

# gamedeveloper

THE LEADING GAME INDUSTRY MAGAZINE

THE KINEMATIC INTEGRATOR

EQUATIONS! GRAPHS!  
IS THIS TRUE PHYSICS!?

THE HORROR!

SPOOKY CRUNCH TALES THAT WILL  
CHILL YOU TO THE BONE!



# TALES OF MONKEY ISLAND

POSTMORTEM





# 3Di is here.

[www.scaleform.com/3di](http://www.scaleform.com/3di)



© Copyright 2010 Scaleform Corporation, All Rights Reserved. Epic Games is a registered trademark of Epic Games, Inc. in the United States of America and elsewhere.

### POSTMORTEM

#### 20 TELLTALE GAMES' TALES OF MONKEY ISLAND

As a studio founded by LucasArts veterans, episodic gaming pioneer Telltale Games went back to its roots with *TALES OF MONKEY ISLAND*. Telling a story over five chapters allowed the team to fill out the game's narrative but presented unique challenges to the art and production pipeline.

*By Emily Morganti*

### FEATURES

#### 7 TALES FROM THE CRUNCH

The Horror! Crunch time is a fact of life in the game industry; everybody hates it but no one seems to know how to avoid it. Here we gather round the campfire to share some hilarious and horrifying crunch tales to chill your bones.

*By Brandon Sheffield*

#### 13 TRUE PHYSICS

One of the vital components of a physics engine is the code that integrates Newton's laws of motion. There are several techniques that employ approximations of calculus in order to numerically solve this problem. In this article Eric Brown proposes a scheme which can solve this problem without approximation.

*By Eric Brown*

#### 27 AN ISLAND OF DETAILS

Twenty years after its release *THE SECRET OF MONKEY ISLAND* continues to charm players. Here *MONKEY ISLAND* creator Ron Gilbert walks you through some of the design decisions behind its classic gameplay.

*By Ron Gilbert*

### DEPARTMENTS

2 **GAME PLAN** *By Brandon Sheffield*  
Adapt To Survive

[EDITORIAL]

4 **HEADS UP DISPLAY**

[NEWS]

The chipsounds soft synth, the 8th annual Scene.org Awards, and Ralph Baer joins the Inventors Hall of Fame.

31 **TOOL BOX** *By Jeffrey Fleming*

[REVIEW]

Game Developers Conference 2010 show floor report.

35 **THE INNER PRODUCT** *By David Tuft*

[PROGRAMMING]

Plane-Based Depth Bias For Percentage Closer Filtering

40 **DESIGNER'S NOTEBOOK** *By Jordan Mechner*

[DESIGN]

PRINCE OF PERSIA: May 3, 1987

41 **PIXEL PUSHER** *By Steve Theodore*

[ART]

Blind Alleys

44 **DESIGN OF THE TIMES** *By Damion Schubert*

[DESIGN]

Win Expectancy

47 **AURAL FIXATION** *By Jesse Harlin*

[SOUND]

The Global Asset List

48 **GOOD JOB!**

[CAREER]

Akira Yamaoka interview and new studios.

52 **EDUCATED PLAY**

[EDUCATION]

Joshua Nuernberger's BORYOKUDAN RUE

56 **ARRESTED DEVELOPMENT** *By Matthew Wasteland*

[HUMOR]

Phone Tag With The Art Contractor From Hell



# ADAPT TO SURVIVE

GAMES ADAPTED FROM OTHER MEDIA TEND TO BE SUB-PAR. THEY DON'T HAVE TO BE.

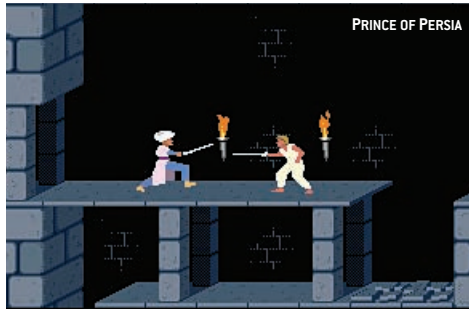
SOME TIME AGO I HAD THE OPPORTUNITY TO INTERVIEW FAMED MOVIE PRODUCER JERRY BRUCKHEIMER (*Beverly Hills Cop*, *Pirates of the Caribbean*, *CSI*) regarding the production of the *Prince of Persia* movie. The movie was put together in typical Bruckheimer style, with the aim more toward blockbuster spectacle and anonymous style than showing an auteur's touch in the direction or editing. He mentioned that he goes through several writers to get every aspect of the screenplay correct, and has each task on the production clearly delineated. It seems as though nothing is simply controlled by one person.

## DEVELOPERS ANONYMOUS

» The way Bruckheimer organizes his film productions is similar to what most game studios do nowadays, with each area of production receiving a lot of iteration, input, and polish. You could hardly point to the art style of a given triple-A title and say "Oh, that's a Steve Theodore game." Auteurism is left to the more independent ventures—you can certainly identify a Dan Paladin game, for instance (*CASTLE CRASHERS*, *ALIEN HOMINID*). Bruckheimer has also put together a game studio, with the help of some game execs, so you can expect that production style to be reinforced.

The thing that struck me about the interview was that someone of his level had the understanding that when you adapt a film to a game, you need to give it the time required to make it good.

"Here's the problem," Bruckheimer began. "To really make a good game, it really takes a long time. By the time you green light a movie, it's a year to a year-and-a-half until it's out. That's too short a period for a video game to be made. It's a three-year process to get a really good game made, and that's where they fail."



"What the studios do is have this business model where they know they'll sell X amount of games on that opening couple of weeks, and a lot of them do that, rather than take their time and create a wonderful game."

Then there are the constraints imposed by the film itself—does it follow the same narrative? What assets can you use? How much access do you have? I know one writer working on an adaptation of a film story into games, who was only allowed one look at the script—he could read it through in a room, one time, but couldn't have a copy, or even take notes or pictures.

To me all of this calls for games that are released based on the theme of the property, rather than some attempt to replicate an experience from another media. *BATMAN: ARKHAM ASYLUM* does this. People love Batman, and they don't care if it's based on a movie. Ditto *GHOSTBUSTERS*. Likewise *PRINCE OF PERSIA*. The story from that game is based on the ancient *One Thousand and One Nights* text. It's not a literal adaptation, but takes a compelling concept and makes it work in a game world. In the case of *THE CHRONICLES OF RIDDICK: ASSAULT ON DARK ATHENA*, I don't think audiences even care that much about the license—but the universe is compelling, and the game plays well, so people buy it.

## GIMME THE CASH

» Certainly studios will continue to capitalize on the release of their films—but perhaps Facebook, the iPhone, or the downloadable console space are better suited to that. Triple-A titles should be given space to breathe and become their own entities. If you want to make an *IRON MAN* game, base it on the comics, and release it at some point between two movies when people are hungry for content. Crossovers like the newly announced *MARVEL VS. CAPCOM 3* offer opportunities to do interesting things with licenses without catering to a certain timeframe.

Perhaps the takeaway is obvious: games need more development time to be good experiences. My hope is that hearing someone like Bruckheimer say it will convince some executives of its truth. Unfortunately, as long as the newest grindhouse-style game production of a popular cartoon continues to sell, people won't stop. But even Pixar is taking greater control of games based on its properties now, and companies like THQ are looking to get out of the direct film license business.

Licenses are compelling to a lot of people, but you have to treat them right and do them justice. That means giving them time, and I would submit not necessarily tying them to a specific release in another media.

—Brandon Sheffield

## SUBSCRIPTION SERVICES

### FOR INFORMATION, ORDER QUESTIONS, AND ADDRESS CHANGES

t: 800.250.2429 f: 847.763.9606  
e: [gamedeveloper@halldata.com](mailto:gamedeveloper@halldata.com)

## EDITORIAL

### PUBLISHER

Simon Carless | [scarless@gdmag.com](mailto:scarless@gdmag.com)

### EDITOR-IN-CHIEF

Brandon Sheffield | [bsheffield@gdmag.com](mailto:bsheffield@gdmag.com)

### PRODUCTION EDITOR

Jeffrey Fleming | [jffleming@gdmag.com](mailto:jffleming@gdmag.com)

### ART DIRECTOR

Joseph Mitch | [jmitch@gdmag.com](mailto:jmitch@gdmag.com)

### CONTRIBUTING EDITORS

Jesse Harlin  
Steve Theodore  
Daniel Nelson  
Soren Johnson  
Damion Schubert

### ADVISORY BOARD

Hal Barwood Designer-at-Large  
Mick West Independent  
Brad Bulkley Nevsoft  
Clinton Keith Independent  
Bijan Forutanpour Sony Online Entertainment  
Mark DeLoura Independent  
Carey Chico Pandemic Studios

## ADVERTISING SALES

### GLOBAL SALES DIRECTOR

Aaron Murawski | [amurawski@think-services.com](mailto:amurawski@think-services.com)  
t: 415.947.6227

### MEDIA ACCOUNT MANAGER

John Malik Watson | [jmwatson@think-services.com](mailto:jmwatson@think-services.com)  
t: 415.947.6224

### GLOBAL ACCOUNT MANAGER, RECRUITMENT

Gina Gross | [ggross@think-services.com](mailto:ggross@think-services.com)  
t: 415.947.6241

### GLOBAL ACCOUNT MANAGER, EDUCATION

Rafael Vallin | [rvallin@think-services.com](mailto:rvallin@think-services.com)  
t: 415.947.6223

## ADVERTISING PRODUCTION

### PRODUCTION MANAGER

Pete C. Scibilia | [peter.scibilia@ubm.com](mailto:peter.scibilia@ubm.com)  
t: 516-562-5134

## REPRINTS

### WRIGHT'S REPRINTS

Ryan Pratt | [rpratt@wrightsreprints.com](mailto:rpratt@wrightsreprints.com)  
t: 877.652.5295

## THINK SERVICES

CEO UBM THINK SERVICES Philip Chapnick  
GROUP DIRECTOR Kathy Schoback  
CREATIVE DIRECTOR Cliff Scorso  
CHIEF INFORMATION OFFICER Anthony Adams

## AUDIENCE DEVELOPMENT

### TYSON ASSOCIATES Elaine Tyson

e: [tysonassoc@aol.com](mailto:tysonassoc@aol.com)  
LIST RENTAL Merit Direct LLC t: 914.368.1000

## MARKETING

### MARKETING SPECIALIST Mellisa Andrade

e: [mandrade@think-services.com](mailto:mandrade@think-services.com)

## UBM TECHNOLOGY MANAGEMENT

CHIEF EXECUTIVE OFFICER David Levin  
CHIEF OPERATING OFFICER Scott Mozarsky  
CHIEF FINANCIAL OFFICER David Wein  
CORPORATE SENIOR VP SALES Anne Marie Miller  
SENIOR VP, STRATEGIC DEV. AND BUSINESS ADMIN. Pat Nohilly  
SENIOR VP, MANUFACTURING Marie Myers





# I WANT YOU



# TO CREATE GAMES



WWW.WARIOWAREDIY.COM

## YOU CAN WIN THE WARIO™ AWARD!

The Wario™ Award celebrates your creativity. Create your own games, compete with others and have the chance to win a trip to the exclusive Nintendo E3 Media Briefing in L.A. You can enter by creating and submitting a microgame with the new WarioWare™: D.I.Y. game, or by submitting a microgame design concept via the website. Challenge your creativity and learn more at [www.WarioWareDIY.com](http://www.WarioWareDIY.com)



PLAY BIGGER!  
Nintendo DSi XL



## MAKE IT YOURSELF! PLAY IT YOURSELF! DO IT YOURSELF!

NINTENDO DS™



Comic Mischief  
Mild Cartoon Violence

NO PURCHASE NECESSARY. PURCHASE WILL NOT INCREASE YOUR CHANCES OF WINNING. Open to legal residents of the 50 U.S., D.C. and Canada (except Quebec). Must be legally able to travel to Los Angeles, CA. VOID WHERE PROHIBITED. Entry deadline: 5/16/10. Grand Prize includes airfare, two-night hotel stay and admission to 2010 Nintendo Media Briefing at E3 on June 15, 2010, for two persons (ARV: U.S. \$2,500). Odds of winning depend on the number and quality of entries received. Only one entry per person is permitted. Restrictions apply. For complete details (including judging criteria and how to submit a microgame design instead of a microgame to enter), see Official Rules at [www.WarioWareDIY.com/TheWarios](http://www.WarioWareDIY.com/TheWarios). Sponsor: Nintendo of America Inc., 4820 150th Ave NE, Redmond, WA 98052.





# infinite bleep

PLOGUE'S CHIPSOUNDS SOFT SYNTH RESURECTS CHIP MUSIC

## MUSICIANS WHO YEARN FOR

the electric buzz of genuine chip music but don't want to compose in machine code on antique hardware have a new creative option with Plogue Art et Technologie's chipsounds software synthesizer ([www.plogue.com](http://www.plogue.com)). The result of four years of research, chipsounds aims to accurately reproduce the unique sonic character of classic console and computer audio within a musician-friendly interface that runs as a standalone application or as a VST/AU/RTAS compatible plug-in.

The chipsounds software emulates a number of venerable chips including the TIA from Atari's 2600 and 7800 console, the Ricoh 2A03 as used in the Nintendo Entertainment System and its custom variant that

**JEFFREY FLEMING:** *Why do think chip music has such an enduring appeal?*

**David Viens:** Well that is one of the most difficult questions to answer considering how personal it is. Some people feel a nostalgic attachment to the sounds and minimal arrangements, but it's important to realize that many current chip music fans were kids in the N64 and PS1 era. For those people, these sounds are new and raw, and are often compared to the new punk. I personally love old school tracks but also believe in the modern chip music incarnation in which chip sounds, textures and tricks are treated as another palette of sounds to be mixed into other styles of modern music.

*he described the chip as having analog warmth that was hard to reproduce. What was your approach to the recreating the SID sound?*

**DV:** Well it is part analog. While the bare waveform generation and envelope calculations are derived from digital parts of the SID circuit, their multiplication is done in analog land, and so is the filter. So it really is a hybrid beast. Not to mention that no two chips sound exactly the same, as proven by many research teams, including us. It's still an ongoing research for me, and for others in this field.

Bob Yannes, the creator of the chip has given plenty of clues on how the digital part works, but it's the subtleties of the analog side that still give people like us headaches. There is really nothing like the SID anywhere, past or present.

**JF:** *What were some of the techniques you used to analyze and reverse engineer the original chips in creating your emulation?*

**DV:** First, because a console is noisy, we extracted each chip from its source environment on to separate breadboards so we could capture test waveforms with the lowest possible level of noise, and with as little filtering as possible. We recorded everything using pro audio cards at a 192Khz/24bit-sampling rate, and then cross-checked everything with a digital oscilloscope for things like DC behavior, which is notoriously hard to monitor with AC coupled sound cards.

The second step was to be able to feed commands to each and every chip and to make lists of such commands to generate all possible waveforms at all possible pitches and all possible amplitudes. From there we were able to take that raw data and make a software model for the generation.

The last step was feeding the model with CPU write dumps of known

games to compare the sound against the real thing. This last part was the most demanding, especially in the case of chip exploits, which have been used and abused in the demo scene.

**JF:** *Have you considered creating a Plogue chipsounds integration for some of the game audio engines such as Wwise or Fmod?*

**DV:** Knowing that our stuff is both portable and modular, I'm just waiting for a call! Actually, if I understand correctly, the issue with contemporary game engines is that there is not much CPU headroom for audio code, whereas in pro audio land, everything goes to the DSP, and then if there are a few cycles left we will refresh the UI. So a different approach would need to be taken in the configuration of our synthesizer scheduling and rendering.

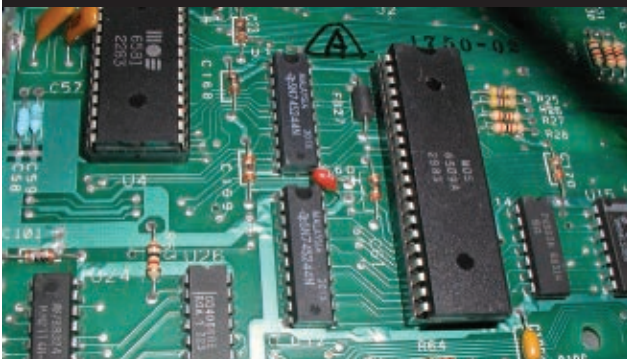
**JF:** *Chipsound files can be edited by users to create new emulations. How does that work?*

**DV:** Our engine, ARIA, is based on the open SFZ text format, so all sound generators and samples can be reconfigured. So you could write a SFZ setup file that would emulate similar chips that I've not yet analyzed myself given that the input clock to the chip, its internal dividers, and exact bit patterns that produce the waveforms are known. Since most chips are already defined this way in plain text in chipsounds, they can serve as template for more.

**JF:** *Do you have plans to include other chip simulations in future versions of chipsounds?*

**DV:** I'm still knee deep in research for future versions (see <http://ploguechipsounds.blogspot.com>). There is just so much stuff that needs unearthing, sound gems and long lost sound tricks that still need to be made easily available to the contemporary musician.

—Jeffrey Fleming



powered the original Game Boy, the POKEY chip from Atari's 400/800 series of computers, and the SID chip behind the Commodore 64's instantly recognizable sound. The software also delves into esoterica with a variety of additional chips rescued from the technological tar pits including emulations of the Odyssey 2's P824X chip, the D18676 from Casio's VL-1 mini keyboard, and the AY-3-8910 that provided sound for the Intellivision console and Atari ST computer.

We spoke with Plogue's David Viens to find out more about this ongoing project.

**JF:** *To what extent are the sounds in Plogue chipsounds being synthesized versus sampled?*

**DV:** In version 1.0 of chipsounds it is roughly a 90 percent synthesis to 10 percent sampling ratio, but we are targeting 99 percent synthesis in the next iterations, leaving the really un-emulatable stuff in sample form like the digital noise interference from neighboring chips that bleeds into the analog signal.

**JF:** *A number of projects have attempted to emulate the unique SID chip sound. I spoke to game composer Charles Deenen and*



# scene.org awards

## ON APRIL 3RD, A LARGE GROUP

of demosceners gathered in a small sports hall in Germany. Celebrating the best of the 2009 demoscene, the legendary Breakpoint demoparty witnessed the prize-giving ceremony of the annual Scene.org Awards. Many more demosceners around the globe huddled over their computers at home to watch the event online via a live video stream.

Now in their eighth year, the Scene.org Awards aim to recognize and reward the best the demoscene has to offer. If demos were films then these awards would be the Oscars—complete with a ceremony fronted by demoscene celebrities and a raft of emotional acceptance speeches. Each year a panel of expert judges takes a look back over the demoscene releases for the previous 12 months and draws up a short list of what they consider to be the best of the best. The nominations are made in categories of Best Demo, Best Demo on an Oldschool Platform, Best 64k, Best 4k, Best Animation, Best Effects, Best Direction, Best

Soundtrack, Best Graphics, Most Original Concept, Best Technical Achievement and Breakthrough Performance. There is also a Public Choice category that is voted for by the international demoscene at large.

Most of these categories will sound familiar but let me explain the others: The Best Demo on an Oldschool Platform category is intended to honor the roots of the demoscene by looking only at those released on platforms that were commercially available before the mid-90s—this includes platforms such as the Commodore 64 and the ZX Spectrum. The 64k and 4k categories recognize small demos (often referred to as “intros” rather than demos) that are written to incredibly tight size limits—64k intros must fit all of their code and data into just 65,536 bytes, while the 4k category gives you only 4,096 bytes to play with. Whilst many of the more impressive looking nominations at these awards consume tens of megabytes of space to store their code and data, for many people these tiny intros are still what the demoscene is all about.

As expected, the quality of the nominations this year was high. The winning demos are stunning examples of what can be done with a computer (even twenty five year old ones!) if you have the talent and the inclination.

The best way to experience any good demo is by running it in a darkened room with your speakers turned up loud. Back in the old days of DOS and proprietary graphics cards, running demos was something of a black art.

These days, thanks mainly to DirectX and OpenGL, things are much easier—though you will still need some serious hardware to fully appreciate some of these demos. It’s lucky for the rest of us mortals that high quality captures of most modern demos usually make it onto YouTube or one of the dedicated demoscene online video sites (such as capped.tv or demoscene.tv) very quickly. This makes it easy for anyone with a passing interest in the scene to dip in and see what the fuss is all about.

There are many ways to catch up with the winners and runners-up of the Scene.org Awards, but probably the best place to start is at Scene.org’s < <http://awards.scene.org/awards.php> > own site. You can also easily find links to this and all previous years’ awards nominees at <http://pouet.net/sceneorg.php>. My personal viewing recommendations would go to *Frameranger* and *Rupture*, two incredible examples of large format demos, and also to *Elevated* and *Puls*, two great lessons to all of us in what can be achieved if you only have 4k (or just 256 bytes in the latter case) to play with.

—Paul Grenfell

Here is a rundown of the Scene.org Award winners:

**BEST DEMO:** *Rupture* by Andromeda Software Development

**BEST DEMO ON AN OLDSCHOOL PLATFORM:** *Andropolis* by Instinct & Booze Design

**BEST 64K INTRO:** *Approximate* by Ephemera

**BEST 4K INTRO:** *Elevated* by Rgba & TBC

**Rupture.**



**BEST ANIMATION:** *The Death Grind* by Endre Barath

**BEST EFFECTS:** *Frameranger* by CNCd, Fairlight & Orange

**BEST DIRECTION:** *Rupture* by Andromeda Software Development

**BEST SOUNDTRACK:** *Blunderbuss* by Fairlight

**BEST GRAPHICS:** *Frameranger* by CNCd, Fairlight & Orange

**MOST ORIGINAL CONCEPT:** *Ballad of a Cluster Bomb - Director's Cut* by Kooma

**BEST TECHNICAL ACHIEVEMENT:** *Puls* by Rrrola

**BREAKTHROUGH PERFORMANCE:** *Extatique* by Adinpsz

**PUBLIC CHOICE:** *Rupture* by Andromeda Software Development

## CORRECTION

The March 2010 Toolbox review of Audiokinetic's Wwise 2009.3 contained a layout error. The correct Pros and Cons for the tool should be:

### PROS

- 1 Fully integrated audio pipeline solution from low level engine to toolset integration.
- 2 Robust toolset features, functionality, and prototyping capability.
- 3 Elegantly solved multi-platform authoring.

### CONS

- 1 UI is deep, complex, excessively gray, and can be confusing during extended authoring sessions.
- 2 Can't copy/paste Real-Time Parameter Control (RTPC) curves or RTPC values between containers.
- 3 Blend container workflow can be clunky.



## ralph baer joins inventors hall of fame

### THE NATIONAL INVENTORS HALL OF FAME

has inducted Ralph Baer, the video game console pioneer who in the 1960s and 70s led development of the technology that would become the Magnavox Odyssey. He's credited with ushering in a new period of home television entertainment.

"People thought I was wasting my time and the company's money for that matter," Baer said. "There's no way anybody could have predicted how fast this industry would take off."

This year, the Akron, Ohio-based Hall of Fame is recognizing him along with 15 other industrious figures whose inventions include GPS, electrothermal hydrazine thrusters for space propulsion, glass ceramics and Post-It notes.

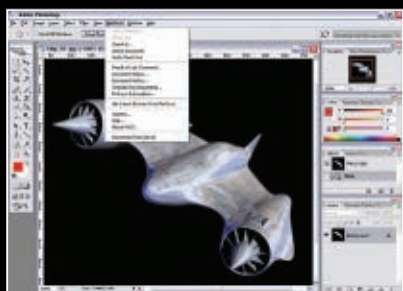
Baer's inventions also include the music and color-based electronic memory game Simon, the interactive stuffed bear TV Teddy, and the "Chat-Mat," which plays a recorded message when stepped on. He was also directly involved with the development of the first light-gun video games.

Baer was born in 1922 and is still running a consulting business.

—Kris Graft



## Introducing P4GT, a productivity feature of Perforce SCM.



P4GT

The Perforce Plug-in for Graphical Tools, P4GT, makes version control painless by seamlessly integrating Perforce with leading graphical tools. Drop-down menus allow access to Perforce from within 3ds Max, Maya, Softimage XSI, and Adobe Photoshop.

Art and development teams can standardize on Perforce to version and manage both source code and digital assets. Enhanced collaboration during the design process helps teams to work together in real time to release small patches or create whole new worlds.

P4GT is just one of the many productivity tools that comes with the Perforce SCM System.

**PERFORCE**  
SOFTWARE

**Download a free copy of Perforce, no questions asked, from [www.perforce.com](http://www.perforce.com). Free technical support is available throughout your evaluation.**



STORIES OF OVERTIME SO CHILLING,  
YOU'LL SWEAR THEY WERE  
ACTUALLY HAPPENING TO YOU!

# TALES FROM THE CRUNCH

**W**elcome children, to my chamber of truths! "Crunch." Does this word scare you? It tends to have negative connotations, but it's not always the worst thing that can happen to a project. Quite often it is! But not always.

In all creative media, you tend to have a period during which everyone is pulling together and going that extra mile to make the project great. Planned crunch can be vital to getting a game polished, or getting in that crucial feature that a small group of developers really believe in. But crunch shouldn't be necessary just to get a game shipped at a basic level of quality, and it shouldn't go without pay.

It's all about how it's used, how it's planned, and how you're paid. Here, we present anonymous stories from the trenches of development—tales of woe and wonder both, outlining the lives of developers living for (and at) work. Like wayward soldiers, these people band together in the face of insurmountable odds and ship a game. Most of these stories are tales of what not to do. But a little creative crunch is sometimes a good thing—so long as you can plan for it!

The sentiment is summed up well by a submission from a developer who calls himself Kawika. "Our crunches were tough but fun," he says. "We had great group dinners, group movies on Friday night, and group gaming sessions. You can have as much fun as you want, but the lack of sleep and downtime takes its toll. I've seen many nervous breakdowns of different types at every company."

So gather round, dear readers, as we weave yarns of terror and titillation. And don't blink—because the next crunch ... could be yours!

—Brandon Sheffield

ILLUSTRATIONS BY VINCENT PEREA

# CRUNCH





## LET SLEEPING DEVS LIE

» In a now-defunct European studio, a programmer's slumber was interrupted by one of the studio's owners. The night before, a meeting room had been designated a sleeping area for the crunching development team. Most of the team members had laid their heads to rest anywhere from 5 to 7 a.m. that same morning, and were hoping to catch whatever Zs they could.

The owner, however, needed the room to woo visiting investors, and was embarrassed to walk in on his sleeping employees, camped out on various levels of bedding. Later on, he let the developers know what an awkward situation he had been put in, and that they shouldn't let it happen again. At this, the programmer spoke up.

"You don't really seem to appreciate that we've been working all night for you."

The owner quelled this insubordination swiftly. "Well YOU don't seem to appreciate that we brought in mattresses!"

## SNAKES ON A CODEBASE

» I had a coder who I needed to stay late, but he refused. Very weirdly. Refused to say why, just kept saying he couldn't stay. So finally I asked, "Why can't you stay late? We really need your feature in..."

He answered, "The pet store closes at six, and I need to buy a rat for my snake."

So I said, "Oh, no problem. I'll run over when I'm picking up dinner!" But once inside the store, I was like, "OMFG, that's sorta gross." I went and bought a rat that was condemned to die.

They put it in a Chinese food container and (of course) I was picking up Chinese food for dinner at Hunan Garden! Naturally, all sorts of fun ideas and prank scenarios ran through my head, but in the end, I just put the rat bag next to the food bags when I got back and told everyone, "No one touch the bag that's moving. It's XXX's." He was happy, the feature went in, his snake ate, and I added a new item to the list of crazy shit I'd do during crunch time to keep everyone happy and productive!

## JUST ONE MORE YEAR!

CRUNCH DURATION: THREE YEARS

» We started working on our project in late 1999. We announced the project in September of 2001. Our first crunch was in preparation for E3 2002. This was a nominal crunch, one month in duration with 12-hour work days. The project lead came up with the idea that if we crunched early, on and off, we could avoid the nasty crunch in the end.

We started with month on, month off of two days per week of crunch, which lasted for one year. Most of us only worked until 10 p.m. or so, and everyone was entitled to a dinner break from 7 to 8 p.m. When we announced the game in 2001, we thought we only had one year of development left, so the thought of doing a mild crunch a year from finishing wasn't that big of a deal. After the first year came and went, we realized that we still had a year of development left (or so we thought at the time). The next year we continued the month on, month off crunch, but progressed to every day of the week. I remember my girlfriend (soon-to-be-spouse) commenting that at least we didn't have to work weekends!

As the second year of crunch came and went, we realized we still had a year of development left. At this time, they moved the crunch to every day until launch—still only to 10 p.m. In March of 2004, management announced to the dev team that a ship date had been set for November, and that we would have to start working 15-hour days. A group of artists met one night and decided that they'd had enough crunching and that they would rather the game slip into 2005 rather than crunch. That group of artists all went home at 6 p.m. that night

after putting in eight hours of work. The next day, they got chewed out by management. The artists held their ground and management went to the execs, who within one week put together a ship bonus if everyone crunched. In order to receive the bonus, the game had to hit a certain number of unit sales within a set amount of time.

The hardest part about the crunch was watching the other development team come in late and go home early. Those guys took long lunches to see new movie releases and the majority of their day was spent "alpha testing" our game (their project still hasn't shipped and is seven years in the making). Management's solution to our low morale was to have an executive sit at a computer station in the hallway each night. One night, it was the president; another night, the CFO. They were usually just playing the game, but showed their support by crunching one night a month with us. This was actually more harmful than beneficial, but we didn't care because at least we had our ship bonus.

Even though this was the hardest ship I have ever endured, it was a fun time. Dinner was provided every night and the bonds that were formed during those long hours will never be forgotten. The only repercussions suffered by the company were having to settle a class action lawsuit akin to the famous "EA Spouse" case, and losing a large chunk of their core development team within a year of launch, which pushed the first expansion out an additional year.

## IN SICKNESS AND IN HEALTH

» As studios go, the one I work for is fairly decent. But there was one year in which two mandatory overtime crunches ran for over a month, one of which involved 10 hours a day, seven days a week "or you lose your job" crunch. This meant between 28 and 40 days without a day off, depending on which team you were on. It was during this time that the flu hit, and I had to take time off. Because of the stress of the crunches, I had used up all my sick days. So after some 120+ hours of unpaid overtime, the company docked me three days of pay for going over the allotted amount of sick time. Outrageous!

## SPACE MADNESS

CRUNCH DURATION: TWO WEEKS

» It wasn't the longest crunch on record by any means, but we were a small start-up and it was our launch title. All our chips were on the table, and we had already pulled several all-nighters to bring the project to gold master.

The founder and CEO was in final code review with the rest of us. It was his 20th hour in the office, and he was looking for a particular function that he had just scrolled past on-screen. Rather than use a search function, he scrolled back to look for it. However, the function had "mysteriously disappeared."

The rest of us rushed into his office after we heard his screams and the sound of the keyboard repeatedly being slammed into the desk. "I've lost the f\*\*\*ing function!!!" he shouted, flailing wildly.

Our producer urged him to calm down, sat him back down in front of his machine, and asked him to walk through what happened. Our CEO pointed at the screen and said he couldn't find the function.

The producer looked at the screen. The function was right there in the middle. "It's right there, Jim..." he said as he pointed to the screen.

"Where?!" screamed Jim.

"Right there, Jim. Right in the middle of the screen."

"I don't see it! Dammit, I don't see it!"

The producer pointed again. "Follow my finger... it's right there."

It took our CEO a good 10 seconds, but he finally saw the line with the function next to the producer's finger. Even with it being in the

# CRUNCH

middle of his screen the whole time, and the project producer pointing to it, the function just wouldn't resolve in his field of view.

"You should go home, Jim."

"I should go home!!"

Our CEO stood up, apologized, told everyone he was going to leave, and urged us to do the same. The next day, over a pot of coffee, he blessed us with this pearl of wisdom: "You know you've been in the office too long when you can't see five inches in front of your face anymore."

## THE DISAPPEARED

» At our studio we were crunching on a real-time strategy game. The managers wanted to speed up production and control it at the same time, so their solution was to have us fill out progress reports every night and have art meetings every morning (going over all new generated assets), then sometimes again at midday. We'd then have our work checked off by the art director before we left at night. On top of that we would have random "all-company meetings" that took hours. During one of these long company meetings, our producer began the meeting by blurting out angrily, "We are going to keep having meetings until we figure out why there isn't enough daily progress!" But of course we felt this was exactly the reason progress had slowed: too many meetings! We barely got to spend time in our offices long enough to complete anything significant; we only made adjustments to existing assets so we had something to show.

I have seen star employees fired for taking their Christmas break instead of canceling travel plans to see family. A cinematic coordinator who came to the studio from the film industry tried to start a union and basically disappeared from work without a trace. One day he was there and everything was fine, the next he was gone with no "goodbye" or email or knowledge from anyone about what had happened to him.

## THE PERILS OF SUCCESS

CRUNCH DURATION: OVER TWO YEARS

» Hot on the tails of a game, which came to be labeled by *Next Generation* magazine as "the most reviled game of all time," our crew got cracking on a new project with a new publisher. Like any small developer looking not only to prove itself but also to simply stay afloat, we 12 members of the team did everything we could to keep things pleasant with our publisher.

At first, we made a big push to get a bunch of new technology together. We were riding the bleeding edge of technology at that time in some ways, with massive streaming worlds and deformed skeletal meshes. It was a lot of work, but of course simply having the technology wasn't enough.

After a year or so of work on the our new title, we were perpetually convinced that what we were doing wasn't good enough for the publisher and started operating in a mode driven in part by fear of getting our project canceled and in part by the pride we had in what we were doing. We wanted to make it even better.

The office layout was such that outside of three managers, the rest of us were in one big room without walls and without cubicles. We could look around and see what others on the team were working on, and many of us were jazzed and motivated to do even better because of what we saw other people doing.

That last year was a pretty solid crunch. Most weeks were seven days of work and it was haphazard in many ways, but we were driven to ship a great game, and that is what we did. After so many deliveries of take out food, we got to know the drivers pretty well.

Once our game was out, the publisher wanted a sequel quickly, and we figured since we'd just made a game, how hard could it be to make

the sequel? Pretty fast though, the feature list started growing. Beyond the scripting system, a multiplayer mode, and new rendering modes for special effects, we were also going to ship more levels that were bigger and would render far more cinematics. The original game shipped on one CD, but the sequel was barely going to fit on two. Not to mention that nearly everything had to be better—more polygons and more shine everywhere we could.

This one was the hard crunch. We had about eight months of development because they wanted the sequel within a year. We staffed up to 15 or maybe even 18. We overflowed the large room that we had. There were folks working in the conference room and a couple of people took over the reception area. Many nights, the whole team pushed through the night into daytime hours, and kept going until it was night again. I remember realizing in one specific instance that I'd worked thirty consecutive days. Where did that time go? It was easy to lose track of the passing days, and it was tough to fit in time to do laundry, but at the end of it all, we finished the project and it became another huge hit.

## "WE'RE IN THE FINAL STRETCH!"

» I work at a well-known developer in the UK which has been in constant crunch for the last several years. A small group of us began documenting our crunch mandate emails, in the event we should need them in the future, and our first request was that we work through the 2008 Easter weekend. It seemed innocuous enough at the time, and a genuine request. But the emails kept coming. We noticed that our schedule never seemed to gel with the reality of what we had to accomplish, and we were always missing milestones in spite of near-constant crunch. To us, this seemed like a management problem, and not so much an issue of us not working hard enough. Still, the emails kept coming, each one promising that the light was at the end of the tunnel—soon we would stop crunching! And so said the next email, and the next, and the next.

The "final crunch," in which we were (for the umpteenth time) to bang out a build for the Wii submission date, lasted until the point where we were not allowed to leave the office without checking with someone. We also had to maintain phone contact in the event that someone needed to ask questions of us. No one believes the emails that tell us crunch is almost over anymore. We always miss the last possible submission date for the Wii on every project, yet it comes out on time due to very skilled workers fueled by coffee and full fat cans of Coke.

It's slowly getting better here, though. We've had a number of good producers that have come in, attempt to share some sort of scheduling and production skill, and then get fired, or resign out of frustration. At the very least, we do get paid for overtime!

## RED LIGHT MEANS GO

» After a big crunch, a big milestone was in. Now it was time for a break ... a quick weekend ferry trip to Amsterdam! Sadly, the U.S.-based producer didn't see the need for the U.K. team to take a break from crunch, and he warned the local producer in no uncertain terms, "You do not take that team to Amsterdam." They'd been crunching for months, and he wanted them to keep grinding until they were done. The team had other ideas, and thanks to one volunteer who had to stay behind for some family function (parents' anniversary dinner or something), everything worked out.

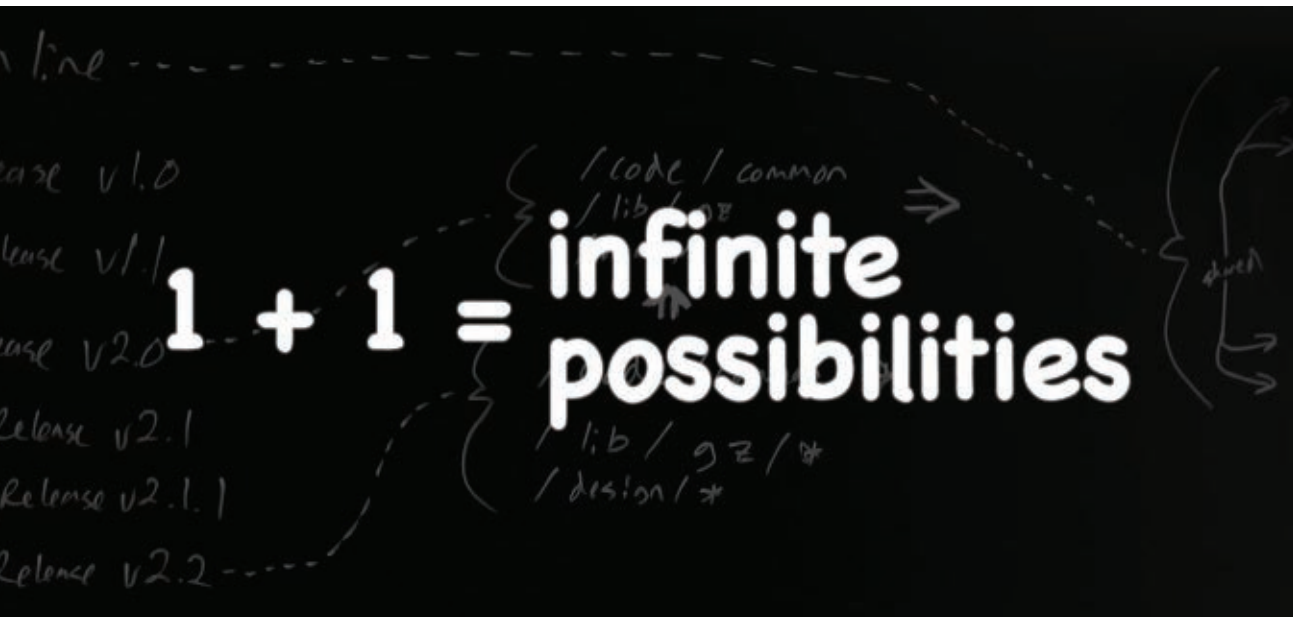
While the team relaxed and recharged in Holland, the volunteer stayed at the office, went around to everyone's machines and sent pre-written emails and IMs, and uploaded broken (and then fixed) builds that had been carefully stockpiled in advance of the trip. The U.S. producer was none the wiser, other than being impressed by how hard the team crunched in the following weeks! 🍌



# TALES FROM THE CRUNCH







© 2009 Seapine Software, Inc. All rights reserved.

## Get More Change Management with Seapine Integrated SCCM

**TestTrack Pro + Surround SCM = infinite SCCM possibilities.** Seapine's integrated software change and configuration management (SCCM) tools do much more than competing tools, and at a much lower price point. Start with TestTrack Pro for change management and add Surround SCM for configuration management—two award-winning tools that together give you the best integrated SCCM solution on the market.

- Link issues, change requests, and other work items with source code changes.
- Manage simple or complex change processes with flexible branching and labeling.
- Coordinate distributed development with RSS feeds, email conversation tracking, caching proxy servers, change notifications, 3-way diff/merge, and other collaboration features.
- Enforce and automate processes with incredibly flexible work item and file-level workflows.

Built on industry-standard RDBMSs, Seapine's SCCM tools are more scalable, give you more workflow options, and provide more security and traceability than competing solutions.

Get more, do more with Seapine tools. Visit [www.seapine.com/gamescm](http://www.seapine.com/gamescm).



# TRUE PHYSICS

USING A KINEMATIC INTEGRATOR  
TO TAKE YOUR PHYSICS TO THE  
NEXT LEVEL

**IF YOU ARE INVOLVED IN MAKING PHYSICS, YOU DO IT** either because it is your job, your interest, or your fantasy. Game physics is hard, and that's part of the reason why it's so fulfilling when you get it right. One rather tricky part of getting it right is the integration of the dynamical equations of motion.

On the surface, it would seem that integration shouldn't be too hard—it usually only occupies a few lines of code. However, the way you integrate the dynamic state of your objects seems to impact just about everything else you do, such as when to check for collisions, how to resolve collisions and constraints, how to apply forces to your objects and so forth.

It is usually very obvious when the integration isn't working; stuff is jittering, popping, or exploding. When it is working, usually, the only person who notices is the person who soothed and tamed the wild physics beast into submission—no one else cares, since everything is just acting the way they would expect it to. If you've ever had to debug an integration-related problem, you know exactly what I mean.

## EXISTING TECHNIQUES

» When it comes to integration techniques, there are several options to choose from. Probably the most universally used and reviled method is the explicit Euler technique. Experienced developers may have heard of, or even prefer, the 4th order Runge-Kutte technique, or RK4. Game physics veterans often prefer a variant of the Euler technique known as the semi-implicit, symplectic,



modified Euler technique. I have messed around with the Verlet and Leapfrog techniques as well. Other Euler-related techniques are the midpoint, implicit, and reverse Euler techniques. There is also the class of predictor-corrector techniques.

There is a lot of outstanding literature on the strengths and weaknesses of each of these methods. I've included a list of references at the conclusion of this article that can give you a general overview of this broad topic.

One thing that all integration techniques have in common is that they seek to derive

"offline" integration were encoded into the equations. It may be more accurate to refer to this set of equations as "stepping equations," since they step the system forward by a single time step.

One feature that these stepping equations have is that you will always get the correct answer no matter how large the time step is. This is not so with the numerical integration techniques. Often, the first step to reducing error when using a numerical technique is to shrink the time step. Shrinking the time step requires more iterations for completion, and thus

has already been done offline through the use of calculus.

You may have noticed that most of the terms in these equations are identical to their counterparts in the kinematic equations for uniform motion. Indeed, the general form of the stepping equations for any arbitrary acceleration function looks like this:

$$v_{n+1} = v_n + \int_0^{\Delta t} a_n dt$$

$$x_{n+1} = x_n + v_n \Delta t + \int_0^{\Delta t} \left( \int_0^t a_n dt \right) dt$$

I call this set of equations the "kinematic integrator."

If you know the functional form of the acceleration, you can perform a little calculus to find out the results of the integrals. I have given the name "integral contribution" to the terms in the position and velocity equations which actually involve integration. For the sake of notational convenience, I usually label these integral contributions "dv" and "dx." I am capable of applying vague mathematical reasoning behind this choice of notation, but ultimately, it's much easier to make a variable called "dv" in my code, rather than one called "integral of a from 0 to delta t."

We can use the integral contributions—dv and dx—to classify acceleration functions. For instance, a constant acceleration would have the integral contributions of the form:

$$dv_{const} = a \Delta t$$

$$dx_{const} = \frac{1}{2} a \Delta t^2$$

And a pulse would have the integral contributions of the form:

$$dv_{pulse} = \Delta v$$

$$dx_{pulse} = \Delta v \Delta t$$

We can see, just by inspecting their form, that if you add two different constant forces together, the integral contributions of the resulting force can be obtained by adding the integral contributions of the two input forces. Likewise, if you add two pulse forces together, you would add the integral contributions of the two forces to get the contributions for the resulting force.

What if there were a constant acceleration and a pulse acting on the same object? Integration is a linear operation, so the integral

## ONE THING THAT ALL INTEGRATION TECHNIQUES HAVE IN COMMON IS THAT THEY SEEK TO DERIVE THE POSITION AND VELOCITY OF AN OBJECT FROM ITS ACCELERATION.

the position and velocity of an object from its acceleration. All the techniques I have listed obtain information about the acceleration by querying the value of acceleration at a very small number of points within the time step interval. Usually, the more points of acceleration you query, the better the method does.

What if you know the acceleration is constant over the course of the time step? Would you need to query the acceleration at more than one point within the time step interval? You would know that the result of the acceleration at any sub-interval time will be the same. In fact, if you know the acceleration will be constant over the course of the time step, you may say to yourself "Why should I mess around with a numerical method to solve this problem? It's possible to solve this problem exactly." The equations of motion for a constant acceleration can be exactly solved to give the equations below.

$$v_{n+1} = v_n + a \Delta t$$

$$x_{n+1} = x_n + v_n \Delta t + \frac{1}{2} a \Delta t^2$$

These are called the kinematic equations for uniform acceleration. Can we use this set of equations to step the state of our system forward in time? Certainly! I have seen the term "ballistic integrator" applied to this technique. Technically, however, this isn't an integrator in the sense of numerical integration. The equations do contain the results of integrating the acceleration over the time step, but the integration was done on paper, using calculus, and the results of this

increases the expense of the method. Having a set of stepping equations that is independent from the time step is a very good thing.

If we have a time step-invariant method, why do we ever mess around with numerical integration methods? The answer, of course, is that this method requires all forces to be constant, and we usually can't be certain that the acceleration is really going to be constant over the course of the time step when it comes to arbitrary forces. Although the "ballistic integrator" gives you the correct answer for constant acceleration scenarios, it gives absolutely the wrong answer if the acceleration changes by any significant amount. Thus, its use is limited to situations where you know that all forces acting in your game are truly constant over the course of the time step.

### THE KINEMATIC INTEGRATOR

» What if we consider another acceleration scenario where the acceleration is known and solvable? What if the acceleration comes as an infinitesimally short pulse at the beginning of the time step? This acceleration function is solvable and produces the following stepping equations:

$$v_{n+1} = v_n + \Delta v$$

$$x_{n+1} = x_n + v_n \Delta t + \Delta v \Delta t$$

The value of  $\Delta v$  is the instantaneous change in velocity caused by the short pulse. These equations are called the kinematic equations for a pulse. If you feel so inclined, you may refer to this as a "pulse integrator," but the term is misleading since all of the integration



contributions of such a combination of forces can be obtained by summing the integral contributions of each individual force, as below.

$$\begin{aligned} dv_{const+pulse} &= dv_{const} + dv_{pulse} \\ dx_{const+pulse} &= dx_{const} + dx_{pulse} \end{aligned}$$

Again, we just add the integral contributions together.

We may now conceive of a physics engine in which an arbitrary number of constant forces and pulses can be applied to an object, and the stepping equations will always give us the correct answer no matter how large the time step is! We merely add together the integral contributions of each of the individual forces to find the overall integral contributions that we feed into the kinematic integrator.

You can get a lot of leverage out of constant forces and pulses, but we might like to increase the repertoire of available forces in our physics engine. No physics engine can live without a spring force, for instance. The spring force is ubiquitous in most discussions of numerical integration, and is often used as a benchmark of performance since the acceleration of a spring-loaded object is constantly changing throughout the duration of the time step.

Like the constant and pulse forces, the spring force is solvable. However, you cannot solve the equations of a spring force by forward integration as with the other two. Since the acceleration of the spring is related to the position of the object, there is a feedback effect to take into account. You must use the mathematics of differential equations to solve the spring problem.

Consider a spring with one end attached to a mass, and the other end attached rigidly to the origin. Our spring is an idealized spring; it's frictionless, and doesn't have any self-interaction. In effect, the ideal spring does not exist as a physical entity, but just as a force acting on an object, pulling it constantly toward the origin. Solving the equations of motion for this spring, we can get the stepping equations:

$$\begin{aligned} v_{n+1} &= -x_n \omega \sin(\omega \Delta t) + v_n \cos(\omega \Delta t) \\ x_{n+1} &= x_n \cos(\omega \Delta t) + \frac{v_n}{\omega} \sin(\omega \Delta t) \\ \omega &\equiv \sqrt{k/m} \end{aligned}$$

These stepping equations will perfectly step the spring forward in time. You should always get the right answer, regardless of how large the time step is. The form of these stepping equations is slightly different than the form of the kinematic integrator due to the feedback mechanism of the spring. We can, however, do a little algebra work to recast these equations into the form of the kinematic integrator in order to identify the integral contributions.

$$\begin{aligned} dv_{spring} &= x_n (-\omega \sin(\omega \Delta t)) + v_n (\cos(\omega \Delta t) - 1) \\ dx_{spring} &= x_n (\cos(\omega \Delta t) - 1) + v_n \left( \frac{\sin(\omega \Delta t)}{\omega} - \Delta t \right) \end{aligned}$$

Now, we may shudder at the thought of inserting trig functions into our integration process. Luckily, the coefficients of the state variables  $x_n$  and  $v_n$  are functions of variables that should not change, so we really only need to calculate them once.

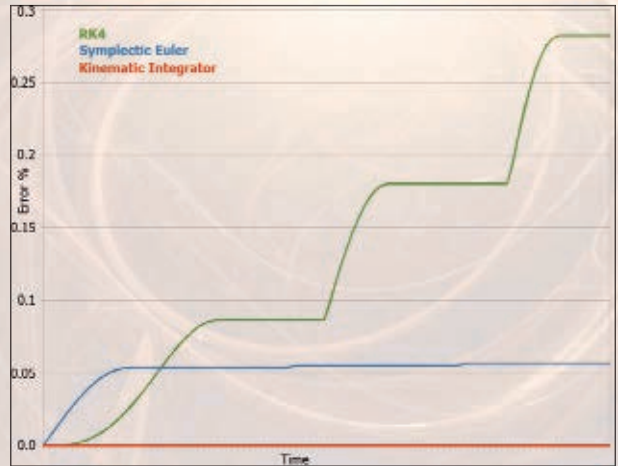


FIGURE 1 The error as a percentage of the maximum amplitude of oscillation is plotted for three integration methods.

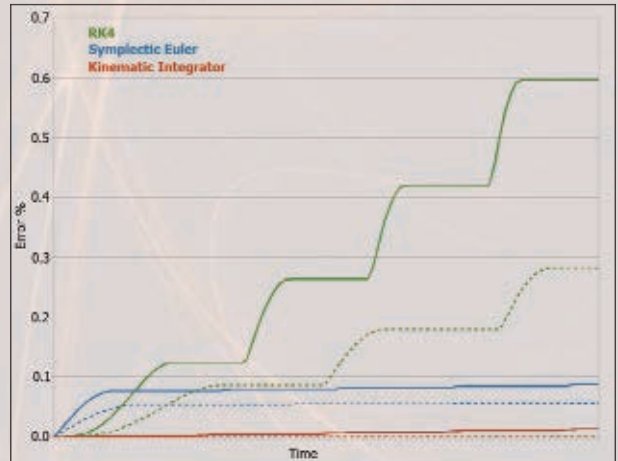


FIGURE 2 The error is once again plotted for three methods with a two-spring scenario. The dotted lines indicate the results of the one-spring scenario.

We may now have constant forces, pulse forces, and spring forces in our physics engine. Any number or combination of these forces can act on an object. We merely calculate the integral contributions of each, sum them together, and insert the resulting totals into the kinematic integrator. If we want to add more forces, we merely need to do a little offline calculus to find the functional form of the integral contributions. Hopefully, you can catch the vision of a full-featured physics engine using this idea.

## A FEW TESTS

» For the sake of comparison, let's contemplate a single un-dampened spring system. Our spring will have a spring strength of  $k = 100$ , and our mass will be  $m = 10$ . We will use a fixed time step of  $1/30$ , since of course our game doesn't drop below 30 fps!

We will compare three different methods: RK4, Symplectic Euler, and the Kinematic Integrator. We will integrate the system for 100 time steps, which is equivalent to about 3 seconds. For our initial conditions, we will start the mass at rest 5 units away from the origin.

Error is calculated as the difference between the integrated position, and the analytically calculated position of the mass, normalized by



dividing by the amplitude of the oscillation, which in this case is 5 units. The error of each of the three methods is plotted in Figure 1.

We see that RK4 starts off much better than Symplectic Euler, but due to the fact that it is an explicit method, and that there's no friction or damping to hold it back, the RK4 method begins to diverge. When plotted on the same plot with RK4 and Symplectic Euler, the Kinematic Integrator appears to have zero error. It does in fact accumulate non-zero error due to finite precision arithmetic. Still, the maximum error accumulated by the Kinematic Integrator in this scenario is more than 7,500 times smaller than the maximum error of the Symplectic Euler method.

Now, I previously stated that because integration was linear, we could safely add the integral contributions together. Strictly speaking, this is true, but when discussing the practical implementation details, it might not be. To see why, you need to remember that some forces, like the spring force, have a feedback mechanism. Practically speaking, when we calculate the integral contributions of individual forces, we don't take into account the feedback effect of other forces that might be simultaneously acting. If we tried to take all of this into account, it would make our integrator impractical for use with diverse systems.

I have made an approximation to solve this dilemma. The approximation is this: Let's assume that this feedback effect is not big—we just calculate the integral contributions of each force, assuming each time that there is only one force acting.

To get a picture of how this feedback mechanism might introduce a non-trivial error into a system, consider a scenario similar to our first example, except now we have two springs instead of just

one. The integral contribution of one spring is calculated with a knowledge of how the force will affect the intermediate position and velocity at all times during the time interval. If there is another force present to muck up these intermediate positions and velocities, the integral contribution becomes incorrect. In other words, the integral contributions of the spring were calculated taking into account only the feedback due to its own action—not taking into account the action of another spring.

Let's run our comparison scenario once more, but this time add a second spring to the system with the same spring strength of  $k = 100$ . Such a system is identical to a single-spring system with  $k = 200$ ; however, we will allow the system to step forward, letting the two springs act independently. The results are shown in Figure 2.

In this plot, we can see that the Kinematic Integrator has a visible departure from the zero line, so it is starting to accumulate some non-trivial error. The question is now, is the error bounded? The error may be small when integrating over 3 seconds, but what about 30 minutes? Unfortunately, we don't even have to go 30 minutes, as Figure 3 shows that the error for the Kinematic Integrator eventually surpasses the error of the Symplectic Euler method and starts to diverge. This plot goes out to 16 minutes and does not include the RK4 method, which we can already see is diverging after only 3 seconds.

This seems to spell doom for the Kinematic Integrator, especially if you want your system to run for more than 6 minutes. Of course, we could try to rescue things by introducing some damping, like we might do for the RK4 method, but the entire reason I began contemplating the



**UBM TECHWEB  
GAME NETWORK**

**MAY 6-7, 2010**

## **GAME DEVELOPERS CONFERENCE™ CANADA**

Vancouver Convention Centre,  
Vancouver, BC

[www.gdc-canada.com](http://www.gdc-canada.com)



**OCTOBER 5-8, 2010**

## **GAME DEVELOPERS CONFERENCE® ONLINE**

Austin Convention Center,  
Austin, TX

[www.gdcaustin.com](http://www.gdcaustin.com)



**DECEMBER 5-7, 2010**

## **GAME DEVELOPERS CONFERENCE™ CHINA**

Shanghai International  
Convention Center,  
Shanghai, China

[www.gdcchina.com](http://www.gdcchina.com)



**AUGUST 16-18, 2010**

## **GAME DEVELOPERS CONFERENCE™ EUROPE**

Cologne Congress Center East,  
Cologne, Germany

[www.gdceurope.com](http://www.gdceurope.com)



**FEBRUARY 28-MARCH 4, 2011**

## **GAME DEVELOPERS CONFERENCE® 2011**

Moscone Center,  
San Francisco, CA

[www.gdconf.com](http://www.gdconf.com)



For updates and more information on our events visit [www.jointhegamenetwork.com](http://www.jointhegamenetwork.com)





Kinematic Integrator was to get super accurate simulations which would be applicable in low-friction environments.

The solution I discovered was the use of an approximation. Since the springs have a feedback mechanism, the error keeps getting fed back into the system. What if I approximated the springs as a force that did not have a feedback? So far, the only other forces we have discussed in this article are the constant and pulse forces, neither of which have a feedback mechanism.

To approximate the springs as a pulse, I must accumulate all the acceleration produced by the spring during the time step and deliver it all at the beginning in a single pulse. This approach does not cause divergence, but it does allow energy to leak out of the system, forcing the spring motion to entirely dampen out within only a couple of minutes.

To approximate the spring as a constant force, I must find the average value of the acceleration during the time step, then use this average as the value of the constant acceleration. This method also has an energy leak like the pulse method, but the leak is much smaller. In fact, using this method, it takes about 1 million time steps, or a little over 9 hours, for the spring to dampen out entirely. The Average Acceleration method is easy to introduce. Simply use the integral contributions given by the below equation.

$$dv = \int_0^{\Delta t} a dt$$

$$dx = \frac{1}{2} dv \Delta t$$

In other words,  $dv$  is calculated however it was previously being calculated, and  $dx$  is always equal to  $dv$  scaled by the time step divided by 2. Thus, the Average Acceleration method does not introduce error into the velocity equation, but does introduce a slight amount of error into the position equation.

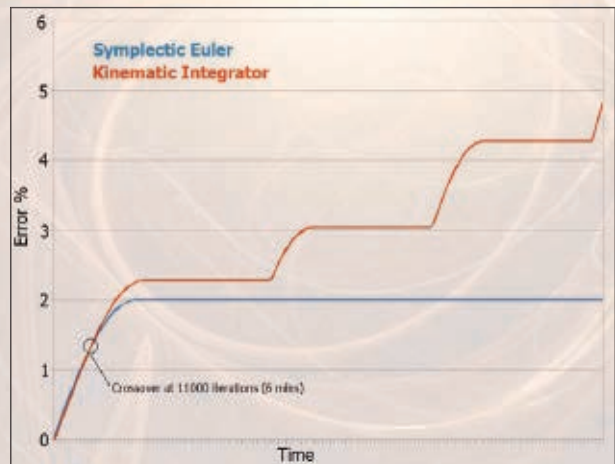
In Figure 4, we can see the actual oscillations of the systems after about 10,000 time steps, or about 5 minutes. The Symplectic Euler method maintains the correct amplitude due to its energy preserving properties, while the Kinematic Integrator using the Average Acceleration method has a slightly diminished amplitude but is in a correct alignment with the phase of the oscillation.

As stated previously, the system takes about 9 hours to dampen out entirely when using the Kinematic Integrator with Average Acceleration. The plot comparing the errors of the Symplectic Euler method with the Kinematic Integrator with Average Acceleration method is given in Figure 5. The Symplectic Euler method has an error that maxes out at 2.0, which represents an oscillation that is completely out of phase. The Kinematic Integrator with Average Acceleration method has an error that maxes out at 1.0, representing an oscillation that has entirely dampened out.

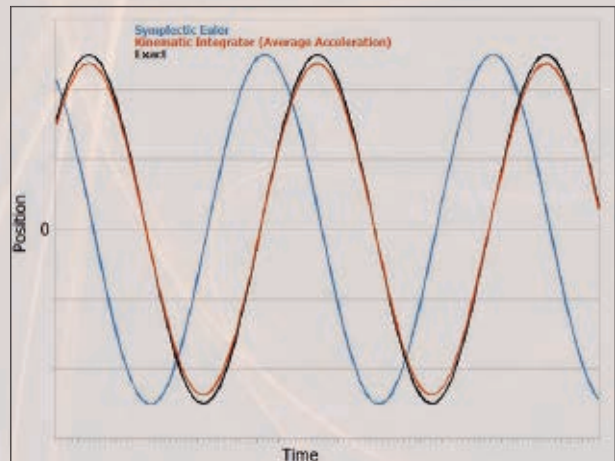
The types of errors introduced by the two different methods (phase vs. amplitude) are not necessarily comparable later in the simulation, but near the beginning there are moments where the error of the Symplectic Euler method is more than 200,000 times greater than the error introduced by the Kinematic Integrator with Average Acceleration.

Although the Kinematic Integrator with Average Acceleration loses energy, I have found the rate of energy loss acceptable for the simulation of real springs. In fact, it is far smaller than the rate that any real spring might lose energy. Any real spring would dampen out long before the 9-hour mark.

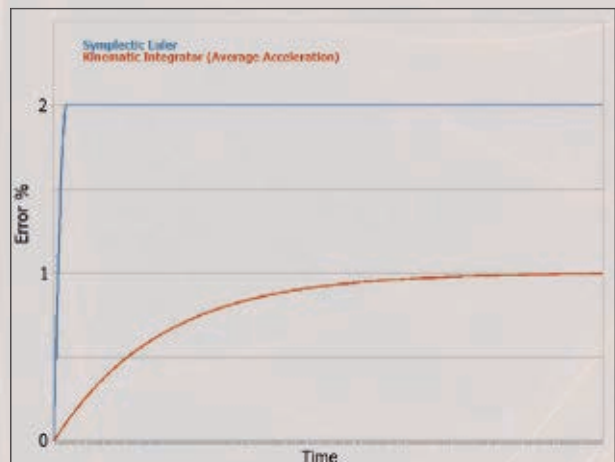
The error present in the Kinematic Integrator was originally due to the fact that we were not properly accounting for the feedback mechanism of other forces when calculating the integral contributions. With the Average



**FIGURE 3** Error plot for a simulation that lasts 16 minutes. After about 6 minutes, we can see that the Kinematic Integrator has diverging error.



**FIGURE 4** The oscillation of the system is plotted for the Symplectic Euler method and the Kinematic Integrator with Average Acceleration method and compared with the exact solution. This snapshot takes place at 10,000 iterations, or about 5 minutes into the simulation.



**FIGURE 5** The Error of the Symplectic Euler method as well as the Kinematic Integrator with Average Acceleration method are plotted for 9 hours' worth of simulation time.

Acceleration method, the error is now due to the representation of forces as constant. This mechanism is entirely different than other numerical techniques, in which error is introduced by approximating the calculus. Error induced by approximating the calculus leads to an established relationship between the size of the error and the size of the time step, leading to the classification of techniques as first order, second order, and so forth. When the error comes instead from approximating the forces, the relationship between error and time step is determined by the force itself, not by the method. Thus the error of the Kinematic Integrator with Average Acceleration cannot be related to the time step, without knowing what forces you are trying to simulate. For the specific example of a system with two springs, it appears to behave like a third order method—if the time step becomes half of what it was, the error becomes an eighth of what it was.

For those who want to see a vision of how you might use a Kinematic Integrator to simulate a dynamical range of forces, I have provided sample implementation code.

## EXTRAPOLATION OF USE

» In this article, we have discussed three forces: constant forces, pulse forces, and a frictionless spring attached at the origin. Here is a list of other possible forces that we can solve analytically offline and plug into the Kinematic Integrator:

- » A damped spring connecting two objects.
- » A network of springs connecting multiple objects.
- » Air friction.
- » A rigid distance constraint, using pulses to enforce a constant distance between objects.
- » A collision constraint, using pulses to resolve the position of the object as well as pulses to introduce restitution and surface friction.
- » Joint constraints can be represented by pulses, generally.
- » Driving forces, such as sinusoidal driving forces.
- » Velocity fields, as long as the functions used to describe the fields are integrable.

So, what happens if we want to simulate a force that is non-integrable? Or perhaps we don't know how to find the analytical solution to a given force, regardless of whether it exists.

What do we do? The Kinematic Integrator actually allows for the simulation of other numerical techniques, which can handle a more general class of forces. For instance, the Kinematic Integrator can emulate the Symplectic Euler method if you use the following integral contributions:

$$dv = a_n \Delta t$$

$$dx = a_n \Delta t^2$$

where  $a_n$  is the acceleration evaluated at the beginning of the time step. We can, thus, simulate any force using the Kinematic Integrator, whether it is solvable or not.

The Kinematic Integrator represents an

**THE KINEMATIC INTEGRATOR REPRESENTS AN INTEGRATION SOLUTION THAT HAS THE FLEXIBILITY TO SIMULATE ARBITRARY FORCES WITH THE OPTION TO LEVERAGE THE EXACT SOLUTIONS TO FORCES TO OBTAIN DRAMATIC PERFORMANCE INCREASE.**

integration solution that has the flexibility to simulate arbitrary forces with the option to leverage the exact solutions to forces, if we know them, to obtain dramatic performance increase. This performance increase is very substantial, reducing the error in the system by a factor of thousands. In some sample cases I've run, I've been able to reduce the error by tens of millions.


We also have the flexibility to employ approximations on the representation of our forces, such as when we introduced the Average Acceleration method. Such an approximation can be enacted globally across all forces, in order to reduce the error due to the feedback mechanism of some forces, like springs.

## FOR WHOSE ADVANTAGE?

» The Kinematic Integrator has the potential to revolutionize the way integration is done for game physics across the entire spectrum of development. From two-person indie projects, to third-party physics engines, to AAA titles.

To access the full potential of the Kinematic Integrator, the forces that are being simulated need to be integrable. Can you think of a single force used in a game that is not integrable? No doubt such forces exist, but I've never had to use them. Nonetheless, such non-integrable forces aren't excluded from use by the Kinematic Integrator—they

just won't get the full benefit of employing an exact solution.

How might your game physics benefit from a dramatic increase in accuracy and stability? How do you think you might go about inserting a Kinematic Integrator into an existing physics engine? How might you design a physics engine to employ a Kinematic Integrator? These are the questions you must answer for yourself. 

**ERIC BROWN** lives the rogue existence of a game physicist, scratching out a life in the bitter streets of Salt Lake City, UT. He is currently working at Rockwell Collins as a senior software engineer in the Simulation and Training division. He is in the final stages of completing a graduate degree in theoretical physics. Eric has lectured at the Game Developers Conference and participated in the publication

of several articles relating to game physics. Eric's work focuses mainly on physics simulation and animation. He believes that a true game physicist must possess the three P's: Passion, Patience, and Perspective. The 4th P is either Pantyhose or Peanut Butter.

## resources

### SAMPLE CODE IMPLEMENTING THE KINEMATIC INTEGRATOR

[www.gdmag.com/resources/code.htm](http://www.gdmag.com/resources/code.htm)

### PHYSICS FOR GAMES BY ERIC BROWN

<http://physicsforgames.blogspot.com/2010/02/kinematic-integration.html>

### ADVANCED CHARACTER PHYSICS BY THOMAS JAKOBSEN

[www.teknikus.dk/tj/gdc2001.htm](http://www.teknikus.dk/tj/gdc2001.htm)

### "PHYSICS, THE NEXT FRONTIER" BY CHRIS HECKER, GAME DEVELOPER, OCTOBER/NOVEMBER 1996

<http://chrishecker.com/images/d/df/Gdmpphys1.pdf>

### INTEGRATION BASICS BY GLENN FIEDLER

<http://gafferongames.com/game-physics/integration-basics>





# Unreal Technology News

by Mark Rein, Epic Games, Inc.

Canadian-born Mark Rein is vice president and co-founder of Epic Games based in Cary, North Carolina.

Epic's Unreal Engine 3 has won Game Developer magazine's Best Engine Front Line Award four times and is also one of the few Hall of Fame inductees.

Epic's internally developed titles include the 2006 Game of the Year "Gears of War" for Xbox 360 and PC; "Unreal Tournament 3" for PC, PlayStation 3 and Xbox 360; "Gears of War 2" for Xbox 360; and "Gears of War 3" for Xbox 360.

## Upcoming Epic Attended Events:

### E3 2010

Los Angeles, CA  
June 15-17, 2010

### Develop

Brighton, UK  
July 13-15, 2010

### Gamescom

Cologne, Germany  
August 18-22, 2010

Please email:  
[mrein@epicgames.com](mailto:mrein@epicgames.com)  
for appointments.

## CCP DEVELOPING A GENRE MASHUP USING UNREAL ENGINE 3

CCP's DUST 514™ is an MMO and first-person shooter hybrid for consoles that will be set in the same deep universe as EVE Online, their long-established PC game that has become a staple of the MMO genre.

Developed for Xbox 360 and PlayStation 3, DUST 514 allows players of both games to interact seamlessly across platforms. If that sounds ambitious, that's because it is — nothing of its kind has been attempted before.

CCP's decision to use Unreal Engine 3 was based on Epic's reputation, speed and functionality.



CCP's cross-platform online world DUST 514

"We needed to provide our development team with a solid foundation to work from, an engine that allows rapid prototyping and iteration of our core FPS mechanics," said Atli Már Sveinsson, DUST 514's creative director.

"With the scope of certain elements of the game, such as terrain size and lighting, we needed an engine that was agile enough to provide quick iteration within the provided framework. Combining that flexibility with the fact that the Unreal Engine has been battletested by some of the greatest titles ever crafted, made Unreal Engine 3 the absolute best choice for us."

The team members at CCP and at Epic Games China worked collaboratively, creating a strong relationship from start to finish.

According to Sveinsson, hands down, accessibility of knowledge was Epic's greatest asset. Unreal's long-standing reputation for excellence and convenience has produced a multitude of seasoned users, including members of their own development team. The Unreal Developer Network (UDN), with its various mailing lists and access to the talented engineers and artists at Epic Games China, gave them an edge, allowing CCP to train its team quickly on how to use the tools and hit the ground running.

Paul Meegan, Epic Games China's CEO, leads efforts to make sure that Epic's licensees can walk through a process in a way that's efficient and fast.

"We found Epic's licensing process to be outstanding, starting with a tremendous level of support during CCP's evaluation stage," said Thor Gunnarsson, CCP's vice president of business development.

"The DUST 514 team in CCP's Shanghai office was able to prototype and iterate early proof of concept with close proximity to and support from the Epic Games China team, which proved to be invaluable.

As we completed this phase and moved to licensing, confidence and trust in the working relationship was cemented, leading to a rapid and streamlined commercial licensing pipeline."

In just six months, dynamic lighting and massive mega terrains were

implemented in DUST 514 due to the standardizing on Unreal Engine 3 that gives CCP's development team the time and flexibility to focus on perfecting the optimal client-side engine and pipeline to fully realize its creative and artistic vision.

With its own core technology platform, CCP created a fully scalable and persistent online gaming experience that meets the visual, technical and creative demands for DUST 514.

"Unreal Engine 3, which complements and integrates easily with our own technology, provides CCP with precisely the sort of elegant solution we favor," said CCP's CEO Hilmar Veigar Petursson.

"Having a proven framework for consoles supporting our first venture into that genre allows the DUST 514 developers to focus time, talent and energy squarely on making an incredible game," said Petursson.

For more information about CCP or DUST 514, visit [www.ccp.com](http://www.ccp.com) or [www.dust514.org](http://www.dust514.org).



For UE3 licensing inquiries email:  
[licensing@epicgames.com](mailto:licensing@epicgames.com)

For Epic job information visit:  
[www.epicgames.com/epic\\_jobs.html](http://www.epicgames.com/epic_jobs.html)

WWW.EPICGAMES.COM

Epic, Epic Games, the Epic Games logo, Gears of War, Gears of War 2, Unreal, Unreal Development Kit, Unreal Engine, Unreal Technology, Unreal Tournament, the Powered by Unreal Technology logo, and the Circle-U logo are trademarks or registered trademarks of Epic Games, Inc. in the United States of America and elsewhere. Other brands or product names are the trademarks of their respective owners.

# TELLTALE GAMES TALES OF MONKEY ISLAND

postmortem



IN MAY 2009, TELLTALE GAMES CELEBRATED ITS FIFTH anniversary. In this short time, we'd already released over 20 games across six franchises—most of them on a consistent monthly release schedule—and had gained a reputation for being the only company to really get episodic gaming right.

Our founders and many members of our staff came from LucasArts, where they worked on popular adventure games such as GRIM FANDANGO, DAY OF THE TENTACLE, and, of course, the MONKEY ISLAND series. Although we had long believed that these classic franchises would be perfect for our episodic format, LucasArts holds the rights. Then something pretty amazing happened: we reached an agreement with LucasArts and their new president, Darrell Rodriguez, to develop their best-known and most beloved adventure game franchise into an episodic series. TALES OF MONKEY ISLAND would give us the rare opportunity to take the best of the games we'd built in the past and combine that legacy with the episodic model we're pioneering today.

A few weeks after our anniversary, we announced TALES OF MONKEY ISLAND at E3, and hunkered down for our most ambitious episodic production yet.

## WHAT WENT RIGHT

**1) TELLING A STORY IN FIVE CHAPTERS.** MONKEY ISLAND is famous for sending Guybrush Threepwood, Mighty Pirate™, on sprawling adventures. It may be a comedic franchise, but below the surface, it's fairly serious and epic. To do it justice, we felt TALES OF MONKEY ISLAND needed to be one big story told in five chapters, rather than five little stories loosely strung together (as we have typically handled other episodic

series). Approaching the story this way meant evolving our design process and staying hyperconscious of our story decisions, right from the beginning.

Episodic games provide a unique chance to keep the audience's attention for several months as the series runs its course. While we wanted each episode to provide a gratifying experience on its own, we also wanted to leave players dying to find out what would happen next—the same way an addictive television series like *Lost* keeps viewers guessing week after week. Early in the design process, we put a lot of thought into the questions each episode should raise and the cliffhanger it would end with. Our intent was not to abruptly end the episode at the height of the action but to bring it to a satisfying conclusion, and then tantalize the player with a tiny bit more. In the first episode, this cliffhanger came in the form of a mysterious woman putting a sword to Guybrush's neck. We knew we'd made the right choice when our forum exploded with speculation about who she was and what she would do to him.

This season-wide approach to the story gave us interesting opportunities to develop character relationships as well. Rather than revealing up front that Guybrush would enter into a complicated friendship with this new woman, we allowed this relationship with mercenary Morgan LeFlay to unfold naturally throughout the series' five-month run. The demon pirate LeChuck suddenly denouncing his evil ways would have been a much harder sell if we hadn't been able to gradually reveal his new personality. And the shocking events of the fourth chapter, which set up our grand finale, were heightened by the fact that terrible things were happening to people the players had grown to love not over









the course of a few hours, but over several months. *TALES OF MONKEY ISLAND* showcases some of our most intricate storytelling to date, and we couldn't have pulled it off if we hadn't thought so much up front about how to use the five-chapter format to tell this story.

## 2) WORKING WITH MONKEY ISLAND "OLD TIMERS."

Since many on our team got their start working on (or playing) classic LucasArts adventure games, we're in a pretty good position when it comes to reviving a franchise like *MONKEY ISLAND*. However, we didn't have the benefit of working alongside *MONKEY ISLAND* creator Ron Gilbert. (He was busy over at Hothead Games with his own project.) Fortunately, we were able to pick his brain for a few days last spring, when he graciously agreed to visit our office.

Brainstorming with Ron was incredible. He cast new light on the characters' personalities and helped us understand what their motivations would be in the storyline we were devising. His most influential input came when we discussed Elaine's character. We had originally imagined her as a typical damsel in distress who had to rely on Guybrush to save her. In Ron's mind, though, Elaine was smarter than that; she should always be one step ahead. This simple revelation completely changed our perspective on her character and had a huge impact on our story.

Instead of being clueless, Elaine was now playing the long game—going along with LeChuck's plans because she had an inkling of the bigger picture that Guybrush, and even the player, did not. In fact, if we recall correctly, it was Ron's idea for Elaine to give Guybrush her wedding ring early in the season, an item that he carried around with him through the remaining episodes which became crucial to his eventual success.

Other "old timers" with whom we had the pleasure of working were several returning members of the voice cast, including Dominic Armato as Guybrush Threepwood and Earl Boen as LeChuck. Dominic knows Guybrush's character so well that during voice recording, the designers just stepped back and let him do his thing. (If he hadn't been available for this game, we'd probably have an entirely new "What Went Wrong" section for the postmortem!) We also got to work with Michael Land, the original *MONKEY ISLAND* composer, who did a fantastic job on the music and met our expectations with almost every piece.

And of course, our development team has quite a few series veterans as well. Our design director, Dave Grossman, co-designed and co-wrote the first two *MONKEY ISLAND* games with Ron Gilbert and Tim Schafer. Many other members of the design, art, and programming teams had been involved in previous *MONKEY ISLAND* games at LucasArts. Considering it had been nearly 10 years since a

new *MONKEY ISLAND* game was released, it was a really special feeling to bring together so many people who made the original magic happen.

## 3) THE CREATION OF A NEW CINEMATIC DIRECTOR ROLE.

We want our games to be cinematic experiences, like you're playing an episode of a TV show. Our versatile development tools allow us to achieve this by making it easy to move cameras around and add effects, but of course someone needs to be available to take a hard look at the scene, decide what needs to be done, and make sure this happens. Normally, this is part of the designer's job—along with designing and implementing puzzles, writing all the dialogue, and overseeing the voice recording process.

Due to *TALES OF MONKEY ISLAND*'s ambitious schedule, certain scenes weren't getting the attention they required. The problem first became apparent as we reviewed the series' opening, which sets the entire story in motion. Set on ships at sea, the scene was supposed to be a high-stakes battle between Guybrush and LeChuck, but initially it felt static and not very exciting. The designers were too swamped to focus on it, so Jake Rodkin, our graphic designer, stepped in and provided additional direction to the choreography team to help get the scene up to snuff cinematically. Thunder and lightning were added, the sky was changed from dusk to



# TALES OF MONKEY ISLAND

dark, and movement and camera shakes were implemented to send the ships bucking and rolling through the choppy waters.

The changes gave the scene the impact it needed, and led us to create a new “cinematic director” role—someone who could work with the designers to identify the big story beats that needed this type of attention, then with the animators and choreographers to make sure the scenes came together. This provided continuity across the series that is sometimes lost due to the design lead changing from episode to episode, and it freed up the designers to focus on script writing and puzzle implementation, which became even more critical as the season progressed and the schedule got tighter. The new role proved to be an important addition for TALES OF MONKEY ISLAND, and we are now including it as a regular part of our teams moving forward.

#### 4) A GOOD BALANCE OF OLD AND NEW CONTENT.

We were lucky to have a vast MONKEY ISLAND tradition to draw from. It was tempting to bring back every character and gameplay mechanic that fans enjoyed in the old games, but we also had an incredible opportunity to evolve a series that had been dormant for years. So we made an effort to embrace the old tradition while also building upon it.

We knew that the game should include Guybrush, his wife Elaine, his nemesis LeChuck, and the enigmatic Voodoo Lady—it wouldn't be Monkey Island without them. We also wanted to include other popular characters, like Stan the used ship salesman and Murray the demonic talking skull, but wanted their appearances to be surprising for the fans and to make sense to the story. We looked for places in our overarching plot that would accommodate them, and held off on bringing them back until later in the episodic run.

Instead, we populated the first few episodes with new characters who were true to the MONKEY ISLAND universe, like Reginald van Winslow, Guybrush's faithful first mate, and Morgan LeFlay, an adversary-turned-ally as the season progressed. By season's end, both had become fan favorites. When Murray and Stan made their highly anticipated appearances, we brought back elements we knew the fans would expect, such as Murray's voice actor Denny Delk, and Stan's waving arms and crazy, dancing-plaid jacket. (It's not easy to make 3D plaid dance!)

We took a similar approach to beloved gameplay mechanics such as Insult Swordfighting, the practice of overpowering an opponent through calculated quips such as the classic “You fight like a cow.” From the day we announced TALES OF MONKEY ISLAND, we were constantly being asked whether the game would include it. Our designers maintained that we would only revisit Insult Swordfighting if it made sense for the story, and if we could evolve the mechanic rather than simply copy it. When Insult Swordfighting eventually

appeared in the final episode, we put a new spin on it by making the player compliment one character while insulting another. Ironically, the puzzle was so well integrated that many players didn't realize it was Insult Swordfighting at all. It was simply good gameplay that fit into the MONKEY ISLAND universe—and that was our goal all along.

**5) STRONG ART DIRECTION.** Art direction tends to be a hotly debated topic among our audience, and TALES OF MONKEY ISLAND was no exception. Some fans wanted the game to look exactly like the pixelated THE SECRET OF MONKEY ISLAND (but better!), or to have the hand-drawn cartoon style of THE CURSE OF MONKEY ISLAND. Others thought it should use the best available 3D technology. What we wanted was the best of both worlds: 3D art with a distinct cartoon style that made the best possible use of our development tools. We're proud of the outcome.

Defining the art style started with nailing down the characters. Our concept artist, Ryan Jones, worked closely with Jeff Sengalli from LucasArts, who provided valuable feedback about the characters' silhouettes, facial features, and clothing. Since Guybrush had presumably been on many adventures since we last saw him, we were aiming for an older, wiser Mighty Pirate. Many people on the team liked the coat he wore in MONKEY ISLAND 2: LECHUCK'S REVENGE, so our

Guybrush was given a similar coat with a pattern that looked like gold embroidery. Jeff noticed this and encouraged us to use similar details on the rest of the characters' clothing, which helped provide visual interest and a cohesive design.

Our art directors Derek Sakai and Dave Bogan had worked on the THE CURSE OF MONKEY ISLAND and ESCAPE FROM MONKEY ISLAND, so they knew what they wanted—and what they didn't want—in a 3D cartoon style. Our plan was to take the best aspects of the first two games, which had a more realistic (if pixelated) look, and mix them up with the whimsy of the third game to create a visual style that was distinctly ours. A lot of consideration went into the color palette, details around the environments, and the use of light and shadow.

All of our series are built with the same engine, and new tech that had been developed for our WALLACE & GROMIT and CSI games allowed the MONKEY ISLAND team to push the art direction. Updates to the lip sync and character acting systems allowed for more detailed facial expressions which benefitted all of the TALES characters—especially Guybrush, who we think is our best-acted character to date. New lighting tech allowed us to do effects we previously couldn't have achieved, like having a character walk in and out of pools of light in the Voodoo Lady's hut. All these additions contributed to a game that looks unlike any other Telltale project.







## WHAT WENT WRONG

### 1) THE SCHEDULE WAS TOO CLOSE FOR COMFORT.

TALES OF MONKEY ISLAND is our fifth monthly episodic series, and by this point, we have a pretty good system. We usually take several months up front to build assets, plan the story, and get a head start on episode production before the monthly releases begin. Once the series launches, we follow overlapping schedules, with work on a new episode ramping up as the previous episode wraps.

The TALES OF MONKEY ISLAND schedule followed the same principles but encountered problems in practice, starting with a very ambitious release date. From a marketing standpoint, revealing the game at E3 in June and announcing its July 7 premiere at the same time was a good move. We maximized the MONKEY ISLAND love by putting out our game at around the same time as LucasArts' THE SECRET OF MONKEY ISLAND SPECIAL EDITION. However, our agreement with LucasArts wasn't finalized until sometime in February, so we were on a very fast-tracked schedule to meet this date. Thanks to hard work from everyone on the team, we were able to launch our first episode on time, but this situation sent a ripple effect through the rest of the schedule.

We believe that for an episodic game to be successful, it must follow a consistent and reliable schedule. This makes scheduling a critical factor in

our process, and keeps us under pressure to stay on track and release our games on time. We did initially try to make up for the scheduling issues by planning a six-week gap between the first two episodes, instead of four weeks. Since we were still getting up to speed with the series, we needed that time to get Episode 2 out the door, but we still couldn't get a head start on Episode 3. By the end of the season, the schedule was extremely tight, and we ended up taking a few extra weeks to get the final episode out.

The external conditions that drive announcements can't always be controlled, and even in hindsight, we'd probably still announce at E3 and launch soon after. But we now know all too well that if the schedule is compromised up front, it will be very tight in the home stretch.

### 2) MULTIPLATFORM DEVELOPMENT CAUSED PRODUCTION TRAFFIC JAMS.

TALES OF MONKEY ISLAND released episodically on both PC and WiiWare. We had some prior experience with multiplatform development, but we weren't yet at the point where we could easily release an episode on two platforms simultaneously. For this reason, we decided to lead with each episode on PC and follow on WiiWare as soon as we could.

Since the PC version always launched first, this is where the team's focus was concentrated during production. Once the PC version of an

episode was complete, most of the team rolled onto the next episode while our QA group and producer got the WiiWare version out the door. This often meant that testing of the newest episode began late, by which time the people capable of fixing bugs may have already moved on to the next episode. So when WiiWare submissions were kicked back to us, we had to take resources off the latest episode to fix them.

The music pipeline presented another set of problems. Since cutscene music can't be finished until scenes are time-locked, it's delivered late in the production cycle. The TALES OF MONKEY ISLAND music was created in MIDI to allow a smaller data size for the WiiWare version, then converted to WAV for better quality in the PC version. It's a time-consuming process that couldn't get underway until the cutscene music came in at the very end of an episode's development, and often wasn't completed until days before the episode's scheduled release date on PC.

Multiplatform development—and in particular, simultaneous episodic launches on multiple platforms—is an important goal, so these are issues we'll have to continue to anticipate and plan for as we work on new series.

### 3) OUR "GENERIC" PIRATES TURNED OUT TO BE TOO GENERIC.

Reuse of assets is critical in episodic games. By developing useful libraries





# TALES OF MONKEY ISLAND

of art and animations that can be used across a series—like a Hollywood back lot and prop department—we can focus our resources on the most important areas of each episode. We can also reduce our data size this way, which is necessary for downloadable PC games and especially crucial for WiiWare titles.

Since TALES OF MONKEY ISLAND's storyline would send Guybrush on an epic pirate adventure, we decided not to reuse locations as much as we had in previous games. To compensate, we developed a system that would easily allow us to reuse character skeletons and animation suites. We wanted the illusion of a big, bustling world full of pirates—similar to THE SECRET OF MONKEY ISLAND, where random pirates are roaming the jungle and standing around on the streets—so we conceived a "pirate toolkit" with four different body types that would allow the so-called "generic" pirates to share animation suites and to be visually differentiated by minor changes in their appearances.

The first problem we encountered was that even with only four body types, the data size was too big, and we had to narrow the suite down to two—a tall/skinny pirate, and a short/fat pirate. Then, although we customized them, we didn't customize them enough to make them unique. Even with different outfits, facial features, and skin tones, it was very obvious in the first couple of episodes that these pirates all came from the same stock, which detracted from the overall presentation.

By Episode 3, we started making more drastic changes to the pirates' appearances by diversifying their facial features and giving each pirate a unique silhouette. We tried to make sure that these model changes only minimally increased our riggers' tasks by focusing on areas of the model that did not deform much (i.e. noses, chins, skull shapes, and hats). We also tried to add at least one unique animation to each pirate's acting suite, and differentiated their idles to help set them apart. These changes required help from a character modeler and rigger who otherwise would have been assigned to another project.

**4) CONTROL SCHEME DECISIONS WERE MADE AT THE LAST MINUTE.** Traditionally, Telltale's games have used a mouse-driven, point-and-click interface. Starting with WALLACE & GROMIT'S GRAND ADVENTURES, we implemented a direct control system instead. Direct control makes for a less passive and more engaging player experience, plus it gives us more freedom cinematically. On the downside, WALLACE & GROMIT's direct control was designed for an Xbox controller and was clunky on the keyboard. The MONKEY ISLAND audience is made up of some devoted point-and-click fans, and we didn't want to alienate them with spotty controls.

Unsure of how we were going to proceed, the designers began working with the assumption that TALES OF MONKEY ISLAND would be point-and-click, and had already brainstormed much

of the first episode before they knew for sure that direct control would be used. This didn't cause any major issues with the puzzles that had been conceived, but it prevented the team from incorporating gameplay that took advantage of the control scheme—something that might have helped us sell to those point-and-click die-hards. Fortunately, our episodic development meant that we could improve this in subsequent chapters.

Even after direct control was confirmed, during most of the first episode's development, we didn't really know how it would work. We ended up completing the new "click-and-drag" system just days before release, with insufficient review or usability testing. We were generally happy with the system, which works by holding down the left mouse button and dragging in the direction you want the character to move, but there was no time to fine-tune it before the first episode launched. In particular, the wonderfully cinematic opening scene was difficult to control under this system, and since this sequence was our series-wide demo, it might have ended up turning off some of the players we were actually hoping to accommodate.

We do plan to keep evolving "click-and-drag" for future series, but we have to be more diligent about making decisions like this one while there's still plenty of time to implement changes. Which brings us to our final headache.

## 5) WE WANTED EVERY EPISODE TO BE BIGGER AND BETTER THAN THE LAST (AKA FEATURE CREEP).

The drive to make the best game possible is hardly a negative, and we don't regret having a team full of overachievers who take enormous pride in their work. The problem comes when people don't know when to stop. To some extent, our episodic development prevents feature creep, because the schedules are fairly compact and the monthly release dates provide clearly defined endpoints. But if a scene isn't reading right, this often isn't apparent until late in the process. The game doesn't come together in its fully playable format until a week or two before release, at which point there's not much time left for changes.

Spending time perfecting the story's important moments is easy to justify. It's devoting too much time to smaller bits and throwaway gags that can get us into trouble. For example, when Guybrush dives down to the ocean floor in the third episode, we had a perfect opportunity to

reference a similar scene in MONKEY ISLAND 2: LECHUCK'S REVENGE. Doing so wasn't necessary for the story, but it made the game a richer experience for MONKEY ISLAND fans who would understand the reference. On the other hand, it was a lot of work for only about 20 seconds of gameplay, and additions like this can have impacts on other departments, such as the testers who have to make sure the new content works correctly, and the sound team which has very little time to score sound effects. The more we tried to squeeze in, the more those guys were slammed.

Feature creep caused additional problems with our multiplatform development, since last-minute changes to the PC version could send us over the size limit for the WiiWare version. Our compression tools make Wii conversion fairly painless, but when the episodes get too large, we have to start hunting for space wherever possible. Our producer recalls a few stressful days where we needed to squeeze another 600K out of the final episode, and we had

already purged all of the "low-hanging fruit" to get it that far. We ended up reducing or purging assets that added up to a measly 5–10K to find space.

Of course we're going to keep creating the best games we can, but we need to get better about prioritizing and ensuring that any last-minute additions are really important "need to have" changes as opposed to tweaks that would be nice to have.

## MONKEY OFF OUR BACKS

TALES OF MONKEY ISLAND was an ambitious and at times stressful project. It was difficult to meet the schedule and we worried about whether we were doing right by the source material. Nearly everyone on the team has a special spot in their hearts for MONKEY ISLAND—either because they worked on the series previously, or because they played it and loved it long before entering the video game industry—and it was very important to us to get this game right. Thankfully, the hard work paid off; TALES OF MONKEY ISLAND is our best-selling and most successful episodic series so far.

Bringing back MONKEY ISLAND was an amazing opportunity, one that caught us by surprise and forced us to push ourselves and the episodic format farther than we ever had before during Telltale's first five years. And you know what? We'd do it all over again, and we sure hope we get the chance. ☺

**EMILY MORGANTI** handled Telltale's public relations, customer service, and community management for three years. She now works as a freelance writer and PR consultant.

GAME DATA	
	
<b>PUBLISHER</b>	Telltale Games
<b>DEVELOPER</b>	Telltale Games
<b>NUMBER OF DEVELOPERS</b>	50
<b>LENGTH OF DEVELOPMENT</b>	10 months
<b>RELEASE DATES</b>	Chapter 1 – July 7, 2009 Chapter 2 – August 20, 2009 Chapter 3 – September 29, 2009 Chapter 4 – October 30, 2009 Chapter 5 – December 8, 2009
<b>LINES OF CODE</b>	700k or thereabouts
<b>SOFTWARE</b>	Telltale's proprietary game engine, Autodesk Maya
<b>PLATFORM</b>	PC and WiiWare
<b>OTHER FACTS</b>	5 children born, 3,400+ bagels eaten, 1,200 pots of coffee consumed





## Download the NEW xaitControl 3.0 SDK.

xaitControl 3.0 offers a cross-platform live-live debugger featuring conditional breakpoints, per instance realtime-view, history and graphical decision path, manual triggering and blocking of transitions and dynamic FSM updates.

Contact xaitment today for more information about the BrainPack Program under [brainpack@xaitment.com](mailto:brainpack@xaitment.com) or visit our website [www.xaitment.com](http://www.xaitment.com)

 **XaitMENT**  
INTELLIGENCE ENGINEERED



# AN ISLAND OF DETAILS PLAYING THROUGH THE ORIGINAL MONKEY ISLAND WITH CREATOR RON GILBERT

**A COUPLE OF YEARS AGO, WHEN I STARTED DESIGNING THE SOON TO be released DEATHSPANK, I played all the way through THE SECRET OF MONKEY ISLAND to refresh myself on the puzzles and dialog.**

I know this will come as a shock to many hardcore MONKEY ISLAND fans, but I don't spend my evenings playing MONKEY ISLAND. It's probably been 15 years since I sat down and really played it. Doing so brought back a lot of memories and tidbits of facts, so I started keeping notes in celebration of all things MONKEY ISLAND.

Before we begin, a couple of points:

- I was playing the VGA version that was released after the original EGA version. The original version used 16 colors and the inventory was text only.
- These are only "somewhat" in order.
- It's been almost 20 years. My brain is full.

Someone please turn off the lights; I'll start the projector.

» The very small "hot spot" areas are very annoying these days, but they were accepted back then; they were even considered a good thing [1]. It's called gameplay! It would be hard to make an adventure game today where players were forced to hunt for small objects.



Back in the late 80's, the mere thought of a scroll wheel on a mouse would have been crazy talk, but today I find it hard to break the habit of trying to scroll through the inventory with it.

Most people know you can hit the period key to skip a single line of dialog, but surprisingly, some don't know why I chose the period key. It seems obvious to me: a period ends a sentence. Something I added to Humongous Entertainment's adventure games was the cursor changing into a big arrow when you hit the edge of the screen and it was an exit. MONKEY ISLAND would have benefited from this. I was so used to it from the Humongous games that I'd scan the screen expecting to see it over exits.

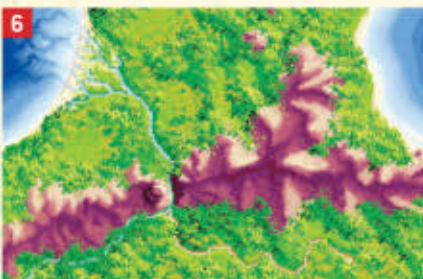


The UI for having to open and close doors independently from walking through them was obnoxious. You'd never do this in an adventure game today, but like pixel hunting, it was accepted back then.

» While Insult Sword Fighting is one of the first things people think of when they hear MONKEY ISLAND, I thought it seemed a little tedious (but fun) when I played through it again [2]. There is a point where you say, "I get it," but you're still forced to go through the motions again and again. If I were going to do Insult Sword Fighting in a future game, I'd make it more free form, allowing the player to be clever and construct their own sentences.

During the early stages of the MONKEY ISLAND design, we watched old Errol Flynn-era pirate movies. One thing that stood





out was that during the fights, they always taunted each other with insults. I knew we needed to have sword fighting in the game—it was about pirates after all—but I didn't want to introduce any action gameplay; the old pirate movies provided the perfect solution.

» You only have to follow the shopkeeper to the Sword Master once (3). After that, it's automatic. That was a good design choice. No point in having to solve the same puzzle over and over.

» There was supposed to be ship combat during your voyage to MONKEY ISLAND. It would have been done top-down view with you controlling the ship and firing a cannon. It was right to cut that. It's rare that I will watch a "director's cut" of a film that is better than the original. Most of that stuff was left out for a reason.

The scene behind the walls of the governor's mansion seemed a little long. The original plan was to have real gameplay and puzzles, but the game felt too large and we needed to cut stuff. This gag was perfect and, in some ways, better than making the player solve more puzzles. Never be afraid to edit your game down if it needs it. It will often be better for it.

» Stan's Grog machine was one of three interesting lessons I was given about trademarks and copyright during the production of MONKEY ISLAND (4). Originally, I wanted the Grog machine to be a Coke machine, and barring that, I wanted it to look like a Coke machine. It originally had the "Coke Wave" on it, but said "Grog." The Lucasfilm legal team came back and said it was too close to the real trademarked Coke Wave. I tried to argue parody to no avail. We kept changing it little by little until legal was satisfied it didn't look too much like a Coke machine.

I'm going to admit that I was completely stumped by the grog puzzle. I finally went and looked it up on the Internet. That's a damn good puzzle. Several puzzles gave me pause because I'd remember some previous unimplemented version of them and it would throw me off track. My brain is filled with a lot of old adventure game puzzles, most of which never made it into a game. DEATHSPANK actually has a couple of puzzle ideas that we talked about for the original MONKEY ISLAND.

» Getting stuck on the ocean floor is one of my favorite puzzles because the solution is so obvious most people overlook it (put the anchor in your pocket) (5). The other puzzle I did in the same vein was in the INDIANA JONES AND THE LAST CRUSADE adventure game, where Indy has to have faith and just walk over a ledge. Players that had faith and just clicked on the other side of the crevice had no problem. Players that fiddled around and clicked on other stuff, ignoring the advice, always fell.

» I wanted the time you spent on MONKEY ISLAND to feel more like an RPG, which is why you had a top-down view. As the game progressed, I slowly scaled back those plans, but we were still left with very cool maps (6). I love maps. For me, a game design always starts with a map.

» Back then, the randomly generated forest was cutting-edge technology (7). Disk space was at a premium. Everything had to fit on five floppy disks. Sierra would ship games on 8 or 10 floppy disks. That was always a sore point for us.

» I like the way you meet Otis, Sword Master, and Meathook during regular puzzles, then hire them later. This builds a sense of friendship before they are needed. They aren't just three random people you meet for the first time while looking for a crew.

» The circus tent could have been utilized better (8). It was a bit of a waste for one puzzle, and the Fettucini Brothers didn't add much to the story. That said, this was one of the first screens ("rooms" as we called them) where we used the SCUMM system's exclusive scrolling screens for dramatic effect. Tim Schafer was the programmer on this room, and he spent a lot of time getting it to scroll at just the right time to reveal the tent. I also like that the dialog choices are shown upside down after Guybrush is shot from the cannon. Doing that pushed the SCUMM system to utilize multiple fonts. We didn't have that feature before this gag required it. Guybrush being shot out of the cannon was also foreshadowing for when he needed to be shot out of another cannon later on. OK, come to think of it, this was a pretty useful room.



# AN ISLAND OF STYLED DETAILS



» The “touch the parrot” puzzle was a little lame and linear. I remember being rushed, and we couldn’t think of anything better. The game was feeling good and long already, so I just punted on this puzzle.

» Begging for the necklace from the head of the navigator is a bad puzzle. It’s too easy for players to think they’re on the wrong path [9].

» “I had a feeling in hell there would be mushrooms” is one of Tim’s lines. Tim hates mushrooms. I also hate mushrooms, but unlike Tim, I’m happy to pick them out [10].

» The key to the Monkey Head used to be called a Q-Tip™, but due to my second legal lesson on the project, it was changed [11]. According to our legal advice, it would have been OK if we were using the Q-Tip in a “correct fashion,” but taking a giant Q-Tip and sticking it into a stone monkey’s ear is not “correct usage.” Interestingly enough, a Q-Tip box states: “Do not insert cotton swab into ear canal.” I think we were doing just that.

» Elaine Marley was called “The Governor” until the scene in the church was written. Dave Grossman wrote that scene and put in the gag dialog choice where Guybrush shouts “Elaine!” which is from the movie *The Graduate* [12]. I liked that, so it became her name. In the original design, Elaine was a more ruthless Governor, but she softened up and became a true love interest as the project progressed.

» The last legal snafu we had with MONKEY ISLAND revolved around a “Look At” line in the voodoo shop. When you looked at a statue, Guybrush says, “Looks like an emaciated Charles Atlas.” We got a cease-and-desist letter for that and had to change it in future versions. I don’t know for sure what version it changed in.

» Each of the “actor” sprites had a set of basic animations that included standing, walking, talking, picking up, and reaching. If we needed an animation that would only be used in one place, it was called a “Special Case

Animation”—each one was carefully considered due to the five floppy disk limit [13]. Every pixel had to count. The first SCUMM game they appeared in was *THE LAST CRUSADE*, and we used them to an amazing (well, amazing for 1990) extent in *MONKEY ISLAND*.

» After acquiring the root, we didn’t make the player walk all the way back to the cannibals then all the way to the ghost ship. There was much discussion about this. I think we made the right decision.

» For the most part, MONKEY ISLAND is fairly open-ended. It gives players a lot of freedom to explore and solve several puzzle threads at a time, but there are two big choke points where the puzzles become very self-contained. The first is on the ship as you’re sailing to (or trying to sail to) Monkey Island. The second is while you’re on the ghost ship. All of the solutions to the puzzles can be found right where you are. These were done on purpose to give the player a small break and allow them to focus on one area.

» After you’ve made your way back to Mêlée Island, you are forced to kill two ghost pirates with root beer. This was important because it showed not only how the root beer worked, but that it would work.

» By design, the whole ending of the game is a “gimme.” The player has worked hard to get to this point, I wanted to give them something they could just sit back and enjoy playing.



» In closing, I should mention that I was always bothered by these close-ups [14]. While they were great art, I never felt they matched the style of the rest of the game. Still not sure how I feel about them 20 years later.

**RON GILBERT** helped create *MANIAC MANSION*, *THE SECRET OF MONKEY ISLAND*, and *MONKEY ISLAND 2: LECHUCK’S REVENGE*. He cofounded Humongous Entertainment and has most recently been working with Hothead Games on *PENNY ARCADE ADVENTURES* and the upcoming *DEATHSPANK*.



# GIVE SPREADSHEETS THE BOOT!



## DevTest Studio

The game industry's #1 choice for test management and defect tracking.

### DevTrack

Use DevTrack to track defects/issues

- Track each issue through a definable workflow
- SCM integration-track fixes against their source code deliverables
- Deploy a resolution across multiple releases, versions and products
- Reporting and metrics to illustrate the entire defect lifecycle

### DevTest

Use DevTest to manage your testing

- Create a central repository for your test cases, Knowledge items and automation scripts
- Schedule releases and test cycles using a wizard-driven interface
- Execute test assignments and submit defects from the same interface
- Track results with real-time dashboards and reports

### TestLink

Use TestLink to automate your testing

- Add automated tests to the DevTest test library
- Schedule automated tests along with manual tests
- Launch automated tests from the DevTest interface
- Track automation results with real-time dashboards and reports

Try DevTrack and DevTest live. Download free evaluation software. Watch a recorded overview demo.

[www.techexcel.com](http://www.techexcel.com)





# REPORT FROM THE SHOW FLOOR

## GAME DEVELOPERS CONFERENCE 2010



WE DIDN'T SEE AS MANY HYPE-FILLED ANNOUNCEMENTS AT THIS YEAR'S GAME DEVELOPERS CONFERENCE. INSTEAD, COMPANIES SEEM TO BE TAKING AN EARNEST LOOK AT DEVELOPERS' NEEDS IN ORDER TO REFINE WHAT WORKS AND PROVIDE THE INDUSTRY WITH THE MATURE TOOLSET IT REQUIRES TO GET THE JOB DONE. —JEFFREY FLEMING



Havok Destruction in action.

### HAVOK AI, CLOTH, AND DESTRUCTION

Havok  
[www.havok.com](http://www.havok.com)

Havok AI is Havok's latest addition to its portfolio and aims to solve developers' pathfinding and path following needs. "That's the core thing we're concentrating on," Dave Gargan, Havok's VP of engineering told us. "Because higher-level AI is so game-specific that developers want to keep those things for themselves to make their gameplay unique." The SDK provides automatic nav mesh generation that promises to create optimized navigation meshes for game levels within seconds. Along with nav mesh generation duties, the tool supports nav mesh

streaming for large game worlds that can be stitched together dynamically at runtime. Havok AI is designed for dynamically changing environments and gives NPCs a variety of optimized runtime queries that enable fast pathfinding decisions including line of sight checking, single- and multiple-goal pathfinding, as well as identifying potential cover locations. The tool features a local steering module that can handle large crowds, allowing characters to move and react naturally. "Characters understand how fast they can run and walk and they'll avoid moving objects by slowing down and speeding up," Gargan said.

Responding to the need for greater visual fidelity while reducing animator workloads, Havok Cloth

can be used to simulate the motion of clothing, hair, foliage, flags, and curtains. "People think of it as just garments but it can be used for dynamic skin deformation as well," Gargan noted.

Also on show was Havok Destruction, Havok's tool for simulating rigid body destruction such as shattering, splintering, crumbling, and deforming. "The hard part about simulating physical destruction is the cost that it adds to the game development pipeline. So we've focused on tools that let you do this very quickly in the model," Gargan said. 3D modelers can choose between automatic destruction algorithms that cover a variety of materials such as wood,

stone, or metal, or using the tool to handcraft their own break patterns. The tool can also generate extra debris at runtime for added realism. "With very little artist effort you can have these beautifully destroyed buildings in your game and you can do them without a huge burden on the graphics engine because you're procedurally instancing the same pieces of geometry around, so it's very efficient for rendering," Gargan added.

### MODO 401

Luxology  
[www.luxology.com](http://www.luxology.com)

Luxology was demonstrating its new Studio Lighting & Illumination Kit (SLIK) add-on for modo 401, a pre-built rig that simulates the lighting environment of a real-world photography studio. In addition to SLIK, Luxology has created a number of additional kits that simplify common modeling and rendering tasks such as high-dynamic range panoramas, render-ready background plates, and liquid effects. "It's not taking away or adding anything that a user couldn't do with enough time and knowledge. It is streamlining certain processes," Brad Peebler, president of Luxology said. "Everybody

has the same core modo, but if you want to learn a new discipline, you can add a kit, or you can use your own time to build it. Also, the system we set up to build and use the kits is in every modo. For example, Seneca Menard at id Software builds his own kits with customized UIs, scripts, and tons of content that they use internally for level development."

### AUTODESK DIGITAL ENTERTAINMENT CREATION SOFTWARE

Autodesk  
[www.autodesk.com](http://www.autodesk.com)

Autodesk updated Softimage with a variety of enhancements to its node-based Interactive Creative Environment (ICE) toolset. ICE Kinematics brings custom inverse kinematics, spines, and constraints to help in the creation of advanced rigging elements. New predefined compounds have been added to ICE that cover areas such as kinematics, deformation effects, particle emissions, and skinning. Support for PhysX 2.83 allows artists to use the NVIDIA PhysX rigid body library to create meshless deformations in ICE and also provides new support for springs and dampers. Softimage's



Face Robot facial animation toolset has also been enhanced with automated lip-synching capabilities.

Autodesk Maya

2011 sports a number of improvements including a redesigned user interface that is customizable and features dockable UI elements and improved editors. A new Vector Paint feature has been added to Maya Composite that improves painting and rotoscoping tasks. The skinning workflow for creating 3D characters has been made easier with a Paint Skin Weights tool redesign.

Autodesk 3ds Max has also been refreshed with a new node-based material editor. The software's underlying technology has been upgraded with a multithreaded hardware rendering engine that utilizes both the central processing unit and the graphics processing unit to provide significant speed increases. A high-dynamic range compositor that is based on technology from Autodesk Toxik has also been added to the tool.

## FMOD DESIGNER 2010

Firelight Technologies  
[www.fmod.org](http://www.fmod.org)

Firelight Technologies is readying FMOD Designer for a soon-to-be-released 2010 update. The new version of the sound design tool will feature extensive integration with Unreal Engine 3 that will allow designers to access their content in Unreal Script, Unreal Kismet, and Unreal. All FMOD disk reads, including streaming audio, will go through the Unreal IO, streamlining disk access tasks. Loading and unloading of event data and sound banks will be handled automatically by the C++ side of the integration which will eliminate the need for designers to manually

manage their memory usage during Unreal Script and Unreal Kismet sequences. Working in Unreal Script and Kismet, designers will also be able to directly manipulate FMOD events, categories, parameters, and music cues to create dynamic music scores and complex event behavior.

## KORE

Kore Virtual Machines  
<http://kore.net>

Scripting languages are fast becoming an essential component in game development. Seeing a need for fast and efficient virtual machines to embed scripting languages in, Kore aims to become the VM of choice for demanding console developers.

"We're focusing on the LUA language which is a fantastic, accessible, small, easily integrateable language, and we've re-written the virtual machine and the compiler," Hugh Reynolds, Kore's CEO

told us. "We see people using scripting languages in lots of different ways. We've seen some teams that have chosen to write all of their game in scripting language. You can produce a world-class product and it doesn't have to be bleeding-from-your-teeth samurai assembly. It can be other languages."

A key advantage that scripting languages provide over lower-level coding is the speed of iteration. "Five years ago, build time was the problem. Now, link time is the problem. You make any change whatsoever that requires just a relink and it's going to take five minutes," Reynolds said. "The half-life of an idea is 10 to 12 seconds. If a designer or a technical designer can get feedback within that time frame, they will stay there and they will iterate, and they will polish until they get it right. But if you've got a three- to five-minute link time on your game, then it's very hard to get a

productive flow. Anywhere that you have to iterate, we're going to see scripting languages more and more," he predicted.

## VISION ENGINE 8

Trinigy  
[www.trinigy.net](http://www.trinigy.net)

Trinigy's Vision Engine 8 has been updated with a host of new features including DirectX 11 and Shader Model 5.0 support, a water shader that enables streamlined creation of realistic water surfaces from rivers to oceans, a LUA remote debugger, and an extended audio system that supports additional sound formats and streaming.

Third-party support has been enhanced with an extended Havok Physics and Perforce integration. Vision 8 has been optimized to run on Intel six-core hyperthreaded processors and the engine's Console Resource Viewer has been extended to support Xbox 360, PlayStation 3, and Wii.

Also new to the engine is the Windows-only browser plug-in WebVision. "We see a rise in higher end casual games and that was definitely something we wanted to address. The idea is that people have the full capabilities of the Vision engine, the full feature set and the same tool chain that they use to make XBLA, PSN games, and large-scale boxed productions. They don't need to adapt to a new feature set to make browser games that reach a completely new level of visual quality due to the fact that we are running the whole Vision engine inside the browser," Dag Frommhold, Trinigy's managing director said.


Vision 8 offers a new post-processing framework that integrates with the engine's forward and

deferred rendering pipeline allowing developers to dynamically switch between the two. "We take care of all the switches between shaders. People can use this to extend our standard rendering pipeline regardless of whether it is deferred or forward with their custom effects and rendering features. The big advantage is that it allows developers to very easily, without having to modify any of our code or do anything from scratch, implement whatever rendering solution they want to have in there," Frommhold said.

## XAITCONTROL 2.6 AND XAITMAP 2.6

xaitment  
[www.xaitment.com](http://www.xaitment.com)

As part of xaitment's modular suite of AI tools, the company unveiled recent updates of xaitControl and xaitMap at the conference. xaitControl is a tool for designing hierarchical, probabilistic finite-state machines that uses a unique node-based graphical interface. By drawing states and connecting them with transitions, designers can intuitively create complex NPC interactions that can be grouped into hierarchies of sub-behaviors.

The tool's new graphical debugger also promises to make fine-tuning AI a less painful process. "This was previously a major concern for developers," Alexander Seifert, xaitment's field application engineer told us. "Especially for designers who were meant to script state machines. They typically had a really hard time reading through tons and tons of debug output just to figure out what had happened. Our graphical debugger makes it very easy to trace back what happens inside the state machine." 





June 15-17, 2010 \* Los Angeles Convention Center \* [E3Expo.com](http://E3Expo.com)



entertainment  
software  
association

# CREATIVITY UNLEASHED

© 2010 Entertainment Software Association

# E3 Expo 2010

E3 Expo is the world's most important annual gathering for interactive entertainment.  
It's where business gets done, connections are made and the future of the industry is revealed.

International Media  
Sponsor

**MCV**

Official Show Daily  
Sponsor

**GAMEPRO MEDIA**

**REGISTER NOW AT [E3EXPO.COM](http://E3EXPO.COM)**

Produced By

**IDG**  
WORLD EXPO

E3 Expo is a trade event and only qualified industry professionals may attend.  
No one under 17 will be admitted, including infants. Visit [www.E3Expo.com](http://www.E3Expo.com) for registration guidelines.



Supported by



European  
Games Developer  
Federation

gamescom

BIU

**GDC Europe returns to Cologne in 2010**

# GDC<sup>10</sup> Europe

Game Developers Conference® Europe

**August 16-18, 2010**

Cologne Congress Center East | Cologne, Germany

Visit [www.GDCEurope.com](http://www.GDCEurope.com) for more information



UBM





# PLANE-BASED DEPTH BIAS FOR PERCENTAGE CLOSER FILTERING

## CALCULATING A CUSTOM OFFSET TO REMOVE SELF-SHADOWING FOR LARGE PCF KERNELS BY FITTING THE UNDERLYING GEOMETRY TO A PLANE

A YEAR AGO, I DECIDED TO DIVE DOWN ON SHADOW MAPS AND WRITE A SAMPLE THAT ATTACKED THE most common artifacts. When I got to Percentage Closer Filtering (PCF), I found myself playing with offsets and slope-scaled depth bias to try to find a sweet spot that would satisfy the evil twins: Peter Panning and shadow acne. I had created a large outdoor scene to demonstrate how important it is to conserve depth buffer precision by setting the near and far planes as close as possible. However, this scene was so large that in the end, I could not find a combination of offsets that removed self-shadowing without making objects appear to fly.

The crux of this problem is that PCF is fundamentally flawed (see Figure 1). Comparing neighboring pixels in the depth map is invalid because these pixels correlate to different geometry. The solution that I found was to use derivatives to calculate custom offsets based on the orientation of the light with respect to the orientation of the geometry in screen space.

### SLOPE-SCALED DEPTH BIAS IS NOT SUFFICIENT

» As you can see in Figure 2, without a bias, pixels are as likely to erroneously fail the depth test as they are to pass. The rasterization hardware has to quantize the depth of a primitive for each pixel it passes through. A view-space depth is then tested against its corresponding texel in the depth map. This quantization error results in 50 percent of the pixels being erroneously self-shadowed.

Slope-scaled depth bias solves the problem for single-tap filters by adding a custom bias to the depth based on the orientation of the polygon with respect to the camera. In other words, the depth is scaled by the slope of the polygon. Slope-scaled depth bias does not remove self-shadowing artifacts when large PCF kernels are used because we're comparing neighboring pixels in the depth map against the current depth in view space.

### PERCENTAGE CLOSER FILTERING

» Filtering ordinary shadow maps does not produce soft, blurred shadows. The filtering hardware blurs the depth values, and then compares those blurred values to the light-space texel. The hard edge resulting from the pass/fail test still exists. Blurring shadow maps only serves to erroneously move the hard edge. PCF enables filtering on shadow maps. The general idea of PCF is to calculate a percentage of the pixel in shadow based on the number of subsamples that pass the depth test over the total number of subsamples.

Direct3D 10 and Direct3D 11 hardware can perform PCF natively in hardware while older hardware can do PCF filtering in shader code. The input to a PCF sampler consists of the texture coordinate and a comparison depth value. For simplicity, PCF is explained with a four-tap filter. The texture sampler reads the texture four times, similar to a standard filter. However, the returned result is a percentage of the pixels that passed the depth test. Figure 3 shows how a pixel that passes one of the four depth tests is 25 percent in shadow. The actual value returned is a linear interpolation based on the sub-texel coordinates of the texture reads to produce a smooth gradient. Without this linear interpolation, the four-tap PCF would only be able to return five values: { 0.0, 0.25, 0.5, 0.75, 1.0 }.

It is also possible to do PCF without hardware support or extend PCF to larger kernels. Some techniques even sample with a weighted kernel. To do this, create a kernel (such as a Gaussian) for an  $N \times N$  grid. The weights must add up to 1. The texture is then sampled  $N^2$  times. Each sample is scaled by the corresponding weights in the kernel.

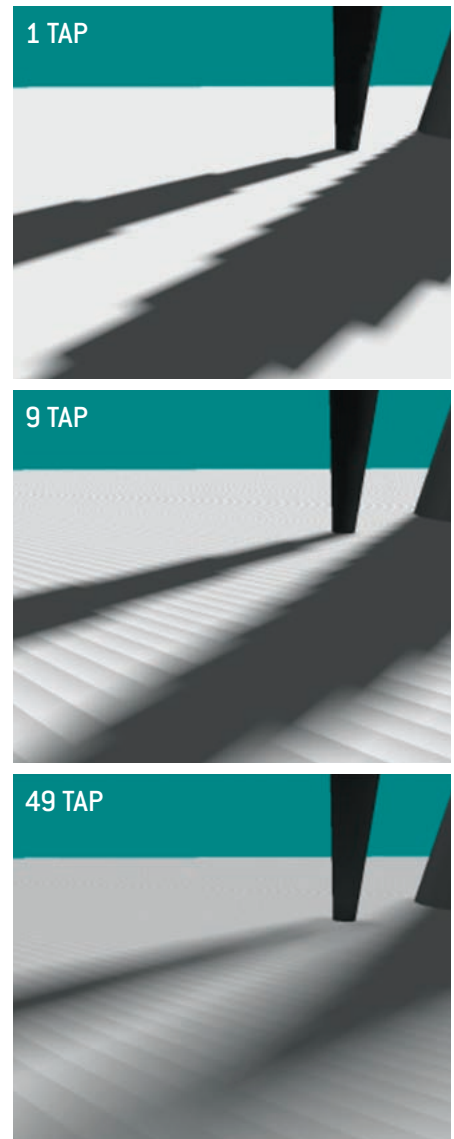


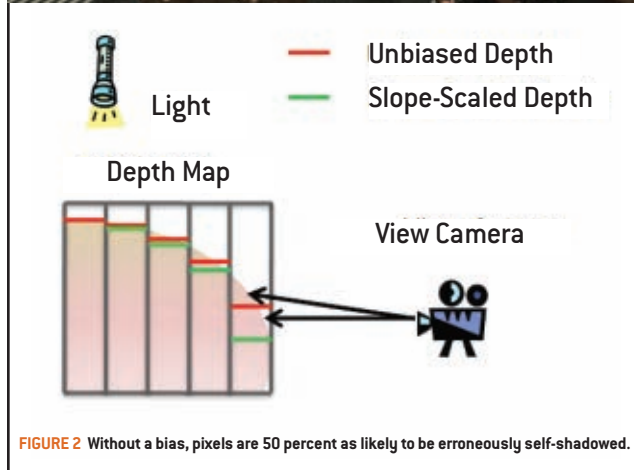
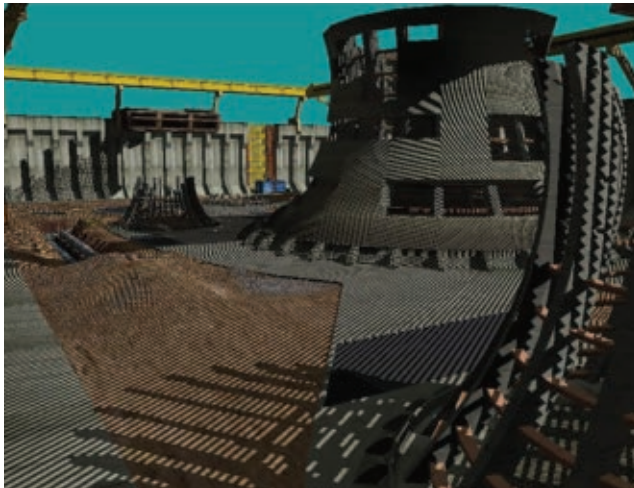
FIGURE 1 Large PCF kernels have problems with self-shadowing.

CONTINUED ON PAGE 36





CONTINUED FROM PAGE 35



## THE PCF FLAW

» It is only valid to compare a pixel's light-space depth against the pixel it maps to in the depth map. The depth-map texel's neighbors refer to a different position in post-projected view-space. This depth is likely to be similar, but can be very different depending on the scene. Figure 4 highlights the artifacts that occur. A single depth is compared to three neighboring texels in the shadow map. One of the depth tests erroneously fails because its depth does not correlate to the computed light-space depth of the current geometry. One solution to this problem is to use a larger offset. It can be frustrating to try to find the perfect offset that removes self-shadowing without causing Peter Panning.

## CALCULATING A PER-TEXEL DEPTH BIAS WITH DDX AND DDY FOR LARGE PCFS

» Calculating a per-texel depth bias with ddx and ddy for large PCFs is a technique that calculates the correct depth bias—assuming the surface is planar—for the adjacent shadow map texel.

This technique fits the comparison depth to a plane using the derivative information. Because this technique is computationally complex, it should be used in conjunction with deferred shadows or some other technique that skips regions that are clearly inside or outside of shadows.

Figure 5 highlights the problem. The depth in light space is known for the one texel that is being compared. The light-space depths that correspond to the neighboring texels in the depth map are unknown.

At a high level, this technique uses the ddx and ddy HLSL operations to find the derivative of the light-space position. This is nontrivial because

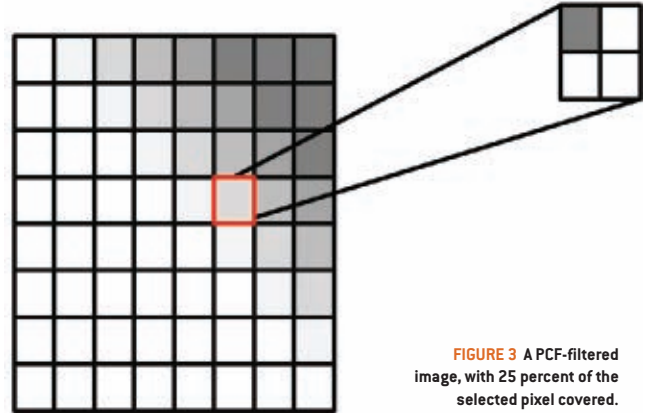


FIGURE 3 A PCF-filtered image, with 25 percent of the selected pixel covered.

the derivative operations return the gradient of the light-space depth with respect to screen space. To convert this to a gradient of the light-space depth with respect to light space, a conversion matrix must be calculated.

## EXPLANATION WITH SHADER CODE

» The details of the rest of the algorithm are given as an explanation of the shader code that performs this operation. This code can be found in the CascadedShadowMaps11 sample in the DirectX SDK. Figure 6 shows how the light-space texture coordinates correspond to the depth map and how the derivatives with respect to X and Y of these light-space coordinates can be used as a matrix to transform from screen space to light space.

The first step is to calculate the derivative of the light view space position as below:

```
float3 vShadowTexDDX = ddx
    (vShadowMapTextureCoordViewSpace);
float3 vShadowTexDDY = ddy
    (vShadowMapTextureCoordViewSpace);
```

Direct3D 9 class GPUs calculate these derivatives by running 2 x 2 quad of pixels in parallel and subtracting the texture coordinates from the neighbor in X for ddx and from the neighbor in Y for ddy. These two derivatives make up the rows of a 2 x 2 matrix. In its current form, this matrix could be used to convert screen space neighboring pixels to light-space slopes. However, the inverse of this matrix is required. A matrix that transforms light space neighboring pixels to screen-space slopes is needed:

```
float2x2 matScreenToShadow = float2x2(vShadowTexDDX.xy,
    vShadowTexDDY.xy);
float fInvDeterminant = 1.0f / fDeterminant;

float2x2 matShadowToScreen = float2x2 (
    matScreenToShadow._22 * fInvDeterminant,
    matScreenToShadow._12 * -fInvDeterminant,
    matScreenToShadow._21 * -fInvDeterminant,
    matScreenToShadow._11 * fInvDeterminant );
```

The Light Space to Screen Space matrix.

$$\begin{pmatrix} \begin{matrix} \text{ddx(LSP.x)} & \text{ddx(LSP.y)} \\ \text{ddy(LSP.x)} & \text{ddy(LSP.y)} \end{matrix} \end{pmatrix}^{-1}$$

CONTINUED ON PAGE 39

## LINEAR PCF

## POINT PCF

1 TAP



9 TAP



25 TAP



FIGURE 4 Linear PCF produces smooth gradients.





CONTINUED FROM PAGE 36

This matrix is then used to transform the two texels above and to the right of the current texel (listing 2). These neighbors are represented as an offset from the current texel:

```
float2 vRightShadowTexelLocation = float2( m_fTexelSize, 0.0f );
float2 vUpShadowTexelLocation = float2( 0.0f, m_fTexelSize );
float2 vRightTexelDepthRatio = mul( vRightShadowTexelLocation, matShadowToScreen );
float2 vUpTexelDepthRatio = mul( vUpShadowTexelLocation, matShadowToScreen );
```

The ratio that the matrix creates is finally multiplied by the depth derivatives to calculate the depth offsets for the neighboring pixels:

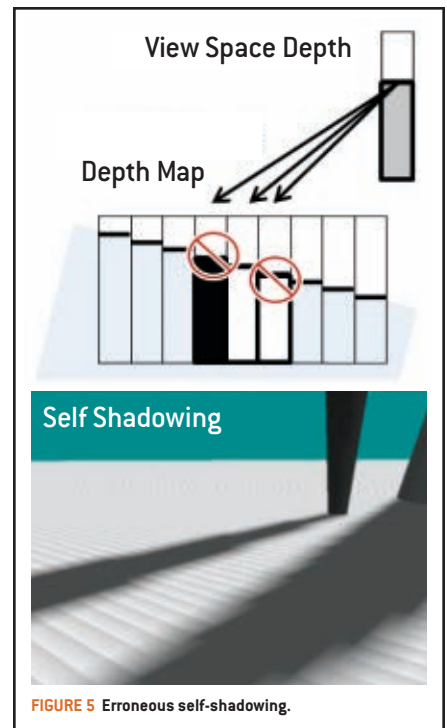
```
float fUpTexelDepthDelta =
    vUpTexelDepthRatio.x * vShadowTexDDX.z
    + vUpTexelDepthRatio.y * vShadowTexDDY.z;
float fRightTexelDepthDelta =
    vRightTexelDepthRatio.x * vShadowTexDDX.z
    + vRightTexelDepthRatio.y * vShadowTexDDY.z;
```

These weights can now be used in a PCF loop to add an offset to the position as seen here:

```
for( int x = m_iPCFBlurForLoopStart; x < m_iPCFBlurForLoopEnd; ++x )
{
    for(int y = m_iPCFBlurForLoopStart; y < m_iPCFBlurForLoopEnd; ++y)
    {
        if ( USE_DERIVATIVES_FOR_DEPTH_OFFSET_FLAG )
        {
            depthcompare += fRightTexelDepthDelta * ( (float) x ) +
                fUpTexelDepthDelta * ( (float) y );
        }
        // Compare the transformed pixel depth to the depth read from the map.
        fPercentLit += g_txShadow.SampleCmpLevelZero ( g_samShadow,
            float2(
                vShadowTexCoord.x + ( (float) x ) * m_fNativeTexelSizeInX ),
                vShadowTexCoord.y + ( (float) y ) * m_fTexelSize ),
            depthcompare
        );
    }
}
```

## EXISTING TECHNIQUES

» After implementing the technique in this paper, a colleague of mine sent me a link to a previous GDC session by John R. Isidoro. For the most part, Isidoro's technique is the same. His technique transforms the projected texture coordinates, while my technique uses the derivatives of the light-view space, which allows the technique to be used with cascaded shadow maps without performing derivatives in divergent flow control. Since Isidoro's GDC technique was revealed, deferred shadows have become commonplace. Deferring shadows enables computationally complex techniques for filtering to only be performed on the silhouettes of the shadows. Additionally, linear interpolation of

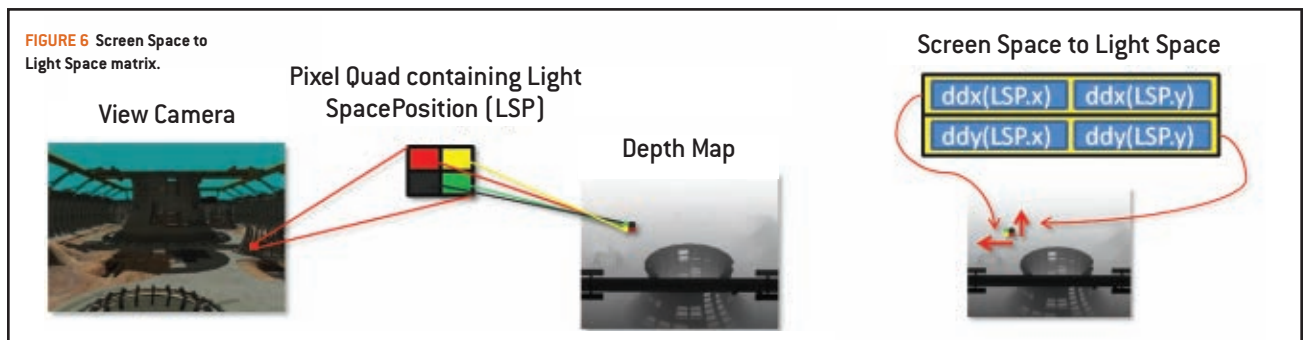


the texture coordinates used by the PCF filter can be used instead of point-based PCF filtering to give a smoother gradient.

## WRAP UP

» Correct shadows tell us when objects are hovering or securely planted to the ground. Shadows don't need to be correct, just believable. However, when too much offset pushes the shadow away from the base of an object, the eye is tricked and suspension of disbelief is lost. Additionally, the self-shadowing pattern that occurs from incorrectly comparing neighboring depth-map texels to the current view-space texel looks out of place. A plane-based depth bias mitigates these two artifacts.

**DAVID TUFT** works in the advanced technology group at Microsoft. In the past he has worked on Direct3D 11 and various GPU projects.





## BLIZZARD IS HIRING

We are actively recruiting across all disciplines for the following locations:

IRVINE, CALIFORNIA	AUSTIN, TEXAS	VELIZY, FRANCE	CORK, IRELAND
SINGAPORE	SHANGHAI, CHINA	TAIPEI, TAIWAN	SEOUL, SOUTH KOREA
SAO PAULO, BRAZIL	BUENOS AIRES, ARGENTINA	MEXICO CITY, MEXICO	

[jobs.blizzard.com](http://jobs.blizzard.com)

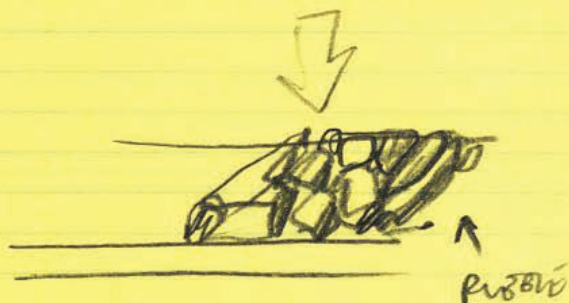
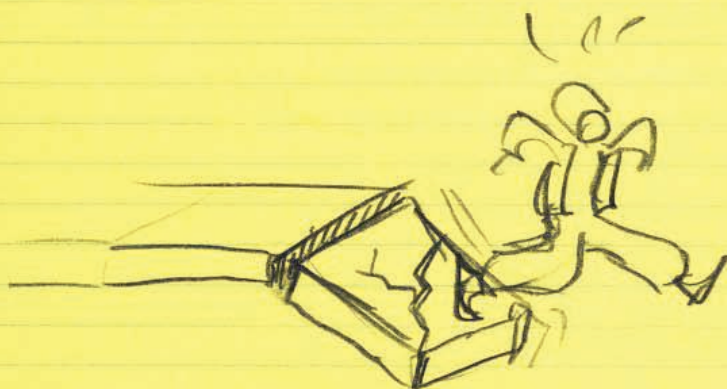


MAY 3, 1987 I'm back in work mode. Whatever the reasons, the long dry spell that began with Corey's and my exile to the attic ended the day Sensei moved in with us. I got a hell of lot done this week, and I'm actually starting to look forward to arriving at work every morning, sitting down at the Apple to make things happen.

4/87

## CRUMBLING BLOCK —

YOU STEP ON IT,  
IT BREAKS OFF  
& CRUMBLES.  
(POWERS UPKX)



LOOKS LIKE  
ANY OTHER  
BLOCK\*, BUT  
MAY NOT HAVE  
SUPPORTS OR POSTS  
(IN FACT, ONLY  
SPACE & FLOORBOARD)  
UNDERNEATH.

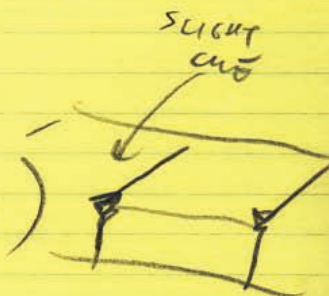
ONLY USED ONCE.

YOU CAN USE IT TO PASS, OR TO  
BREAK A FALL, BUT ONLY ONCE.

(WHEN YOU LAND ON IT,  
A HORRIBLE CRACKING SOUND \* (OR SHOULD IT?)

BEGINS. SIGNAL:

GET OFF FAST!)





# BLIND ALLEYS

## PRODUCTION SHOULD BE A TWO-WAY STREET

### "PIPELINE" HAS LONG BEEN THE GO-TO METAPHOR

for the magic that helps us turn hand-wave-y design docs and whiteboard doodles into finished game art. When you think about it though, a better metaphor for a production environment might be a city street map, particularly if the city is old and confusing, with lots of gnarly back alleys, meandering routes, and rundown neighborhoods where outsiders fear to tread.

Like a city, game production doesn't involve a simple linear flow—there's lots of data moving in various directions. You've got designers commuting in from the suburbs to drop off story notes and character ideas. There are engineers tearing up the streets to lay new pipes, blocking

renegade, bike messenger chic to it—if you don't get run over by the art director in his Escalade. And don't forget to wave "hi" to the tech artists on their Segways!

### MERGE AHEAD

» Seriously, the street-grid metaphor is a better way to understand what we do than the usual Visio diagram of pastel rectangles connected by arrows. Game studios, like cities, evolve and grow in surprising ways. Plenty of studios work in rhythms based on vanished technology, much like Boston's streets follow colonial cow paths and the routes of horse-drawn trolleys. Politics reroute real-world traffic to represent the balance of power; rich folks get nice smooth highways,

committing to a character skeleton, adding a UV map, and baking lightmaps are all examples of these one-way streets. Going down a one-way street is a serious proposition, because it involves many possible ramifications. Animating a character with bad proportions, or texturing on a badly done UV map means wasted work, schedule trouble, and the risk of shipping something lame. Revisiting these kinds of decisions is always expensive because new assets are constantly created that depend on the flawed originals; any fix will involve updating all the dependencies too.

One-way streets are a fact of life. Some decisions really are harder to unwind than others. You can twiddle a single texture all day in pursuit of artistic perfection without sending a tremor through your whole team's schedule, but if design wants to move all those buildings you painstakingly hand-stitched into the map terrain grid, there are going to be a lot of repercussions. Knowing how to spot the one-way streets in your production roadmap and handle them safely is one of the most important and least respected skills in game production.

### ONE WAY

» UVs, skeletons, and lightmaps are all classic examples of production milestones which are hard to revisit without serious costs, but there are other examples too. Carefully stitching geometry to make watertight volumes? Check! Distributing a complex character rig? Hell yeah! Normal casting? You betcha! Any decision which lays the groundwork for subsequent work is a potential one-way street. One-way streets are a fact of life—some decisions really do set the stage for others (it's hard to imagine, for example, animating a character that hasn't been given a skeleton yet). Even so, you can—and should—design your workflow to minimize the costs and likelihood of expensive do-overs.

One-ways are doubly difficult to work with. The rightness or wrongness of a one-way route is rarely obvious until new content comes later in the game. The flaws in a skeleton, for example, are usually revealed when the character's animation set is really taking shape—which is to say, alas, only after enough work has been put into the character that going back to fix the skeleton will be painful and costly.

Try to keep in mind that the first pass of any one-way process is the beginning, not the end of a process. Never build schedules



A game production is a lot more like a complex street map than it is like a pipeline.

off old routes and promising that the new bypass will shave minutes off the drive from Concept Street to Gold Master Boulevard. Deep in their underground bunker, the producers are playing with the stoplight timings, hoping to ease the traffic flow (while management keeps lobbying for new red-light cameras). The artists, meanwhile, are frantically pedaling their bicycles across town, trying to deliver their assets on time despite choking traffic, route changes, and an irrational street layout. Hunting for shortcuts and safe routes has some

poor folks get potholes and detours—just as our toolsets reflect the relative power of different departments. Most importantly, driving an asset from concept to completion—like getting cross-town at rush hour—requires a good map. All routes are not created equal; pick the wrong turn and you'll end up wasting a lot of time.

Every toolset includes some steps which are very hard to re-trace—the digital equivalent of one-way streets. As we've pointed out before (in past columns like "The Boneyard," March 2009, and "Most Likely To Succeed," November 2009),





on the assumption that laying down a skeleton or a UV map is something you do once and forget. Instead, try to push the asset as far as you can go in a loose, unpolished state first, so you can make sure that the one-way streets are really safe to travel. Build a few key cycles and run them on an untextured mesh and provisional skeleton before committing to months of animation production; play your level for a few weeks with low-res lighting before you get down to massaging the lightmaps by hand. You don't want to make big commitments of time and energy until your one-way street has been thoroughly surveyed.

Making this work demands trust between the art team and other departments. Nobody wants to charge down blind alleys, but artists are emotionally attached to polish work, and our clients in other departments often have trouble visualizing final quality from rough beginnings. The art leads need to be able to convince design and engineering that preliminary work really will be shippable before the game is ready to go. Resisting pressure to polish up assets too early is hard, but vital. It's also important to placate the frazzled egos of line artists who don't want any of their babies to go out the door looking less than perfect. The art staff needs to trust that they'll be judged on the final, shipped product and not on placeholders, tests, and trial balloons.

### OBEY POSTED LIMITS

» Another important tactic for dealing with one-way streets is clear signage. It's important that everybody, both inside and outside the art team, be well educated on which kinds of changes do and don't have big downstream costs. Say, for example, you've got a system for assigning parts of a model to different hit regions so the game can translate a hit on the head into a headshot at runtime. This is a small example of the one-way street problem, since odds are whatever region modifiers you paint onto your character won't survive serious mesh edits.

Naturally, the designers will want you to add this kind of markup right away when the character is coming online. It's important, though, that they get some education on the maintenance costs of adding this kind of detail early. Grabbing a few face triangles and hitting a button might not take that long to do once, but because this happens on the wrong end of a one-way street, that little extra increment of work will be added on to every future revision of the character.

Mesh and skeleton-dependent markups always create a one-way route, so adding this kind of data should be put off as long as possible. Perhaps you can convince the designers that a system of hitboxes attached to the skeleton will work as well as per-face control, or generate the data off of the ragdoll. Or, if per-face control really is needed for gameplay, is it certain that it has to be added early in the dev cycle? Just postponing the markup step until polish time will remove a lot of schedule pressure. In the worst-case scenario, if this fragile bit of data really must be available as soon as the character is in the game, talk to engineering or tech art about tools for preserving the markup between model edits (and get the designers to back you up!). The key goal is to keep the iterative part of developing the character as cheap as possible, and to add one-way dependencies only after the character is in its final form.

### CONSTRUCTION AHEAD

» Of course, the best way to deal with one-way streets is to eliminate them altogether. For example, if your production is consistently handcuffed by the costs of offline lightmapping, maybe it's time to consider an all-dynamic lighting system. It's certainly true that the technical quality of dynamic lights and shadows isn't a match for the gorgeous effects you can achieve with an offline radiosity solution and a big render farm. Even so, when artists get to iterate—even on a system that's short on bells and whistles—the results can compete well against technically perfect systems that are too slow and cumbersome to tweak. More importantly, should you discover that one of the firing spots in your level is useless due to sun glare, you'll be able to fix it instead of shipping useless content because you don't have a week to manually re-do all the lightmap UVs. Naturally, a lot depends on the nature of the game—competitive multiplayer games need to be more flexible; cinema-driven, story-based games can do more up-front planning. In any case, though, it's always valuable to check your assumptions once in a while.

There are lots of similar situations where “the best” really is the enemy of “good enough.” Quick procedural content isn't as good as carefully crafted art—but it can be a huge timesaver for early iterations. For example, a smart artist can build better and more efficient collision models than the automatic convex-hull generators that come with Havok or PhysX. But is “better” enough to justify the cost? Automatic tools are often fine for anybody except the artist who built the asset in the first place. In the heat of gameplay, when the polygons are flying, who will really notice that the tailpipe on your go-kart clipped through track walls by an inch as it tumbled out of control?

Using automatic tools to do a first pass or to generate placeholders is a great way to cut down on the costs of one-way decisions. By making the iteration cheaper, they allow you to make really important fixes to the look and feel of the game without busting schedules. By all means, replace


the computer-generated filler with beautifully tweaked hand-built hulls—just tolerate the filler until you're certain the content is locked. Besides saving wasted iteration time, you'll be making the resource calls at the end of the project when the tradeoffs really stand out in stark relief instead of blowing your budget on minutiae when the project is young and the team is overly-optimistic.

### DEAD END

» The last, and perhaps most important survival tip for navigating one-way streets should be obvious: Iteration is good, one-ways are bad—so don't create one-way streets of your own. It's surprisingly easy for a hastily designed tool or process to add hidden iteration costs. Be vigilant and don't let new features (even ones you really want!) drag down your agility.

Imagine you've finally convinced your engineers to add tweakable animation compression. Hurrah, at long last, artists have some fine-tuning control over the animations! No more stuttery pantomime! Still, don't let excitement over the new tech lead you down a blind alley for production. It seems natural to add compression right into the exporter; it's probably what the engineers will offer to do. Unfortunately, doing the compression there means that a last-minute change in the budget (or, for that matter, in the compression technology!) requires you to re-open and re-export every existing animation file. It's infinitely better to run the compression in a separate tool so you can experiment, make bulk changes, and react to last-minute resource crunches without spending massive amounts of time revisiting old work.

It's easy to get carried away by enthusiasm when new tech comes along. Even the swankiest new effects need to fit into our overcrowded, chaotic schedules. Make sure to stand up for flexibility and iteration whenever a new tool or trick is on the table. If you don't, you'll pay for it later.

Flexibility is an easy thing to praise and a hard thing to achieve (if you don't believe it, try taking a few weeks of Pilates). The complex web of interdependencies that makes up a game is never going to look like a rational, streamlined assembly line. Maybe we don't want it to—creativity and experimentation aren't usually found on the factory floor. However, we can certainly make our own lives easier, and make better art, by making sure that we—and our teammates—know the real costs of the decisions we're contemplating. We may never get satellite navigation for our crazy, constantly changing environment, but we can certainly learn to read the street signs and keep out of dead ends. 

---

**STEVE THEODORE** has been pushing pixels for more than a dozen years. His credits include *MECH COMMANDER*, *HALF-LIFE*, *TEAM FORTRESS*, *COUNTER-STRIKE*, and *HALO 3*. He's been a modeler, animator, and technical artist, as well as a frequent speaker at industry conferences. He's currently a consultant helping game studios perfect their art tools and pipelines.

FROM THE CREATORS OF "SCRIBBLENAUTS", "DRAWN TO LIFE" & "LOCK'S QUEST"

WORK ON SOMETHING

**TRULY ORIGINAL**

Now hiring for Xbox 360:

Lead Environment Artist

Senior Environment Artists

Lead 3D Level Designer

Senior 3D Level Designers

Lead Animator



**5THCELL™**

Visit [5thcell.com/jobs](http://5thcell.com/jobs)





# WIN EXPECTANCY

## HOW PLAYERS PERCEIVE THE ODDS

**IN 2004, THE RED SOX WERE DOWN 4–3 IN GAME FOUR OF THE AMERICAN LEAGUE CHAMPIONSHIP** Series in the bottom of the ninth inning. Their hated rivals, the Yankees, had won the previous three games and were three outs away from going to the World Series. But in one of the most thrilling championship series in all of sports, the Red Sox managed to tie the game in the ninth inning, win the game in the twelfth, and proceed to win the ALCS and then the World Series. What are the odds?

Thanks to a bunch of fans, we know exactly what the odds are. Baseball as a sport lends itself to statistical analysis—a hobby advocates call Sabermetrics. Unlike other team sports, baseball has a relatively low number of variables—it really comes down to the duel between the pitcher and batter—so it's easy to quantify the value of any given player in any given situation. There are few parts of the game that have not undergone statistical analysis.

So when we ask, “What are the odds the Red Sox would win game four?” we know the answer exactly. In the history of all baseball games ever played, a team that has been down one run with no outs at the bottom of the ninth inning has won 23 percent of the time. When Kevin Millar got his walk, that percentage jumped up to 37 percent. One stolen base later—meaning there was a runner on second with no outs—the odds shot up to 47 percent. When Bill Mueller knocked him home to tie the game, the odds that the Red Sox would finally end their accursed run of bad luck against the Yankees shot up to 73 percent. There are two takeaways for game designers. The first is that the odds of victory (what Sabermetricians call “win expectancy”) can absolutely be quantified. There are, in fact, iPhone apps that you

On top of all this, a win expectancy that is too high may indicate a game that is ultimately not very interesting. A game where victory is effortless is one where the player fails to get emotionally invested and, as such, it's very easy for him to put the controller down.

### THE ILLUSION OF THE UNDERDOG

» We want players to feel like they are riding on the edge of failure, without actually putting them there. This isn't insurmountable—movies do it all the time. Cognitively, you know that Luke is going to blow up the Death Star, but in the midst of the roller coaster ride, you're fooled.

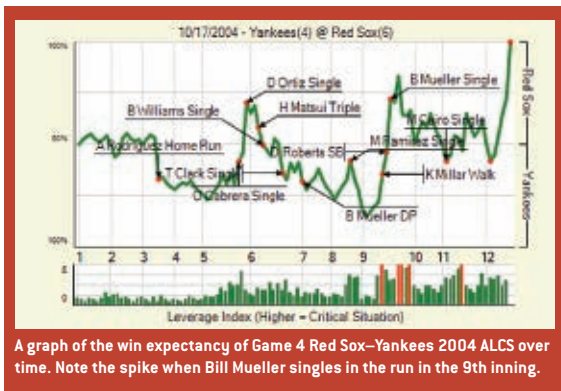
One easy way is to conjure the illusion of toughness—most commonly, this is done with puffery. Scaling up the boss in size, giving him an intro speech and extremely showy particle attacks, or even doubling the size of his health bar can all instantly create the sense that you're expected to fail. Another method is to simulate attrition, even fake attrition. *Star Wars* does this by winnowing down Luke's wingmen one by one. In games, this can be done by knocking off a player's armor, breaking his weapon, or killing a companion.

Most games, like the *GOD OF WAR* series, set up boss fights with a good amount of easily dispatched enemy fodder that can easily be overcome. Just as a low win expectancy can make the resulting victory that much higher, having an easy baseline experience can make a boss feel much tougher than he actually is.

Playing with the player's health bar is another way to create an illusion that the player is closer to death than he actually is. Potions in *DIABLO* allow the player to see a health bar that empties much faster than he normally expects, while still giving him a statistical edge to recover. The health bar can also unlock functionality: Tanking Paladins in *WORLD OF WARCRAFT* get access to a talent that results in them taking less damage when they are below 35 percent health (which helps put them in the “danger zone” more often), and certain fighting games, such as some entries in the *KING OF FIGHTERS* series, give players access to super attacks that are only available if they are below a certain health threshold.

### PLAYER VS. PLAYER ACTIVITY

» These days, most single-player games are designed to let the player win. They might have an overall win expectancy for most of their content at about 95 percent. However, this is not the case in player vs. player scenarios. For example, in a one-on-



A graph of the win expectancy of Game 4 Red Sox–Yankees 2004 ALCS over time. Note the spike when Bill Mueller singles in the run in the 9th inning.

can take with you to determine your team's win expectancy at the ballpark. The second, more important takeaway is that the low win expectancy of the Red Sox (that 23 percent I mentioned) is what made the game great. It's a game that is still discussed today, out of hundreds of championship baseball games that have been played. It is the underdog triumphing over long odds.

### GAMES AND WIN EXPECTANCY

» Movies play with win expectancy all the time. When Darth Vader is trailing Luke Skywalker in the trench near the end of *Star Wars* and breathes, “I have you now,” you feel like Luke's chances are hopeless (perhaps Sabermetric geeks would say sub-10 percent). Then Han Solo comes in like a cowboy and clears Luke for the shot. Han's appearance is epic, but it is the emotional low beforehand that makes his appearance so fist-pumping.

Overcoming great odds is just one of the scenarios that can provide an emotional high to the player, one that can make a gaming experience even more memorable and compelling. However, game designers have some real difficulties to overcome when attempting to manipulate win expectancy. The first is the role of player skill: A 90 percent win expectancy for a seasoned *NINJA GAIDEN* player may plummet to 10 percent for a casual player.

The more pernicious problem is that underdogs lose most of the time. The Red Sox–Yankees game in 2004 is unusual because the Red Sox statistically should have lost. But if a boss fight has an actual win expectancy of 10 percent, this suggests that the player should flat-out fail 9 attempts out of 10. In practice, most players will throw their controllers through their flat screens in frustration. Still, many who have finished a boss fight with a sliver of remaining health feel the resulting emotional peak is worth some frustration along the way.



The Leviathan in GOD OF WAR III. Turns out, he just looks big.

one STARCRRAFT match, a player's chances of winning start at 50 percent, and that is with the assumption that the players are evenly skilled. Add more contestants, such as in a QUAKE deathmatch or a large game of *Risk*, and the individual's chances drop with each additional contestant. This is especially problematic in games where both kinds of content exist. A PvE [player vs. environment] player in an MMO like WORLD OF WARCRAFT or WARHAMMER ONLINE who is used to winning 95 percent of the time may have trouble adjusting to the sudden increase in failure when playing PvP [player vs. player].

This is one of the reasons that team gameplay is important—having only two teams in capture the flag mode means a QUAKE player is much more likely to experience victory than in a deathmatch. Furthermore, to some degree, it disguises a player's own contribution to failure. If you lose a one-on-one match, you have no one else to blame. If you lose a four-on-four match, that sense of failure is ameliorated. And sometimes, a low-skill player will experience victory by osmosis.

Calculating win expectancy in situations with asymmetric power can be extremely complex. In the board game *Illuminati*, the Gnomes of Zurich are by far the most powerful Illuminati faction, but according to Steve Jackson, they don't win disproportionately often. It turns out that most other players recognize that power and gang up on the hapless sap playing the gnomes.

## THE ILLUSION OF CLOSENESS

» The polling for the presidential election of 2004 was never even close—nearly every poll in the election gave George Bush leads over John Kerry. This was even more true in 2008; other than a short spike around the Republican convention, Barack Obama had a convincing lead over his opponent John McCain from the moment he secured the nomination. In both cases, the frontrunner played defense, avoided making any mistakes, and secured the election with relatively little doubt.

You wouldn't know that from the election coverage. Every news story made it clear the underdog was within arm's reach of the frontrunner, and that the election was anyone's game. Close elections sell more newspapers than blowouts. The candidates were complicit, too. The frontrunners wanted the election to feel close to create pressure on their voter base to turn out. The underdogs have to seem to have a fighting chance if they want to attract donations and volunteers.

Close matches are compelling because the expectations around a contest remain in doubt. In blowouts, there is precious little doubt, and the contest ceases to be interesting. Ensuring that there is doubt, that there is always an angle for an underdog to get back into the thick of things, is a time-honored design principle. Consider MARIO KART: there are a lot of power-ups in the game that allow you to attack other racers on the track, but by and large most of them fire forward. One well-aimed tortoise shell can knock a frontrunner to the back of the pack.

Having slightly opaque game mechanics can also create this doubt. In *Settlers of Catan*, the player may purchase cards that offer secret victory points. The leader may have a convincing lead over other players, but anyone who has cards in his hand may actually be able to surprise the leader and steal victory, which maintains interest in the game.

## THE FUTURE OF WIN EXPECTANCY

» Game designers are already becoming more and more sophisticated about building a good win expectancy cadence in their experience, ensuring that players alternate between assured victories and more challenging content. I expect we will see more sophistication of this nature in the future.

Already, games are starting to notice that players are failing more often than expected and asking them if they want to adjust their game's difficulty on the fly. In the future, more will start

doing this behind the scenes—to have AI determine the worthiness of the foe and adjust automatically. This is already commonplace in racing games like KART RIDER, although designing systems to do this without cheapening the accomplishments of your more hardcore players takes more design thought than you might expect. LEFT 4 DEAD similarly uses an "AI Director" in order to provide an appropriate level of challenge to whichever four players have found their way into a match.

Game mechanics like Xbox Achievements and Boasts in FABLE are already helping players to make this choice manually. More casual players can just play the game as presented and experience a default win expectancy that was balanced for them. Hardcore players who can blow by that challenge as-is can choose to make the fight harder: in FABLE, players can boast they are capable of defeating a boss unarmored, or without taking damage. In WORLD OF WARCRAFT, players can earn achievements for killing bosses without losing any raid members, within a time limit, or in other ways that provide hardcore players a challenge without bending the difficulty curve to a place where more casual players can never conquer it.

Surprising players is important—they need to expect the unexpected. But it is equally important for game designers to cater to what players do expect. Doing so will give the players greater understanding of their own failure, and increase their satisfaction when they succeed. Win expectancy is a powerful tool, and designers should be willing to cheat and manipulate it in order to provide the appropriate sense of challenge and tension in their game experiences. 🎮

**DAMIAN SCHUBERT** is the lead combat designer of STAR WARS: THE OLD REPUBLIC at BioWare Austin. He has spent nearly a decade working on the design of games, with experience on MERIDIAN59 and SHADOWBANE as well as other virtual worlds. Damian also is responsible for Zen of Design, a blog devoted to game design issues.



VVoh!N[D]3r

Join the people behind the pixels in the world capital of digital imagination.

The People Behind the Pixels

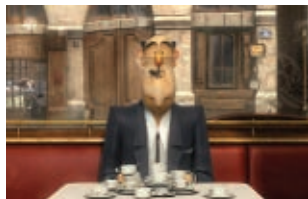
# SIGGRAPH2010

## Los Angeles



share ideas   build relationships   advance your career   be inspired

French Roast © 2009 Fabrice O. Joubert, The Pumpkin Factory; Touch the Invisibles © 2009 Junji Watanabe, PRESTO Japan Science & Technology Agency



The 37th International Conference and Exhibition on Computer Graphics and Interactive Techniques

**Conference 25-29 July 2010   Exhibition 27-29 July 2010**

Los Angeles Convention Center   Los Angeles, California   USA

Art Gallery • Art Papers • Birds of a Feather • Computer Animation Festival • Courses • Emerging Technologies  
Exhibition • Exhibitor Tech Talks • Featured Speakers • Game Papers • International Resources • Job Fair • Panels  
Posters • Research Challenge • SIGGRAPH Dailies! • Talks • Technical Papers • The Studio

FOR COMPLETE DETAILS:

[www.siggraph.org/s2010](http://www.siggraph.org/s2010)

SPONSORED BY ACM SIGGRAPH



FOLLOW US ON:





# THE GLOBAL ASSET LIST

## DATA INFRASTRUCTURE FOR THE LITTLE GUY

**BIG DEVELOPMENT STUDIOS** function because of their infrastructures. FTP servers, T1 connections, and networked source control software are commonplace tools used to connect studios to their global development partners. These days, big studios aren't the only ones with global partners. Freelance composers and sound designers can make use of the Internet

script of in-game dialogue, the last thing you want is for critical data to become unsynchronized. To combat this, a number of online document services are available.

The two main services are Microsoft Office Live and Google Docs. Both allow users to upload existing Microsoft Office documents. Once uploaded, documents can be viewed, edited, and

can set up user privileges just like Office Live, and the tool sticks to Google's simplistic UI aesthetics. Unlike its Microsoft competitor, though, there is no version control. As such, audio leads using Google Docs may want to ensure that they routinely download and archive evolving versions of their documents manually.


A note for freelancers on the go: neither Google

orchestral recording. Two composers were located in rural northern California, two in San Francisco, and one in Los Angeles. Each composer had his own orchestration team with members spread across California. Additionally, a number of cues were further outsourced to ghostwriters across the country. The lead composer created a Google doc to track the composition status of each track and

without having to skip any cues. As we moved into post-production on the score, columns for mixing and mastering status were added to the same central document. Trying to tackle this project with a non-networked spreadsheet would have instantly become a version control nightmare.

### A WORD OF WARNING

» The security of online document services is nebulous. As such, IT departments of large developers and publishers will likely be opposed to their use. With the risks of cyber attacks, network outages, or the simple vagaries regarding what information is captured and shared through online services, IT fears over the leaking of confidential or proprietary data are well founded. Smaller studios and indie developers may not have the luxury of an IT department, but their contractors will still need ways to share data and communicate in sync with each other across the globe.

With the gulf in price between secure data sharing systems and free online alternatives, the chances are good that—IT-sanctioned or not—online document solutions outside of the control of corporate IT departments will continue to gain ground as part of the standardized suite of modern development tools. 

**JESSE HARLIN** has been composing music for games since 1999. He is currently the staff composer for LucasArts.

If a Pro Tools prelay or instrumental part was missing, we were instantly able to call up its status on the spreadsheet, track down who was responsible, and find the missing asset without having to skip any cues.

for collaboration with team members from Los Angeles to Louisiana to London. Large developers hire smaller contractors who can subcontract to any number of satellite team members. Thankfully, a slew of online tools exist to provide anyone with a solid infrastructure comparable to those of the largest development houses.

### INFORMATION CENTRAL

» One of the largest hurdles facing small networks of freelancers is version control. Simultaneously emailing Word or Excel files to a number of recipients is a prime recipe for allowing critical data to get out of sync. When coordinating cues between a team of orchestrators, tracking sound effect iterations between sound designers, or maintaining a changing

shared with a vast number of users—100 for Office Live and 200 for Google Docs. Security groups can be created so that users have editing or read-only privileges. As such, anything from design documents to asset lists can be created, managed, and maintained in one central location.

Microsoft Office Live has a familiar web interface and a clear UI. Office Live also makes use of version control and options that allow users to track changes. New documents can be easily created and shared within the browser. However, creation and uploading of Office Live documents requires installation of Microsoft Office.

Google Docs doesn't require the downloading or installation of any additional software. Document owners

Docs nor Office Live works completely with smart phones. Until specific smartphone-centric applications are created for these services, expect limited functionality and the potential for lost data.

Google Docs and Microsoft Office Live aren't the only options available. Other products such as Zoho.com or ThinkFree.com fill the same niche. However, it should be kept in mind that less established companies always run the risk of being eaten by larger companies. EtherPad.com was a popular alternative until it was bought by Google and absorbed into Google Docs.

### DATA IN ACTION

» On a recent project, I supervised the work of a five-composer creative team preparing for six days of

shared it with the entire team. The doc had columns for composition status, final file names, cue assignments, whether approved cues were orchestrated or not, if parts were printed and shipped to the stage, and lastly, a "Notes" column where we could communicate centrally without the fragmentation of email.

As we recorded, we had a laptop set up in the control room that updated the "Recorded" column of the doc at each break. Rather than dealing with printouts or handwritten lists, all our project-long documentation was at hand any time we needed it. If a Pro Tools prelay or instrumental part was missing, we were instantly able to call up its status on the spreadsheet, track down who was responsible, and find the missing asset





# YAMAOKA'S HETEROPHONY

## SILENT HILL COMPOSER JOINS GRASSHOPPER MANUFACTURE

*As reported in these pages in January, SILENT HILL composer Akira Yamaoka has left Konami—and now he's turned up at NO MORE HEROES developer Grasshopper Manufacture. We caught up with Yamaoka to ask about the switch.*

**BRANDON SHEFFIELD:** *What made you decide to go to Grasshopper Manufacture?*

**AKIRA YAMAOKA:** Well, Grasshopper is one of those Japanese companies that makes very original games, even by global standards. Their games always have a very unique worldview, and with the sort of toolset I had in the fields of music and audio, I thought I'd be able to contribute to what they're doing in a really constructive way. That was what inspired me. They have the sort of projects going on that I want to be involved with, and that was a big aspect behind the decision.

**BS:** *How do you think your personal style will fit into the Grasshopper style? There's a lot of innovative sound work there—will you be managing that as well, or will you only be composing?*

**AY:** I think I could do it either way. Of course, when you're talking about "style," it's not like I'm going to demand a particular type of music or a kind of output in our games' audio. We think in terms of what the game requires; it's not a matter of me needlessly pushing my perspective into projects.

**BS:** *I notice that you don't often use symphonies, or a lot of orchestration. Why is that? Do you think that'll change in the future?*

**AY:** I do think that a change of pace in that respect could be a good thing. I think that, but ... going back to the first question for a moment, whether you're using a symphonic or heterophonic approach in Asian or Western music, first you have a rhythm—beating out time in one way or another. To that you add harmonies, the chords that are played at set times to this rhythm. Melody is the thing that puts these chords together. In heterophony, you think of each aspect

as being on its own axis; you come up with melodies, then you come up with the rhythm and harmonies to make them work. Western music often begins with the composer thinking about harmonies first, but within music, there are only a limited number of harmonies (about a hundred or so) that are suitable to use. It's a matter of combining those together. I thought about this for a bit once, and I'm not saying that one is better than the other, but for example, the English alphabet has 26 letters, while there's a practically infinite number of kanji characters—not infinite, but quite a lot.

So in English, you take this small set of characters and form words like "god" and so on with them, but playing with these same letters can give you very different results, like "dog." The Eastern line of thought simply has a character, "kami," for god, and then a completely different and unrelated character to signify dog. In much the same way, Western music often takes this given set of chords and comes up with new and novel ways to combine

them together, while Japanese heterophony is more concentrated around melody. So thinking symphonically is actually pretty difficult for me—it's not like I couldn't do it if I studied a bit, but it's tough. I am interested in symphonic composition, though, definitely.



PHOTO BY VINCENT DIAMANTE

## new studios

2D Boy co-founder Kyle Gabler (WORLD OF GOO) and Experimental Gameplay Project cohort Kyle Gray (HENRY HATSWORTH) have founded the "authentic indie labor"-powered developer Tomorrow Corporation. The duo has teamed up with fellow Carnegie Mellon University's Entertainment Technology Center graduate Allan Blomquist, who contributed to WORLD OF GOO. All three developers are also co-founders of the influential rapid prototyping-centric Experimental Gameplay Project.

Renowned RESIDENT EVIL creator Shinji Mikami plans to launch a new Tokyo-based studio, Tango, as soon as his current project wraps. Mikami is currently at work on Platinum Games' shooter VANQUISH.

Following the closure of Melbourne, Australia-based HEROES OVER THE PACIFIC developer Transmission Games last year, former employees of the studio founded Trickstar, which claims to have a number of games in production.

## who went where

MONKEY ISLAND creator Ron Gilbert has left Hothead Games, the studio at work on his upcoming DEATHSPANK. Gilbert says that now that the game's creative and production phase has wrapped, he's finished with his work at the studio.

UK-based development conglomerate Kuju Entertainment (CHIME, HOUSE OF THE DEAD: OVERKILL) will see co-founders Ian Baverstock and Jonathan Newth leave their current roles for non-executive director positions, as a new CEO, Nigel Robbins, steps in.

David Reeves, formerly the longtime president and CEO of Sony Computer Entertainment Europe, has joined Capcom as chief operating officer of the company's European arm. He had spent 14 years at Sony.

Activision chief financial officer Thomas Tipll will become the publisher's chief operating officer, continuing to hold the CFO role until a replacement is hired to fill that spot. This busy gentleman was also previously CCO.

After spending 15 years with Electronic Arts, executive Steven Chiang has left the publisher to join social gaming firm Zynga (FARMVILLE) as the president of its development studios.



**ARE YOU READY TO EXPLORE  
THE MOST SOUGHT AFTER JOBS IN ENTERTAINMENT?**

**Visit our site: [www.activision.com](http://www.activision.com)**





# SCORE YOUR DREAM JOB AT GAMELOFT.

AT GAMELOFT, WE ALWAYS  
KEEP OUR EYES OPEN FOR  
FRESH TALENT.

WE EMPLOY CREATIVE  
AND AMBITIOUS:

- \* 2D & 3D ARTISTS
- \* GAME DESIGNERS
- \* PROGRAMMERS
- \* PRODUCERS
- \* AND MORE!



[GARGANTIAN GAMERS, DARING DAY DREAMERS, HILARIOUS HIPSTERS,  
SUPER-DUPER SMARTY PANTS, WONDROUS WHIZ KIDS, BRILLIANT  
BRAINIACS AND CLOSET SUPER HEROES ALL NATURALLY, WELCOME]

For more info about specific job opportunities in Barcelona, Buenos Aires, Montreal, New York City and Shanghai (Redsteam), please visit the career section at [www.gameloft.com](http://www.gameloft.com)

We will contact you if we believe your experience is a good fit for an open position,  
but if you don't hear from us right away, don't fret. You might be our next hire!

**gameloft**  
[www.gameloft.com](http://www.gameloft.com)





# gearbox<sup>®</sup> software CAREERS



**Join the Team!\***

## —HAPPINESS—

- Be part of a Passionate Team
- Career Not a Job • Make Your Path
- Achieve Your Goals
- Free of Earthquakes and Wildfires!

## —CREATIVITY—

- Independent and Unbound
- Dream Projects & Original Games
- Collaborate with Top Talent
- Make Meaningful Contributions

## —MONEY—

- We Ship Bestsellers & Award Winners
- Over \$14 Million in Profits Paid Directly To Our Talent
- Exceptionally Low Cost of Living • Get Paid to be Passionate about Work

\*The Awesome are welcome here.

### **Gearbox Original Games**

Brothers in Arms • Borderlands

### **Games from Gearbox<sup>†</sup>**

Halo • Half-Life • Aliens: Colonial Marines  
Tony Hawk • Samba de Amigo  
• 007: Nightfire

**Visit [gearboxsoftware.com/jobs](http://gearboxsoftware.com/jobs)**

<sup>†</sup>All games and franchises listed are the property of their respective owners and may be trademarked in the United States and other countries. All inquiries regarding such trademarks should be made to their respective owner. All inquiries regarding Skagzilla specimens and Rakk Hive husbandry should be directed to the appropriate zoological authorities.





## EDUCATED PLAY!

# BORYOKUDAN RUE

## GAME NOIR

**THE MASS MARKET MAY** have passed the adventure game genre by in recent years but the form continues to attract a small cadre of designers and players who crave a well-told story. Joshua Nuernberger's 2010 IGF Student Showcase finalist BORYOKUDAN RUE is a new game with a compelling sci-fi noir setting that reaffirms the timeless appeal of 2D point-and-click adventures.

**Jeffrey Fleming: How did you like working with Adventure Game Studio to create BORYOKUDAN RUE?**

**Joshua Nuernberger:** Working with AGS is like working with an old friend. It can pretty much do everything you need it to do as far as point-and-click adventures resembling MONKEY ISLAND or any other early '90s adventure games are concerned, and it continues to improve over the years and with a great community to boot. So I've never seen the need to go on to look for other methods of game creation.

**JF: In addition to the AGS forums, have you found other online resources that have been useful to BORYOKUDAN RUE's development?**

**JN:** Pixeljoint.com is always a favorite source of inspiration, in addition to other art-related sites or blogs. The musings of other game developers—professional or otherwise—such as Ron

Gilbert or Ben "Yahtzee" Croshaw also prove useful.

**JF: Was learning AGS' scripting language difficult?**

**JN:** The learning process was relatively straightforward, and the AGS community is always there if you don't understand something.

**JF: Have you been able to express all of your ideas with it?**

**JN:** Yes, I was able to convey all of my ideas properly, particularly because they didn't rely on the existence of particle systems or physics-based puzzles—minus some handy-dandy box-moving sequences, that is.

**JF: You seem to be pushing at the limits of traditional adventure game mechanics by incorporating action sequences into BORYOKUDAN RUE as well as your earlier game, LA CROIX PAN. How difficult was it to construct these sequences in AGS?**

**JN:** I'm by no means a master programmer, so it was basically reiteration off of what I knew at the time, and then construction of a basic prototype during the first couple of weeks of production to make sure that I could actually achieve what I wanted to do.

In all my games, I like to try and incorporate different aspects of gameplay into the adventure genre, rather than just reusing existing mechanics from the heyday of twenty-or-so years ago.

Wow, was it really that long ago? So that's what led to these different amalgamations of gameplay scenarios.

**JF: Has being at UCLA helped you in developing the game?**

**JN:** I would like to say yes, but for the most part, BORYOKUDAN RUE has been a side project that I've worked on for about two years now, with a lot of the work having been done before I even started at UCLA. Being at UCLA, however, has been a great opportunity to get to know others who are interested in game design as well.

**JF: BORYOKUDAN RUE has a fantastic look to it. The desaturated color palette and rough-edged sketch aesthetic communicates the noir atmosphere really well. What was your process in designing the game's visuals?**

**JN:** One of my main inspirations in creating the visuals was *Cowboy Bebop*. There's a specific scene set in a gloomy cathedral that really captured the mood I was looking for.

**JF: What art tools did you use?**

**JN:** Tools of the trade include Photoshop, a Wacom tablet, and many, many blank canvases of 320x200 with a pink, beige, and blue palette to jump-start the drawing process.

**JF: Nathan Allen Pinard's moody electronic soundtrack adds an extra dimension to the game.**

**JN:** Even though we've never met, Nathan has been extremely collaborative with

the project. I have been very fortunate to work with Nathan as he's a great composer and his work really adds a lot to the atmosphere.

**JF: A good narrative is crucial in adventure games. Did you write BORYOKUDAN RUE first and then design the gameplay or have they evolved together?**

**JN:** Like probably most other game narratives, it evolved together as time went on. Creating a narrative

things are working or not until you get a secondary opinion on the game. Someone who can specifically tell you "Yes, I get it," or "No, I have no idea what's going on, who that character is supposed to be, or why I'm here." It's just one more reason to playtest your games.

**JF: What is appealing to you about telling a story through games rather than print or film?**

**JN:** Interactivity. Or that's



for games always depends upon the gameplay, because the story has to set up natural barriers that will reinforce whatever gameplay mechanics you have in mind for that specific game.

So, writing a story about organized crime naturally sets up an enemy force that would impede the player's progress. In addition, setting a story in a prison-like facility automatically creates some opposition to the goals of the second character, Delta-Six.

**JF: What has been the most difficult part in creating BORYOKUDAN RUE?**

**JN:** Probably the writing. It was a real challenge developing the characters, dealing out the exposition, and just making sure everything in the story "worked." It's hard to tell when

what I keep telling myself anyway. It's a hard balance to find what is unique to games and exploit that, rather than trying to deceive yourself into thinking that the story you're telling is different just because it's in a game, rather than in a novel or film.

What's compelling for me in an interactive narrative is the idea of choice, which is a theme that also plays out in BORYOKUDAN RUE. I'd really like to play with choice more in future projects—how does a player's choice or rather, the illusion thereof influence a narrative? Games are different because of their interactivity, so it's important to exploit that in order to provide unique narrative experiences.

—Jeffrey Fleming

## resources

**BORYOKUDAN RUE Trailer**  
[www.bigbluecup.com/yabb/index.php?topic=35594.0](http://www.bigbluecup.com/yabb/index.php?topic=35594.0)  
**Adventure Game Studio**  
[www.adventuregamestudio.co.uk](http://www.adventuregamestudio.co.uk)

# TURN YOUR PASSION FOR GAMING INTO A CAREER

## Game Art

Bachelor's Degree Program  
Campus & Online

## Game Design

Master's Degree Program  
Campus

## Game Development

Bachelor's Degree Program  
Campus

## Game Design

Bachelor's Degree Program  
Online



© 2010 Full Sail, Inc.

## Campus Degrees

### Master's

- Entertainment Business
- Game Design

### Bachelor's

- Computer Animation
- Digital Arts & Design
- Entertainment Business
- Film
- Game Art
- Game Development
- Music Business
- Recording Arts
- Show Production
- Web Design & Development

### Associate's

- Graphic Design
- Recording Engineering

## Online Degrees

### Master's

- Creative Writing
- Education Media Design & Technology
- Entertainment Business
- Entertainment Business: with a Sports Management Elective Track
- Internet Marketing
- Media Design

### Bachelor's

- Computer Animation
- Entertainment Business
- Game Art
- Game Design
- Graphic Design
- Internet Marketing
- Music Business
- Music Production
- Web Design & Development



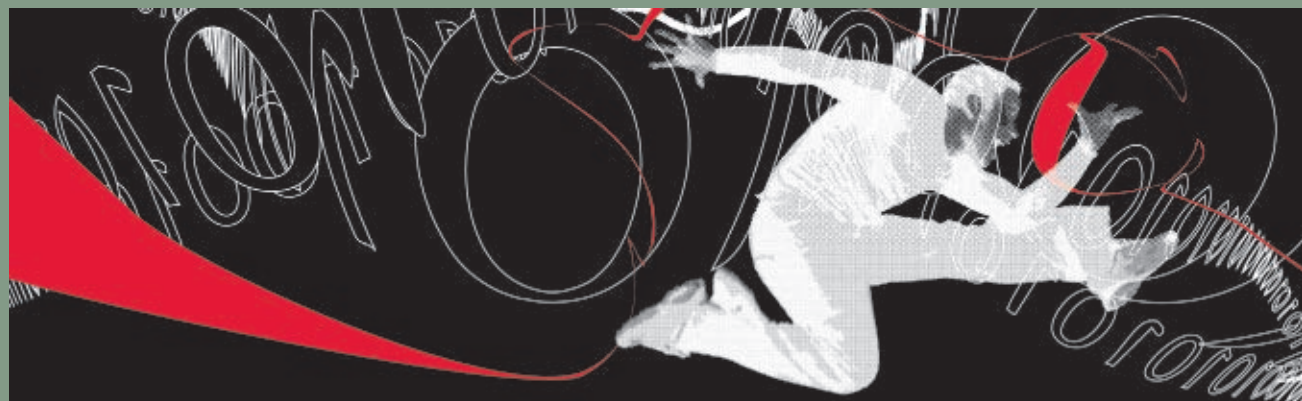
**FULL SAIL**  
UNIVERSITY

[fullsail.edu](http://fullsail.edu)

Winter Park, FL

800.226.7625 • 3300 University Boulevard

Financial aid available to those who qualify • Career development assistance  
Accredited University, ACCSC



## BECOME A MASTER OF DIGITAL MEDIA

We offer a 20-month master's degree in entertainment technology and digital media. This powerhouse graduate program combines industry-facing curriculum, real world projects, and a 4-month internship in Vancouver, Canada: videogame capital of the world.

Learn more. Contact [alison\\_robb@gnwc.ca](mailto:alison_robb@gnwc.ca) or visit [mdm.gnwc.ca](http://mdm.gnwc.ca)

**CENTRE FOR  
DIGITAL MEDIA**

Masters of Digital Media Program  
@ Great Northern Way Campus



**emily carr**  
university of art + design



a collaborative university campus environment



# MAKE MORE ENEMIES

**Game Design at VFS** shows you how to make more enemies, better levels, and tighter industry connections.

In one intense year, you design and develop great games, present them to industry pros, and do it all in Vancouver, BC, Canada, a leading hub of game development.

Our grads' recent credits include *Prototype*, *Mass Effect 2*, and *Dawn of War II*. The LA Times named VFS a top school "most favored by video game industry recruiters."



VFS student work by Maximilian Gordon Vogt



VFS

Find out more.  
[vfs.com/enemies](http://vfs.com/enemies)

"The staff at VFS provided a foot in the door that gave me an opportunity to prove myself."

ARMANDO TROISI | GAME DESIGN GRADUATE  
LEAD CINEMATIC DESIGNER, MASS EFFECT 2

## THE LEADING PROFESSIONAL GAME SOURCE FOR JOB SEEKERS AND EMPLOYERS

Access the Web's Largest Game Industry Resumé Database and Job Board



**GAMASUTRA**  
The Art & Business of Making Games

Take Control of Your Future Today!  
Log onto [www.gamasutra.com/jobs](http://www.gamasutra.com/jobs)





[ GEEKED AT BIRTH ]

[ IT'S IN YOUR PULSE ]

**Bachelor of Science** > Game Programming  
**Bachelor of Arts** > Game Art and Animation, Game Design, Serious Game and Simulation  
**Master of Science** > Game Production and Management

LEARN:

Advancing Computer Science	Enterprise Software Development	Robotics and Embedded Systems
Artificial Life Programming	Network Engineering	Technology Forensics
Digital Media	Network Security	Virtual Modeling and Design
Digital Video	Open Source Technologies	Web and Social Media Technologies



You can talk the talk. Can you walk the walk? Here's a chance to prove it. Please geek responsibly. [www.uat.edu](http://www.uat.edu) > 877.UAT.GEEK

## ADVERTISER INDEX

COMPANY NAME	PAGE	COMPANY NAME	PAGE	COMPANY NAME	PAGE
5th Cell .....	43	Gearbox Software .....	51	Seapine Software .....	12
Activision .....	49	Havok .....	C3	SIGGRAPH .....	46
Blizzard Entertainment .....	39	Masters of Digital Media Program .....	53	TechExcel .....	30
E3 Expo .....	33	Nintendo of America .....	3	University of Advancing Technology .....	55
Epic Games .....	19	Perforce Software .....	6	Vancouver Film School .....	54
Full Sail Real World Education .....	53	Rad Game Tools .....	C4	xaitment .....	26
Gameloft .....	50	Scaleform Corporation .....	C2		

*Game Developer* (ISSN 1073-922X) is published monthly by United Business Media LLC, 600 Harrison St., 6th Fl., San Francisco, CA 94107, (415) 947-6000. Please direct advertising and editorial inquiries to this address. Canadian Registered for GST as United Business Media LLC, GST No. R13288078, Customer No. 2116057, Agreement No. 40011901. **SUBSCRIPTION RATES:** Subscription rate for the U.S. is \$49.95 for twelve issues. Countries outside the U.S. must be prepaid in U.S. funds drawn on a U.S. bank or via credit card. Canada/Mexico: \$69.95; all other countries: \$99.95 (issues shipped via air delivery). Periodical postage paid at San Francisco, CA and additional mailing offices. **POSTMASTER:** Send address changes to *Game Developer*, P.O. Box 1274, Skokie, IL 60076-8274. **CUSTOMER SERVICE:** For subscription orders and changes of address, call toll-free in the U.S. (800) 250-2429 or fax (847) 647-5972. All other countries call (1) (847) 647-5928 or fax (1) (847) 647-5972. Send payments to *Game Developer*, P.O. Box 1274, Skokie, IL 60076-8274. For back issues write to *Game Developer*, 4601 W. 6th St. Suite B, Lawrence, KS 66049. Call toll-free in the U.S./Canada (800) 444-4881 or fax (785) 838-7566. All other countries call (1) (785) 841-1631 or fax (1) (785) 841-2624. Please remember to indicate *Game Developer* on any correspondence. All content, copyright *Game Developer* magazine/United Business Media LLC, unless otherwise indicated. Don't steal any of it.





# PHONE TAG WITH THE ART CONTRACTOR FROM HELL

WATCH A MASTER AT WORK!

**WEEK 1** Hey guys, I'm really looking forward to working with you on your game. I love modern military hardware and modeling intricate detail is totally my specialty. I am "all systems go," and ready to bust ass on these models and get them to you fast and looking extra hot! I should have something to show you by next week. This is going to be, like, a master at work! Alright dudes, talk to you later!

**WEEK 2** What's up? How are you? Oh yeah, the models. They're coming along pretty awesomely, even though I did run into some delays I couldn't do anything about. There was, uh, band practice on Thursday, so I had to stop working early, and that was something I just couldn't miss because we have a pretty major gig coming up—the open mic night at O'Malley's Irish Tavern, right next to the library. You heard of that place? Also, my cat got sick and I had to take care of her. It's all good, though. Just a couple more days and we'll be good to go.

**WEEK 3** Yo! Making progress! Making really good progress,

actually, considering the fact that I only have one monitor. See, it's actually kind of difficult to work on my single-monitor setup here. I envy you guys in your big studio setup, haha! I was talking to my friends and they were like "Woah, you don't have a dual-monitor setup?" So I'm sure that once I get that, my productivity will skyrocket and I can get all those models done on time. I actually just spent the last couple of days researching monitors and I found one I want. Just need to order that sucker and then it's productivity city.

**WEEK 4** Huh? Who is this again? Oh! Hi! Sorry, yeah, you called pretty early in the morning. Do you all get up before 11 over there? Haha! Well, I've got something to show you. If you point your browser to this URL here, you can check out this orc head I modeled in ZBrush the other day. Isn't that incredible? Look at all the crazy detail on his mottled skin there. I know you guys are making a game with a realistic modern military setting, but I was just, you know, evaluating

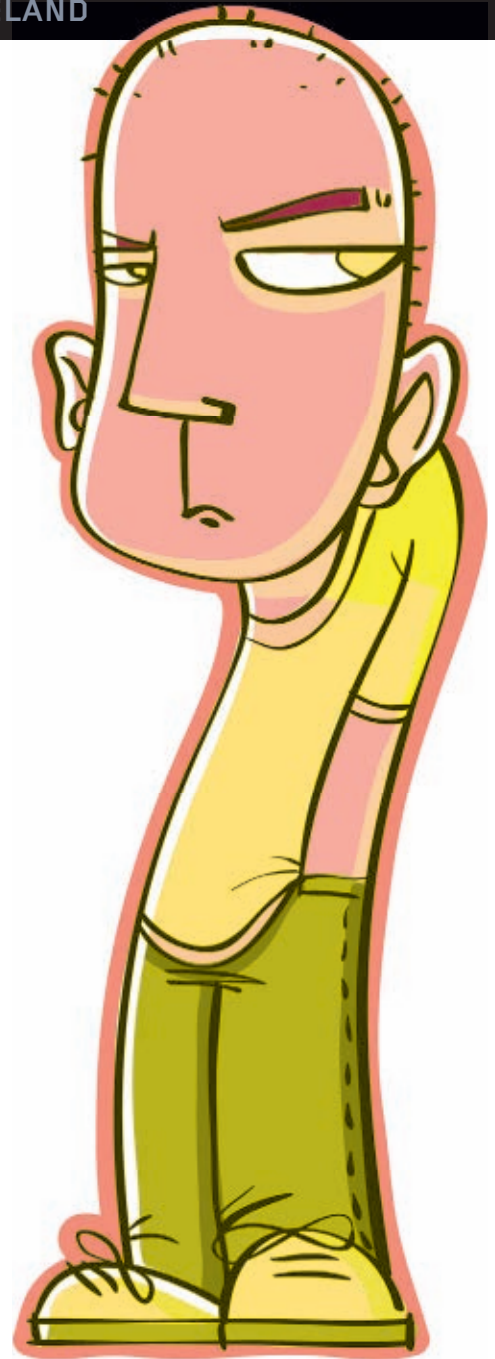
potential tools. I mean that orc head modeling technique will translate directly over to the weathering I'm going to put on your jeeps and trucks and stuff. It's almost the same exact thing, you know, with, like, the texture and whatever. So yeah, as soon as I get back from the workshop I'm attending in San Francisco on orc head modeling techniques, I'll be ready to start cranking.

**WEEK 5** Got a great update for you guys this morning. I've been working out an all-encompassing generic vehicle system that should allow me to easily create any vehicle you want at a moment's notice. I've got some generic chassis assemblies and a big MEL script I've been working on over the last week that automatically puts together the basic components of any vehicle. You get a dialogue box where you can input the number of wheels the vehicle has, and it figures out the best placement for all of them. You should see what it does when you put in numbers like 7 or 3.5! It's crazy, dude!

**WEEK 6** Hey, it's me, the contractor. Hope you remember me, haha. I was just wondering if you could send me the, uh, asset list again? I can't find the original one you sent and I just thought, hey, if I'm going to be making these models, I should know what they are, haha! Thanks bro.

**WEEK 7** Hi guys, it's me, the art contractor. I, uh, haven't heard from you in a while, so I just thought I'd check in and like, say hey, what's up. I've got stuff cooking, stuff on boil, and lots of pots and pans in the kitchen, haha. Cookin'. Cookin' like a maniac. Sorry I called late—I just finished a wild set with the band, like, out of control. The drummer told me he thought I couldn't keep down as many shots as he could and, uh, uh, uh, anyway, I'm sure you'll get this message in the morning. Give me a call when you can. I know you guys are busy making your awesome video game. And your game will have sweet models. I guarantee this. Sweetness. Okay, bye.

**WEEK 8** Yo, it's me again. I hope this isn't awkward, but I



just saw the trailer for your game and it looks like there are, like, vehicles in it already. Are they placeholders? I noticed you didn't even use the hubcap model I turned in the other day. It's too bad because I think my hubcap is better than the one you can see in the video. And there's more where

that came from. I've got a tire coming in the next couple of days, too, and maybe even more than—oh, crap. My cat just threw up again. Hey, I'll, uh, call back later, okay? Alright. Bye.

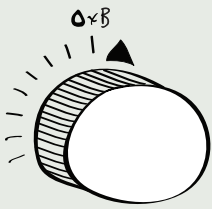
**MATTHEW WASTELAND** writes about games and game development at his blog, *Magical Wasteland* ([www.magicalwasteland.com](http://www.magicalwasteland.com)).

# Reliable Pathfinding Technology. Havok AI.



**havok**<sup>™</sup>  
[www.havok.com](http://www.havok.com)  
*Turning Creative Aspirations  
into Technical Realities<sup>™</sup>*





TURN UP THE

# WICKED



AUDIO IN YOUR GAME WITH

# Miles 8

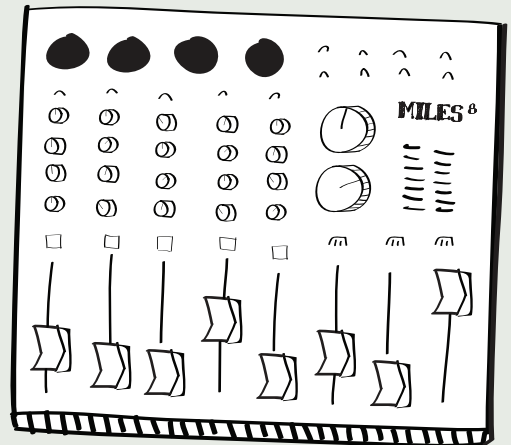
BRAND NEW!



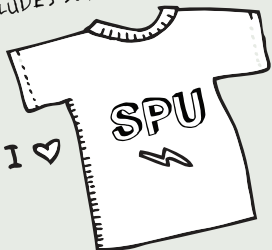
Our multichannel sound  
system features cool, new

high-level audio tools

+ the world's **FASTEST**



INCLUDES SUPPORT FOR THE PS3



# MP3 and Ogg DECODERS.

USING MILES 8 NOT ONLY MAKES YOU

WANT TO TURN IT UP, IT MAKES YOU **rad!**



[www.radgametools.com](http://www.radgametools.com)  
(425) 893-4300