# Vision Engine Case Studies: A Look behind the Scenes

Florian Born
Lead Software Engineer, Havok

KGC 2011

# Overview

- About me
  - Co-founded Trinigy in 2003, Technical director
  - 2011: Acquisition of Trinigy through Havok
  - Working as a Lead Engineer in the Vision Engine product team
  - Closely connected to support team

# Overview

- Very different game titles from three continents…
- Very different budgets…
- Very different team sizes…
- Very different target platforms…
- Very different technological requirements…

- …but all built with the same Game Engine.

# Overview

- Pick some customer projects that we think are representative
- And see the different challenges and requirements


# Arcania – Gothic 4

# Spellbound: Arcania – Gothic 4

- Game Background:
  - Large-Scale RPG, big budget
  - Latest sequel to one of Europe's largest RPG brands
  - Platforms: PC, Xbox360
  - Project Status: Released (2010)

- Challenges:
  - Large, seamless, highly detailed world on both platforms
  - Existing, battle-proven custom Streaming Solution that Spellbound wanted to use
  - Many level designers working simultaneously on the game world
  - AAA visuals

# Spellbound: Arcania – Gothic 4

Engine Requirements:

- Flexible, modular and extensible tech
  - Seamless drop-in replacement for streaming solution
  - Special Solutions for cloud/sky rendering, weather etc.
  - Full access to low-level rendering features
- Fast, low-overhead engine optimized for both platforms
  - Everything performance-critical to be written in C++
  - Scripting (LUA) for high-level game code
- Support for collaborative editing
- Huge environments required 64 Bit versions of tools

# Spellbound: Arcania – Gothic 4

Rendering

- Mix of out of the box features, custom, and third-party solutions
  - Custom features use Vision's low-level rendering API
    - e.g. Weather Effects, Ocean Water
  - Sky/Cloud Rendering uses Simul Weather
    - Full binding with Vision Engine available from Simul
  - Deferred Rendering
    - Tetrahedron Shadow Maps for Point Lights
    - Cascaded Shadow Maps (Variance Shadows) for Sun
    - Time of Day, Sun Glare (custom)

# Spellbound: Arcania – Gothic 4

## Streaming

- Custom streaming solution
  - Already battle-proven in previous projects
  - Spellbound wanted to use this solution in Arcania – Gothic 4
    - Vision Streaming existed at this point, but wasn't as proven yet
  - Full replacement for Vision's standard streaming solution
    - Without touching any of the core engine's source code
    - Built on top of Vision's resource management


Dizzel

# Neowiz (Team Storm): Dizzel

- Game Background:
  - Third-Person Multiplayer Shooter Game
  - Focused on shooting, but also melee combat
  - Platform: PC
  - Project Status:  2nd Closed Beta
- Challenges:
  - Support for lower-end PCs while still providing up-to-date visuals

# Neowiz (Team Storm): Dizzel

- Engine Requirements:
  - Comprehensive tech with all critical components integrated
    - Physics, Sound, ScaleformGfx
    - Visual Shader Editing
  - Good performance on lower-end PCs

# Neowiz (Team Storm): Dizzel

- Uses Vision's standard forward rendering pipeline
  - Performance/scalability reasons
  - Pre-baked Dot3 Lightmapping and Light Grids
  - DirectX9 only, no DirectX11 support
- Extensions
  - Editing AI features (e.g. cover) though Editor plugin
  - Custom forward kinematics solution
    - Presented in a lecture at GDC 2010 by Neowiz


Settlers 7: Paths to a Kingdom

# Ubisoft Blue Byte: Settlers 7

- Game Background:
  - One of Europe's most renowned RTS franchises
  - Large title with extremely high counts of dynamic objects
  - Unique, "cute" visuals which still look outstanding and cutting-edge
  - Custom in-game editing solution for missions/levels
  - Platform: PC
  - Project Status: Released (2010)

# Ubisoft Blue Byte: Settlers 7

- Challenge:
  - Development Team already had code structure defined
    - Engine needed to be able to fit into this requirement
    - Developer affected by Renderware vanishing from the market in 2004
  - Many hundreds of independently animated objects on screen

# Ubisoft Blue Byte: Settlers 7

Engine Requirements:

- Very flexible technology
  - Did not want to build their pipeline around specific engine
  - Instead, engine needed to adapt to their pipeline!
- Exceptionally high performance of the rendering and animation tech
  - Multithreading, low overhead, C++ only
- Squash & Stretch support for animations in XSI (!)
  - Was added as a customer-specific addition to Vision

# Ubisoft Blue Byte: Settlers 7

Rendering

- Pretty heavily customized
  - Motivation: Keep rendering largely independent of underlying engine
  - Uses low-level rendering features of the Vision Engine
  - Dynamic instancing of clutter, trees, rocks,…
    - Only use HW instancing when instance count > threshold
    - Lots of shader permutations!
  - Extreme care regarding vertex formats and -attributes
  - Static LOD levels
  - Uses custom deferred rendering system with custom atmospheric effects
  - Custom Water Rendering


Naka Online

# Neowiz (Team Sion): Naka Online

- Game Background:
  - Fantasy MMO RPG
  - Platforms: PC

- Challenges:
  - Heavy on the Game Effects side
    - Needed solution for timing animations, particles,…
  - Mixing of heightmap-based terrain and mesh-based terrain
    - To match the unique look of the game, rocks, mountains etc. needed to be meshes
  - Lush environments (lots of vegetation)

# Neowiz (Team Sion): Naka Online

- Engine Requirements:
  - Comprehensive out of the box functionality
    - Effects editing
  - Time of Day and Deferred Rendering
  - Streaming Solution
  - Efficient terrain/vegetation editing and rendering system

- Vision was a very good fit:
  - Fast vegetation rendering through hardware instancing
  - Deferred Rendering Pipeline “out of the box”
  - Effects framework didn’t fully meet the team’s requirements
    - But editor’s extensibility made up for it

# Neowiz (Team Sion): Naka Online

## Custom Effects Tool

- Neowiz created a custom Game Effects tool
  - Vision has a powerful particle editing solution
  - Additional Requirement: Combine Entities, Particles, Special Effects Shapes, etc. and animate them over time
  - Solution: Build Game Effects Tool on top of existing editor infrastructure
    - Layered editor architecture allows using editor framework classes in custom tools
    - Integrated Engine View to see results in realtime (WYSIWYG)

# Neowiz (Team Sion): Naka Online

- Customizations
  - Some customizations were provided by Trinigy for particle rendering:
    - Terrain Constraint (particles colliding with terrain)
    - Texture Mask Emitter
    - Particle Rotation / Ringwaves
  - All these customizations are now part of the regular Vision SDK

- Third-party bindings
  - PhysX
  - Scaleform Gfx for UI


Bus & Cable Car Sim San Francisco

# TML: Bus & Cable Car Simulator

- Game Background:
  - Bus and Cable Car driving simulation
  - Not intended for training, but a “real” game
  - Realism and high recognition value are critical
  - Platform: PC
  - Project Status: Released (2011)

- Challenge:
  - Accurately recreate a huge real-world environment (San Francisco)
  - Realistic visuals
  - Very (!) limited budget and time frame (~9 months)
  - Small team (less than 10 people, almost all of them artists)

# TML: Bus & Cable Car Simulator

- Engine Requirements:
  - Comprehensive “out of the box” feature set
  - Streaming of huge scenes (100 square kilometers!) at high detail
  - Deferred rendering (thousands of switchable lights)
  - Very fast prototyping and short turnaround times
    - Playing the game in the editor
  - Collaborative editing
    - Whole environment is a single, large scene…
- Given these constraints and requirements, the results are quite impressive!

# TML: Bus & Cable Car Simulator

- Statistics:
  - > 100 km² of Terrain, fully streamed
  - 1 200 Bus Stations
  - More than 10 000 different meshes (Many Gigabytes of data)
  - Millions (!) of static/dynamic mesh instances
  - Ten thousands of switchable/dynamic lights (traffic lights, car headlights, …)
  - Heavily using prefabs to manage the scene creation effort
    - Prefab parameters very useful for composite objects!
  - Everything is streamed dynamically
    - > 1 000 separate streaming zones

# TML: Bus & Cable Car Simulator

## Streaming System: Setup

- Uses Vision's built-in streaming solution
  - Scene separated into 1096 streaming zones using the editor
    - "Zone Grids" feature
  - All resource types can be streamed in Vision
    - Textures, Meshes, Shaders, UI, Particle Effects, Animations, …
    - Hard requirement – meshes alone account for ~4GB of data!
  - Default: Cache-in / Cache-out / Force-in radius
    - Based on distance from camera
  - Multi-user editing across zones

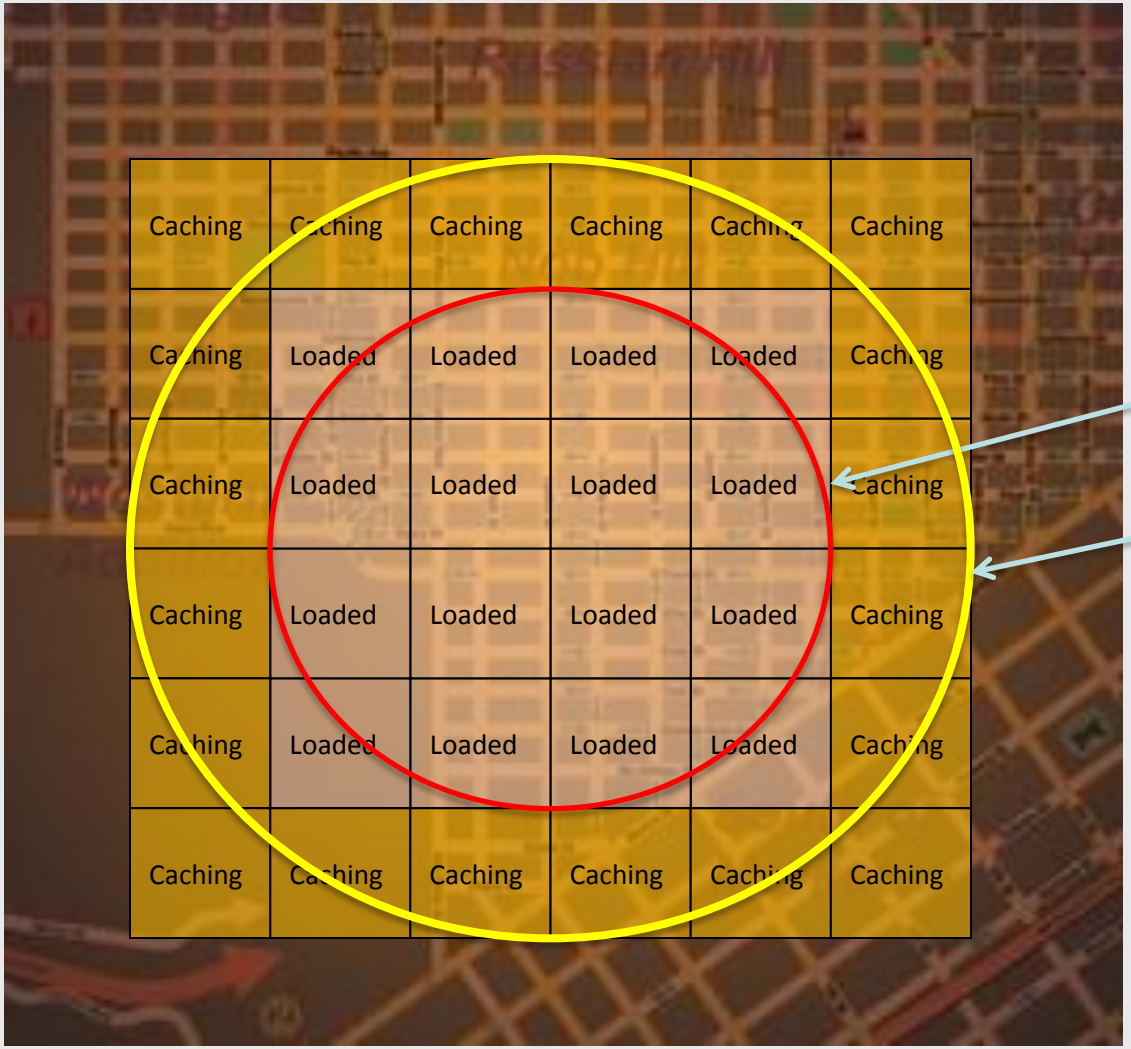
# TML: Bus & Cable Car Simulator

## Streaming System: Challenges

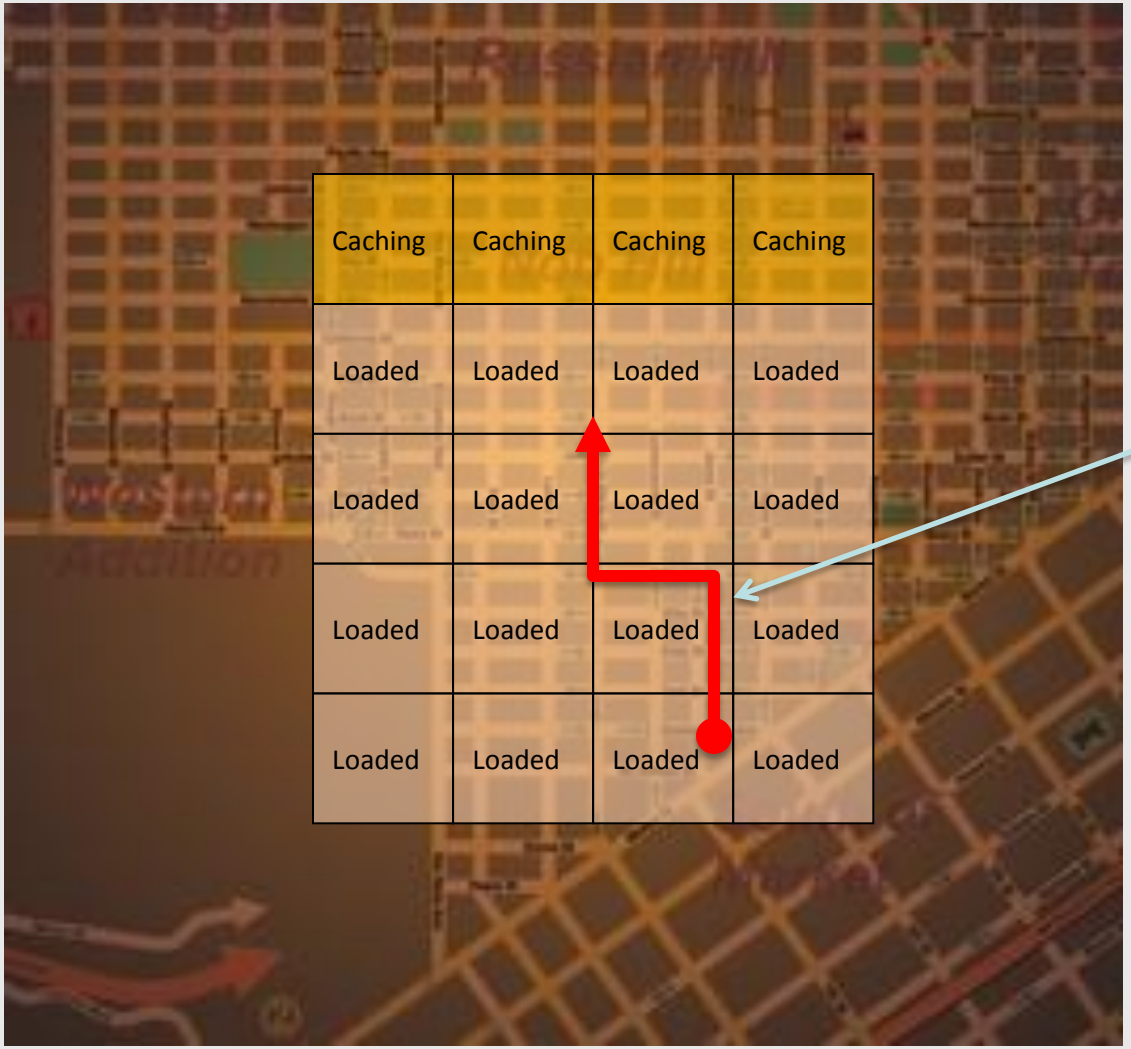
- Some zones contain a *lot* of data
  - Too much for some systems to handle
  - Solution: Streaming zones around highly detailed areas have to contain less unique data
  - Since the cache-in areas overlap, this balances streaming load

- Application-specific optimization
  - Prefer streaming zones located in the driving direction of the bus
  - Simply override the default behavior for streaming priorities

# TML: Bus & Cable Car Simulator



# TML: Bus & Cable Car Simulator



# TML: Bus & Cable Car Simulator

Rendering

- Heavily uses Vision's out of the box features
  - Deferred Rendering, Postprocessing Effects (DOF, SSAO, Tone Mapping with Color Grading)
    - Every street light, traffic light etc. has 1..n switchable light sources
  - Volumetric Cones
  - Cascaded Shadow Maps
    - Large view distances
  - Time of Day
    - No pre-baked lighting at all – everything is dynamic
  - Dynamic Cube Map Reflections

# TML: Bus & Cable Car Simulator

## Rendering

- Plus a few custom-built rendering features
  - Very lean – built on top of Vision's powerful infrastructure
  - Windshield wiper / raindrop effect
    - Postprocessing Effect using modular component approach
    - Simple plugin – immediately accessible in the editor
  - Dirt rendering on vehicles (Shader)
    - Custom shader class interacting with game code
  - Dynamic Weather System (Particle Effects, Fog, Wind)
    - Particles: Modified versions of Vision's default weather particles
    - Wind: Vertex Shader modifies vegetation geometry depending on wind speed (no SpeedTree!)


Trapped Dead

# Crenetic: Trapped Dead

- Game Background:
  - Award-winning Tactical Action Game (think Commandos)
  - Survival Horror – a group of humans against hordes of Zombies
  - Dark environment, hard lighting effects to create creepy atmosphere
  - Single-player & multi-player mode
  - Game Demo available as a Browser Game through WebVision
  - Platforms: PC, Browser

- Challenge:
  - Dynamic lighting and many animated characters
  - Goal: Game Demo fully playable inside a web browser

# Crenetic: Trapped Dead

- Engine Requirements:
  - Deferred Rendering Solution
    - Full-fledged rendering pipeline without the need for customizations
  - Web-enabled technology
  - Efficient, multithreaded animation system
  - Fully integrated bindings with third-party AI solutions
    - Crenetic ended up choosing Xaitment

# Crenetic: Trapped Dead

## Trapped Dead Core Features

- Rendering: Uses Vision's built-in feature set
  - Multithreaded animations and particle effects
  - Vision supports various skinning modes (CPU, Vertex Shader, Stream Out, Compute Shaders, …)
    - Crenetic chose Vertex Shader Skinning on DirectX9
  - Deferred Rendering System
    - With dynamic shadows (tetrahedron shadow maps, CSM)

# Crenetic: Trapped Dead

## Trapped Dead Core Features

- Game demo available as a browser game
  - Based on WebVision
  - Uses the same code base as retail product
  - Web-enabled in less than a day!

Orcs Must Die!

# Robot Entertainment: Orcs Must Die

- Game Background:
  - Fast-paced third-Person Action Game with tactical elements
  - Humorous Fantasy Setting
  - Cartoony, but high-fidelity visuals
  - Platforms: PC and Xbox360 at launch
  - Project Status: Released October 2011

- Challenges:
  - Consistently high frame rates on consoles (60fps)
  - Seamless interaction of character animations with in-game physics
  - “Building block”-based approach to creating levels

# Robot Entertainment: Orcs Must Die

- Engine Requirements:
  - Optimized Console versions
  - Physics-enabled character technology
  - Pre-baked lighting
  - Dozens of enemies on screen simultaneously
    - Fast multi-threaded character animation system
  - High performance even when levels are built from many small pieces
    - No “draw call explosion”

# Robot Entertainment: Orcs Must Die

- Rendering
  - Uses Vision's Forward Rendering System
    - To achieve consistent 60fps on consoles
    - Custom "Über-Shader" approach to handle specific materials and create special cartoony look of OMD
  - "Subtractive" Shadow Maps
  - Lighting is pre-baked into Dot3 Lightmaps and Light Grid
  - Many custom particle effects built with the editor's particle editor
  - Visibility Culling: Uses Vision's Portal System
  - Memexport Skinning on Xbox360

# Robot Entertainment: Orcs Must Die

- Level Building
  - Levels are built from many small pieces (some only have a few triangles)
    - Would be too slow if rendered separately
    - Editor supports “mesh baking”:
      - Select arbitrary number of static mesh instances
      - Have editor generate a new mesh for you
      - Thousands of triangles per draw call instead of a dozen
      - Comes at a certain memory cost
- Characters and Physics
  - Physics-enabled characters (Ragdoll, reaction to hits, …)
  - Vision Engine provided bindings with Morpheme + PhysX

# Summary

- Teams chose the Vision Engine for very different reasons
  - Comprehensiveness
  - Flexibility and Modularity
  - Extensibility
  - Performance

# Summary

- Vision's design ensures that you won't encounter roadblocks
  - Out of the games that were discussed here, only Orcs Must Die used a modified version of the engine
    - Minor special-case performance optimizations to the math code for Xbox360
  - Vision provides you with a comprehensive feature set…
  - …but still leaves full control in your hands

# Summary

## Questions?