Video Reality™: Creating Navigable Feature Film Environments

John A. Toebes, VIII

Vice President, Research and Development

toebes@southpeak.com

SouthPeak Interactive

One Research Drive, Cary, NC 27513

Synopsis

Interactive video is considered somewhat of an oxymoron in the gaming industry. Much of the effort to date has been of the form of switching video streams based on user input. During this time, polygon-based games have been created to provide the user with a more interactive experience while sacrificing the quality and realism that the video environment provides. With this increase in realism, the collaboration necessary to produce these video based titles has grown.

The Video Reality™ technology is both a technology and a development system targeted at addressing the needs of both the game creators and the game players.

For the players, Video Reality delivers continuous freedom of movement with a 360 degree field of vision throughout the game environment. Some games limit players to multi-directional vision only at pre-defined spots, then move them from one of these spots to another without giving them the immediate opportunity to change their minds, back-track, or even choose what they are looking at while they go. Video Reality completely immerses the players in the game, letting them control where they want to go, when they want to go, and what they want to look at on the way there.

For the creators, Video Reality delivers a multi-user, highly automated tool to facilitate the *collaborative* process of planning, designing, assembling, optimizing and running immersive, video-based computer games. The system has been developed as a technology for creating a wide range of interactive titles and not just for a single title.

Origins

Two traditional gaming environments: Video Click-and-Play vs. Rendered-on-the-Fly

We created Video Reality by looking at both ends of the spectrum for presenting a gaming environment. On one end of the spectrum, we have what we call the "Video" style of *click the mouse and a video or show a picture* which has been used quite successfully in a number of titles because it offers the opportunity to provide a high quality image to the user. On the other end, we have what we call the "Reality" style -- virtual reality polygon based environments which are rendered on the fly to provide the greatest degree of freedom of movement.

	Click and Play ("Video")	**Render on the Fly ("Reality")**
Color Quality	High Quality (16-bit color and better or 8-bit optimized palettes)	Based on memory and machine performance (Typically 8 bit)
Detail	Rich detailed environments. Ray traced, 100,000+ polygons or real environment.	Limited by machine performance 10,000+ polygons
Navigation	Limited to predefined shots and locations	Unlimited (within constraints of the rendering engine)
Variability of Environment	Typically limited to display on pre-rendered backgrounds	High degree of variability as provided by rendering/animation engine.
Game Play	Typically puzzle solving/searching or twitch based action	Typically shoot-em-up action

While both of these have their advantages, there is an opportunity to provide the best of both environments. In fact, a constant demand from the render-on-the-fly crowd is more, faster, better polygon rendering to which there has been a response of hardware 3D accelerators. No matter what happens, the performance will be limited to how fast we can make the hardware. If we are to assume that it typically takes about one hour to render a single high-quality frame on a high-end workstation (short cuts can certainly cut this time down), and we were to assume that machines double in performance every year (Moore's law implies a doubling in performance every 18 months), then we should see the average machine able to render at 30 frames/second sometime around the year 2014[i]. ☺

Given the desire to provide the best of both worlds in the near future, we decided to attack the problem of making the video based environment more navigable. We rejected a number of techniques which involved simply putting the user on a path and allowing them to make 'twitch' choices. Instead we focused on techniques that allow a user to

essentially control a camera in an environment. The problems were then reduced to figuring out how to capture the environment and then allow the user to play it back. Along the way, we read many quotes from people that said that video can not be interactive.

Planning a technology and not a single title

We decided to focus first on making a technology and then on the titles using the technology. This was an easy ordering for us since we already had in mind quite a few potential titles. The approach of planning for a technology allowed for the freedom of sitting back and designing a complete system without having to worry about delivering a title yesterday. It allowed us the necessary time to smooth out the rough edges and to plan for a system that is capable of handling multiple titles efficiently instead of being tied to a single title.

Our work broke out into several important parts:

1) Architecting the development system. We assembled a team of seasoned system designers (and gamers) to lay out the overall system components. This first structure chart occupied an entire wall in the conference room and is still valid today. We looked at making the system flexible to handle a wide variety of titles while optimizing those things that were commonly done.

2) Planning video production methods. Because we are part of a video production facility that has been working for 15 years with video and film, we had the opportunity to plan the technology around the way that production people work instead of forcing them into a programmer mindset.

3) Testing the technology. This is certainly one of the most fun parts. Over the course of the first two years, we shot a number of sample productions and put them together with the toolset to learn what was necessary to be efficient as well as to iron out problems with shooting a Video Reality production. Of course, none of these productions were ever planned to be shipped to a customer.

Only when we were satisfied that the technology would produce a viable title did we actually start on our first production - *Temüjin™: The Capricorn Collection*™. This title was designed from the start to take advantage of the strengths of our technology, while at the same time offering the richness, characters, and gameplay that people have come to expect in a top-quality game.

It is worth noting that if we were only attempting to produce a single title (or a small series of titles), this approach is not a very cost effective one. Much of the effort in creating a reusable tool-set could have been more productively focused on the individual titles.

Introducing Video Reality

The result of this is a system that allows us to film (or render in high quality graphics) a complete and richly populated environment. The user can seamlessly navigate through this environment interacting with objects and characters as part of a well orchestrated experience. This resulting “spatial video” is a fundamentally new concept which we have invested more than a hundred person years in developing and fine tuning.

Targeting a minimum platform

Very early on in our development cycle, we looked at the minimum machine configuration necessary to deliver an acceptable experience and had initially settled on a DX4/100 with a hardware MPEG playback assist. However, after attempting to work with the MPEG hardware manufacturers with little success, we ended up targeting a Pentium® 90 class machine. While this seemed like abandoning part of the market, we felt that it was more important to preserve a minimum level of experience. In order to maximize our success in the marketplace, we decided to hold back the technology until this was an acceptable minimum platform.

Designing for collaboration

The most important part of our technology was to ensure that the titles can be created efficiently. We did not want to be held back waiting on a single person to edit a title, so we instead focused on a truly multi-user collaborative environment. By treating the production as a database, we were able to allow a number of people to work on the title at once. In fact, we have had a dozen people working on hotspots for the same production all at one time.

This collaboration goes beyond the tool-set. The process of creating a title involves taking the best that everyone has to offer. It is easy to spot the titles that were created primarily by an artist, or a programmer, or a film producer. It is when you have several of them working together to balance a game that the best titles are created. In doing this, we broke down the different types of people who participate in the process:

Person	**Responsibilities/Talents**
Writer	Creates story, dialog, characters, and settings
Game Designer	Creates logic and connective gameplay

Title Engineer	Creates high level logic and provides game play balance
Lead Programmer	Creates low level logic and 'flash'
Tools Developer	Automates inefficient game creation tasks
Graphic Artist	Creates artwork and backdrops
Director/Producer	Creates film/video environments that fit into the game
Audio Engineer	Creates sound and music for the game
Video Engineer	Provides balance and quality in the video environment
Navigator (Unique to Video Reality)	Ensures navigability of the captured environment
User Interface Designer	Creates GUI and user interface metaphors

By allowing each person to do their job within the environment, we were able to maximize the talents of all the people involved. For example, the User Interface Designer can quickly import bitmaps and try out the created controls without having to go through the often tedious process of creating separate control component bitmaps. The Audio Engineer can test out sound components within the runtime environment. The Title Engineer can focus on the important task of play balancing the title -- without having to worry about how a particular bitmap is going to be rendered on the screen. Even the writer is given an opportunity to test out the script in a playable mode.

While we went to great lengths to provide for collaboration, we did not want to recreate the wheel. There are many excellent tools out there for creating bitmaps and editing together video. For these types of operations, we chose to continue to use those tools in their native form, but to quickly incorporate their output into the Video Reality system.

Capabilities

The Basics (bitmaps, sound, music, mouse)

Because Video Reality was designed as a interactive title development system, it has all of the basic capabilities that one would expect. You can display and move bitmaps, play sound, play music, respond to mouse moves and clicks, save and restore productions, and even manage inventory.

Navigable Video

What makes the technology particularly unique is that it offers a method of creating highly navigable video. This means that the end user can move through the environment at will. They can choose to go somewhere, look around as they are going there, stop at any time, and even go back to where they came - all on their own terms. If you were to video tape a house to show off to someone, they would be forced to go through it in the order that you taped it. However, with Video Reality, they can choose to go through in any order looking where they want to.

Hotspots

If all someone could do was to walk through a scene, it could be pretty boring (although the pictures would be pretty). Clearly as someone is looking at something they would like to be able to interact with it. Recognizing that such an environment would be far richer, we developed a technology that allows us to manage a very large number of hot-spots across the entire video environment. For example, we have been able to shoot a greenhouse and put a hot-spot on every plant in the environment on every frame that it appears in - over 27,000 clickable areas - in just under three weeks. What the hotspots do is completely in the control of the Title Engineer.

Integrated spatial/dramatic scenes

Once you have clicked on a hotspot or performed some actions, there is often some sort of a dramatic event that may need to be carried out. However, instead of jumping to a cut scene as is typical, the dramatic video is seamlessly integrated into the user's navigation. While we often refer to these as dramatic scenes for production purposes, it is often difficult to separate them out from the truly navigable video.

Object integration

By itself, video may not offer enough variability in the environment. There are only so many replenishable items (stacks of paper, cups of pencils, an infinite supply of anything) that you can put into a title. Often there is a need to provide a unique item which the user can take with them. We have approached this in one of two ways. For central items which may have a dramatic effect on an environment, we can pre-create the environment both with and without the object. At runtime we can automatically select the appropriate clips. For other objects, we can dynamically adjust the video image to include a picture of the object anti-aliased into the image. The placement of these objects is done at edit time and can allow for a very high quality integration.

Of course, this allows our graphic artists to really stretch their talents as they create models of actual objects which were encountered on the set to match the environment exactly.

GUI design

Another important aspect to creating a gaming environment is to allow for a user interface which is consistent with the gaming environment. Typically this requires the creation of organic design elements which fit into the character of the experience. With this constraint, the typical drag a button out of the toolbox and slap it into the environment just doesn't cut it. However, when a graphic designer has to create custom buttons and button states for each spot that the user interacts with, it can quickly be overwhelming. There are many examples of titles where the final graphic was misplaced by a single pixel, marring what would have otherwise been a very beautiful effect. With our GUI Designer™ tool, the graphic artist can quickly import a series of bitmaps and then identify the active areas of the bitmaps. The system automatically handles cutting them out and even defining the basic logic.

Basic logic

This basic logic can range from something as simple as playing a sound to showing a picture. In fact, it is extremely common that the same action or series of actions applies to a wide range of elements in a title. To make this more efficient, the Video Reality system was designed with a simple flow language (with a syntax that is familiar to people who use Microsoft's Visual Basic) that has been extended with object oriented concepts. Hence a simple class can be created for a button, yet the actions can be overridden for a button which may need special handling. Because of this, while we originally expected our first title to have 3000-5000 of these tiny flows, we are finding that the reuse paradigm is so strong that we are currently anticipating fewer than 1000 of these 3-15 line subroutines to be written for the entire title.

ActiveX integration

We recognized that there are two types of programmers. Someone who is good at play balancing the logic of any title is not a likely candidate for the high performance graphic intense routines. Instead of having to find such a rare person, we can instead take advantage of the people who are good at one of these areas and have them work together.

Through the use of Microsoft's ActiveX technology, we have the ability to separately build (and test!) any of these graphically intense components and then integrate them into the production. The flow language can readily communicate with these components and provide the high level control of the applications. We were then able to have separate programmers write these custom components such as a page turner, newspaper reader, jigsaw puzzle, navigation map, and even a scorpion game. In fact, the jigsaw puzzle OCX turned out to be so much fun that we decided to send it out as a separate product (at this writing, we at SouthPeak Interactive have released four Virtual Jigsaw™ products) and to incorporate it into our web page.

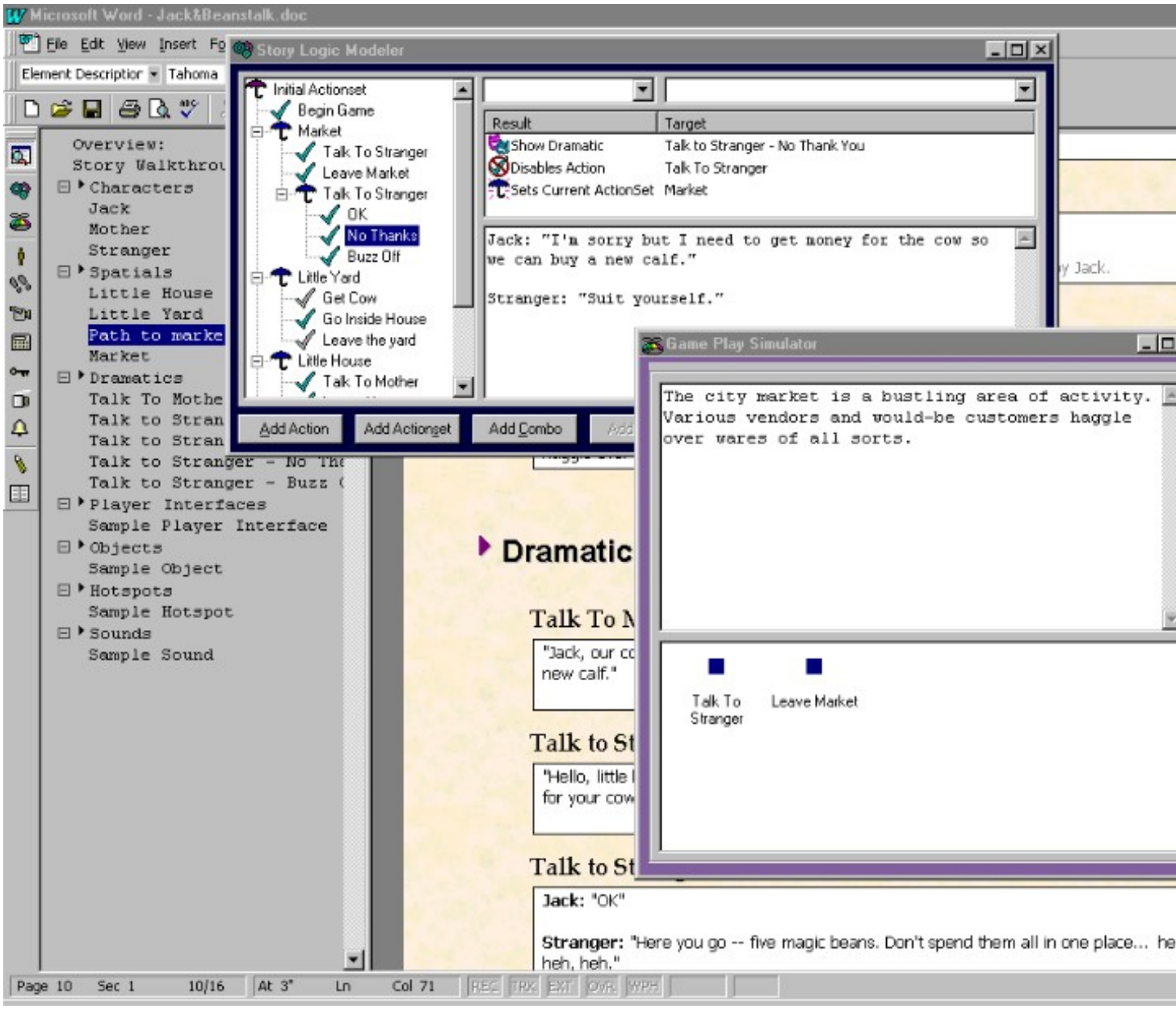
The Process

Having the capabilities in the system is not sufficient to bringing out a title. We also set in place a complete process for putting together a production. While every system has some sort of process to creating a title, what we have done is put a lot of effort into the planning and collaboration.

1. Planning the experience

All great titles start out with an idea. With StoryWriter™, we are able to put together a playable script for the title quickly while identifying assets and components. The output of this process directly feeds into the production database. Because we recognized that many writers prefer to work in Microsoft Word, we used the automation capabilities of Word97 to allow us to interactively define a script and run it right in that environment. This ability to test a script in development is very important early in the process to identify holes in the logic and to tighten up game play.

The StoryWriter system allows the game designer to identify all of the assets, characters, logic, conversations, environments, and custom components. In this way the script is as much a programming design document as it is a game play description.



As the script becomes more refined and detailed, the assets and basic components get more defined in the system. The title engineering team can take the refined script and add real assets to fill in the placeholders at any time. In this way, there is a smooth progression from a prototype to a finished product.

2. Designing the environment

During the later phases of the story development, the title engineering team also starts the task of creating the basic programming glue that holds the title together. Video Reality Studio provides them with a large number of basic classes and components to start from providing the basics behind button logic, navigation, some inventory management, and even video control. During this phase, the user interface is designed and polished using the toolset.

This is where the real game play is put in. While some tools have attempted to take the programmer out of the picture, we recognized that there are really two types of programmers that you want involved in the product. For all of the high level logic and glue that makes a game playable and balanced, the title engineering team uses our high level flow language and objects. However, we didn't forget that there is a need for a level of interactivity that goes beyond the basics. For these, we were able to take advantage of component technology to allow a C++ programmer to create the more interactive components.

The result of this is a playable title with very rough assets (yes, that blue watering can is supposed to represent the Amulet of Knowledge) and plenty of placeholders. We can mark these assets as placeholders instead of having to remember to replace them later on (or worse, shipping a product with a temporary asset).

3. Assembling the components

As the title is being laid out, the production crew is working rapidly to create all of the assets: video clips, bitmaps, sound effects, etc.. With any interactive product, there are tens of thousands of individual assets, sometimes with many iterations on the same asset until it is right. In fact, sometimes the new asset doesn't work out as well as the old one. For this, the system being designed around a database works very well. We track multiple candidates for a single asset and allow the title engineering team to choose which candidate will be used for the final product.

The multi-user nature of Video Reality Studio really helps in this phase because of the overwhelming number of assets that need to be handled. Because the system allows for multiple concurrent users working on the same production, the bottleneck of waiting for a single person to check in assets is reduced.

One important place where this bottleneck is completely eliminated is the addition of hotspots to this extremely rich environment. With the ability to have multiple people working on the same project, it is easy to assign the tasks of different areas to different people to achieve the value of the proverbial "Mongolian horde" approach to solving a task.

We have had as many as 20 people all working on the same production at one time defining hotspots and their actions. Most importantly here is that the Lead Title Engineer is freed from this monotonous task to be able to concentrate on the overall gameplay.

4. Optimizing the layout

While we designed a multi-user database for creating and maintaining a production, we knew that this would not be optimal for running a final production on the user's machine. Furthermore, we did not want the title engineering team to have to go through a series of tedious steps in creating a final CD only to find that something has been left out.

To solve this problem, we have a binder process which takes the database, assembles all of the assets (performing a final sanity check on all of them), converts them to their final representation, and lays them out for the CD. It puts everything under a single directory (or multiple directories for multiple disks) which includes all of the assets and executable components. All the engineering team has to do is burn the final CD straight from that directory.

The binder system also is responsible for optimizing the placement of the assets on the CD in order to ensure that the system runs more efficiently as well as trimming away any excess in the assets (extra frames, unused audio, unreachable logic).

5. Running the final product

What is probably the most impressive part that the user never sees is that the final CD has everything necessary to just run on the end user's system. Through our proprietary auto-load technology, the system dynamically adapts to whatever the user has installed on their machine - including incompatible versions of DLLs. If the user's machine is already in a running state, nothing is replaced. Instead the runtime system dynamically loads the appropriate versions from the CD for the duration of the game.

Once up and running, the Video Reality Engine goes through the task of allowing the user to seamlessly navigate through the environment. The engine manages access to the disk, dynamically pulling in assets and logic as they are needed - even taking advantage of times that the user is pausing to think.

An extra bonus that was designed into the system from the start is the ability to debug even the final production. Through the use of special side files (which are built into a separate directory), the title engineering team has the ability to view source code, set breakpoints, examine variables, and generally trace the flow of the environment. This system also allowed us to build in a complete testing environment so that all of the actions that a tester takes can be captured and played back at will. Since this capturing can be directed over the network to another machine, any crashes can be diagnosed and reproduced by simply playing back the captured file. This makes quick work of any of those intermittent crashes.

By combining this technology with our ActiveX integration, we have also built into the beta portion of the cycle the ability for our testing lab to simply press a key and not only enter the description of the problem, but have the system track where they were in the game. This has greatly reduced our turnaround for fixing problems reported by the testers.

The Results

In the end, it is not the technology or the amount of planning that went into the title that makes it appealing to the end user. They must find the experience to be enjoyable and challenging. With the Video Reality technology, we have created the opportunity to put them in real (or created in painstaking detail) environments. The game designer is offered an opportunity to concentrate on the gameplay aspects to provide a balanced game.

The end result is that the user gets the richness of a video environment, the rendered navigability of a virtual reality environment, and the gameplay that only a well balanced team can provide.

Where Do We Go From Here

Video Reality is a long way from being finished. The industry and platforms will continue to evolve. Around the corner is DVD which we feel is an excellent medium to deliver Video Reality titles on. The usage of MPEG-2 immediately quadruples the usable resolution of the environment with virtually no impact on the development process.

While it might not have been initially obvious, the inserted objects and even characters in the environment could be rendered by taking advantage of many of the advances in 3D hardware. Instead of dedicating the majority of polygons to the environment and a scant few to the characters and objects, we can create richly detailed and animated characters by applying thousands of polygons to them instead.

Because we are focused on the R&D aspects of the technology, we will continue to evolve Video Reality to provide richer and more interactive game experiences. Temujin: The Capricorn Collection is only the first of many titles to come.

[i] At 1 hour/frame, you end up with 1hour=60minutes=60*60seconds=60*60*30 frames=108,000 times slower than we need to be. If we double each year, we have 1998=2, 1999=4, 2000=8, … 2012=32768, 2013=65536, 2014=131072. Really applying Moore's law of every 18 months and using the more conservative figure of 4 hours per frame, the actual break even year is 2042, but we like to be optimists.