by clicking back on the TextBar.

Figure 81. Clicking on a picture centers it on the screen and shows its caption.

Knowledge and intention. The system has knowledge of the content through the thematic links connecting layouts to different sections of the text. The author can encode a sense of intention into the story in the dynamics of the interface. The author's intention is a procedure for handling the story content which is applied by the actions of the user to create a dynamic story form. The computational medium makes it possible to encode that procedure as codes which can be run by a reader.

What I like most about ActiveArticle is that most users understand how to use the interface after seeing it *once,*[1] because the interface is made solely of the content of the story (Rule 1). The system responds to your dragging and clicking intuitively, bringing the text you are interested into focus, or activating the photographs. Although users can not affect the content, they have satisfying control over how they access it (Rule 3). The content is compelling, Kiernan's text is fun to read, and Walker's photographs tell visual stories (Rule 2). Finally, users are provided with smooth, dynamic visual presentations which help them build intuition about the content. Less important things are smaller, more important things move to places of prominence (Rule 4).

[1] Louis Weitzman, a former Media Lab student liked ActiveArticle so much, that he took it with him back to IBM. They are developing the code into a more scalable, generalized solution for any kind of text. They have created a system which generates ActiveArticles automatically from a news feed by searching the text for proper nouns, then searching for photographs to match the text, and generating an XML template. Pretty cool.

After learning these rules and exploring a visual language with my 3D experiments, I modeled my thesis after ActiveArticle. I shot, edited, and collected my own content. The ActiveStories are ways that I communicate my intention as an author in intuitive, dynamic interfaces.

5.0 Experiments in ActiveStories

And now for something completely different.

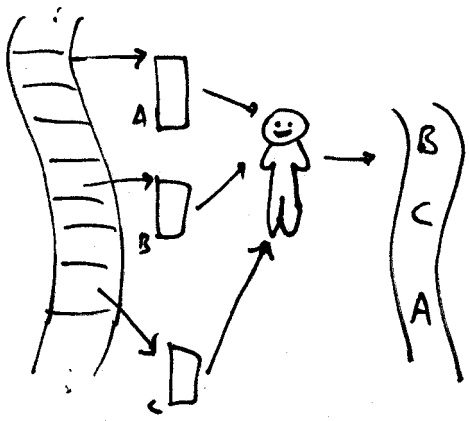
In this section I present the intent and functionality of my thesis projects, ActivePoem and ActiveXmas.

The ActiveStories are unique because each story is inseparable from the tool that I used to make it. ActivePoem combines a poem by Caraway Seed and four performances of it into a interactive single form. ActiveXmas takes two home movies filmed simultaneously in Paterno, Sicily and Atlanta, Georgia and portrays them as a single event. The content of my stories is personal: I know the subjects intimately, and that puts me in a unique position to tell their stories. In ActiveXmas, I connect the viewer closer to me and my family, or in ActivePoem, to the voice and words of a poet.

5.1 A little commentary before we go on

I have provided you with the history of Interactive Cinema. I have also walked you through the projects that I made and learned from.

Figure 82. A human editor.



Films are made by placing film into cameras, developing it, and editing it together. Film is easily cut into pieces, rearranged, and spliced together. I think that we are easily fooled into thinking that we can do the same with narrative. Logically, if the story is on film and you cut the film into pieces, then you can cut the story into pieces. However, before the computational medium, it was necessary for a *person* to splice the film together eventually. Usually you only splice it together once, but sometimes in films like *Clue*,[1] you actually splice together three endings from the same footage. In any case, with film or television, a person with a sense of story puts the pieces back together.

[1] *Clue*, 1985, writer/director Jonathan Lynn

In the computational medium, a human being no longer has to assemble the pieces. It is tempting to cut stories up into pieces and try to "teach" the computer how to put them back together. However, story is not very easily cut into granules. LogBoy and FilterGirl tries to assemble stories at the shot-by-shot level. [see "DMO, LogBoy, and FilterGirl" on page 45] Contour and Dexter[1] tries to assemble a bigger story from lots of little tiny stories. [see "Contour and Dexter" on page 49] In both cases, the authors of these systems tell us that it is a significantly more difficult task to build an infrastructure to support telling an extended story with plot, conflict, and resolution.

[1] Murtaugh discusses how he would build more comprehensive storytelling engine from Contour. However, it requires a much more complicated annotation scheme which could identify concepts like conflict and plot-points. I think that these concepts are difficult to encode into content.

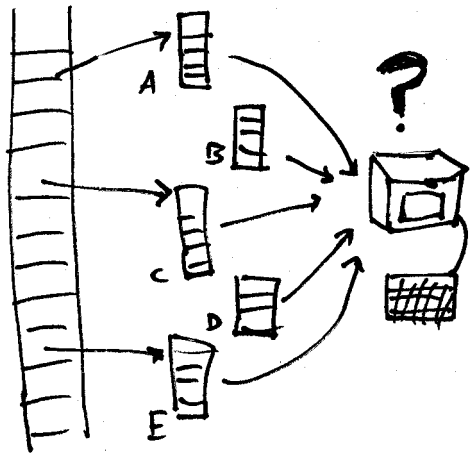


Figure 83. The computer editor

I believe that treating stories like streams empowers the author. A stream is a continuous non-segmented perspective of a story. John Maeda gave me an example of a real world streaming story. In New York, there was a theatre piece performed on an entire side of an apartment building. The audience were on the street and paid for binoculars and headphones. They peeked into any of the windows to eavesdrop on the conversation inside. Every room in the building was a thread in the same story. The story was told in real-time streams and could not be broken into story granules. The user's exploration of the windows determine the story, and the author depends on the viewer to construct the meaning of the narrative.

I have little faith in branching narratives or "Interactive TV" because they do not enhance how a story is told. If anything, the technology distracts from the story. I think that by the time kids have the attention span to read Shakespeare that their interest in Choose-Your-Own-Adventure books has disappeared. Translating the branched narratives from Choose-Your-Own-Adventures into DVD-ROMs cannot be the answer.

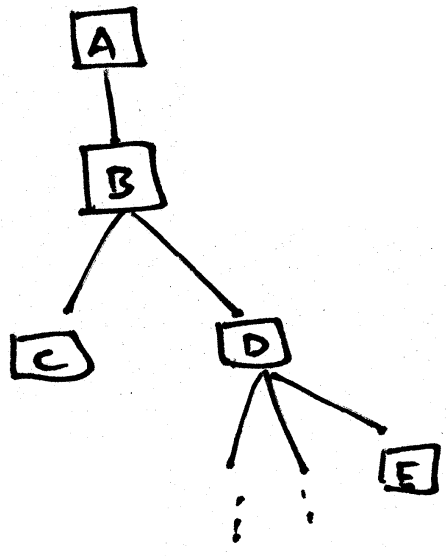


Figure 84. A branching narrative structure

Branching narratives allow the user to interact only at nodes of the story graph. In contrast, my stories invite the user to take control at any time when they want to. Branched narratives can only be explored a set number of repetitious ways. If Choose-Your-Own adventure books taught me anything, it was to keep my fingers in the pages so I didn't have to reread the stories to get to all of the endings. The ActiveStories may have fixed story content, but the user can have a unique experience every time. In ActivePoem, the different performances of the poem can be played in different orders, in tandem, or alone. In ActiveX-mas, users can take control of the "video spotlight" at any time and explore the video frame on their own.

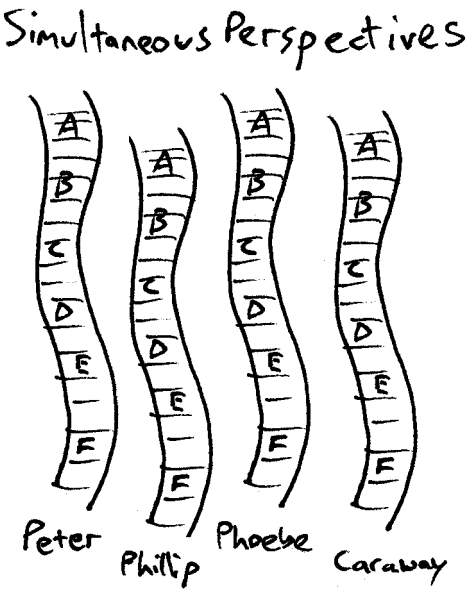
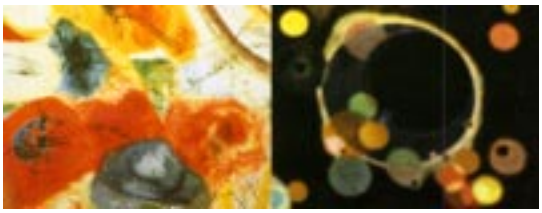


Figure 85. Four simultaneous stories from four unique perspectives

Both of my thesis projects are based in multiple streams of simultaneous story. Instead of cutting these stories into granules, I use the multiple streams to give you a larger sense of the unfolding event. The systems do not hide content from the user like most games do or use computational slight-of-hand to try to author stories. Instead they illustrate how an author can exercise more control over a story instead of how to take control out of the author's hands and give it to an algorithm.

It is clear that the choice of object that is one of the elements in the harmony of form must be decided only by a corresponding vibration in the human soul.

Wassily Kandinsky



In order for the form of a story to have harmony, it must be informed by the human need to communicate. I am searching to find my voice, to understand content, and to use computation to communicate my understanding. In ActivePoem, you hear Caraway's voice clearly telling you the poem's words, structure, and images. In my home movies, you see what I am seeing: my nephew ignoring my Dad on the phone and my mother's smile after saying good-bye to my sister. These are stories which mean something to me, and I am just beginning to learn how to tell them.

5.2 ActivePoem

Figure 86. The first line of Caraway Seed's poem

To give the words a voice. When I approached Caraway Seed to do the project, I asked her to pick a poem that used literary devices like multiple voices or repetitive echoes in it. Poems are expressed first as a voice in the poet's mind, then on paper, then in performance. I wanted a poem which she felt could not be performed well by a single voice. I was already thinking ahead about the ease of the computational medium to handle multiple streams of data. On the computer, Caraway's voice could exist in multiple spaces in overlapping times. ActivePoem is both a performance and a reading. It tries to bring the act of reading a poem and experiencing a performance together into a single active experience.

As it happens, Caraway and her poetry group had performed their poems a couple months prior to my experiment, and they had videotaped that performance. I watched that performance as Caraway and a fellow poet read alternate verses of Caraway's poem, echoing each other. I asked Caraway what she thought about the recording, and she said that although she liked the performance, the echoing voice was "not quite right." She wanted a voice more like hers. However in live performance, she could not do both voices by herself. There are other disadvantages to live

poetry performance: the audience is literally in the dark, in a crowded room, perhaps unable to hear or see the poet clearly; the audience does not have the text of the poem to reflect on. The audience has only one chance to hear the poem, and it is gone. However, live performance is not all bad, the audience hears the voice of the poet express her own words.

The poem. In ActivePoem, I worked with the author to film four distinct readings of the poem. One reading is of the odd-numbered stanzas, one reading is of the even stanzas, and then two readings of the entire poem. The poem can be broken into two distinct sections—the Voice and the Echo. The Voice is in the odd-numbered stanzas, the first voice you hear, and the dominant one. It is the first to tell you new things. The Echo is a softer voice, responding gently to the speech of the first. Sometimes repeating, sometimes changing the words of the odd stanzas, the even stanzas represent the Echo in the distance. Caraway and I chose this poem because of its unique structure which leant itself to a new kind of performance.

I wanted to emphasize the repetition of certain words and phrases throughout the poem, as well as the echoing aspect of the stanzas by using multiple performances to create echoes.

untitled poem by Caraway Seed

something about writing while kneeling
and the bed where I wasn't sleeping
or maybe I was
dreaming

I'm explaining to you but
these lines are perfect

it's my birthday
there is a yellow house
with the beach outside her

I knew that someday this hand
would be beautiful

I dreamed I had a brother
I did
he came to visit

it's my birthday
there is a yellow house
with the beach outside her

now he is the one who sleeps
or maybe not

I am in love with this hand
therefore she loves me too

we dream the same dream
you are getting close now

I dreamed I had a brother
I did
he came to visit

you say maybe detrimental
blah blah blah you say
I learned a lot I poured in
a ton of love you say I
had been playing poker
all day without even
realizing it you say
it had been a very bad
day you say until I
realized

now he is the one who sleeps
or maybe not

the dream says now I am
a yellow house a beach ball
sound moves me toward light
this is because I love you
and I am afraid

we dream the same dream
you are getting close now

your face had been made
of something beautiful and frightening
you say it's horrifyingly powerful
I thought you were talking about
the movie but you're not
dreaming
about a yellow house with the ocean
inside her
or maybe
you are

the dream says now I am
a yellow house a beach ball
sound moves me toward light
this is because I love you
and I am afraid

Figure 87. Three simultaneous performances turned on at once. On the left are three video windows drifting toward the bottom of the screen. On the right, in the text panel, the position in the poem of each poem is represented by the tinted words.

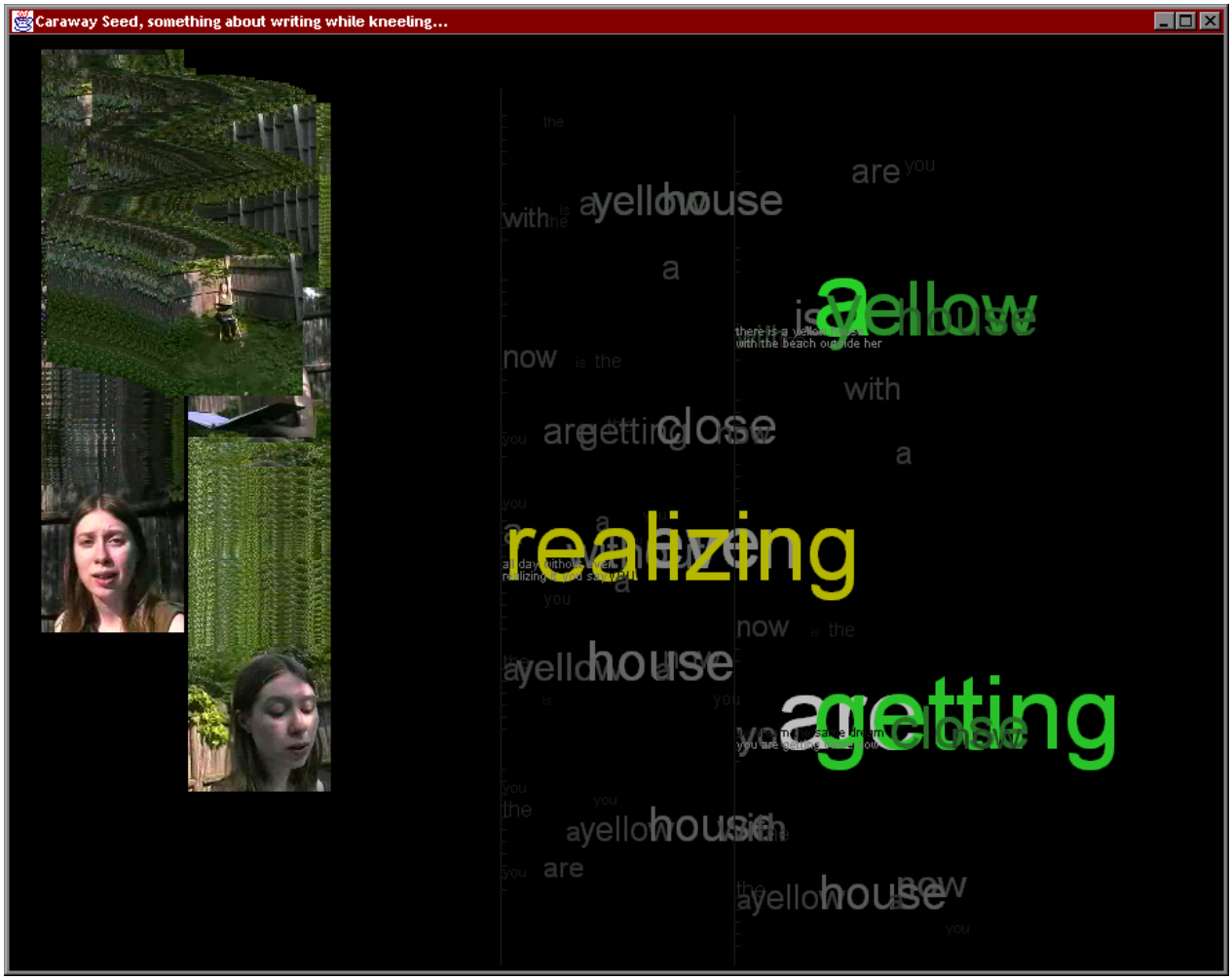


Figure 88. Below, the initial ActivePoem window with no performances playing

The visual interface. The interface consists of two separate panels. On the left-hand side is the video panel, and on the right-hand side is the text panel. In the video panel hang four paused videos each containing Caraway photographed from a different angle. The four videos are laid out into two columns of two. In the text panel are two grey vertical lines with tick marks down them. Each tick represents a line of the poem. The poem is laid out on the page in two columns left-justified against the gray lines, but initially hidden from view. In the left-hand column, the odd stanzas of the poem will appear and in the right-hand column, the even stanzas.

Active reading. The user can play any of the videos by clicking on them. As the video plays, Caraway reads the words of her poem. When Caraway begins reading a new line of the poem, that line appears in the text panel on the right. At the same time, as she utters each word, the word appears by itself and grows in size and in color saturation. The word "pops" out at you for a fraction of a second, then fades away into the background. The net effect is that you hear and see the words of the poem as they are spoken, as if the words themselves were aware when Caraway calls them. This is the active reading effect that I wanted to give.

Figure 89. A dramatic re-enactment of the "popping" effect

I also wanted to emphasize the repetition and echoes in the poem. So when Caraway reads a word, any other occurrences of the word in the poem also grow in size, but peak at a smaller size, then recede back into the background. The visual effect is that when common words are spoken like "yellow house" or "I" all the references jump out for a second around the page to give you a feel for where they are, and how many times things are repeated. Because almost every word in the poem is repeated, the whole page vibrates with motion as Caraway reads.

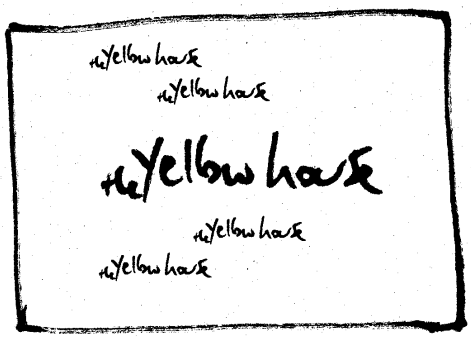


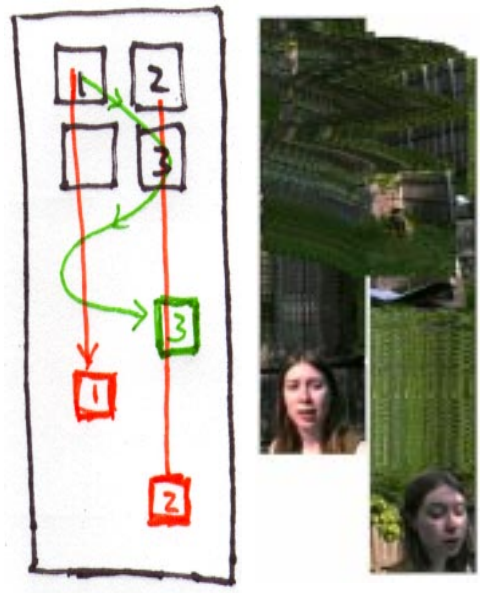
Figure 90. The text panel reveals repetition

Because each video can be stopped and started independently, up to four virtual Caraways can be speaking at a time causing an audio-visual barrage of poetry. The user has explicit control of every voice and can layer multiple readings. The text panel responds to each voice equally, activating the words from each reading accordingly. The emergent behavior is a transient visual pattern of rhythmic text synchronized to the voices of the poet.

Figure 91. The four video windows

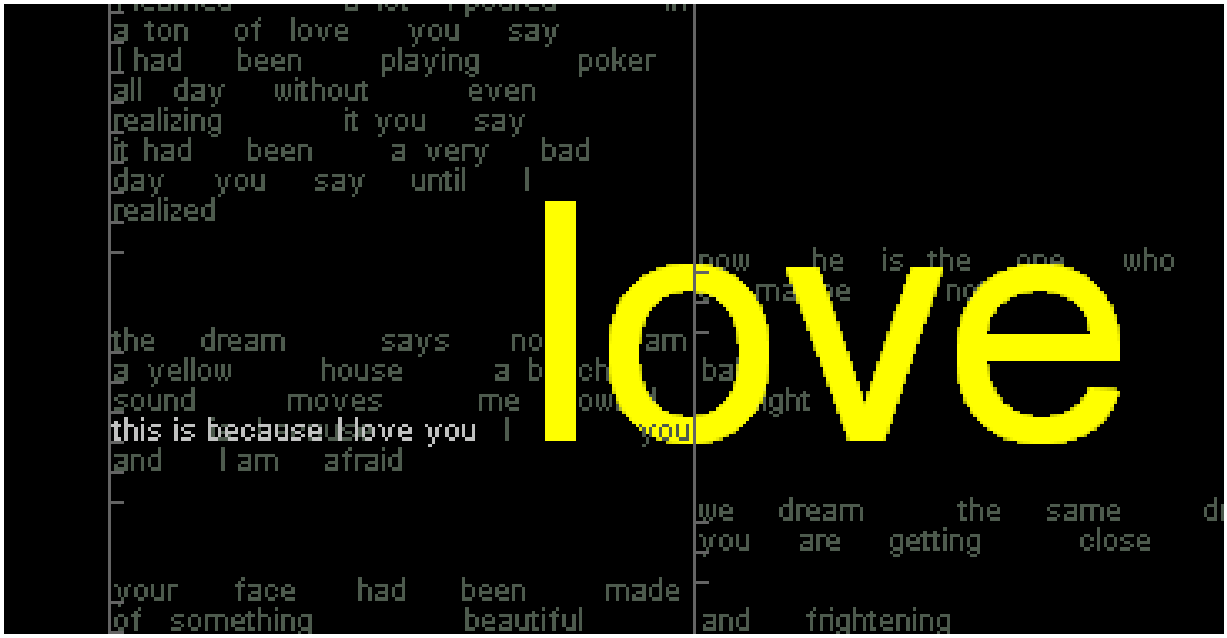
Four views. The video panel contains four different video streams laid out in two columns of two clips each. In the upper-left clip, Caraway performs only the odd-numbered stanzas of the poem. In the upper-right, she performs the even-numbered stanzas. In the bottom two clips, she performs the whole poem. Clicking on a clip toggles whether the video is paused or playing. As a clip plays, it moves spatially according to the line of the poem that Caraway is reading. The horizontal center of the clip aligns itself with the position of the line of the poem displayed in the text panel.

Figure 92. The VideoStreamer effect



As Caraway reads, the clip drifts toward the bottom of the screen stopping when she reads the last line of the poem. The vertical center aligns itself in the left column if Caraway is reading from the odd-numbered stanzas, or in the right column if she is reading from the even-numbered stanzas. So the video clips try to follow the movement of the words, moving left or right to follow the stanza, and up or down to follow the line. The video also leaves a trail behind it, a trace of each video frame, to create a VideoStreamer-like effect, to show the clip's path around the page. This provides a visual history of the path of all the video clips. The (x,y) position of a video clip indicates exactly which line Caraway is reading in that clip's performance.

Figure 93. A close-up of the user choosing the word "love."



Reading the words. The user can also click and drag in the text panel to see all the words of the poem appears in small grey type. While the user is dragging around the panel, all the videos pause waiting for the user to choose a word. If the user drags over one of the words of the poem the word "pushes out" at you for a moment indicating that it has been selected. This is the same effect that happens when Caraway reads a word in performance. Thus you can browse through the words of the poem. Activating a word also sends a message to all the videos to cue or rewind to where that word is spoken in that performance. The videos also float to the proper (x,y) positions to match the word, line, and stanza chosen by the user. When the user releases the mouse, the videos resume playing from the new word chosen by the user. In this way, the user can find a line, stanza, or just a word and fast-forward all the performances to start from that word. The reader has complete random access to the videos and the text.

Figure 94. The videos on the left spatially align themselves with the position of the word "love"

The typography design draws on the Temporal Typography work of Yin-Yin Wong. [see "Dynamic typography: Small & Wong" on page 58]

Poet's intention. The goal of this project is to give the audience a closer connection to the poem and the poet's intention. By simultaneously providing a performance with the text of the poem, the viewer can experience a performance while actually reading the poem, experiencing its text as well. Because the poem is a poem, its text is the center around which the performance and interface is built. Yet this interface tries to reflect visually the internal structure of the poem's dual voices and echoes. It also gives the user an responsive way to navigate through the poem and the multiple performances.

Figure 95. This is not a hand-drawn yellow house.

ActivePoem does not reveal the "true meaning" of the words. It does not provide a literal interpretation of the meaning or present concrete images in place of the words. An interpretation locks a particular image into the form and discourages readers from interpreting the words themselves. If when the words "yellow house" appear, I presented an image of a particular yellow house, that image robs from the abstract nature of the words. Everyone who reads the poem imagines a different yellow house that is tied in with their own experience, making meaning for themselves.

The only visual elements in ActivePoem are the poet's own performances and her own words. Poetry as a literary form, takes time to understand—time to make sense of the words. The system provides a way for a user to study each word of the poem and how the poet performs every word. The system also gives users a way to explore a multiplicity of ways the poet reads her poem. The system is designed for readers who want to come back and reread the poem, just as they might return to a book of poetry.

Building a mystery. As an experiment, I wanted to try out the system with some different content, especially to see how the system's performance would change with only a single stream of content. I used this thesis's theme song Sarah McLachlan's *Building a Mystery*. I found a digitized video and the song's lyrics on the Web and dropped them into the ActivePoem system.

Figure 96. A temporal sequence to show the dynamic display of the words "Now you're working, building a mystery."

As I hoped, the system performed much faster with a smaller text panel and a single stream of video. Thus the frame rates were faster, and the animation was more satisfying. The pop song's structure of verse and refrain worked well in the text panel because whenever McLachlan would sing a refrain, all the echoes of that refrain would also appear. Another feature was that the VideoStreamer effect in the video panel was more pronounced, because of the cuts in the music video.

However, the use of the music video in the system felt forced, because the music video is a polished presentation of the music itself. The video is reproduced is a tiny window that presents a single interpretation of the song's lyrics. The video is a finished piece, shoehorned into my display engine. Because the video is a single edited stream, there is not much to explore. Perhaps if there were multiple simultaneous streams of story in the video, my system would fit better. Perhaps, this is the answer to my problems with Fluid Transition. [see "Fluid Transition" on page 82] Still, it was interesting seeing a different set of content inside my form.

5.3 A recipe for ActivePoem

1. It starts with fresh content.
2. Stir in a little representation.
3. Grind out dry-roasted Java, coat liberally.
4. Add a cup of Quicktime.
5. Code until done.

Four readings, one poem. After Caraway and I decided on which poem to use, we chose a setting for the video shoot. Caraway's back yard was sunny, green, and only occasionally did a monster truck rumble by. [see "Shooting the video" on page 120]

Caraway read her poem differently each time we recorded it. To create the echoing effect, we recorded the even and odd stanzas in separate video tracks. Then I asked her to read the whole poem from two different angles in a medium shot and a long shot. In the medium shot, she decided not to say the last stanza of the written poem because according to Caraway, "Sometimes, that *is* the end of the poem." In the long shot, she reads all the stanzas including the last echo.

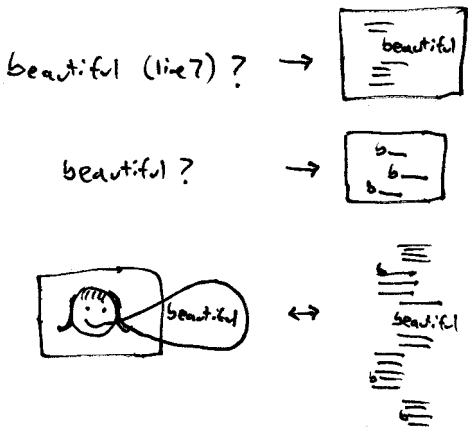
Representation without taxation. When initially thinking about the system, I assumed that I was going to cut the video up into smaller clips into "poem granules" and annotate each granule. I thought that it would be useful to have each stanza in a different video clip. From the beginning, I wanted to have random-access to each word in the poem, so I knew that I would have to create a representation to store the time of each spoken word in the video. I realized that it would be much less taxing to leave the performances unedited and create a stream-based representation of the whole poem instead. I needed to determine at what precise time in each video Caraway spoke each word. I wrote a tool to do just that.

A spacebar track. The tool reads a movie file and a text file. In the text file is a list of all the words spoken in the movie. This was easy to make because the text of the poem is fixed. I thought that the easiest way to determine when Caraway says a word listen to the soundtrack and hit the spacebar every time she said a word. Creating a "spacebar track" would allow me to synchronize external events to the performance of a word.

This representation is useful in three ways:

1. You can search for an individual word in the poem. Find the word "beautiful" in line seven.
2. You can search for all the occurrences of a word. Find the word "beautiful" in all lines.
3. You can use the video to find a word in the text or use the text to find a frame in the video—the annotation is bidirectional.

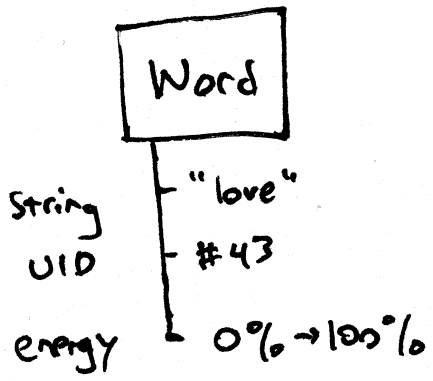
Figure 97. The eternal searches for beauty



Passive playout. I use the first two properties in the passive playout of the poem. The system reads the log for each movie and polls each movie to check its current time. If the current time is greater than or equal to the time of a Log Entry, the word in that Log Entry has been spoken, and the system queries the system to activate the Word object using its UID and String attributes.

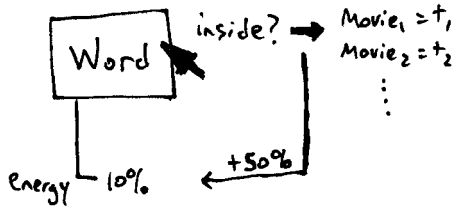
In my system, a Word object has three attributes: an energy level, a String, and a UID. The energy level determines the size and color of the word when it is eventually drawn to the screen. The String and UID are used during queries. If the query matches the UID, then the word's energy level is increased 50% of the maximum. It the query matches the String, then the word's energy level is increased 10% of maximum. Over time the energy level decays down to a minimum energy level.

Figure 98. Representation of a Word



The effect is that when Caraway speaks the word “beautiful” in line 7, that “beautiful” grows to its maximum size and brightest color, then slowly fades back into the background. Any other occurrences of the word “beautiful” grow also, but to a smaller size and darker color. By changing the growth and decay values, the author can affect how quickly words fade in and out. The author can also tweak how intensely words are activated to change the effect of the spoken word and the echoes.

Figure 99. The process of dragging across a Word’s bounding box.



User browsing. When users click and drag around the text panel, the panel checks to see if the mouse is over one of the words. The Word objects have an “inside” function which determines if the cursor is inside a Word’s bounding box. If it is, the system sends a message to that word to activate itself and uses the UID to set the time of each movie to the time the word is spoken.

The system has very simple rules but the emergent effect from all these simple interactions is a complex and beautiful visual display.

The Quicktime canvas allows you to set the background color to null which means that it does not clear the background. This is how I implement the VideoStreamer-like effect. [see “VideoStreamer: video in x, y, and t” on page 56] Wherever it goes, the movie leaves a trail made up of the edge pixels of each frame. Though visually interesting, it is chaotic and unpredictable because I am not in complete control of the pixels that are left in the background.

Limitations of the interface. The visual display of the videos is chaotic because of the VideoStreamer effect, and I would like a more controlled process to maintain the visual history. The user should be able to drag the movie windows in the video panel to set their position in the poem. This functionality would give users a mechanism to set up

different juxtapositions of words and sounds from different readings. I have found myself wanting to set up a relationship of videos to hear certain phrases together, which is difficult to do, given the current interface.

I would also like to improve the text browsing interface. It would be nice if there were a mode of the text panel which made the whole text of the poem legible. When you drag around the text panel, you can see the whole text, but at a small size. I had to make a trade-off between the ability to read a whole stanza, line, or word at a time. I found that if I made more than one line available during the dynamic text presentation, it would distract from the dynamic elements. However, only having the words grow and fade alone made it difficult to focus on the sense of a line. For this reason, I decided to fade in one line of the poem at a time, as well as performance reasons outlined above. This device gives the reader a visual anchor for each line as well as a whole phrase to read and think about. On the other hand, the reader should have the freedom to read a whole stanza or the whole poem.

The ActivePoem environment allows for a different kind of exploration of the poetic space of performance and text. Other forms of CD-ROM poetry scroll the text of a poem along with a video but do not allow you to index the video to the performance of a single word or have multiple simultaneous performances of a poem. The poem's performance is repeatable and offers many different ways to experience the poem, so that readers might come back to reflect on the poem. With every performance, ActivePoem offers a unique visual experience because of the different possible recombinations of the performances.

The tool is the story. This project is an illustration of the power of a "shallow" knowledge representation. The representation was just enough information for me to get the job done. The "annotation tool" was little more than a MoviePlayer with a spacebar. However, it would have been prohibitively difficult to build that representation using any ready-made tools. For instance, to use Premiere or MoviePlayer, you would have to advance the movie manually and record the times and frames of each word in each video. That is a lot of clicking and dragging. If you did get the information, creating the interface in HyperCard or Director would be nearly impossible. HyperCard and Director both use the operating system's default text entry area to display text. It is difficult to animate that area dynamically. If you chose to turn the words into sprites, you would have to generate bitmaps for each word in each size and color. The process would have to be performed manually in Adobe Photoshop or Macromedia Freehand. All of this work is an artifact of using tools for fixed media (cel animation, photographs, movies) to make a dynamic, responsive form.

From ActiveArticle with love. This work draws on ActiveArticle in the sense that I wanted to create a completely different kind of scrollbar to access the content. In ActiveArticle, I create a TextBar on the left-hand side that you can drag up and down that is also the text of the article. In ActivePoem, you can drag up and down the words of the poem. Dragging actually has the effect of cueing the video clips to that point in their performances. Rather than having a meaningless scrollbar underneath each clip to drag back and forth, I give you access to the content of the video with the content of the poem, drawing a one-to-one connection between the words and the frames of video.

5.4 ActiveXmas

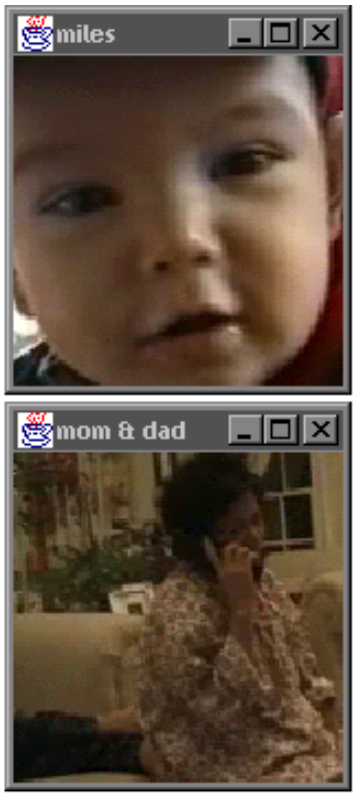
ActiveXmas started out as a lucky coincidence. My parents called my sister at 2:00AM Eastern Standard Time which was 7:00AM in Paterno, Sicily. My nephew, Miles, was eight months old, experiencing his first Christmas. I was shooting my Mom during the call, and my brother-in-law Bink was filming in Sicily simultaneously. We captured both halves of the phone conversation between Atlanta and Paterno. Months later when I got Bink's footage, I synced the two streams together using two VCRs, and suddenly I could see the whole picture. I could see my mother's reaction to Miles opening his first Christmas present. Even though we were separated by 3000 miles, we were together as a family. I wanted to figure out a way to make these two videos into a single event. I wanted to tell the story of the event to my family, so that they could experience the event brought together. I also wanted a way to share the event with my friends, who could not see the same details that I could see, simply because they did not know where to look. I made ActiveXmas to have a way to tell a personal audience about that special Christmas morning.

ActiveXmas is built out of two separate video clips which have about five minutes of common time. When synced up properly, you can hear both ends of the phone conversation perfectly. The clips are both unedited, continuous streams.

All I want for Xmas is my in-ter-face. ActiveXmas uses three video windows. The main window holds the two original video clips placed side-by-side. The window is dimmed except for a mouse-controlled "spotlight." The spotlight is a small square inside the video frame that shows the area inside it at its normal brightness.

Figure 100. ActiveXmas main video window: on the left is the scene from Sicily, and on the right is the scene from Tennessee.

Figure 101. The two spotlight windows



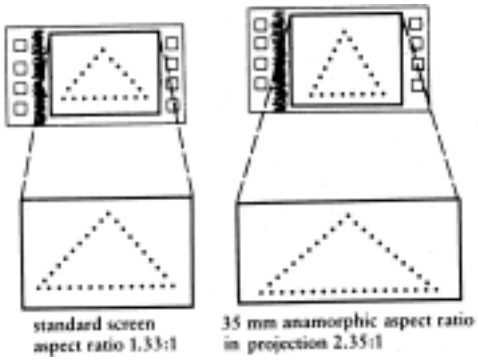
The area underneath the spotlight is "projected" into the other two windows. One window holds the spotlight on the video of Miles, the other holds the spotlight on the video of Mom and Dad. By dragging the mouse around the main window, you can grab control of the spotlight interactively and move it around inside the movies.

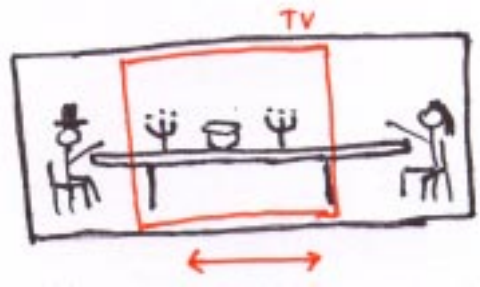
The spotlight is a virtual camera shooting inside the frames of the original video clips. The external windows are real-time viewscreens which display the effect of your new camera. By changing the size of the spotlight, you can "zoom" into and out of details in the video frame. The spotlight allows you to "reframe" the action in a video, highlighting details that you think are the most important. In a way, this is similar to the "pan and scan" technology used when a film is transferred to video.

Because a modern film's aspect ratio is at least 1.85:1 [CinemaScope 2.35:1] and the standard television's aspect ratio is 1.33:1to transfer a film to video, you must either "letterbox" the movie or crop the sides.

In most movies, subjects are usually placed on the far left-hand or right-hand part of the frame and cropping the edges out would cut out the actors. To solve this problem, telecine machines allow the operator to move the frame around during a shot and selectively crop the scene, a process called "panning and scanning." The other process, letterboxing, scales the whole film to fit in the TV frame with black areas filling the top and bottom parts of the frame. Letterboxing is considered the superior way to transfer a movie to video, because it shows the movie's whole frame. However, most videos "pan and scan," which to the filmmaker's chagrin, allows the telecine operator to redirect the camerawork of the film. Reframing a video is a powerful technique for guiding a viewer's attention. In ActiveXmas, the author is given control of the reframing window, to frame the video to show the most important details.

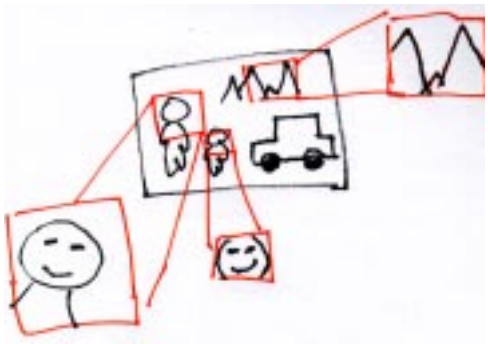
Figure 102. Academy Frame (TV) vs. Cinemascope (top) A crucial scene from *Citizen Kane* would be tragically reframed by panning and scanning. (bottom)





Reauthoring and reframing. The ActiveXmas system is the realization of the idea of a self-modifying film. Viewing the movie can change the way the next person sees it. The movie data itself is fixed, but viewers can save how they perceive the story and become authors by storing those perspectives. Then viewers can share their viewings with others. For instance, I could author different viewings for my friends than my family. The author has a chance to reframe his or her point of view and clarify it, using the spotlight windows. It is a way to make sense of all the data that we try to capture in a story. When I try to explain a video to someone, I inevitably end up physically pointing at details and moments, to try to distinguish them from the

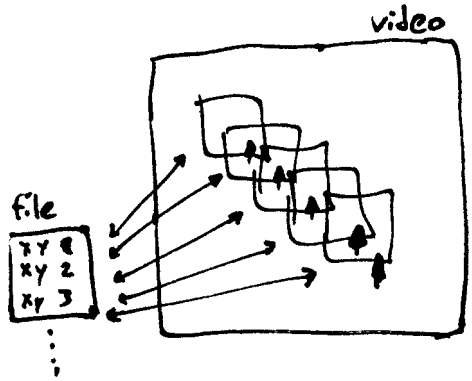
Figure 103. Reframing a video can pull out different stories



rest of the information in the frame. However, when your audience is an ocean away, there is no physical way to indicate what is important.

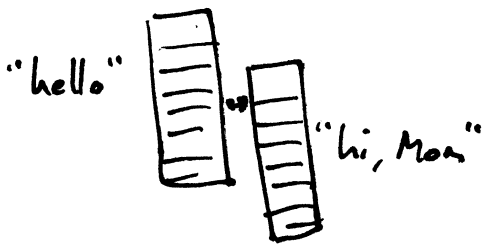
By using a combination of audio clips and visual annotation, authors have new ways to reauthor their content. I used this multi-linear story about my family to illustrate this point. While watching ActiveXmas, you have a chance to annotate the video while you are watching it. Those annotations can then be replayed for other viewers who may be curious to see your version of the story.

Figure 104. Mouse positions are stored in a file to be played back on request.



Annotating the video. In the ActiveXmas main window, clicking on a video toggles whether it is playing or paused. The user can play a movie and use the mouse to control the position of a "spotlight" over the video. The user can also zoom in and out to cover different amounts of the frame. To annotate a video, you simply have to watch it with a mouse in your hand and your fingers on the zoom buttons (like a video game). The control of the "spotlight" allows the user to see the area of the video frame they want to see and dim out the rest. Every mouse motion stores an (x, y) position, the current time in the movie, and the size of the spotlight window (the zoom value). I wrote this tool to annotate my videos quickly. The system can read the log and replay the events, reconstructing the visual path of the user through the video in real time. In ActiveXmas to watch my version of what happened that morning, you hit "R" to Read the data, and the system loads how I watched the movies.

Figure 105. Automatic synchronization



The two home movies sync themselves up automatically so that the phone conversation can be heard from both sides. In the two spotlight windows, the user watches the conversation unfold seeing the details of the videos through my eyes.

ActiveXmas started with the story, and I developed the technology to support its telling. The representation is light-weight, real-time, and visual, because I needed those attributes to tell this story. Because the design revolved around the content, it was easy for me to design a shallow representation for my needs. [see "Representation without taxation" on page 104] ActiveXmas provides a novel way to add visual narration to a story.

> The plot may arrange cues in ways that withhold information for the sake of curiosity or surprise. Or the plot may supply information in such a way as to create expectations or increase suspense. All these processes constitute narration, the plot's way of distributing story information in order to achieve specific effects. Narration is the moment-by-moment process that guides us in turning the plot into a story. [Bordwell 1990]

Narration and narrator. ActiveXmas brings up the question: "Who is the narrator?" The original footage is the unedited point of view of an observer in the story—me. A home movie does not necessarily have a narrator making sense of the actions that happen and turning key events into a story. That is usually why watching your friends' home movies is so boring even when they are narrating the events. They can not narrate fast enough, and 90% of the time, you do not know what to look at in the frame or why it is important. ActiveXmas provides authors a way to visually narrate a story.

However, ActiveXmas also gives every viewer the chance to become author. While watching a movie, you can take control of the visual narration from the previous author. In the case of my movies, each member of my family will be interested in different aspects of the video. (Knowing my family, everyone will want to take control at the same time.) The viewer can control the moment-by-moment process of specifying the important elements in the frame.

The experience. ActiveXmas is a new way to experience a home movie about my family, through my eyes. Undoubtedly, it is not going to be interesting to a wide audience of thousands. ActiveXmas illustrates a new way to share personal stories to small audiences. The networked, computational medium no longer needs a movie theatre full of people to support its economic needs, but can serve an audience of seven people, distributed around the world.

I want to share my stories with people who are close to me—the people who have heard the stories about my family and know that my mom loves to give presents. You already know that Bink and Gigi have been dating since I was three years old, and persistence, more than anything else, convinced my parents it would be OK to "marry American." You already know that Miles is their first grandchild, the newest baby boy in the family, finally relieving me of the position. And if you don't know these things, I hope that by watching ActiveXmas, you might want to know more about the cute baby's face that my eyes are drawn to.

Finally, what more do you need to know about my mother's smile, as she says good-bye on Christmas morning. I give you a glimpse of my family, using these computational tools, to share with you an emotional moment for me.

6.0 Conclusion and the future

6.1 Why is there Interactive Cinema?

> [Film makers] have forgotten even [the film's] basic aim and function: the rôle set itself by every work of art, *the need for connected and sequential exposition of the theme, the material, the plot, the action,* the movement within the film sequence and within the film drama as a whole.…This is all the more necessary since our films are faced with a task of presenting not only a narrative that is *logically connected*, but one that contains a *maximum of emotion and stimulating power*.
>
> Montage is a mighty aid in the resolution of this task. [Eisenstein 1947]

Eisenstein talks about the role of art, to evoke emotion and power. Perhaps today Eisenstein would say, "Computation is a mighty aid in the resolution of this task." We have a responsibility to bring narrative into a new medium in a way that maximizes its expressive power.

Analog: A cut is forever.
Digital: *rm* is forever.

Film montage pushes the film medium to its physical limit, cutting the film medium into strips of a few frames and splicing them back together, once and for all time. Montage editing tests the physical limit of the film medium itself. However, the digitized data of a movie can be separated into the discrete pixels of every frame and be combinatorically recombined, trivially, with no consequences to the author or the media. However, the rub lies in finding the recombinations that enhance the author's expression and the narrative contained within it.

We are searching for ways to change the experience of narrative, inheriting from the legacies of the design, film, and computer science. Though we are on the brink of a new art form, we must not forget the purpose for which we are striving "a *maximum of emotion and stimulating power*."

6.2 What is the point?

> Before them stand the works of today, untainted by the past, primary shapes which identify the aspect of our time: Car Aeroplane Telephone Wireless Factory Neon-advertising New York! These objects, designed without reference to the aesthetics of the past, have been created by a new kind of man: **the engineer!** [Tschichold 1995]
>
> We have been assaulted by television our whole lives, now we strike back!
> Nam June Paik

Figure 106. Fluxhouse by Nam June Paik

I have spent the past four years looking within myself for stories. I love meeting new people and sharing with them my memories and experiences. I have made movies and watched them. I have programmed in scheme, c, and java. I have been searching for a way to express myself using these languages and my own stories.

Slowly, I am starting to understand the connection between technology and expression. I think that the ActiveStories project has been a way for me to explore myself. In the beginning, I found myself searching the technology, looking for ways to make a cool hack or interesting demo. However, the technology has no sense of story and will itself never be a compelling reason to tell a story. Story comes from the human need to share our lives with each other, to learn about each other and ourselves, through our need for communication.

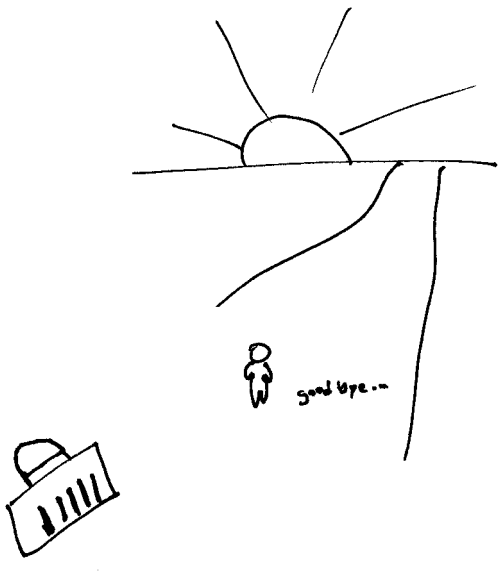
In my search, I had to cast off the technology and look for the stories I needed to tell. It is interesting that although I love to tell the stories of my family to my friends, that it is difficult for me to use my family as content for my projects. I believed that the ActiveStories had to appeal to a wide

audience and telling stories about my family and friends would not be interesting enough. And yet, I know those stories better and more deeply than any story I could make up.

A new process. The process and techniques in this thesis do not point directly to applications. Instead, I think that the ActiveStories result from a new iterative process between technology and author. The new stories approach authorship and viewing from the new perspective of a new medium. In conversation with Teresa Castro, she brought up the idea that one of the goals of an artist is to share with others the "moment of creation." I became excited, because in a way that is what the computational medium can do in a very concrete way. In ActiveXmas, I stored the process of how I experience the story of Christmas into the art form itself. When people watch ActiveXmas, they are in effect, witnessing the moments of recognition, when I began to construct the events in the video into a personal story.

A new medium. The new computational medium can not distinguish between an author and a viewer. Both sit in front of a keyboard. Both use the mouse. Both use tools. A viewer can have as much control over the story as an author, because the viewer is sitting at the same tool used to make the story. The computational medium presents a way to share processes of creation, as well as the artifacts of those processes. ActiveXmas and ActivePoem let both the author and the viewer share a new kind of experience.

Thus ends six years of exploration at MIT, and just as surely, it also begins a new search.


good bye..

7.0 Appendix

7.1 ActivePoem post-production

Shooting the video. I used a Lectronics UHF wireless lavalier microphone and set up in a yellow chair in the corner of the yard in the warm sun. I used a JVC GR-DV1 Digital Video camera. It was small and unobtrusive, and I wanted to put Caraway at ease. The wireless microphone made it easy for me to set up anywhere in the yard and get good sound.

There was a lot of ambient wind noise and when playing multiple performances, the wind begins to overpower Caraway's voice. I shot video in medium shot (head and shoulders), long shot (whole body), and extreme close-up (lips only) of Caraway reading the poem. Unfortunately, the JVC camera's optics are poor for close focus, making the extreme close-ups difficult to shoot. I was unable to capture an entire performance in extreme close-up. If I had time, I would try again, with a different camera, to capture this unique perspective of performance.

I digitized the clips using a PowerMacintosh 7600/180 using the on-board digitizing board. I cropped the video to 107x112 to match the vertical composition of the readings. I did not use a full-frame digitizing system, because I could not present full-frame [640x480] movies at 30 frames-per-second [fps]. Displaying full-frame video stresses the processor and memory subsystems to the point that the system can not do anything else. I decided to trade off resolution and frame rate for other functionality. I wanted to play multiple streams of video, drive an animated text display, and manipulate the videos' position in real time.

I used MovieCleaner Pro to crop, compress, and flatten the movies for cross-platform use. I used the new Quicktime 3.0 Sorensen Movie codec, which provides extremely good compression and image quality compared to the old standard, Cinepak. I used 44.1 KHz, 16 bit mono audio with IMA4:1 compression, but some of the resulting movie files would not play correctly on the Windows systems, instead giving me static. On those I used 22 KHz, 16 bit mono audio, with no compression. Note that all the files worked on the Macintosh correctly.

Quicktime 4 Java. For the four videos, I use a new Application Programming Interface [API] called Quicktime for Java. Although the software is, as of this writing, still pre-alpha, the API is quite comprehensive, giving developers complete access to all the Quicktime functionality. I use Quicktime to do all the video rendering in the video panel, and Java's built-in Abstract Windowing Toolkit [AWT] to do all the text rendering. In my first implementation of the system, I used Quicktime to render everything including the text in the same panel. This meant that I could have the text float over the video windows. However, performance suffered. Performance is an important issue in dynamic presentation, and I determined I needed to get at least 10 fps for the system to "feel" right. The Quicktime movie rendering is so efficient that the time to render each frame does not increase as you add movies to the panel. However, the total area of the Quicktime canvas was directly proportional to the overall performance of the system. I had two choices, either shrink the entire window size to 320x240 or use the AWT to do the rendering of the text in a separate panel than the video. I chose to use the AWT because it allowed me to keep my large window to hold the entire poem on-screen. Separating the video from the text also made the text was more legible.

The AWT Text Panel. Performance dictated how much of the poem the user could see at a time. The AWT drawString() method is slow and drawing each word causes the whole panel to choke. Even trying to draw only the active stanza affected the frame rate significantly, especially when there was lots of other activity on the screen. By drawing only the lines currently being performed, the frame rate was satisfactory. The lines of the poem give the user an easy visual anchor to scan back to when looking for the next word.

Getting the representation. Caraway reads quickly, so I set the tool to play the video back at half-speed, so that it would be easier for me to mark exactly when a word begins. The custom tool makes annotating streams of video a fairly trivial process. The tool shows you two words: the word that you are waiting for and the lookahead of the next word. You listen to the video, and every time the speaker says the word you are waiting for, you hit the spacebar. The tool records a unique identifier (UID), the word, and the current time in the movie. At the end of the movie, you hit "w" to Write the data to a file on disk. Now you have a complete log of the words and timings for that clip.

However, not every movie contains Caraway speaking every stanza, so it was important to have a UID for each instance of a word in the poem. By counting the poems' words, I created a UID for each word and stored it in a registry that contained all of the words and UIDs.

It was necessary to reconcile the UIDs of the registry with the UIDs in each of the individual movie files. I edited each movie log and adjusted the UIDs to match the ones assigned by the registry. I use the words to lay out the text panel, and by counting the extra carriage returns as I read the poem in, I can determine which lines belong to the even or odd stanzas.

7.2 ActiveXmas and Quicktime

Back to the Java grind. ActiveXmas is built using the same technology as ActivePoem—Java and Quicktime. Quicktime provides many different ways to control the display of a movie. You can control the size, the position, the scale, and the cropping rectangle of a movie. I use the information I store in the log about the (x, y) position of the camera and the size of the spotlight to generate a new view of a movie, cropped and positioned correctly in the spotlight window. The time information ensures that the movie in the spotlight window is synced properly to the movie playing in the main window.

To create the visual appearance of a spotlight on the original video, I use a Quicktime compositor object. This object allows me to draw an arbitrary image using the java AWT and composite it on top of a playing Quicktime movie. I can also control how that AWT Image is composited with the movie using the standard Quicktime drawing modes: AND, OR, XOR, BLEND, etc. Unfortunately, since there is almost no documentation on Quicktime for Java's drawing modes, I had to resort to trial and error to find the proper mode that would allow me to create the visual effect of a soft-edged mask.

Additional audio. Although the current version of ActiveXmas does not include this feature, I hope to incorporate it later. In addition to the original two videos, I also recorded audio clips which tell more of the background story of what was going on that Christmas. Using a similar representation to ActivePoem, I recorded the times I wanted to trigger these clips to start playing. Then in a preliminary version, I simply triggered my clips to play at those times. However, the problem was that my commentary clips were long (~20 seconds) and sometimes overlapped each other. Having two or three of commentary clips playing simultaneously

with the two audio tracks of the original movies ended up in an auditory cacophony. I decided that I needed to either add an interface to the clips, or re-record them with shorter clips. Glorianna Davenport gave me feedback on the "flatness" of the clips, and suggested ways that I could make them more interesting. I could either act more like an omniscient narrator or the whisper in one ear. Most importantly, Davenport wanted the commentary to be shorter and more to the point.

In fact, it was after recording these audio clips that I came up with the idea to do the spotlight in the first place. I was showing John Maeda the first version of this story with the audio clips, and we talked about the visual emphasis of the story. He suggested that I clarify what I thought was important about those movies, and so I started coding up a new visual interpretation of the video. We also talked about telling an exclusively visual story rather than resorting to a text display.

8.0 Index of all headings

9.0 References

[Bordwell 1990] Bordwell, David and Kristin Thompson. *Film Art: An Introduction*. 3rd ed. New York: McGraw-Hill, 1990.

[Brand 1987] Brand, Stewart. *The Media Lab: Inventing the future at MIT*. New York: Viking, 1987.

[Bruckman 1991] Bruckman, Amy. *The Electronic Scrapbook: Towards an Intelligent Home-Video Editing System*. MIT MS Thesis, 1991.

[Bush 1988] Bush, Vannevar. "As We May Think." Reprint. ed. Howard Rheingold. *Hyperage*, February-March 1988.

[Cook 1990] Cook, David. *A History of Narrative Film*. 2nd ed. New York: W.W. Norton, 1990.

[Davenport 1987] Davenport, Glorianna. "New Orleans in Transition, 1983-1986: The Interactive Delivery of a Cinematic Case Study. Presentation to The International Congress for Design Planning and Theory. Boston, August 1987.

[Davenport 1989] Davenport, Glorianna and Hans Peter Brøndmo. "Creating and Viewing the Elastic Charles — a Hypermedia Journal." Presentation to Hypertext2. York, England. June 1989.

[Davis 1995] Davis, Marc. *Media Streams: Representing Video for Retrieval and Repurposing*. MIT Ph.D. Thesis, 1995.

[Eisenstein 1947] Eisenstein, Sergei. *The Film Sense*. Orlando: Harcourt Brace, 1947.

[Elliot 1993] Elliott, Edward. *Watch • Grab • Arrange • See: Thinking with Motion Images via Streams and Collages*. MIT MS Thesis, 1993.

[Evans 1994] Evans, Ryan. *LogBoy Meets FilterGirl: A Toolkit for Multivariant Movies*. MIT MS Thesis, 1994.

[Foley 1994] Foley, James and Andries van Dam, et al. *Introduction to Computer Graphics*. New York: Addison-Wesley, 1994.

[Halliday 1993] Halliday, Mark. *Digital Cinema: An Environment for Multi-threaded Stories*. MIT MS Thesis, 1993.

[Halliwell 1994] Halliwell, Leslie. *Halliwell's Film Guide 1994, 9th ed*. ed. John Walker New York: HarperPerennial, 1994.

[Houbart 1994] Houbart, Gilberte. *Viewpoints on Demand: Tailoring the Presentation of Opinions in Video*. MIT MS Thesis, 1994.

[Maeda 1994] Maeda, John. *The Reactive Square*. Tokyo: Digitalogue, 1994.

[Maeda 1994] Maeda, John. *Flying Letters*. Tokyo: Digitalogue, 1994.

[Maeda 1997] Maeda, John. "A Framework for Digital Expression." Published for the Digital Communication Design Forum. Tokyo: International Media Research Foundation, January, 1997.

[Maeda 1997] Maeda, John. *12 o'clocks*. Tokyo: Digitalogue, 1997.

[Maeda 1998] Maeda, John. *Tap, Type, Write*. Tokyo: Digitalogue, 1998.

[Murray 1997] Murray, Janet. *Hamlet on the Holodeck: The Future of Narrative in Cyberspace*. New York: Free Press, 1997.

[Murtaugh 1996] Murtaugh, Michael. *The Automatist Storytelling System: Putting the Editor's Knowledge in Software*. MIT MS Thesis, 1996.

[Negroponte 1979] Negroponte, Nicholas. *The Impact of Optical Videodiscs on Filmmaking*. Unpublished Paper. MIT, June 1979.

[Nielsen 1990] Nielsen, Jakob. *Hypertext & HyperMedia*. Boston: Academic Press, 1990.

[Pavic 1989] Pavic, Milorad. *Dictionary of the Khazars: A Lexicon Novel in 10,000 Words*. New York: Vintage, 1989.

[Richie 1984] Richie, Donald. *The Films of Akira Kurosawa*. Revised ed. Los Angeles: University of California Press, 1984.

[Small 1996] Small, David. *Navigating Large Bodies of Text*. Almaden, NY:*IBM Systems Journal*, 1996.

[Smith 1992] Smith, Thomas Aguierre. *If You Could See What I Mean... Descriptions of Video in an Anthropologist's Video Notebook*. MIT MS Thesis, 1992.

[Segall 1990] Segall, Ricki Goldman. *Learning Constellations: A Multimedia Ethnographic Research Environment Using Video Technology for Exploring Children's Thinking*. MIT Ph.D. Thesis, 1990.

[Tanenbaum 1992] Tanenbaum, Andrew. *Modern Operating Systems*. Upper Saddle River, NJ: Prentice Hall, 1992.

[Toland 1941] Toland, Gregg. "How I Broke the Rules in Citizen Kane." reprinted in *Focus on Citizen Kane*, ed. Ronald Gottesman. Englewood Cliffs, NJ: Prentice-Hall, 1971. orig. *Popular Photography*, June 1941.

[Tschichold 1995] Tschichold, Jan. *The New Typography*. Los Angeles: University of California Press, 1995.

[Winston 1993] Winston, Patrick. *Artificial Intelligence*. 3rd ed. New York: Addison-Wesley, 1993.

[Wong 1995] Wong, Yin-Yin. *Temporal Typography: Characterization of time-varying typographic forms*. MIT MSVS Thesis, 1995.

Figure 107. a VI (6) Frames of a Dead Guy Production