3D Vision Technology
Develop, Design, Play in 3D Stereo

Samuel Gateau

NVIDIA

Can you see it ?

Outline

- NVIDIA 3D VISION™
  - Stereoscopic driver & HW display solutions
- Stereoscopy basics
  - Definitions and equations
- Capcom - Resident Evil 5
  - Words from the developer
- The Look of Depth
  - Controlling stereo parameters

How does it work ?

NVIDIA® 3D VISION™

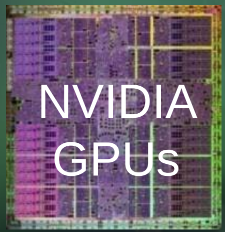
3D Movies

3D Pictures

3D Games

3D Applications

Dimensionalized Experience



The programmability of the GPU allows NVIDIA to import <u>any</u> 3D data format and decode, convert, or transform the data for viewing on a 3D-Ready displays.

120 Hz LCDs

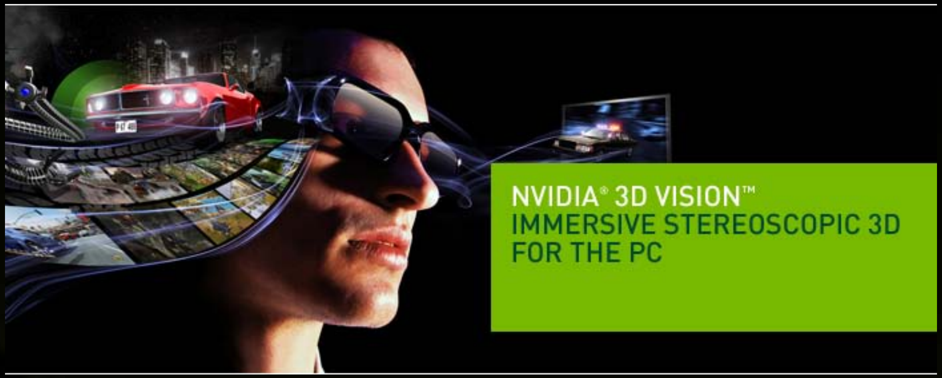
3D DLP HDTVs

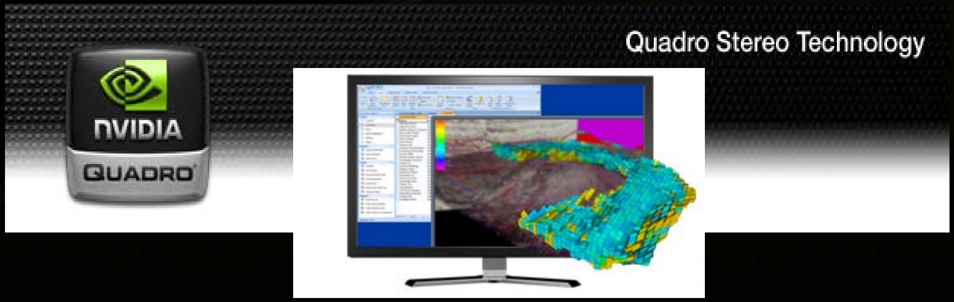
3D Projectors

Anaglyph 3D

Stereo Support

- GeForce
  - Stereo Driver
    - Vista & Win7
    - D3D9 / D3D10
- Quadro
  - GeForce features
  - Professional OpenGL Stereo Quad Buffer
    - Multiple synchronized stereo displays
    - Multi-platform
    - 3D Vision and many other stereo displays





NVIDIA 3D Vision Solutions

	NVIDIA 3D Vision Discover	NVIDIA 3D Vision
Availability	Bundled with select NVIDIA GPUs for a sneak peak at stereoscopic 3D	Sold as a complete kit for full HD stereoscopic 3D
3D Glasses type	NVIDIA optmized anaglyph (red/cyan)	Wireless Shutter glasses
3D Mode	Anaglyph with optimized color and image processing on the GPU	Page flip 120 Hz & checkerboard pattern 3D
Color Fidelity	Limited Color	Full Color
Display requirements	All desktop LCD and CRT displays	3D-Vision-Ready displays
NVIDIA GeForce GPU	GeForce 8 series and higher	GeForce 8 series and higher
Operating System	Microsoft Windows Vista Microsoft Windows 7	Microsoft Windows Vista Microsoft Windows 7
View 3D pictures	Y	Y
Watch 3D movies	Y	Y
Play real-time 3D games	Y	Y
3D consumer applicaiton	Y	Y

3D Vision Industry Support

NVIDIA

Display Partners

SAMSUNG

ViewSonic®

MITSUBISHI ELECTRIC
DIGITAL TELEVISIONS™

InFocus® DepthQ®
STEREOSCOPIC

Software Applications

CAPCOM

BLIZZARD
ENTERTAINMENT

cooliris

How It Works

3D game data is sent to stereoscopic driver

The driver takes the 3D game data and renders each scene twice – once for the left eye and once for the right eye.

Left Eye view

Right Eye view

A Stereoscopic display then shows the left eye view for even frames (0, 2, 4, etc) and the right eye view for odd frames (1, 3, 5, etc).

How It Works

In this example active shutter glasses black-out the right lens when the left eye view is shown on the display and black-out the left lens when the right eye view is shown on the display.

This means that the refresh rate of the display is effectively cut in half for each eye. (e.g. a display running at 120 Hz is 60 Hz per eye)

Left eye view on,
right lens blocked

Right eye view on,
left lens blocked

The resulting image for the end user is a combined image that appears to have depth in front of and behind the stereoscopic 3D Display.

NVIDIA 3D Vision

NVAPI Stereoscopic Module

- NVAPI is NVIDIA's core software development kit that allows direct access to NVIDIA GPUs and drivers
- NVAPI now expose a Stereoscopic Module providing access to developer to the Stereoscopic driver settings
  - Detect if the system is 3D Vision capable
  - Manage the stereo profile settings for the game
  - Control dynamically the stereo parameters from within the game engine for a better stereo experience
- For download and documentation http://developer.nvidia.com/object/nvapi.html

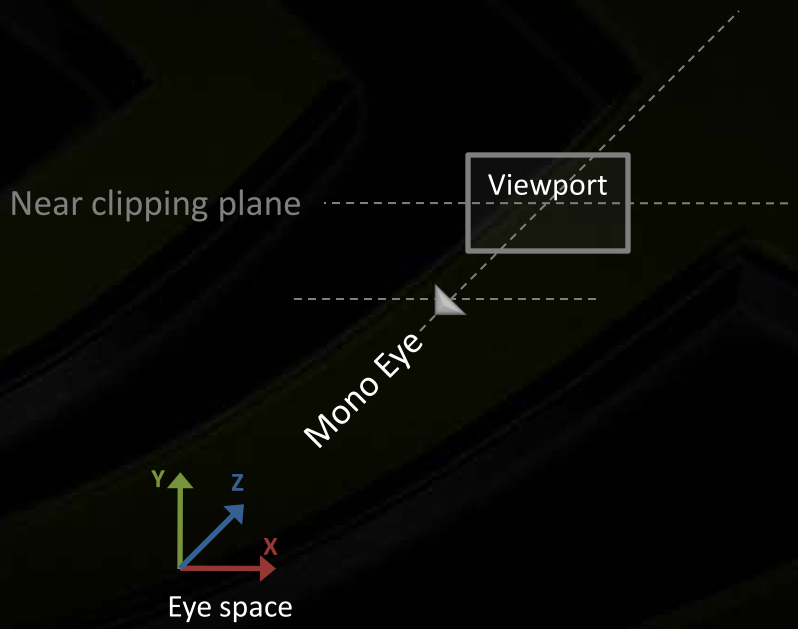
Under the hood

STEREOSCOPY BASICS

Stereoscopy Basics

Standard Mono Rendering

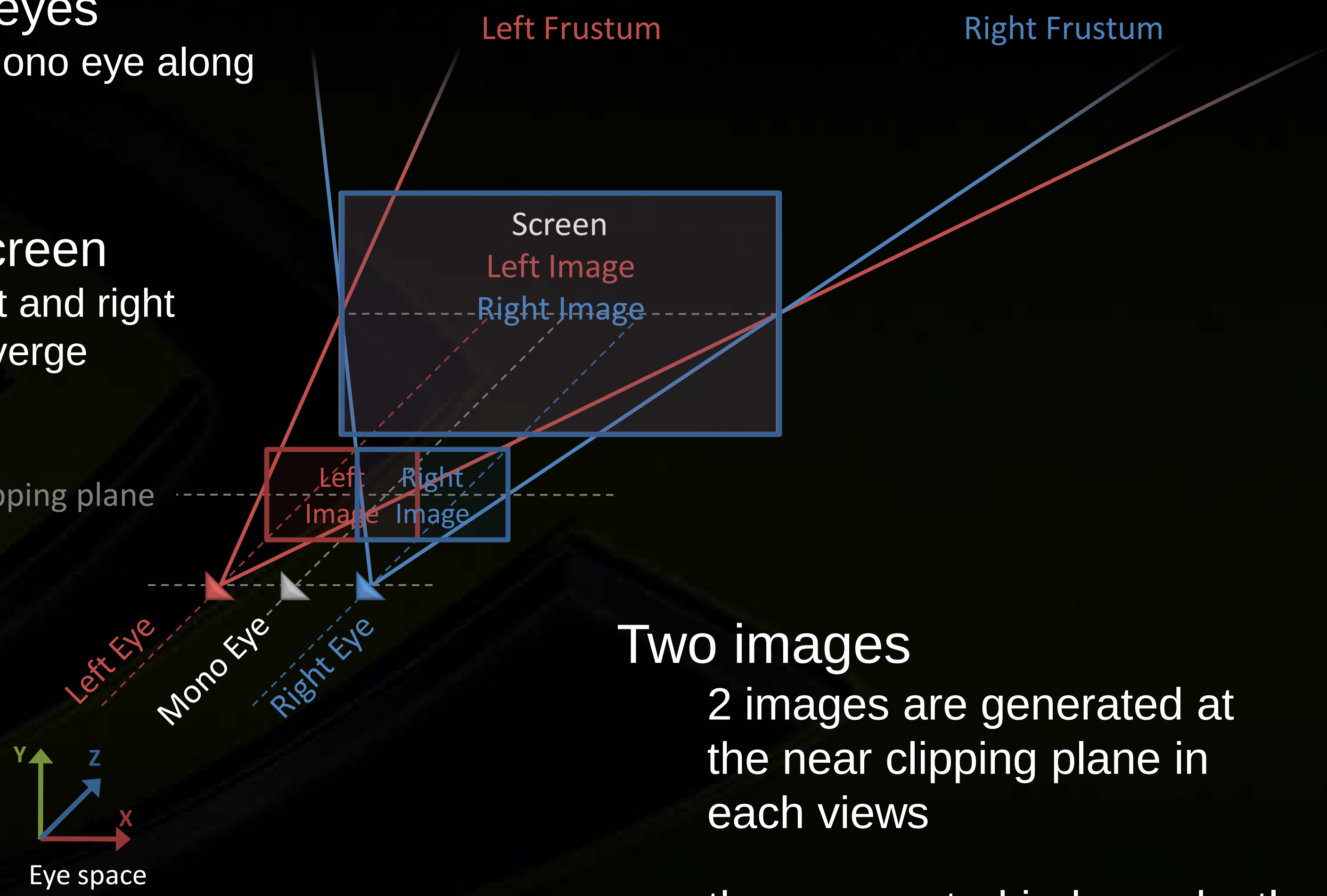
Scene is viewed from one mono eye and projected on Near Clipping plane in Viewport



Two eyes, one screen, two images

Left and Right eyes
- Shifting the mono eye along the X axis

One “virtual” screen
- Where the left and right frustums converge



Two images
- 2 images are generated at the near clipping plane in each views
- then presented independently to each eyes of the user on the real screen

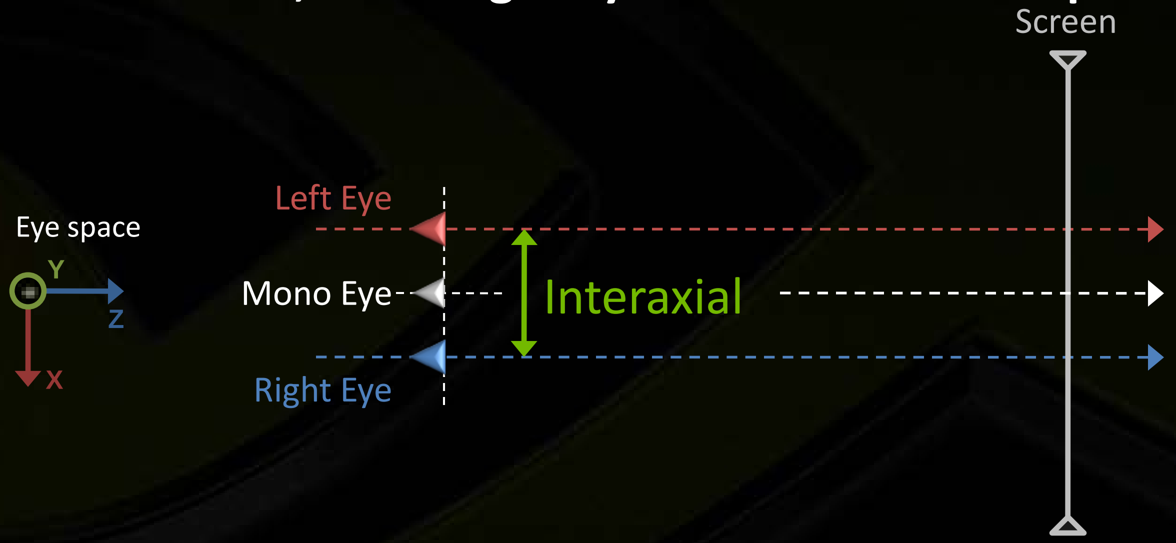
Stereoscopic Rendering

- Render geometry twice from left and right viewpoints into left and right images
- 3 independent modifications to standard pipe
  - Use stereo surfaces
    - Duplicate render surfaces
  - Do stereo drawcalls
    - Duplicate drawcalls
  - Apply stereo separation
    - Modify projection matrix

Stereoscopic Basics

Interaxial

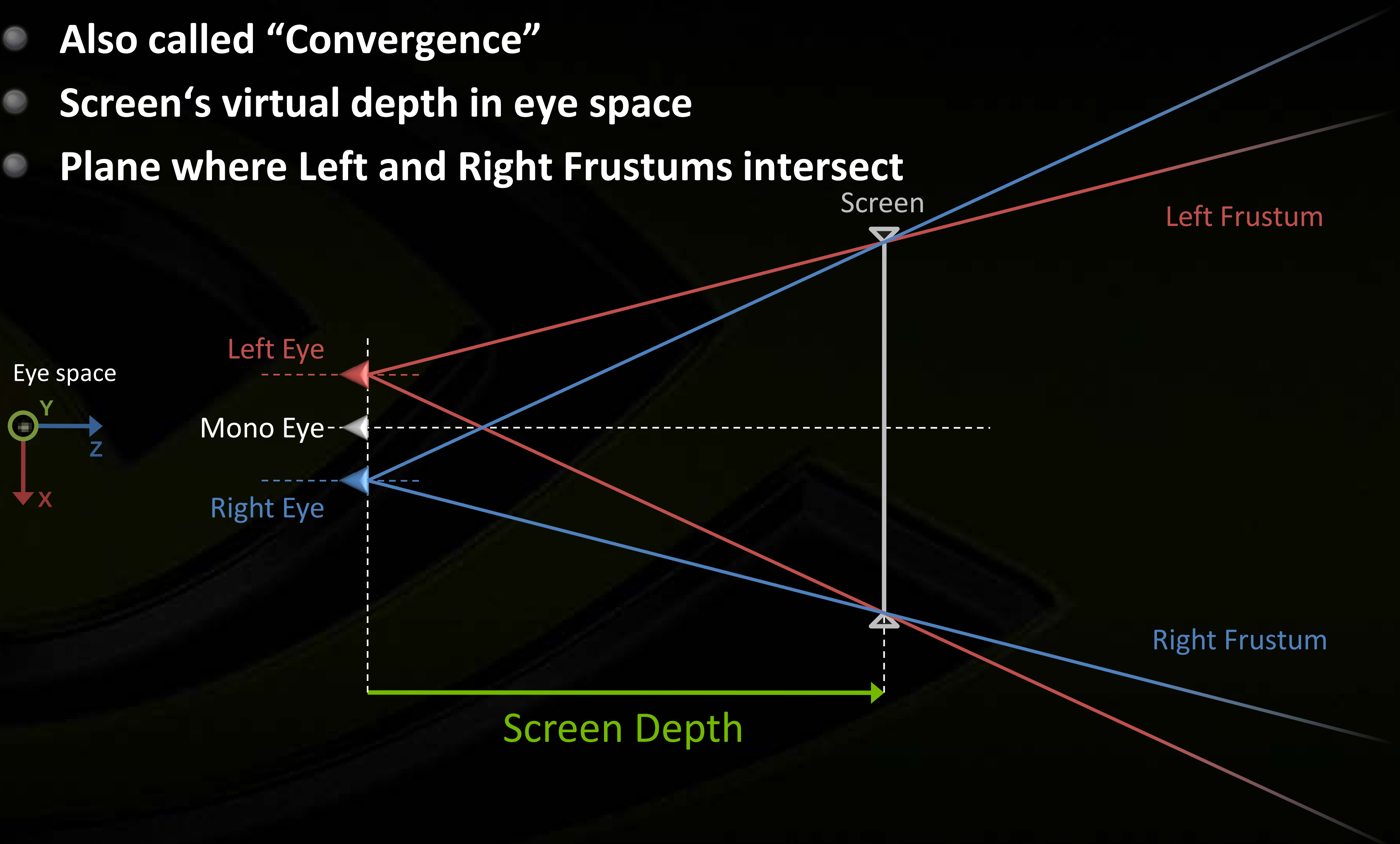
- Also called “Eye Separation”
- Distance between the 2 virtual eyes in eye space
- The mono, left & right eyes directions are all parallels



Stereoscopic Basics

Screen Depth

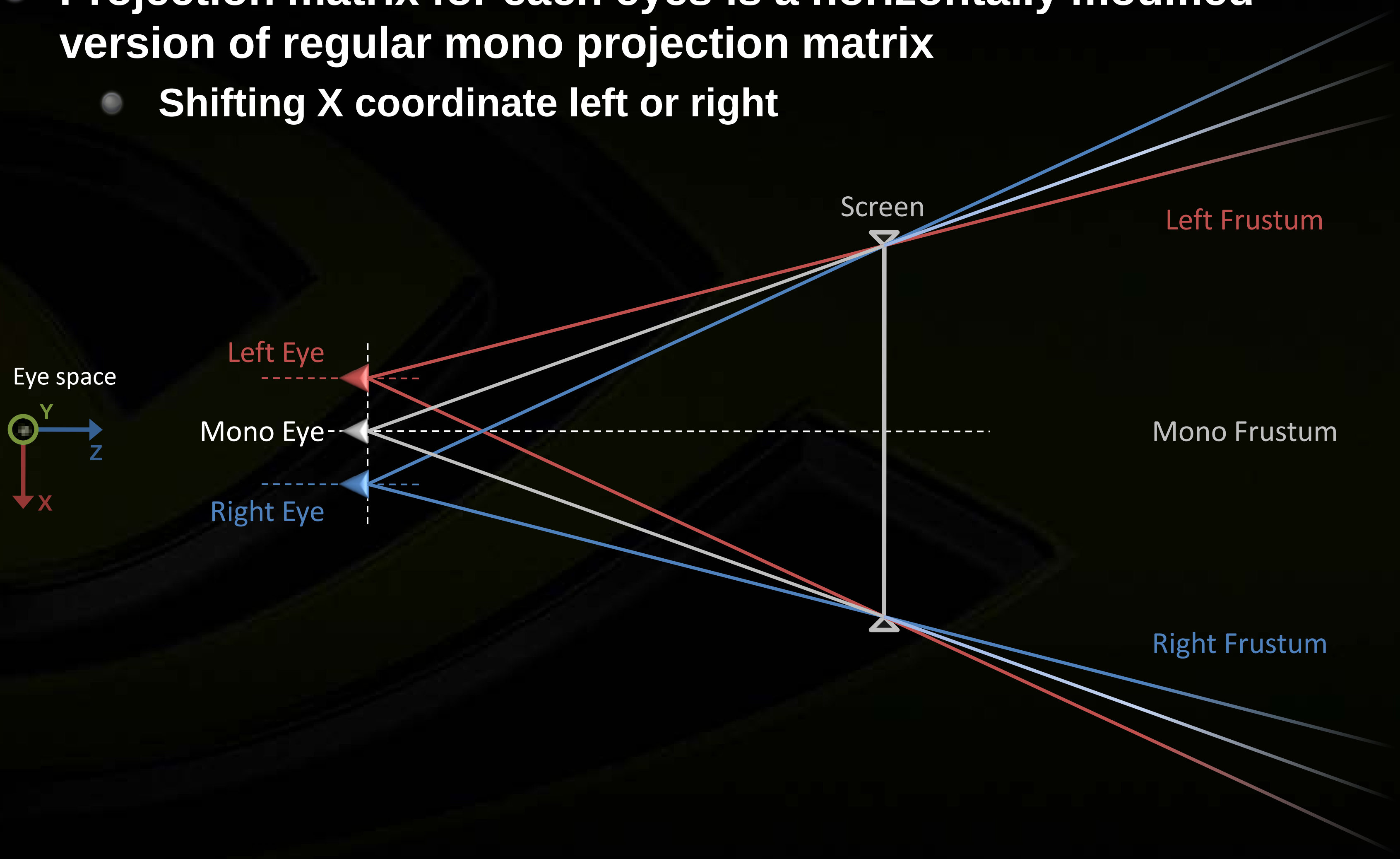
- Also called “Convergence”
- Screen‘s virtual depth in eye space
- Plane where Left and Right Frustums intersect



Stereoscopy Basics

Left / Right Projection

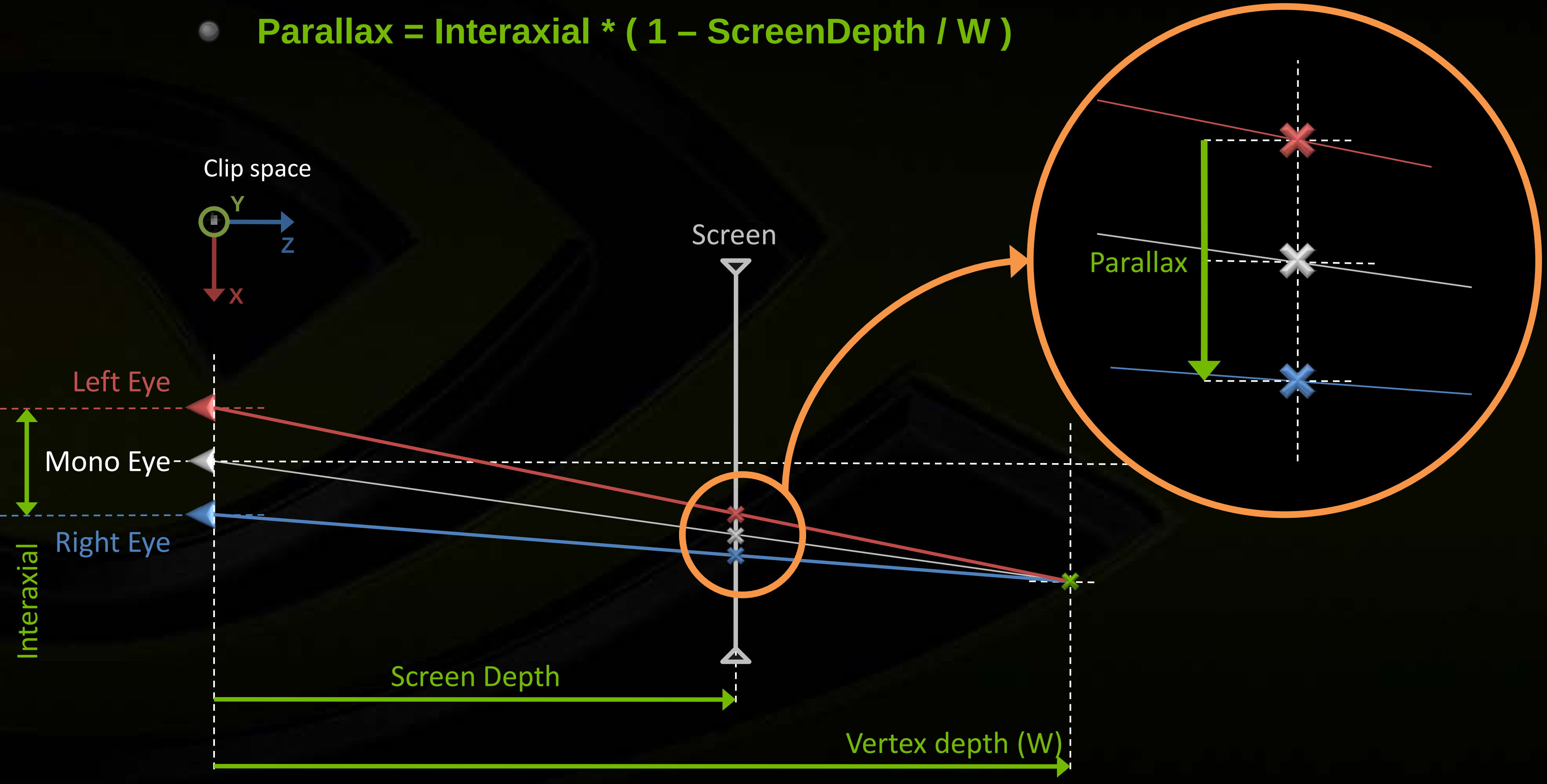
- **Projection matrix for each eyes is a horizontally modified version of regular mono projection matrix**
  - **Shifting X coordinate left or right**



Stereoscopic Basics

Parallax

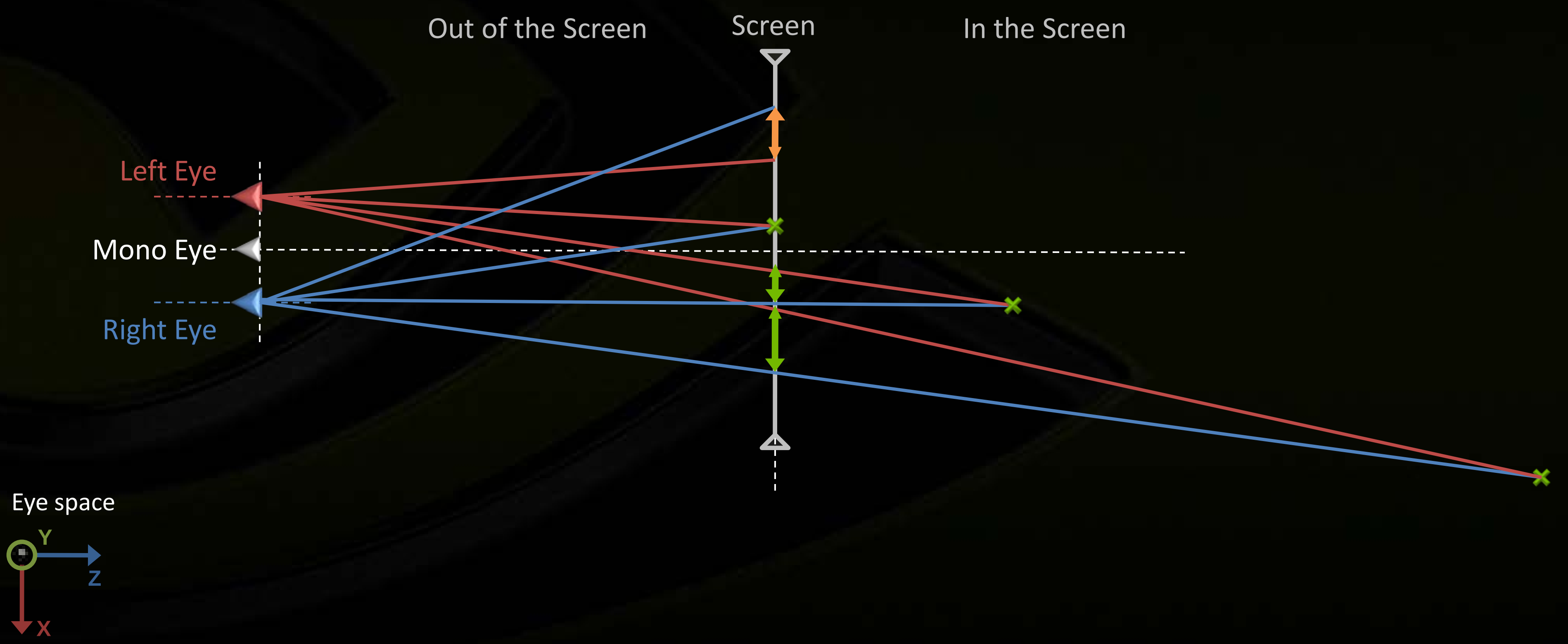
- **Signed Distance on the screen between the projected positions of one vertex in left and right image**
  - **Parallax is function of the depth of the vertex in eye space**
  - **Parallax = Interaxial * (1 – ScreenDepth / W)**



Stereoscopic Basics

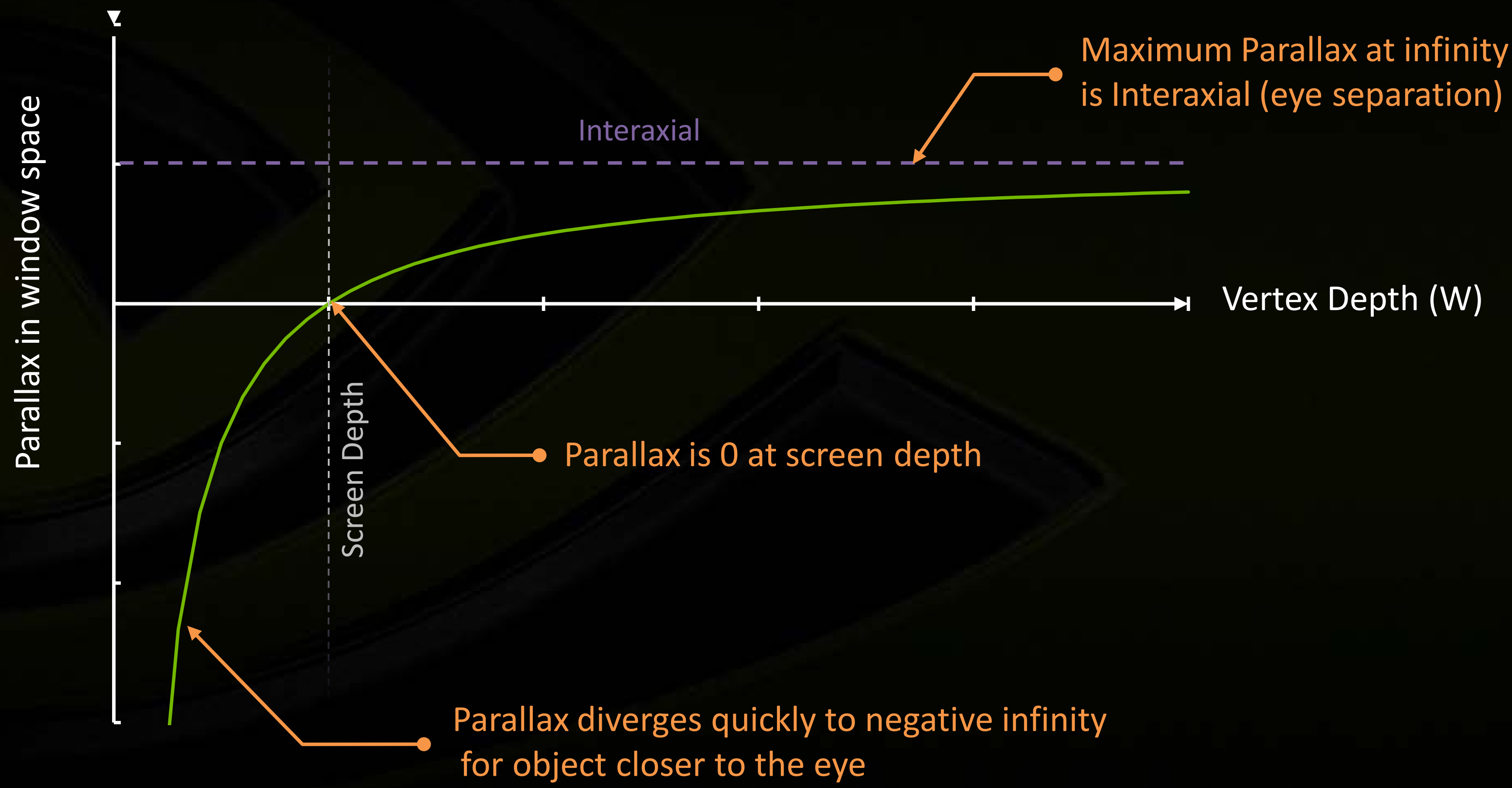
In / Out of the Screen

- **Parallax creates the depth perception relative to the screen**
- **When Parallax is negative, vertex appears Out of the screen**



Parallax in equation

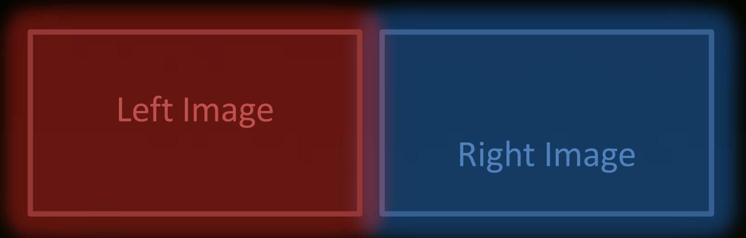
- **Parallax = Interaxial * (1 – ScreenDepth / W)**

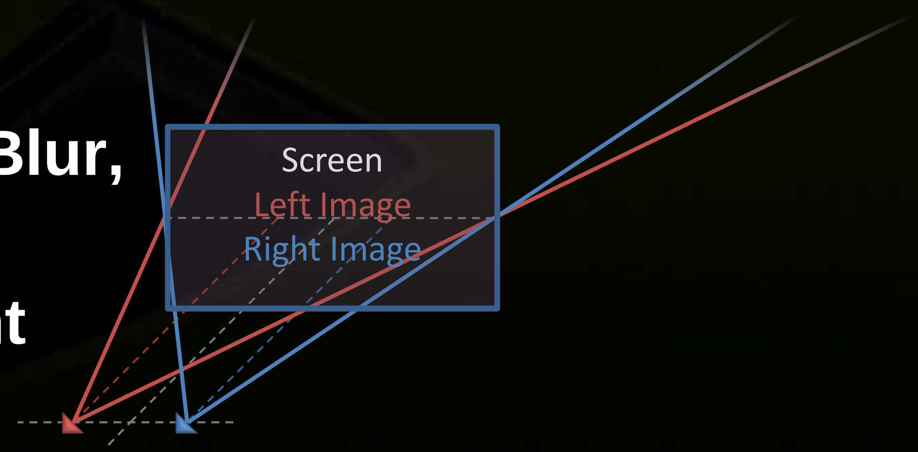


Stereoscopy Basics

Left / Right rendering surfaces

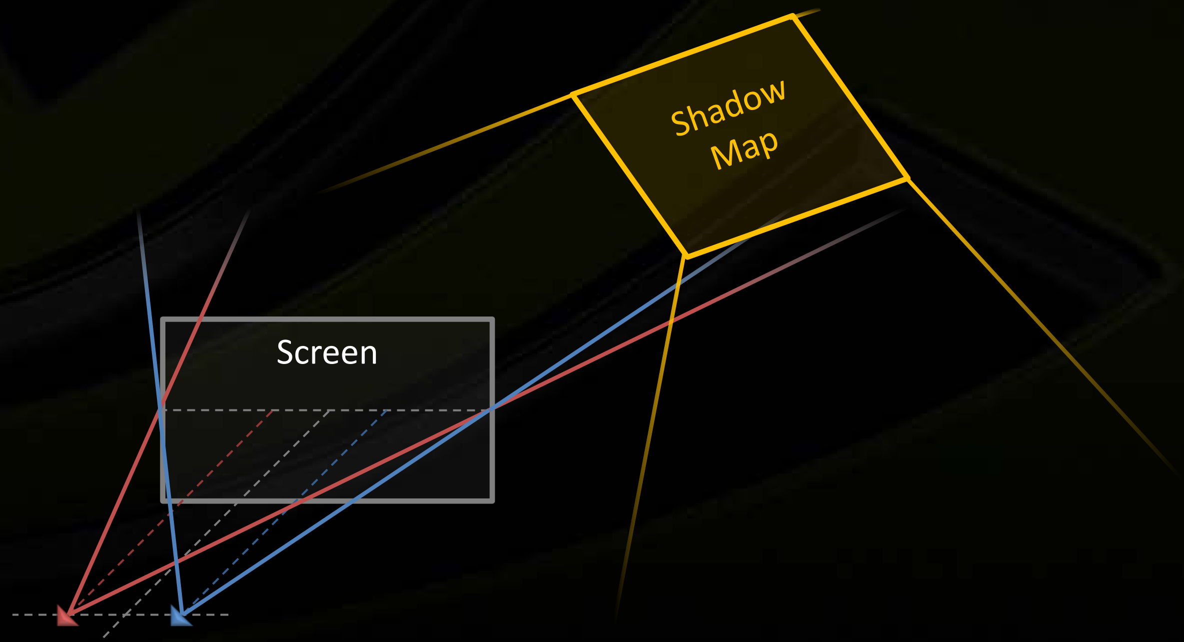
- **View dependent render targets must be duplicated**
  - **Back buffer**
  - **Depth Stencil buffer**
- **Intermediate full screen render targets used to process final image**
  - **High dynamic range, Blur, Bloom**
  - **Screen Space Ambient Occlusion**





Mono rendering surfaces

- View independent render targets DON'T need to be duplicated
  - Shadow map
  - Spot light maps projected in the scene



3D Objects

- All the 3D objects should be rendered using a unique Perspective Projection in a given frame
- The sky box should be drawn with a valid depth further than the regular scene
  - Best is at the Far distance
- Billboard elements (Particles, leaves) should be rendered in a plane parallel to the viewing plane
- All the 3D objects must have a coherent depth relative to the scene

Out of the screen objects

- The user's brain is fighting against the perception of hovering objects out of the screen
  - Extra care must be taken to achieve a convincing effect
- Objects should not be clipped by the edges of the window
  - Be aware of the extra guard bands
- Move object slowly from inside the screen to the outside area to give eyes time to adapt
  - Make smooth visibility transitions
  - No blinking
- Realistic rendering helps

2D Objects

- 2D Overlay elements (defined in window space) must be drawn at a valid Depth
  - At the screen depth to look mono
    - Head Up Display interface
    - UI elements
  - At the correct depth when interacting with the 3D scene
    - Mouse Cursor at the pointed object's depth
      Can not use the HW cursor
    - Crosshair
    - Labels or billboards in the scene
  - The depth is provided by the game engine
- Needs to modify the projection to take into account depth

2D Objects hybrid projection

Proposed vertex shader

```
float4 2DObjects_VertexShader(
    in float2 posClip : POSITION, // Input position in clip space
    uniform float depth           // Depth where to draw the 2D object
    ) : POSITION                  // Output the position in clip space
{
    return float4(
        posClip.xy * depth,   // Simply scale the posClip by the depth
                              // to compensate for the division by W
                              // performed before rasterization

        0,      // Z is not used if the depth buffer is not used
                // If needed Z = ( depth * f – nf )/(f – n);

        depth ); // W is the Z in eye space
}
```

NVIDIA GeForce Stereo driver

Stereo Surface

- Back buffer and Depth Stencil buffer are duplicated
- Automatic duplication of rendering surface is based on the size
  - Surfaces equal or larger than back buffer size are duplicated
  - Square surfaces are NOT duplicated
  - Small surfaces are NOT duplicated
  - Heuristic defined by driver profile setting
    - Consult documentation for fine tuning

Stereo Drawcall

- Drawcall is stereo if rendering surface is stereo
- Drawcalls are issued twice
    - in left surface and in right surface
    - Render target surfaces bound as texture are updated too

Stereoscopic Separation

- If Drawcall is stereo
- Parallax shift is applied in the vertex shader to the vertex's clip position output

- When separation is not required, render geometry at Screen depth
  - Full screen quad to do image filtering

Masaru Ijuuin from Capcom presents

STEREO IN RESIDENT EVIL 5

- Capcom presentation will be added after Siggraph

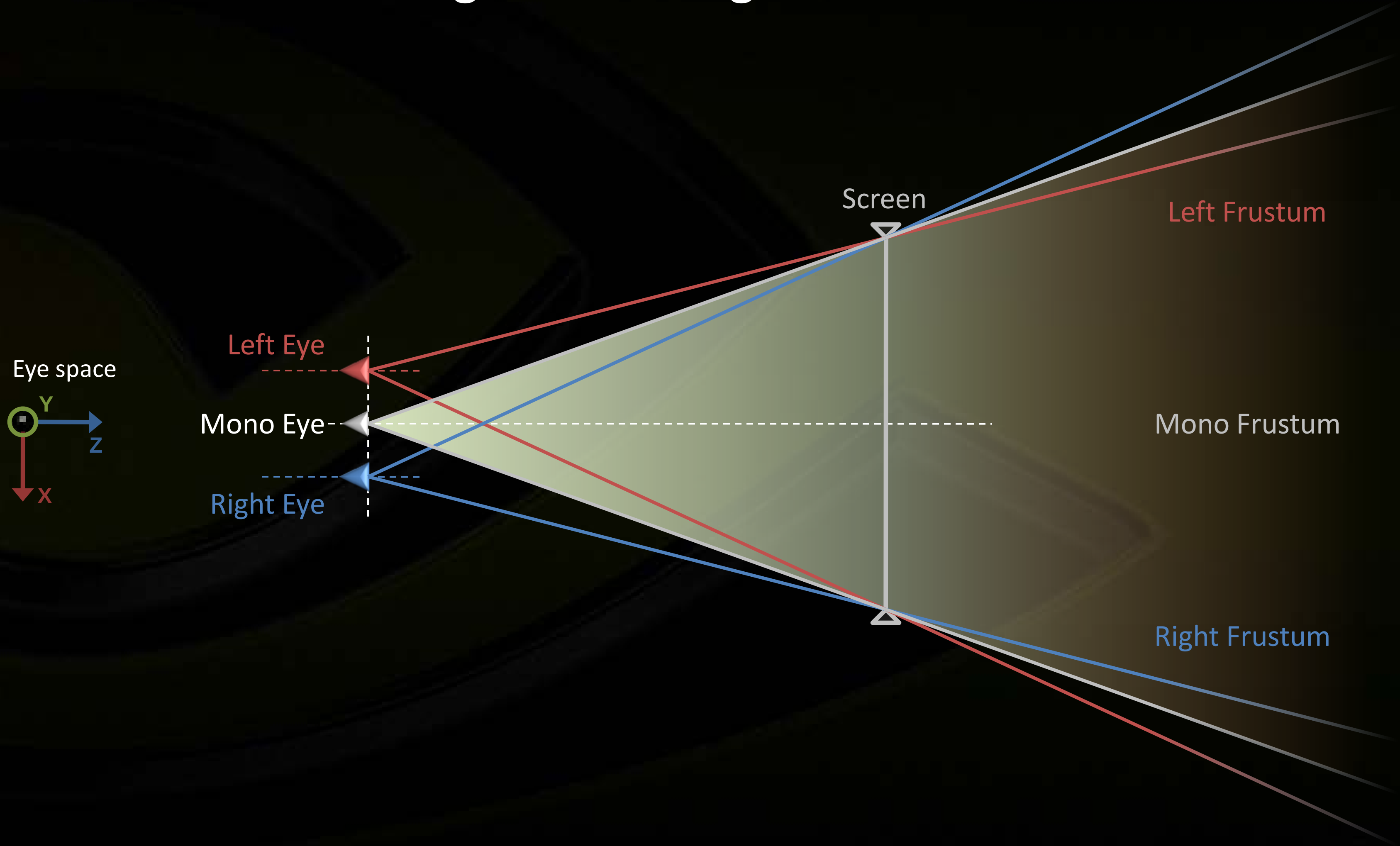
Controlling stereo parameters

THE LOOK OF DEPTH

Rendering Techniques

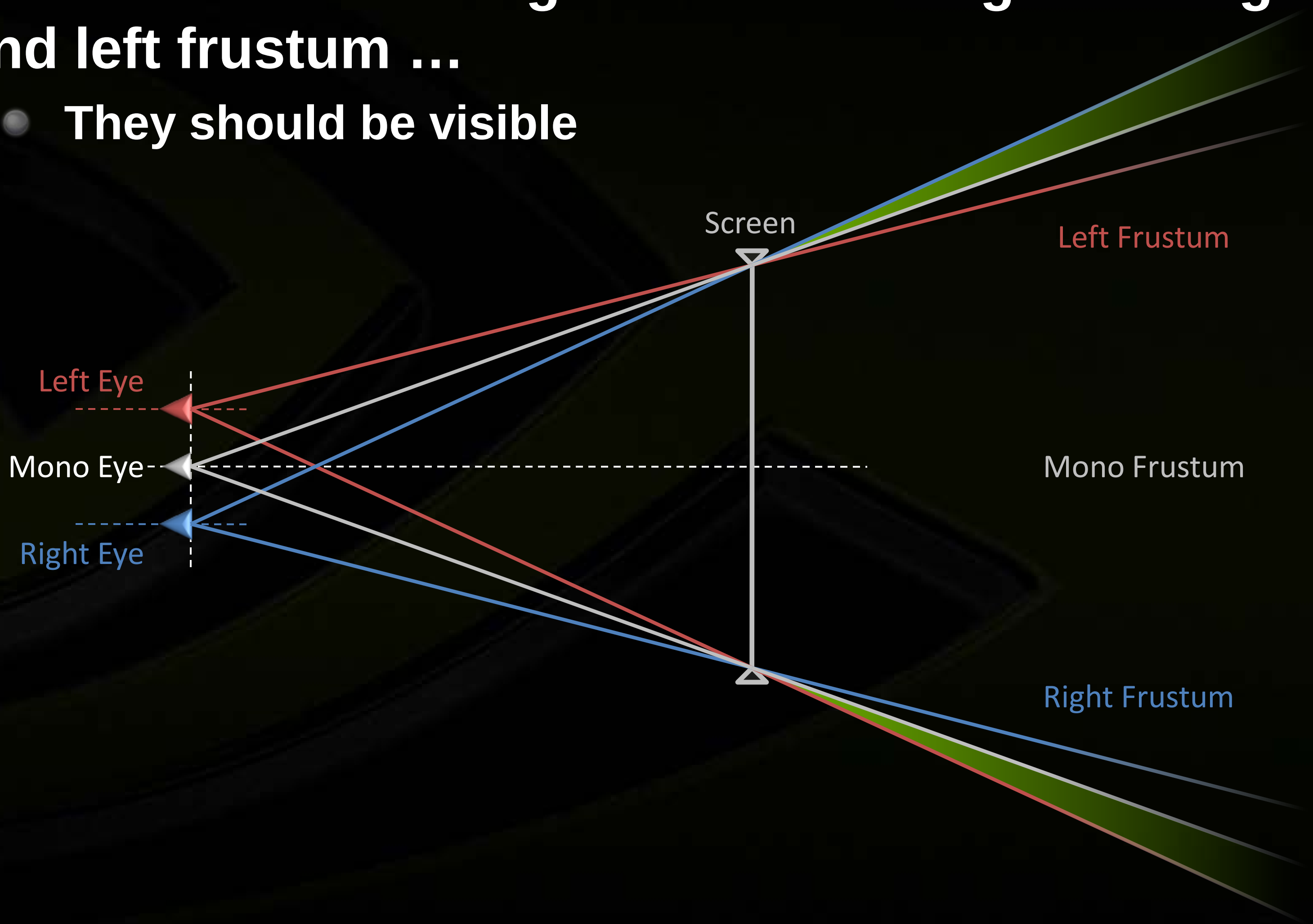
3D Objects Culling

- When culling is done against the mono frustum…



3D Objects Culling

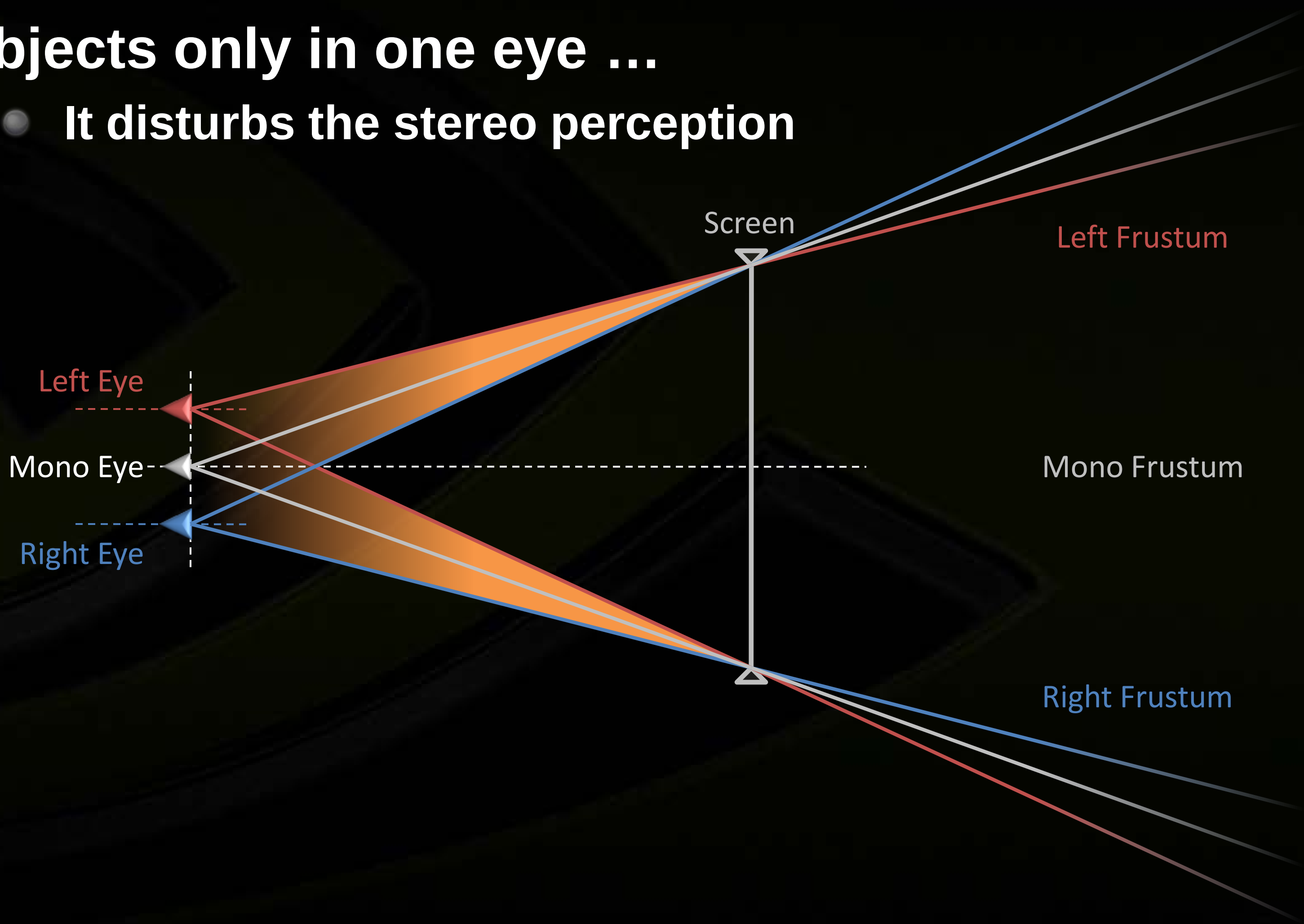
- … Some in screen regions are missing in the right and left frustum …
  - They should be visible



Rendering Techniques

3D Objects Culling

- … And we don't want to see out of the screen objects only in one eye …
  - It disturbs the stereo perception

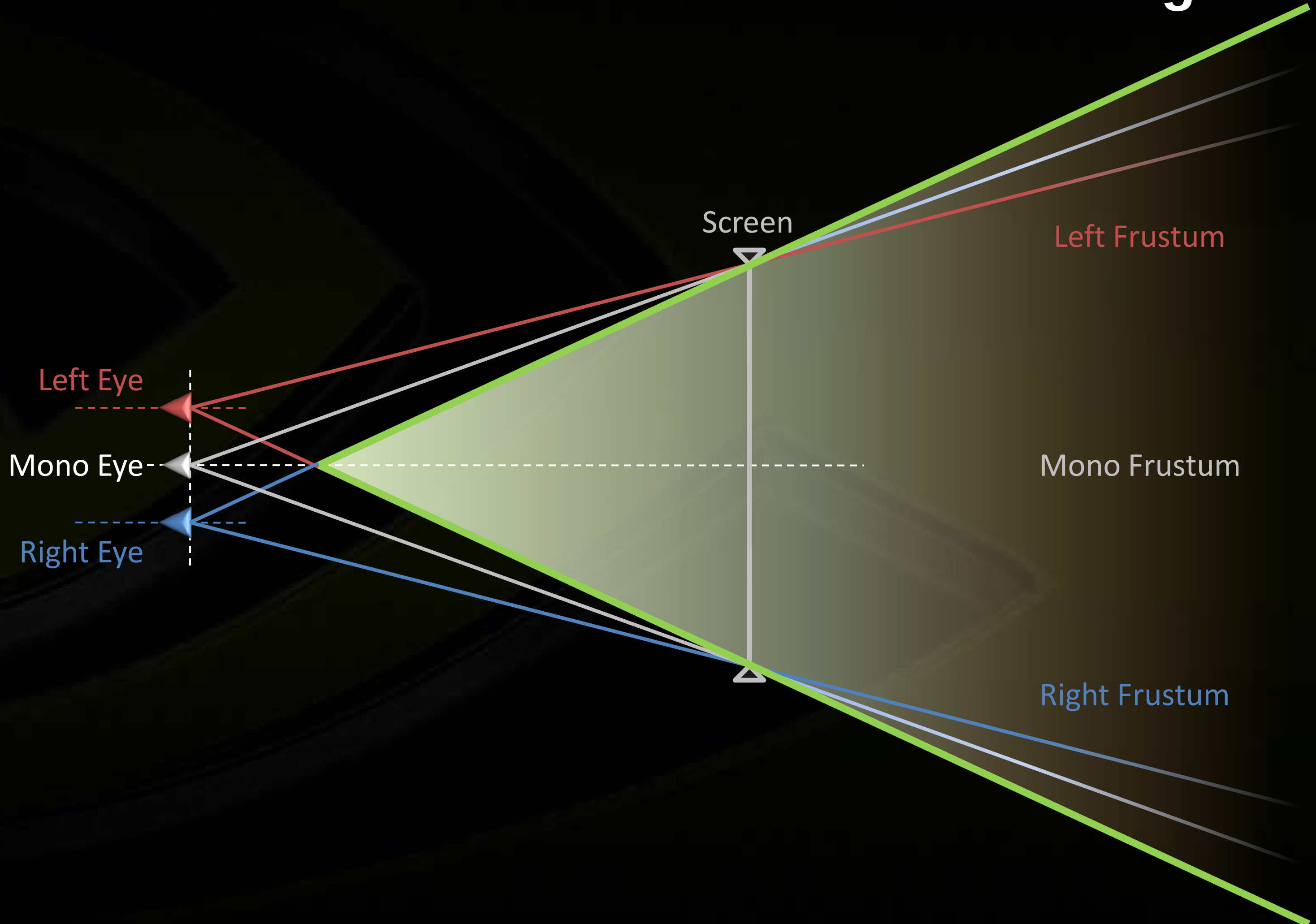


Rendering Techniques

3D Objects Culling

Stereo Frustum for culling

- **Here is the frustum we want to use for culling**

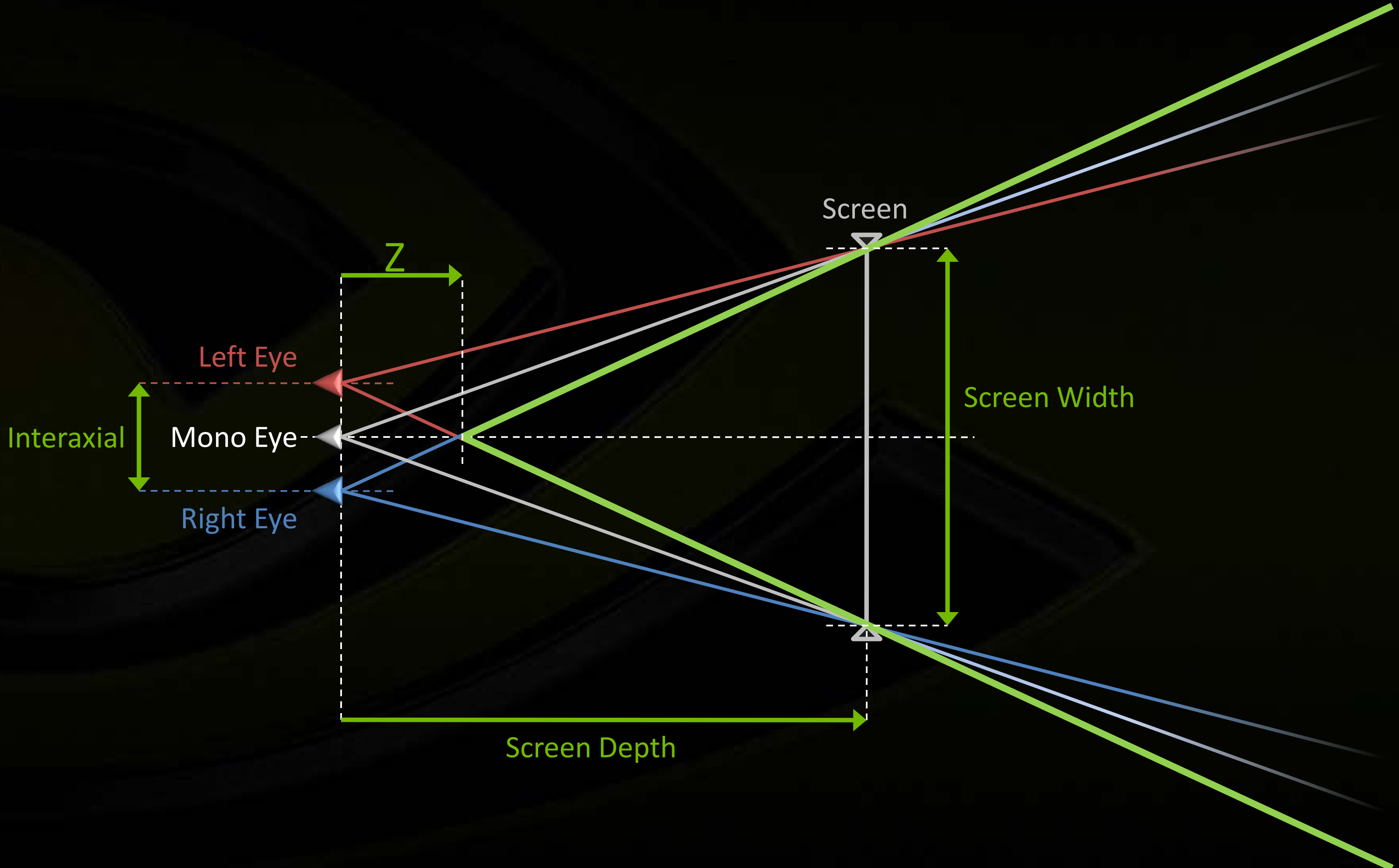


Rendering Techniques

3D Objects Culling

Computing Stereo Frustum origin offset

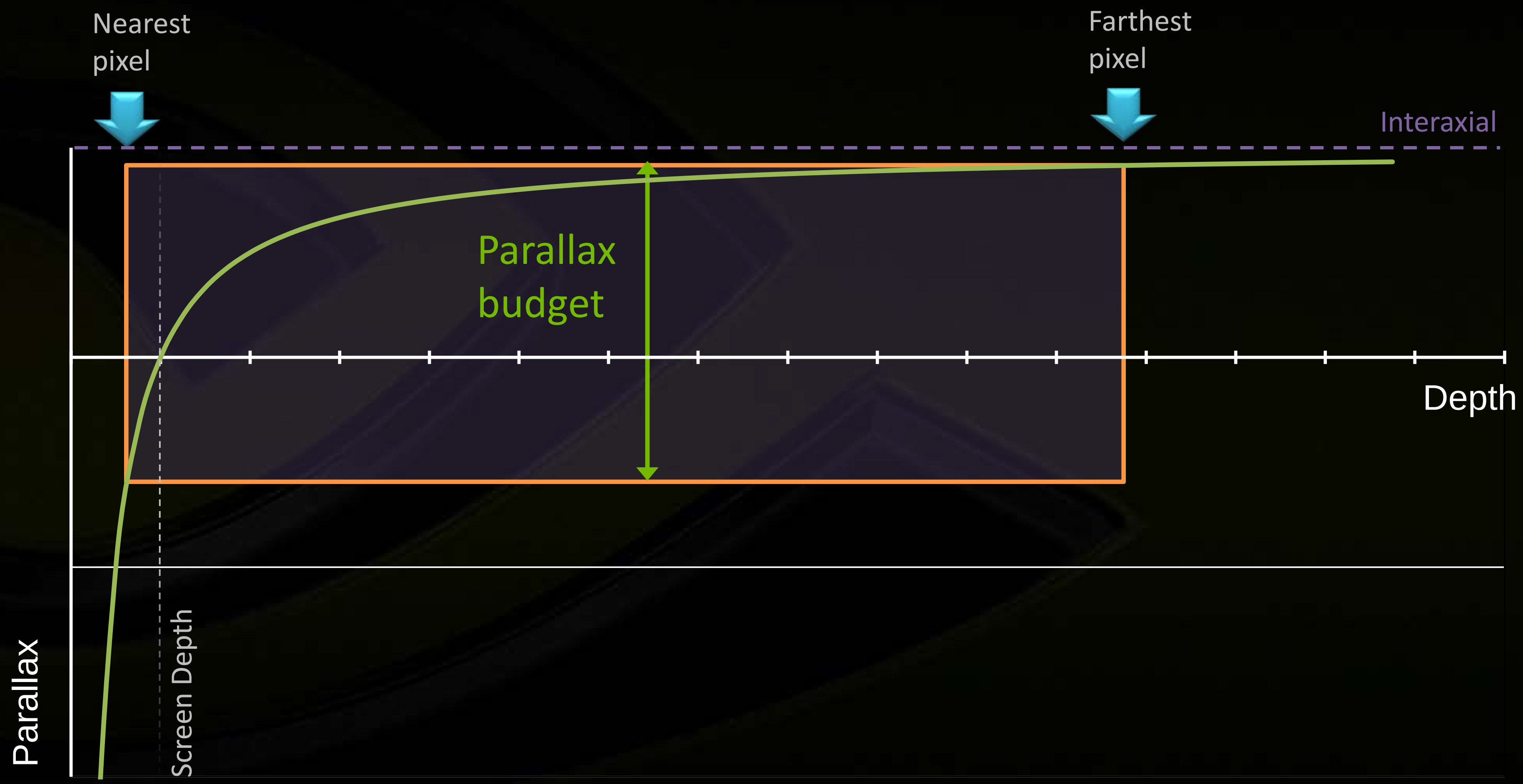
- **Z = ScreenDepth / (1 + ScreenWidth / Interaxial)**



Using Depth

Parallax Budget

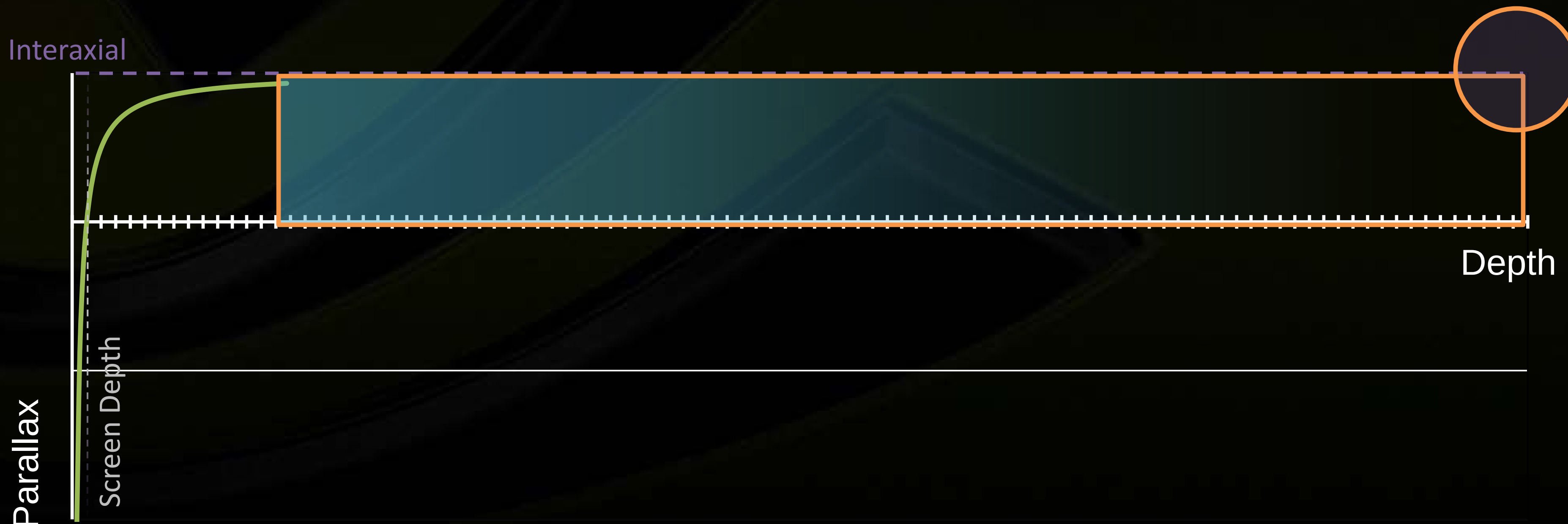
- **How much parallax variation is used in the frame**



Parallax Budget

Farthest Pixel

- At 100 * ScreenDepth, Parallax is 99% of the Interaxial
  - For pixels further than 100 * ScreenDepth,
    Elements looks flat on the far distance with no depth differentiation
- Between 10 to 100 * ScreenDepth, Parallax vary of only 9%
  - Objects in that range have a subtle depth differenciation

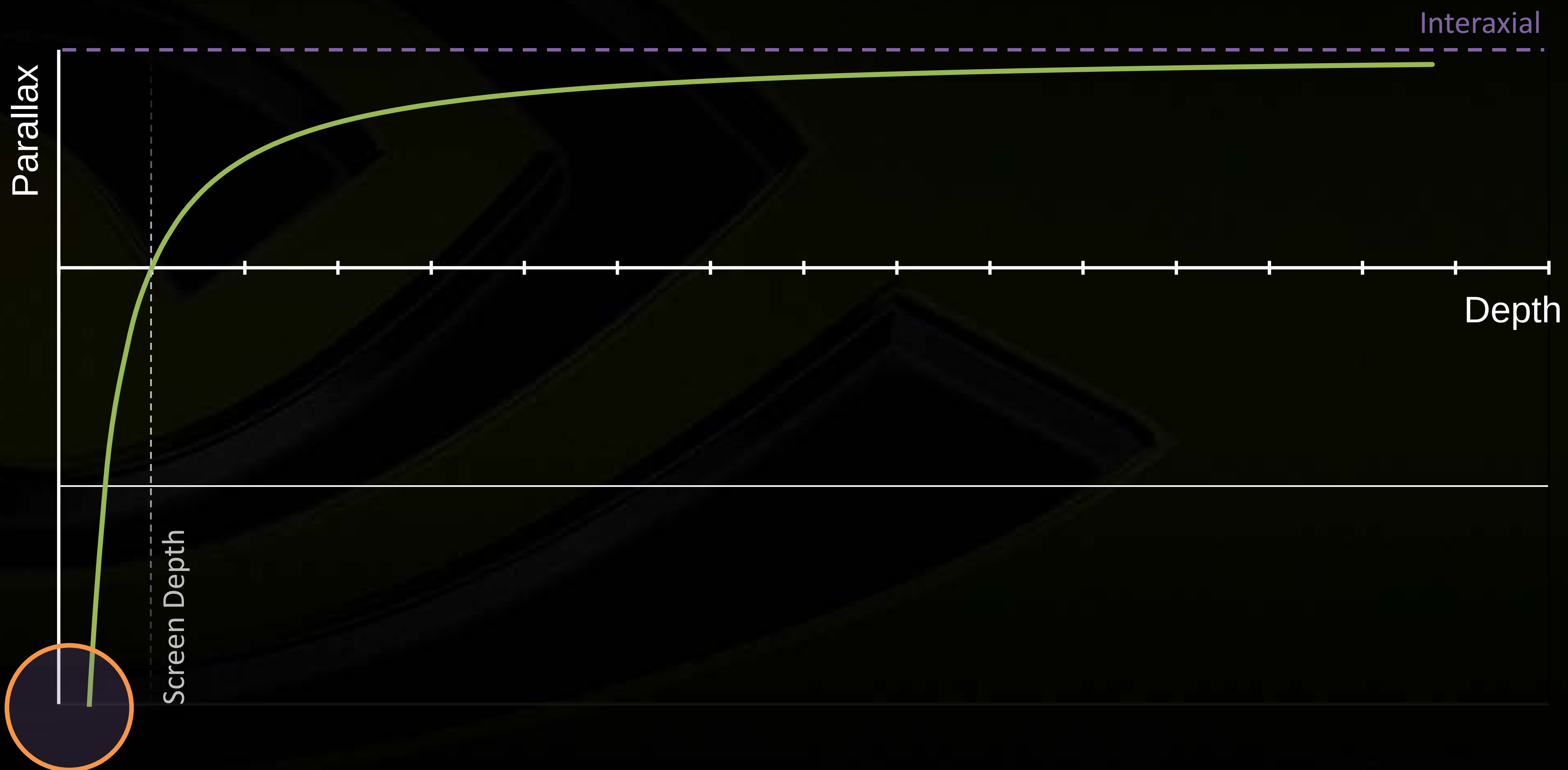


Using Depth

Parallax Budget

Nearest pixel

- At ScreenDepth / 3, Parallax is 2 * Interaxial, out of the screen
  - For pixels closer than ScreenDepth / 3,
    Parallax is very large (> 2*Interaxial) and can cause eye strains

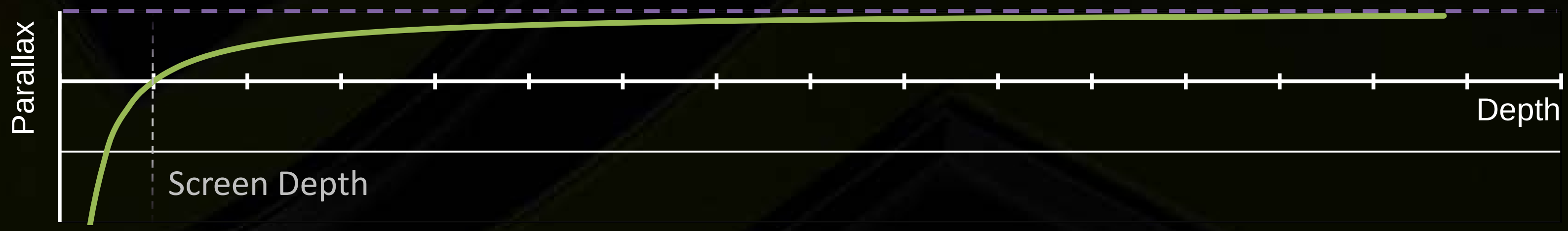


Defining Interaxial

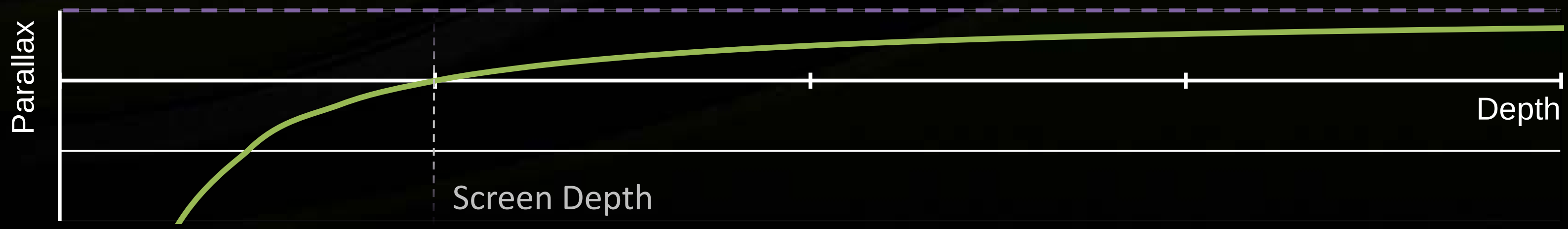
- Interaxial directly defines the amount of depth perception and gives a notion of scale to the scene
- Realistic Interaxial is ideal for first person camera
  - Viewer feels the scene is real
  - Make sure that the position and field of view map the reality of the player setup (field of view, screen size, distance to the screen)
- Large Interaxial makes the scene looks smaller
  - Viewer feels like a giant in front of a small world
  - Good for overlooking camera (RTS)
- Small Interaxial make the scene looks larger
  - Viewer feels very small compared to the scene
  - Flatten 3d effect
- Should be dynamically adjustable for the user comfort
  - Relative to an “ideal” reference value designed for the camera

Defining Screen Depth

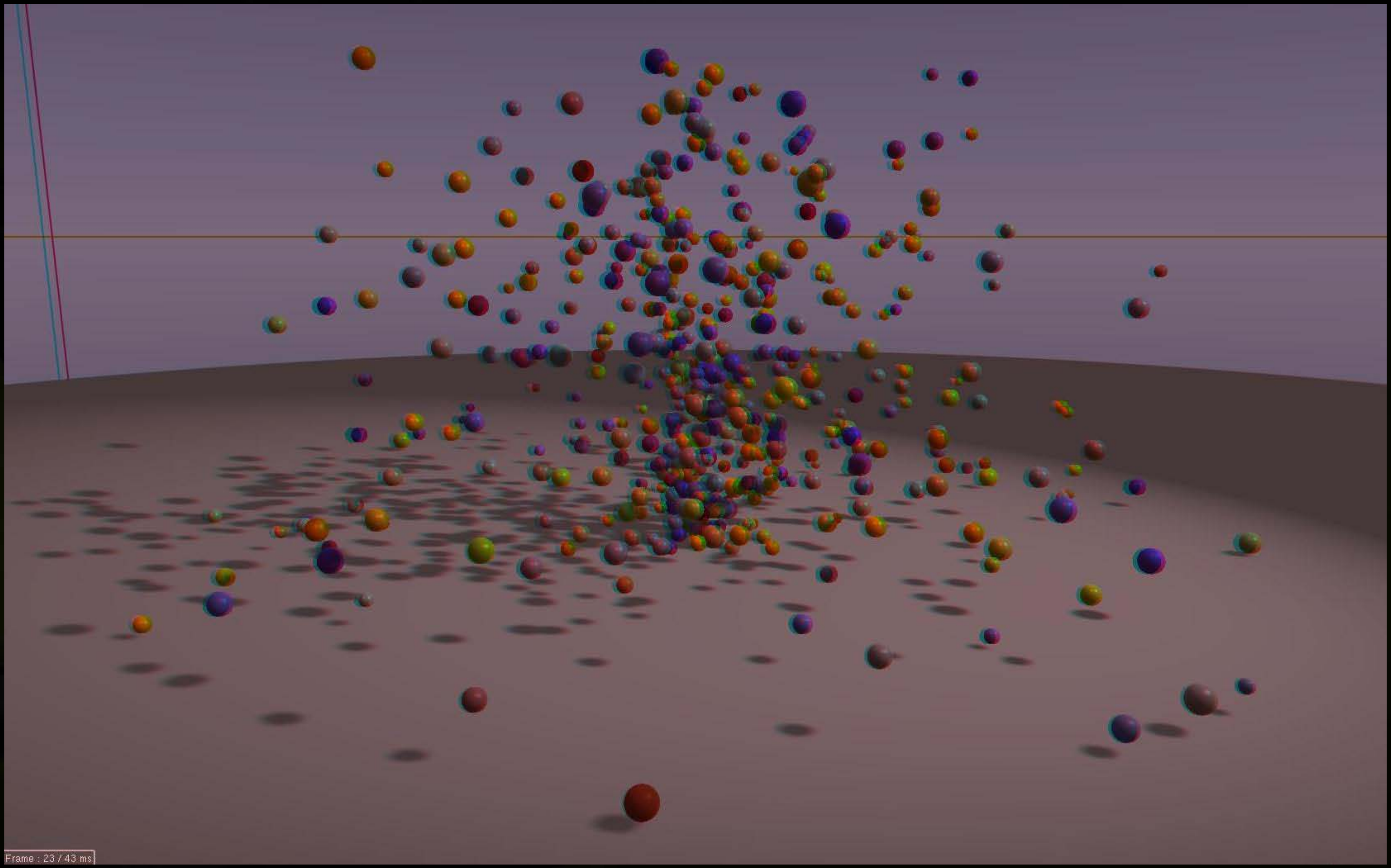
- Screen Depth should be defined by the application depending on the camera and the scene
- Make sure the scene elements are in the range [ScreenDepth / 3, 100 * ScreenDepth]
  - Full range to maximize the 3D perception



  - Smaller range for a more subtle usage of the parallax budget




Sheva
COVER
Chris


Frame : 23 / 43 ms


₦200
Sheva
COVER
Chris
What the fuck are those!? Mathison to
HQ! I'm under attack by flying B.O.W.s!


Sheva
COVER
Chris

QUESTIONS ?

After siggraph
sgateau@nvidia.com

Acknowledgements

- Rod Bogart & Bob Whitehill at Pixar
- Every one in the Stereo driver team !
- The usual suspects in demo and devtech team

How To Reach Us

- **Online**
  - **Twitter: nvidiadeveloper**
  - **Website: http://developer.nvidia.com**
  - **Forums: http://developer.nvidia.com/forums**