[GDC] Festival of Gaming

MARCH 9 - 13, 2026
SAN FRANCISCO, CA

Technical Artist Summit

Tech-Artists.org

Join our Slack!

#GDC2026

Please Remember to:

- Not use flash
- Mute your phone
- Be kind and courteous to our speakers

HOW WE DRAW A 3D SPRITE WORLD:

The Stylized Art of Never's End

Ryan Juckett

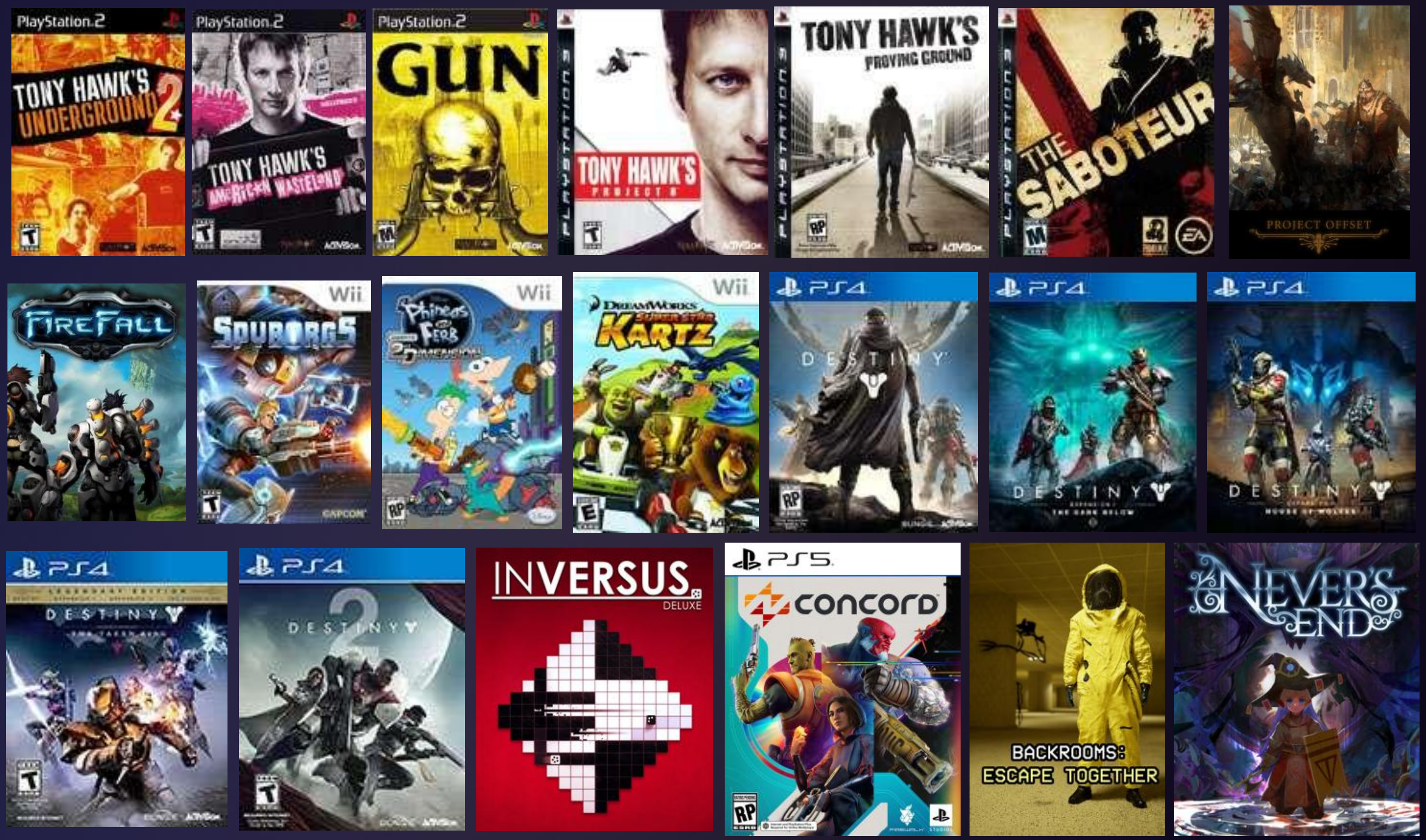
Director, Hypersect

ABOUT ME

- Ryan Juckett

  ryanjuckett@hypersect.com

- Game Director at Hypersect
- Over 20 years in games



ABOUT THE GAME

Never's End

- 3D turn-based tactical RPG
- Hand-drawn 2D sprite look
- Highly dynamic world

WHY?

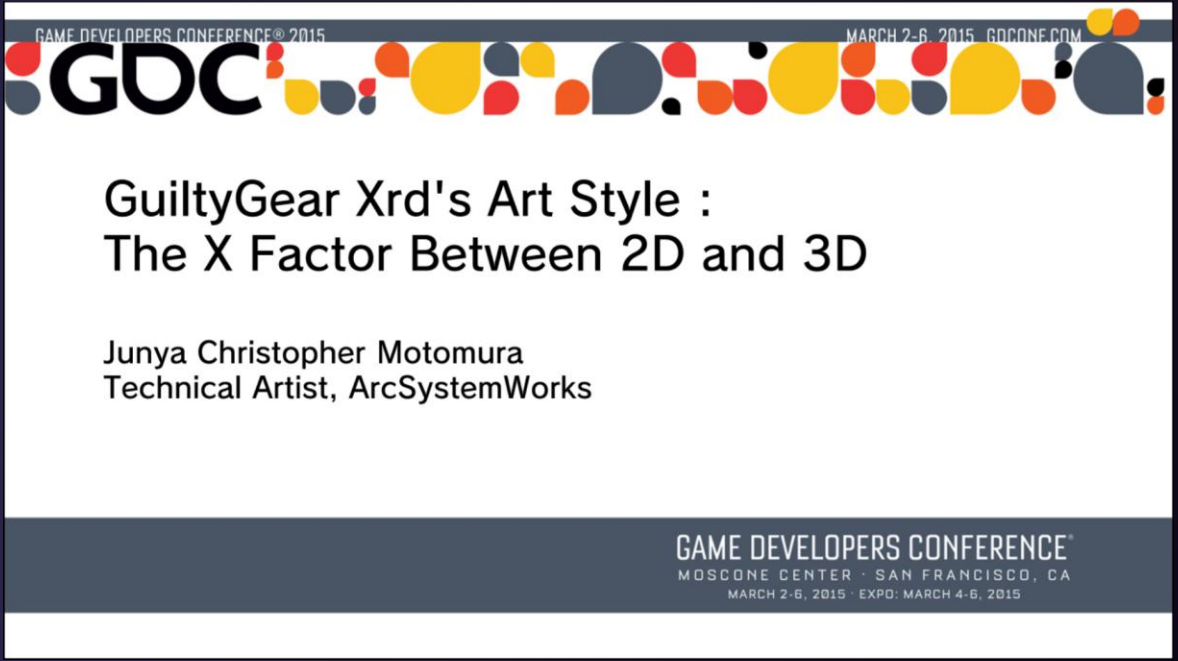
- Stand out of the competition
- Create a visual style not otherwise possible
- Richly animated sprites with customizable gear
- Full shadow and lighting support

DEVELOPMENT

- 7.5 Years
  - First 5 solo + part-time contractors
  - Now 4 full time and 4 part time
- Custom Engine / Editor
- Tools: Maya, Blender and Photoshop

PAST WORK

- Foundation builds on top of the work of ArcSystemWorks for GuiltyGear Xrd.
- See 2015 talk:
  GuiltyGearXrd's Art Style : The X Factor Between 2D and 3D



HOW WE DID IT

- Linework
- Shading
- Pixel Snapping
- Sprite Sorting
- Animation

HOW WE DID IT

High resolution, smooth shaded, perspective render.

LINEWORK

LINEWORK: OVERVIEW

- Rendered as post process.
- Clean single pixel width reads hand drawn.

OFF

ON

Base Color

Base Linework

Translucent Overlays

Selection

Terrain Water

Terrain Water
Linework

LINEWORK: OUTLINES

- Used to separate elements in the scene.
- Three separate outline types

Object Outlines

Depth Outlines

Material Outlines

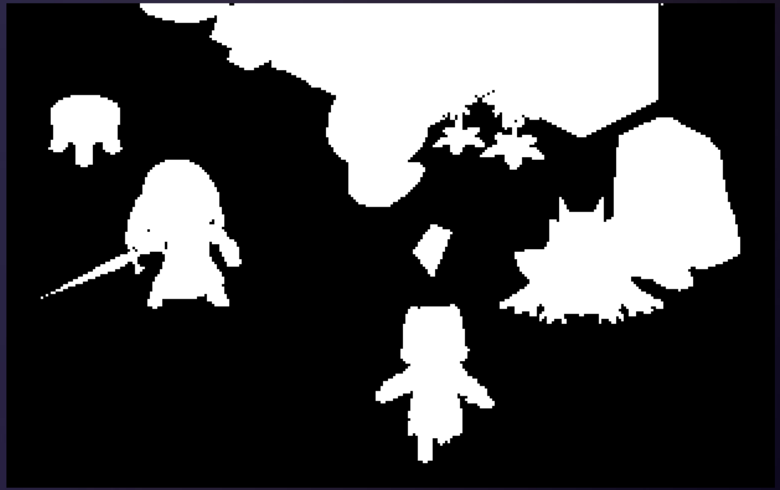
LINEWORK: OBJECT OUTLINES

Specified per entity:

- Share parent's Object ID
- Generate outline around my Object ID

Object ID (16 bits)

Enabled Mask

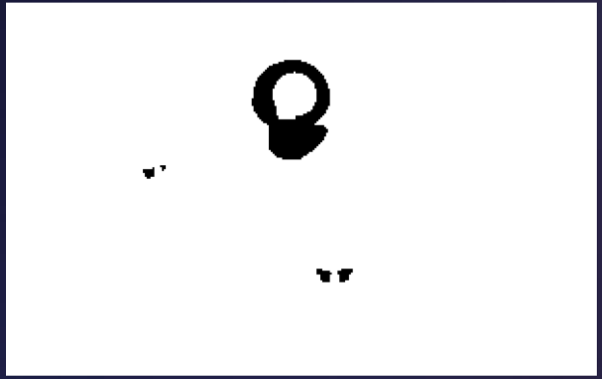


LINEWORK: DEPTH OUTLINES

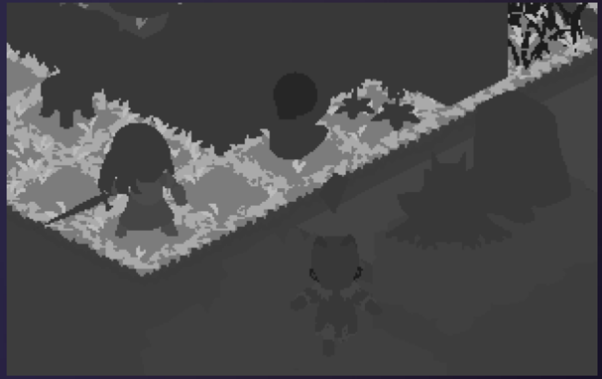
- Material decides if it can generate depth-based outlines.
- Materials specify a baseline depth threshold.
- Normals adjust depth threshold at steep angles.

Depth

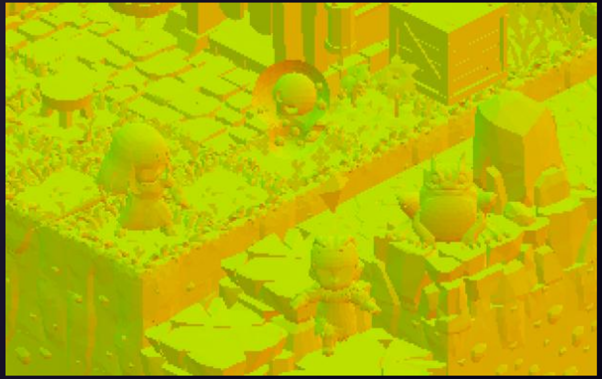
Depth Outline Mask



Depth Threshold



Hard Normals



LINEWORK: MATERIAL OUTLINES

- Material ID marks sections to outline.
- Mask disables outline generation.
- Parameters set by material property or texture data.

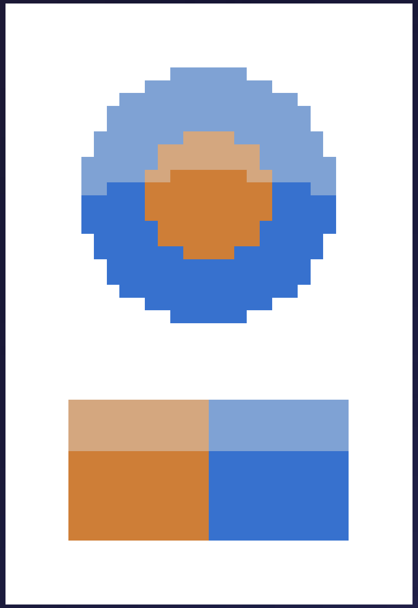
Material ID

Material Outline Mask

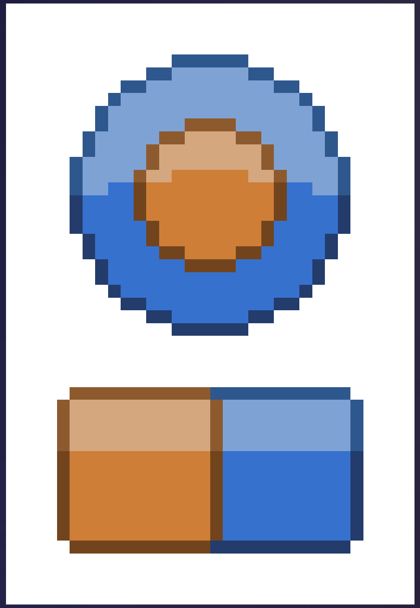
LINEWORK: OUTLINE PRIORITY

- Need to choose which side of a boundary gets the outline.
- Makes one side visually larger and appear in front.

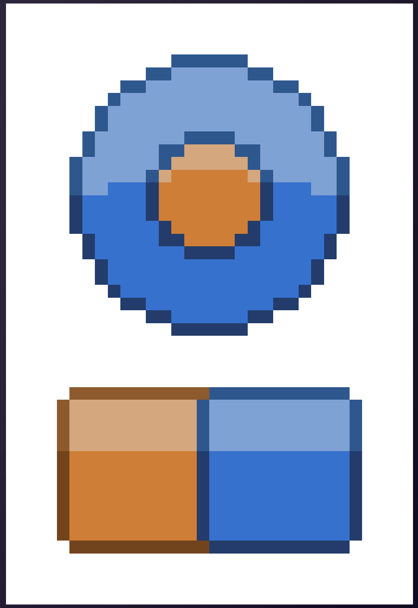
No Outline



Option A



Option B



LINEWORK: OUTLINE PRIORITY

Pri 1: Object

- Object IDs are depth sorted. Outline nearest object.

Pri 2: Depth

- Outline nearest pixel

Pri 3: Material

- Outline nearest pixel if greater than depth threshold (only needed if Depth outlines are disabled)
- Outline lowest Material ID otherwise

LINEWORK: OUTLINE COLORS

Potential outline colors rendered to secondary color buffer.

- Defaults to diffuse lighting result with per-material tint.
- Optional separate color texture for outline color calculation
- Optional blend towards a constant flat color

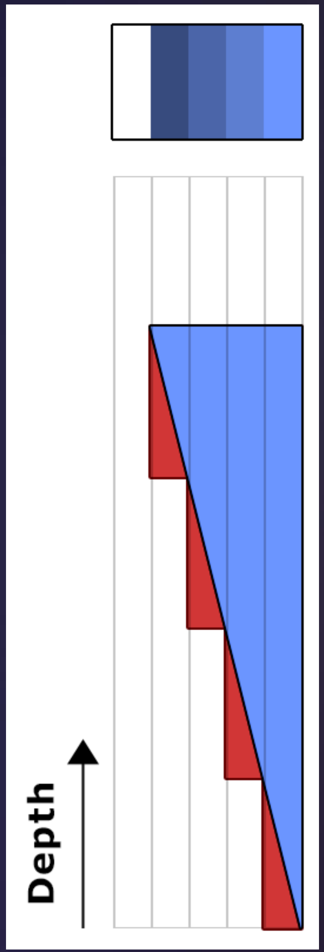
LINEWORK: OUTLINE EXTRUSION

- Extrude depth buffer and all gbuffers for future render passes.
- Depth extension needs to follow geometry plane of pixel being outlined.

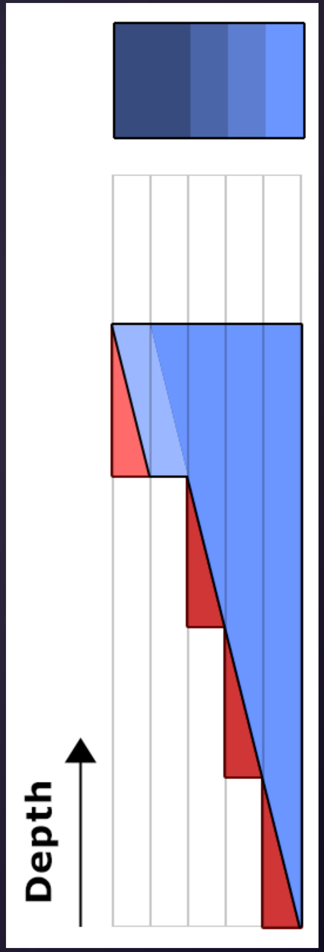
Starting Depth

Extruded Depth

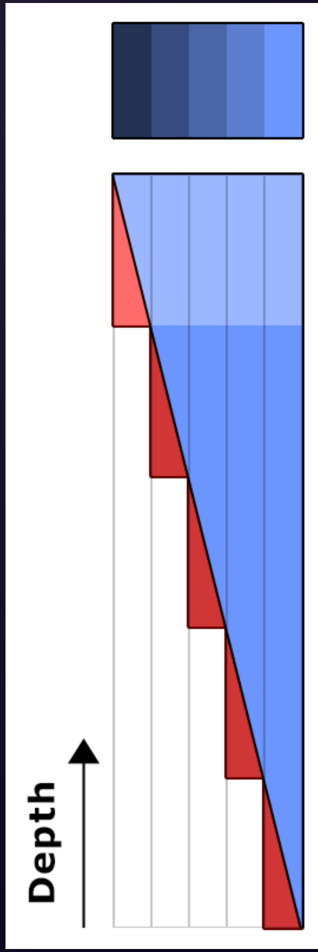
Starting Depth



Naïve Copy Extrusion



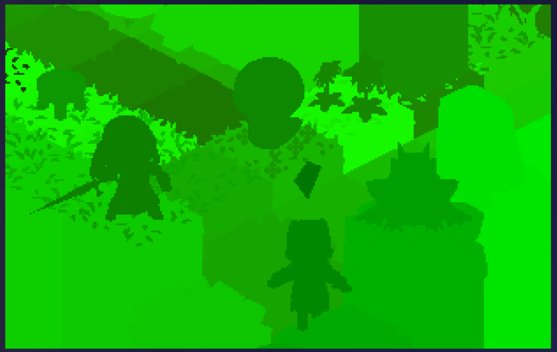
Smooth Plane Extrusion



LINEWORK: INTERIOR OUTLINES

- When Object ID matches, but Material ID doesn't, materials can blend at partial opacity.
- Object Section IDs can further divide a full object into sections of opaque outlines.

Object ID



Material ID

Object Section ID

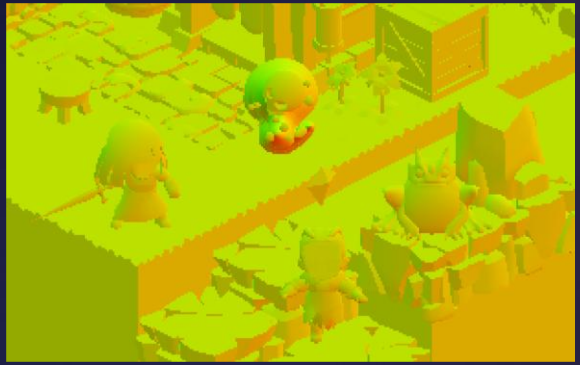
Interior Blend Weight

LINEWORK: INLINES

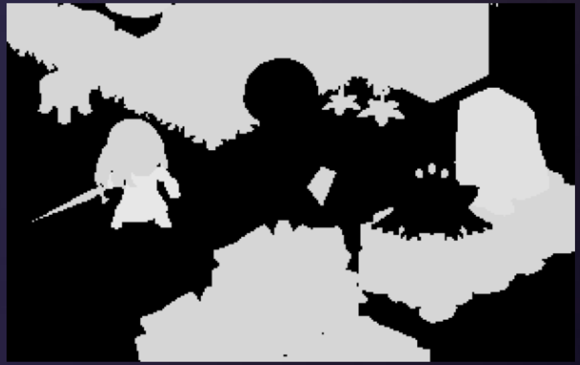
- Detect ridges using lighting normals and depth.
- Materials render an angle threshold and brightness factor.
- Choose the brighter side (more in direction of primary light).

Depth

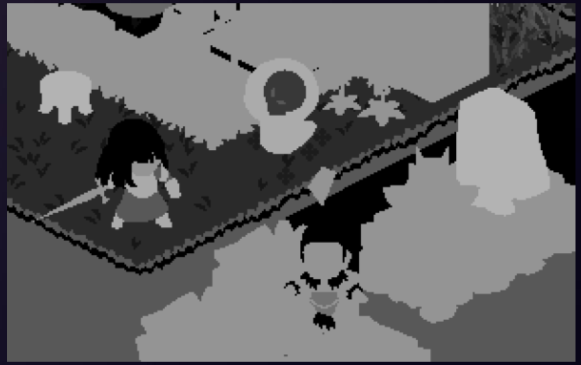
Smooth Normals



Angle Threshold



Brightness



LINEWORK: SELECTION OUTLINES

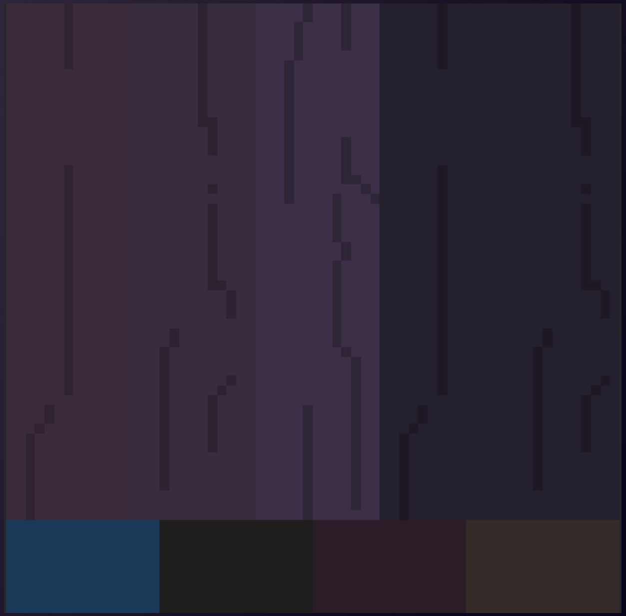
- Second layer of outline based on Selection Category buffer.
- Input is extruded data from first outline pass.
- Selection Category looks up into a table of colors (active character, AOE damage, etc)
- Object ID separates multiple selections in same category.

Object ID

Selection Category

LINEWORK: THICK LINES

- Anything that needs to be more than one pixel is rendered with the scene.
- Carved geometry with a separate colors.
- Nearest-neighbor sampled textures.

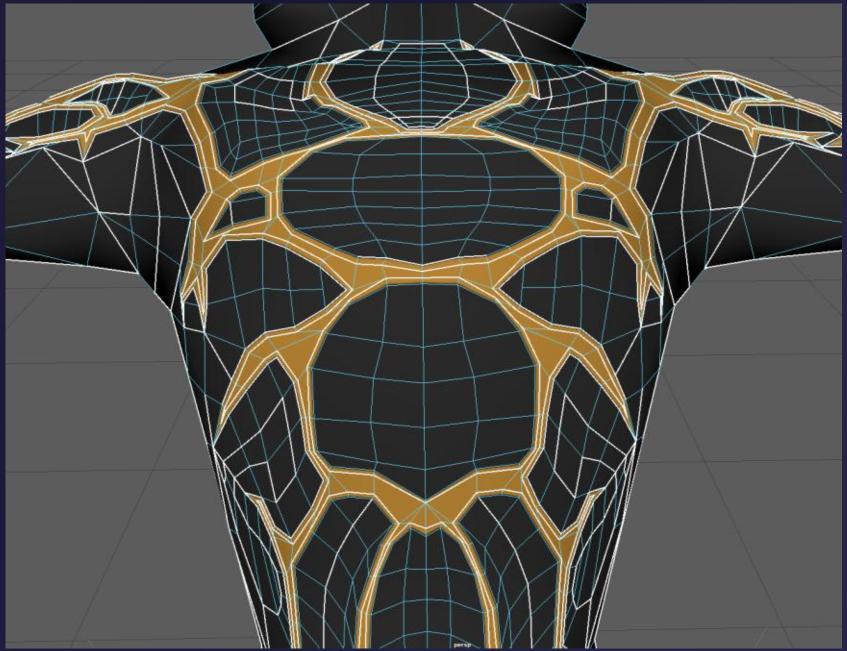


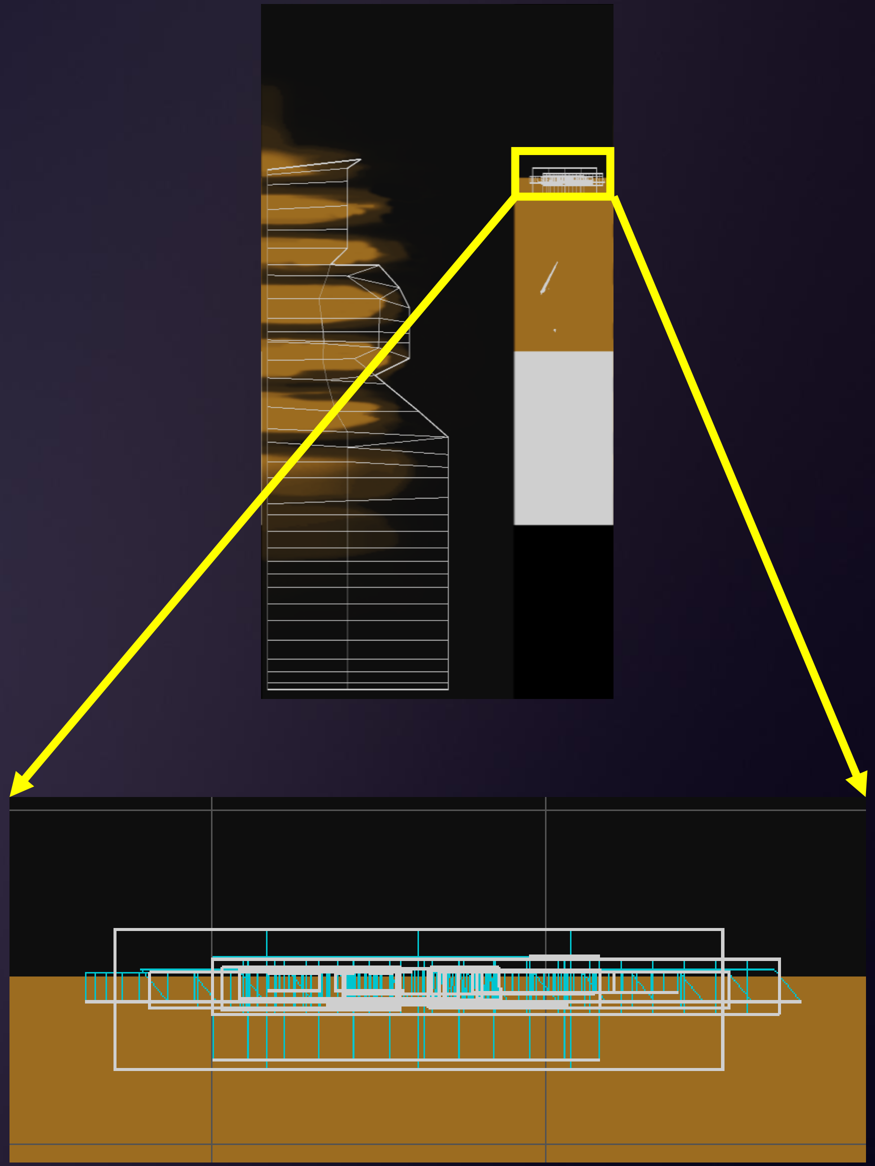
LINEWORK: THICK LINES

- When linework is highly detailed or animated, it may need a softer painted look.
- It must remain sharp, but look like there is some painted anti-aliasing.

LINEWORK: THICK LINES

- Carved geometry around UV line with bilinear sampling.
- UVs are axis-aligned for improved texture sampling results (see GuiltyGear Xrd).





SHADING

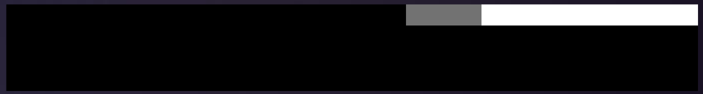
SHADING: TOON RAMP

Three strips of information.

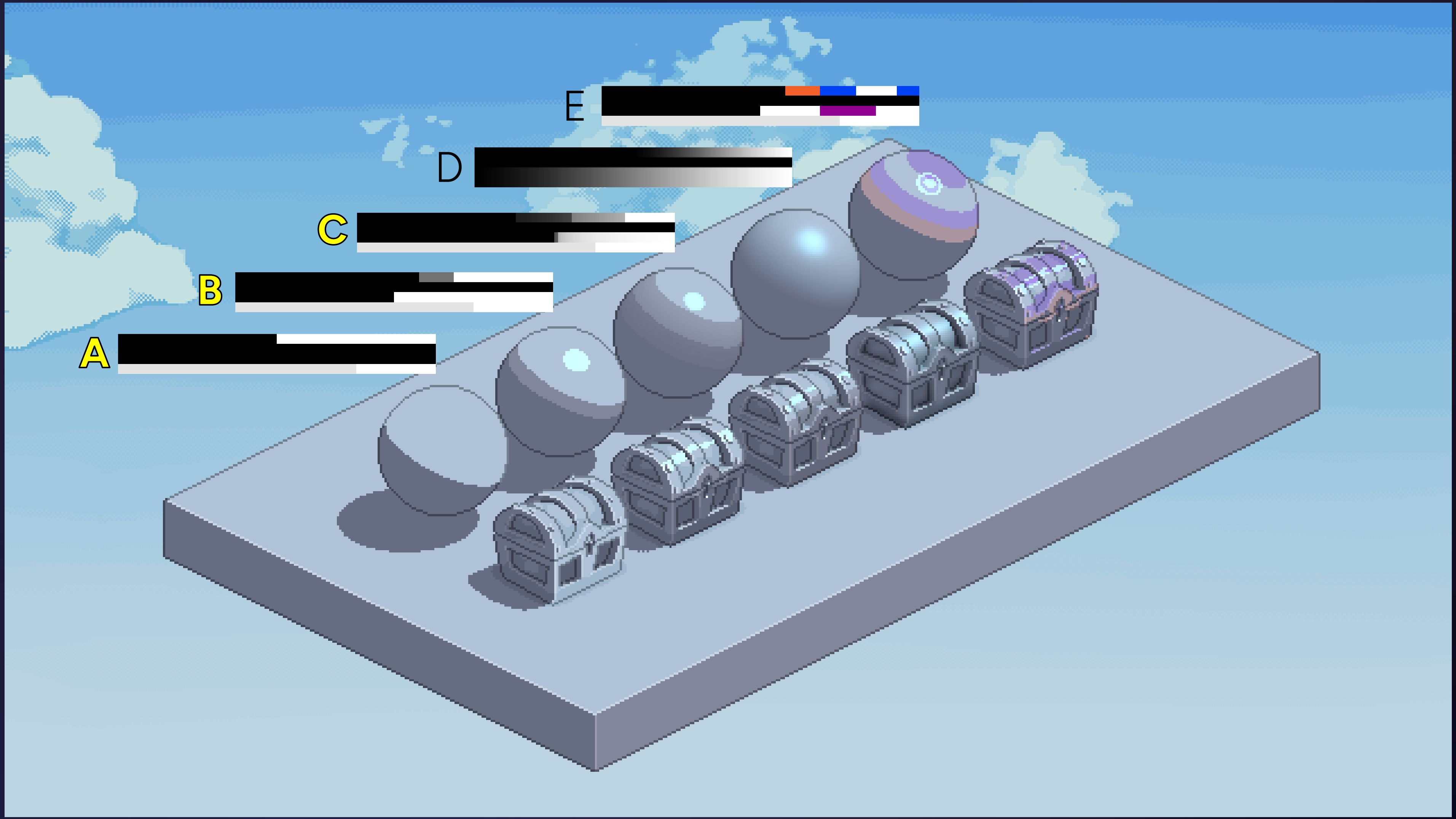
- RGBA Diffuse Ramp
  - Alpha controls “Shade Blend”
- RGB Specular Ramp
- RGB Ambient Occlusion Ramp

Color

Alpha

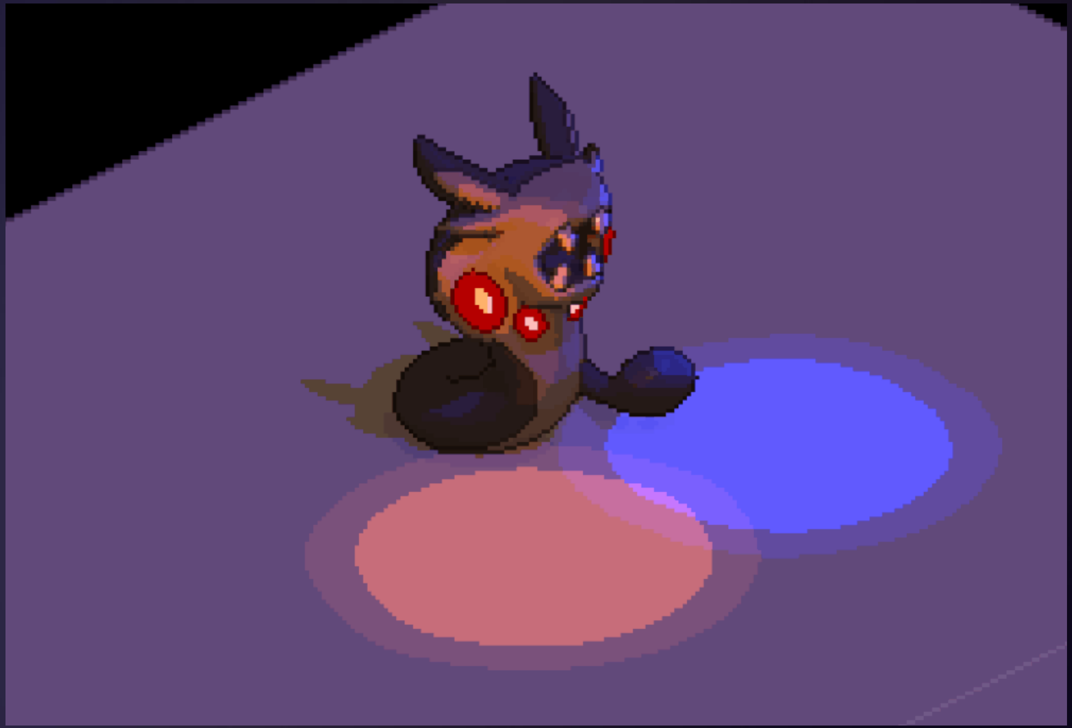


SHADING: TOON RAMP



SHADING: LIGHT SOURCES

- Evaluate toon ramp per light and add
- Distance attenuation scales toon ramp UV towards dark side
- Point light shape
  - Falloff animated by noise.
  - Non-uniform scale



SHADING: SHADE AMBIENT LIGHT

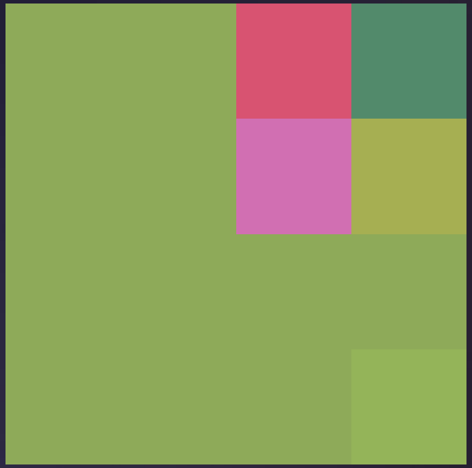
- Time of day specifies separate ambient light color for shade.
- Alpha channel of first row blends to "shade" lighting.
- Cast shadows always considered in full shade.

SHADING: SHADE MATERIAL

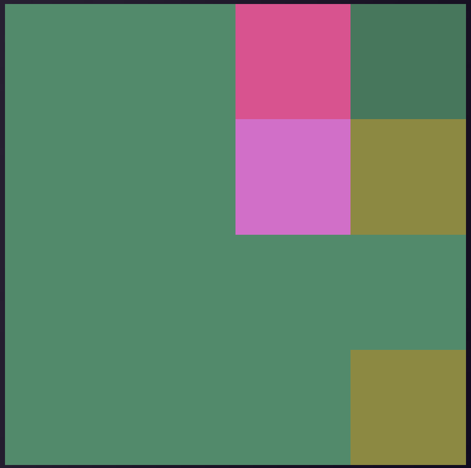
Can also blend material settings in shade.

- Separate albedo texture
- Separate ambient tint
- Separate outline colors

Direct Albedo



Shade Albedo



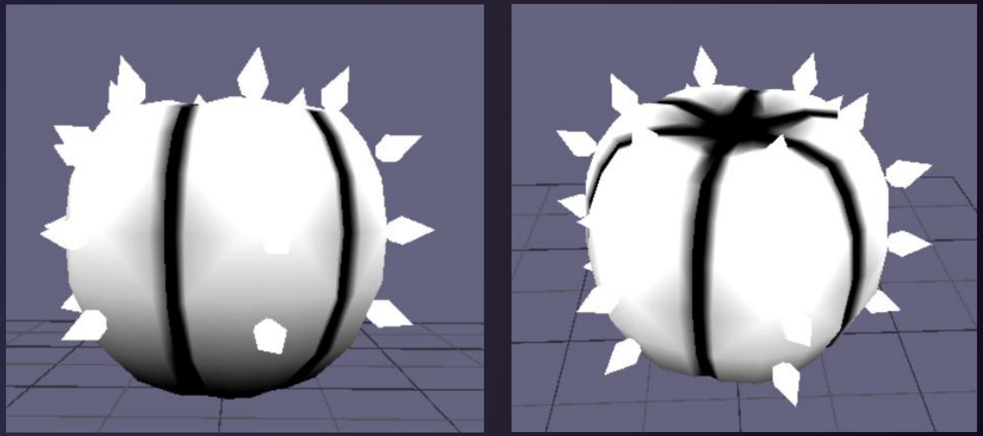
Off

On

SHADING: TOON RAMP BIAS

- Use vertex color to bias the lighting equation passed into toon ramp lookups.
- Can force parts of model into shadow.

Crevices always in shadow



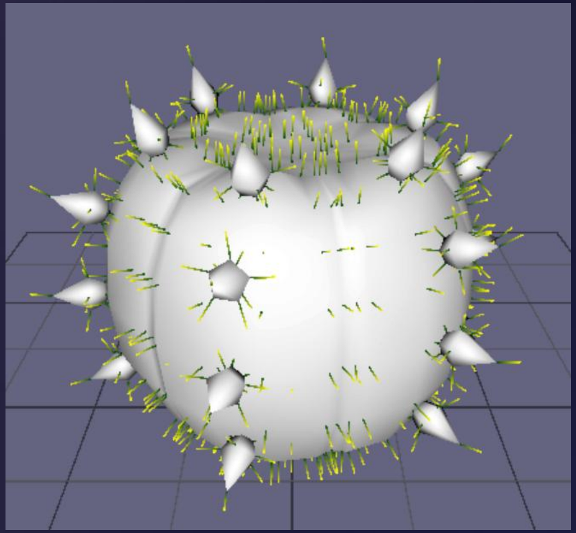
Off

On

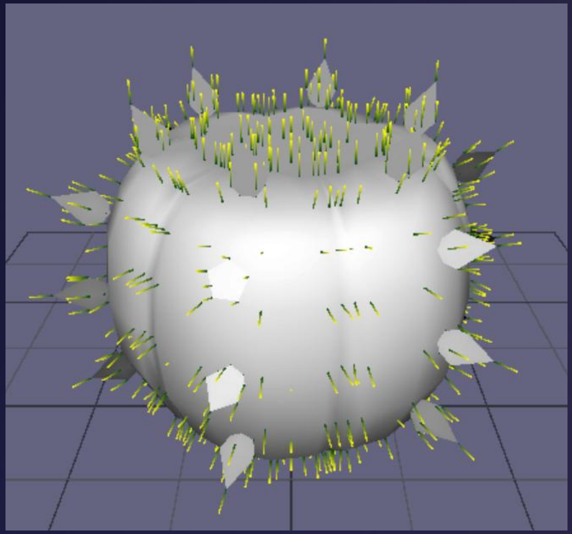
SHADING: NORMAL EDITING

- Controls the shadow line.
- Can "flatten" normals for solid lighting.

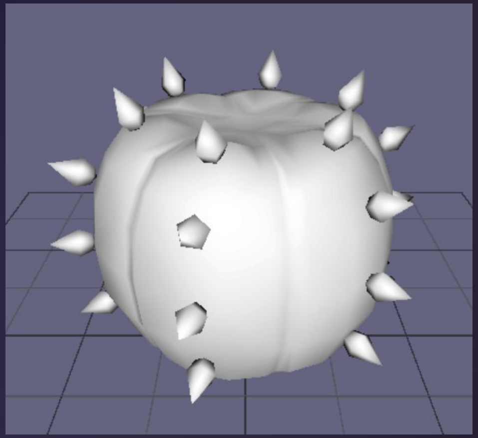
Default



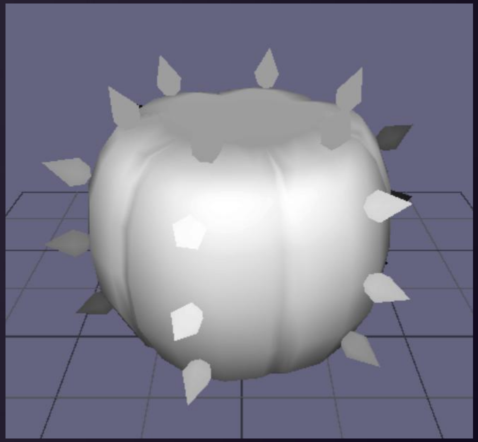
Edited



Default



Edited



SHADING: SHADOW CASTER

- Use separate geometry for cast shadows
- Prevent unwanted self shadowing

SHADING: SHADOW RECEIVER

- Separate shape for receiving shadows
- Prevents shadows in crevices
- Terrain walls projected to flat surface for sampling shadow map

SHADING: AMBIENT OCCLUSION

- Toon ramp applies toon cutoff to SSAO buffer input.

SHADING: SPECULAR

- Three different specular modes:
  - Toon isotropic
  - Toon anisotropic
  - Toon hair
- Result is scaled by gloss map and then passed into third row of toon ramp

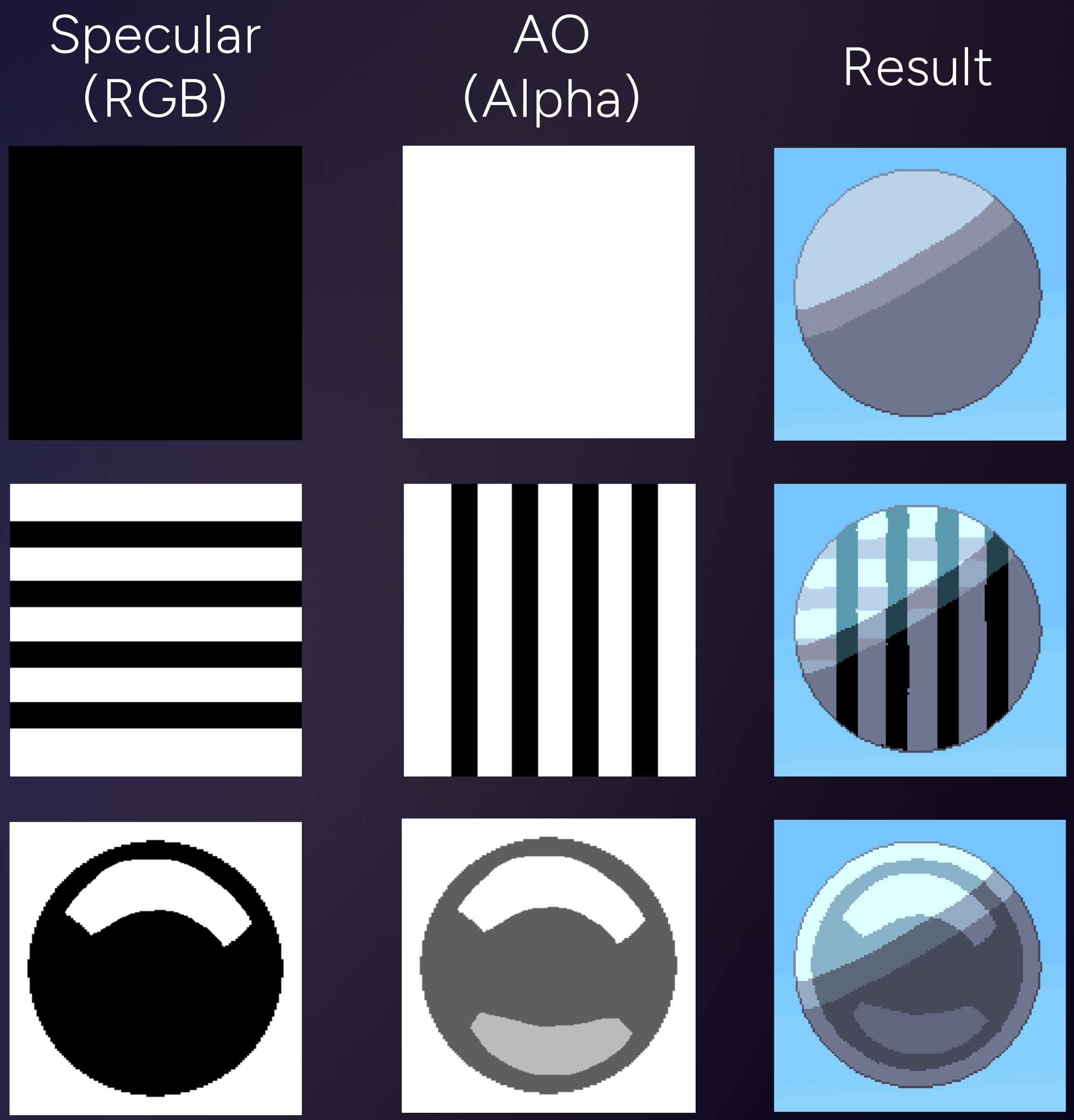
SHADING: MATCAP

Used for reflective materials:

- Metals
- Glass
- Monster eyes

SHADING: MATCAP

- RGB adds specular light.
  - Fades out with diffuse.
- Alpha scales ambient occlusion.
  - Allows for the appearance of reflection in shadow.



SHADING: TOON HAIR SPECULAR

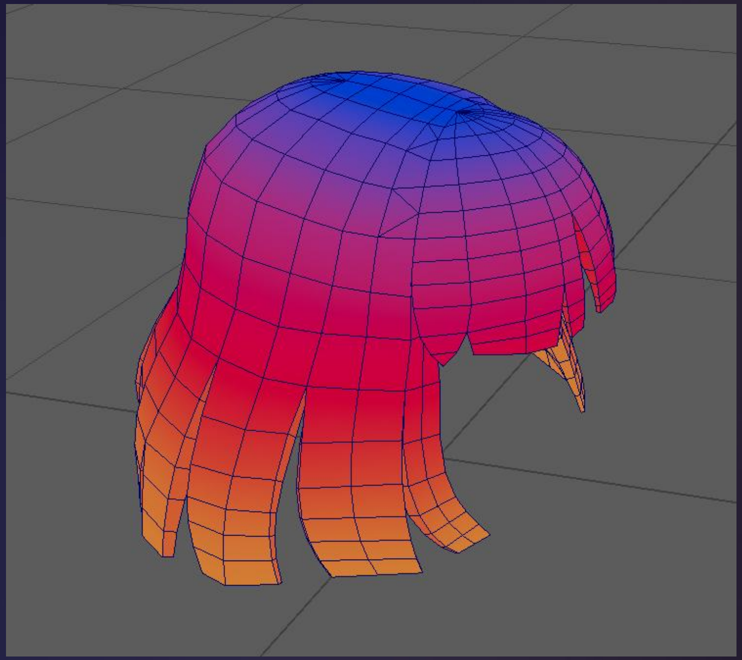
Overview

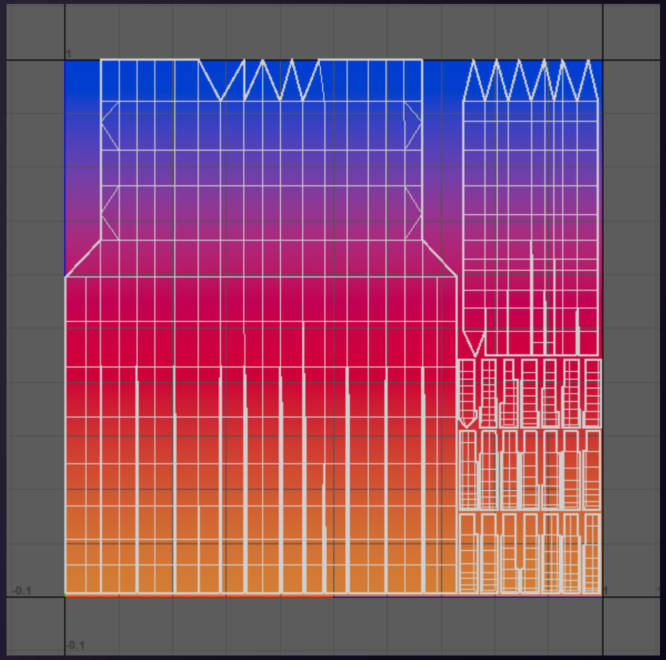
- Toon highlight locations are not baked into model.
- Lighting reacts to animation and camera.
- Parameterized for artistic control over correctness.
- Complicated to tune, but important to final look.

SHADING: TOON HAIR SPECULAR

UV Space

- Hair strand direction is aligned to vertical axis in texture
- Roots at top. Tips at bottom.
- Sample UV coordinate to know distance along strand of hair.





SHADING: TOON HAIR SPECULAR

Baseline Shift

- Tune specular line as UV space distance along hair strands.
- Apply an offset according to camera pitch.

SHADING: TOON HAIR SPECULAR

Curvature

- Curvature offset moves specular line as normal becomes perpendicular to the view.

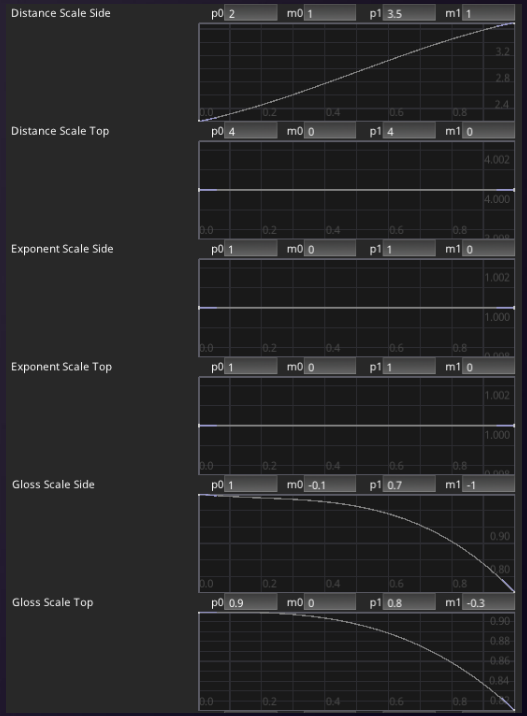
SHADING: TOON HAIR SPECULAR

Spatial Parameterization

- Specular is controlled by three parameters:
    - distance from center line
    - distance exponent
    - gloss scale.

```
specular = (distance ^ exponent)*gloss
```

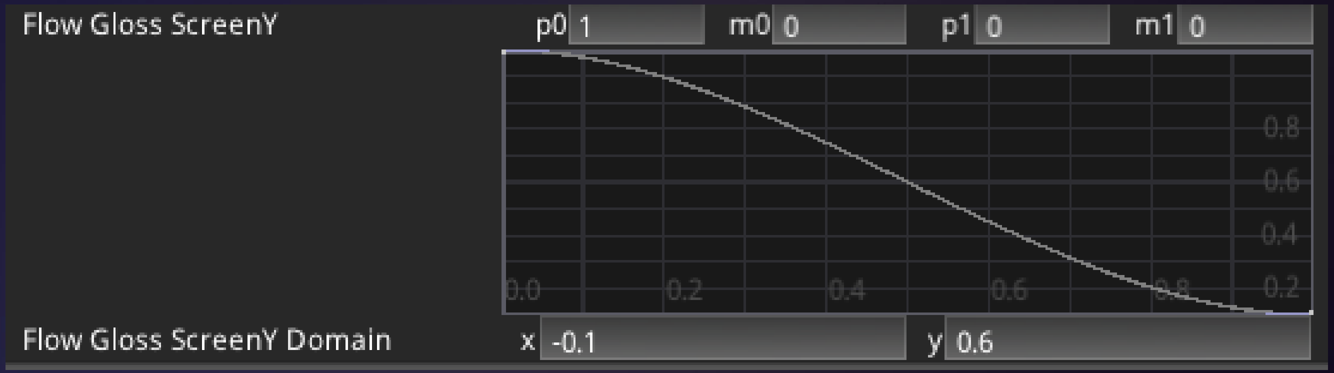
- Simple equation, but parameterized with curves from edge to center of hair.
- Two curves per parameter: side view and top view.

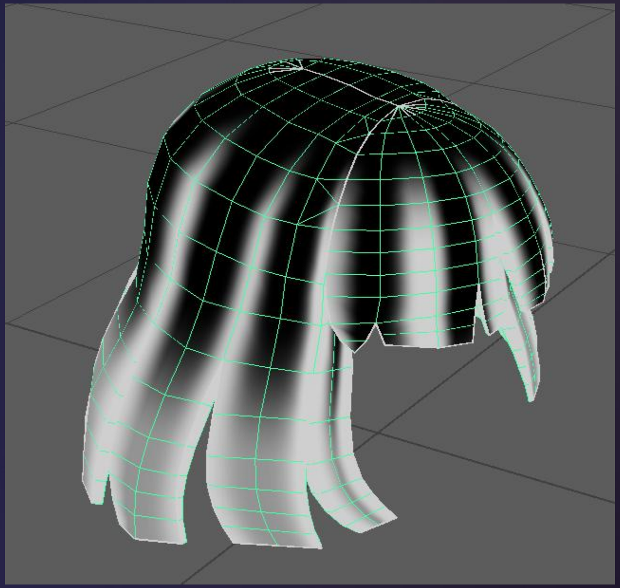


SHADING: TOON HAIR SPECULAR

Masking

- Gloss map used to separate chunks of hair.
- Vertical Flow Mask
  - Curve reduces specular according to how aligned hair strand is with camera up axis.
  - Focuses specular on downward flowing strands





SHADING: SHADOW CONTRAST

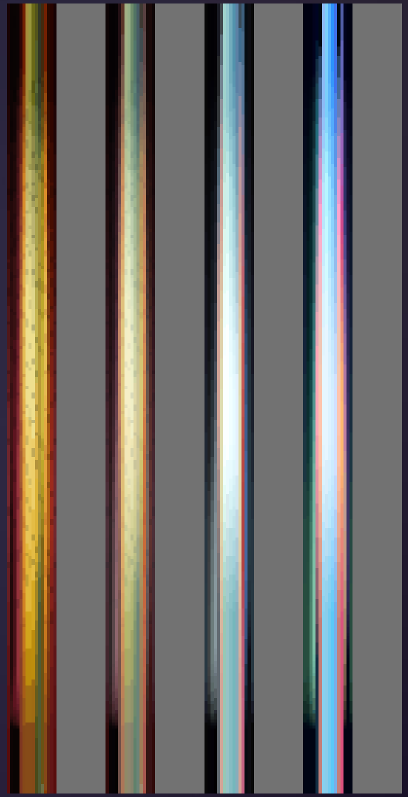
- Reduces shadow contrast is desired for some materials
- Scale down diffuse and scale up ambient
- Diffuse occlusion map can be used to force parts of model into shade lighting

SHADING: SKY

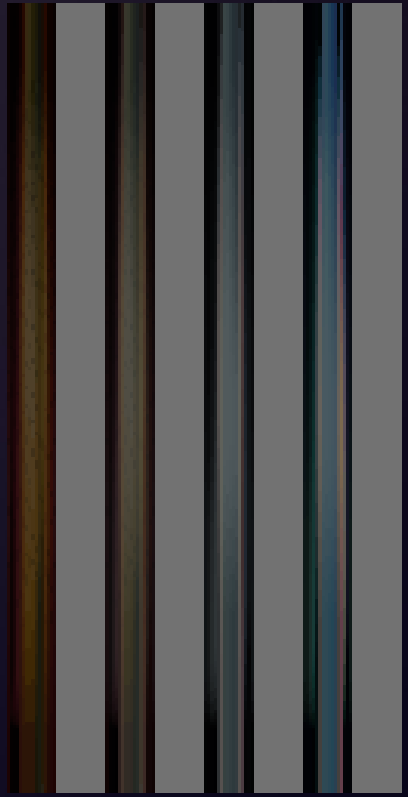
Color is painted as vertical sky gradients for each:

- Time of day
- World state
- Weather extreme

Clear Weather
Four World States



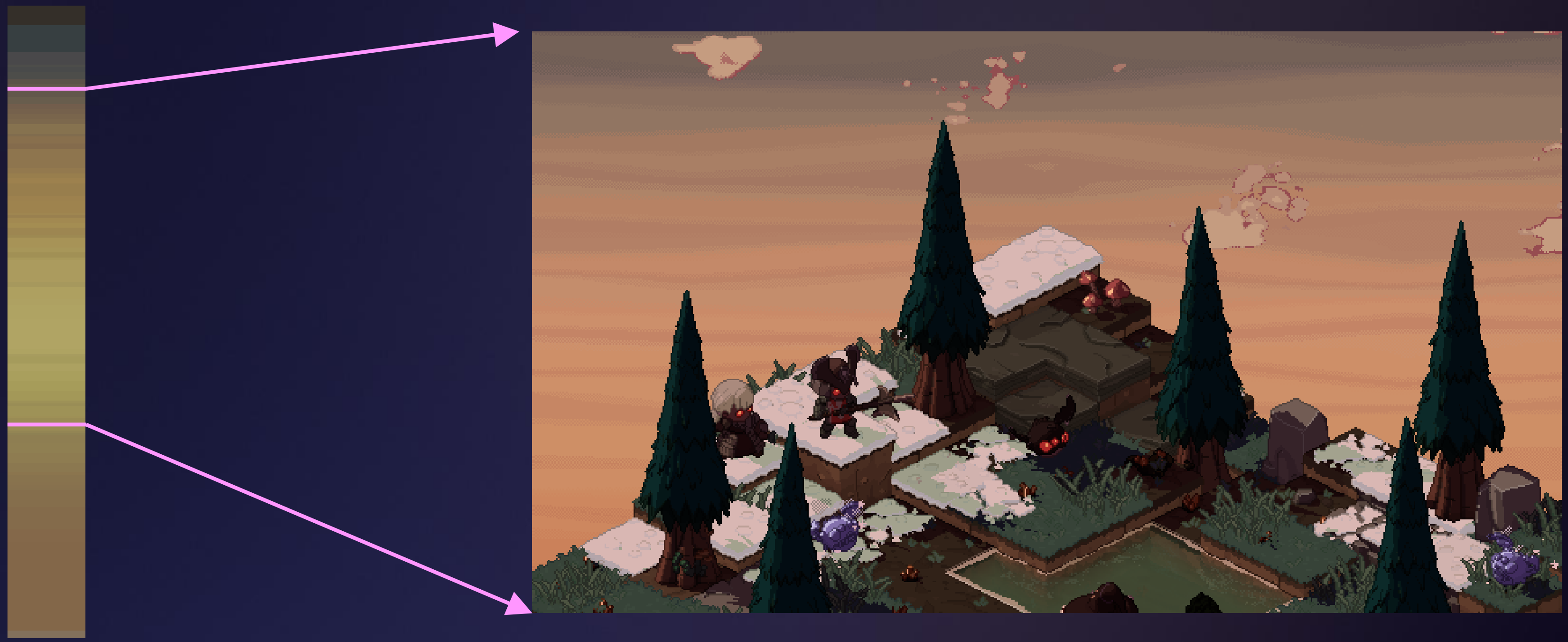
Cloudy Weather
Four World States



SHADING: SKY

- Map a range onto the screen based on camera zoom and pitch.

Blend Result



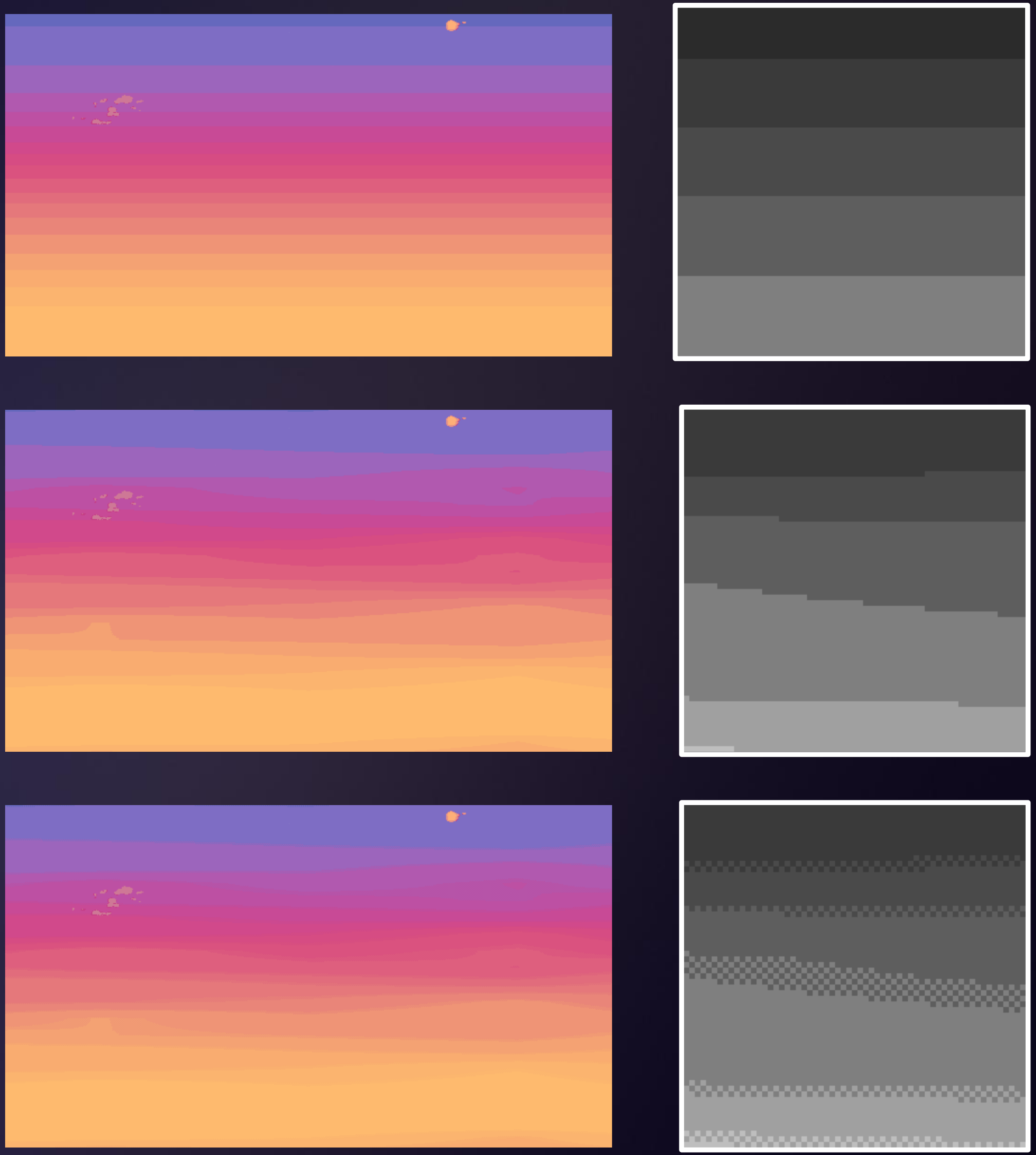
SHADING: SKY

Painterly Shape

- Displace vertical UV with noise map.
- Noise map wraps at with 360° yaw.

Dithering

- Small vertical UV offset on every other pixel.
- Dithered bands also stretch with noise.



SHADING: CLOUDS

Overview

- Needed dynamic cloud system to support dynamic weather and dynamic time-of-day.
- Based on The Real-time Volumetric Cloudscapes of Horizon Zero Dawn (SIGGRAPH 2015) but adjusted to work on a 2D plane.
- Cloud toon shading applied to result.

SHADING: CLOUDS

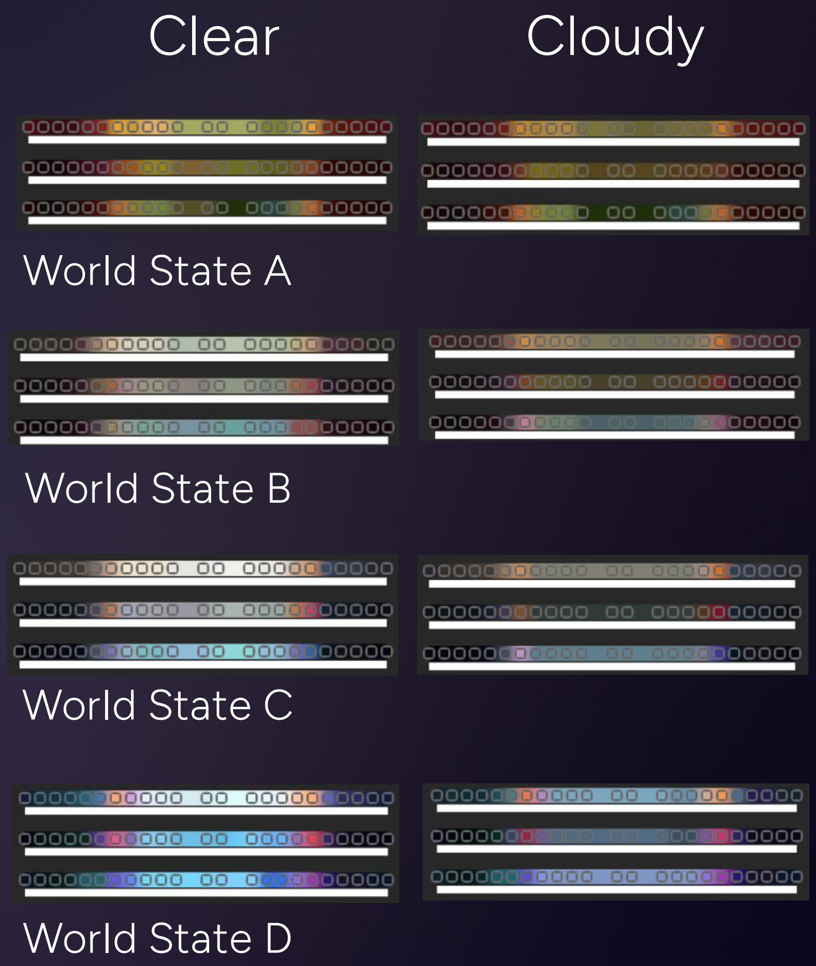
View Region

- Counter scrolls against camera yaw and pitch.
- Noise lookups have a seamless wrap point along horizontal axis.
  - Wraps every 360° yaw.
- Scale UV with depth for parallax effect

SHADING: CLOUDS

Toon Colors

- Cloud lighting takes three color inputs.
  - Direct Light
  - Ambient Light
  - Atmosphere
- Each input is a gradient across time of day
- Sets of gradients defined for each world state and weather extreme



SHADING: CLOUDS

Toon Shading

- Cloud model has three outputs
  - Extinction
  - Ambient Light Energy
  - Direct Light Energy
- Opacity threshold (Extinction < 75%)
- Blend AmbientColor to DirectColor in sRGB

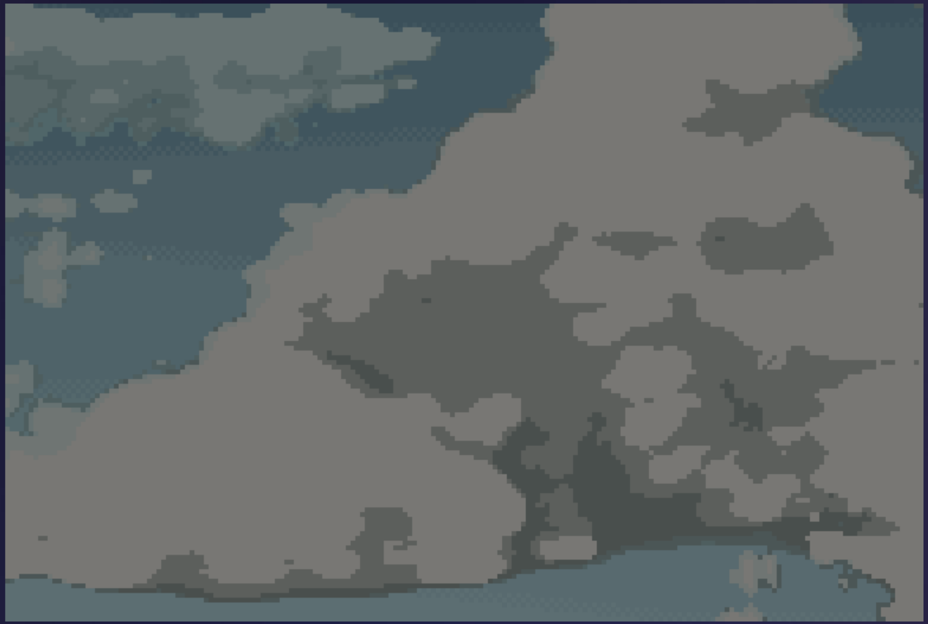
```
100%  DirectLightEnergy > 0.485

 40%  (DirectLightEnergy + AmbientLightEnergy)
          > 0.891

  0%  Otherwise
```

SHADING: CLOUDS

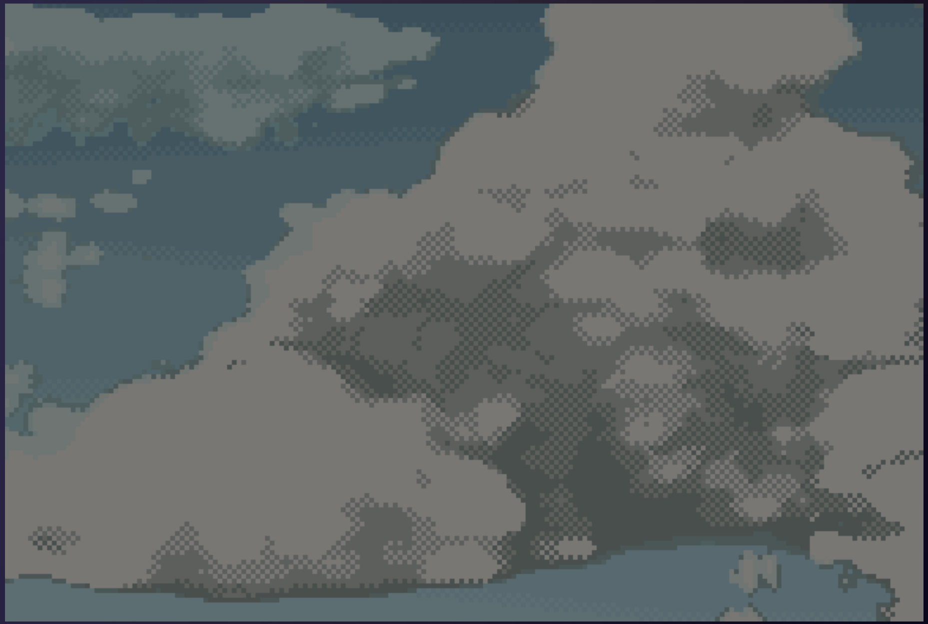
Dithering

- Every other pixel uses an energy shifted lighting calculation.
  - DirectLightEnergy -= 0.087
  - AmbientLightEnergy -= 0.087

OFF



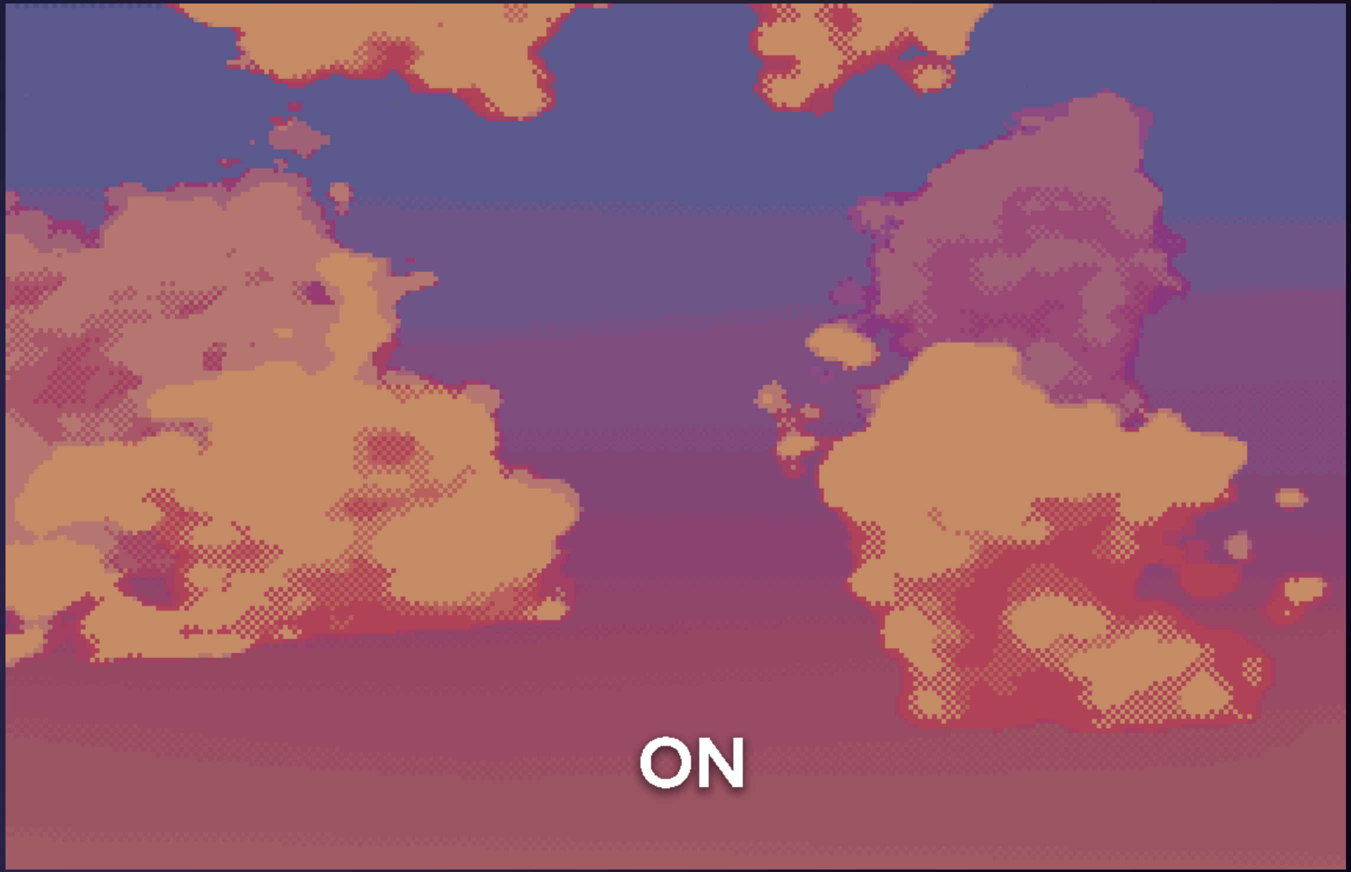
ON



SHADING: CLOUDS

Atmosphere Blend

- Fade towards AtmosphereColor with squared depth
- Step curve with 4 values (0%, 25%, 50% and 75%) is used for toon shading

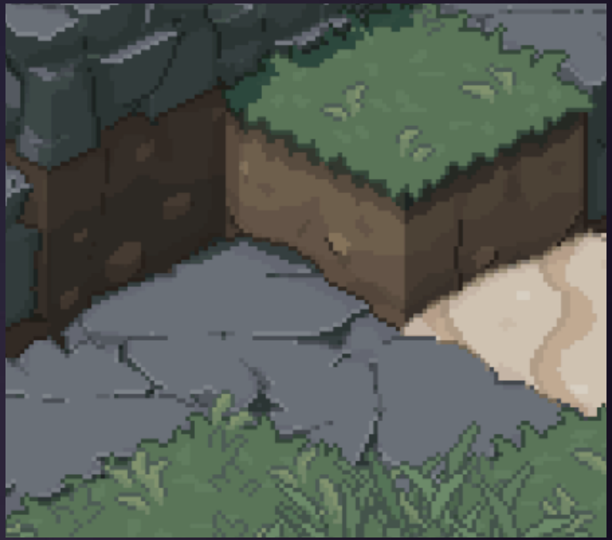


PIXEL SNAPPING

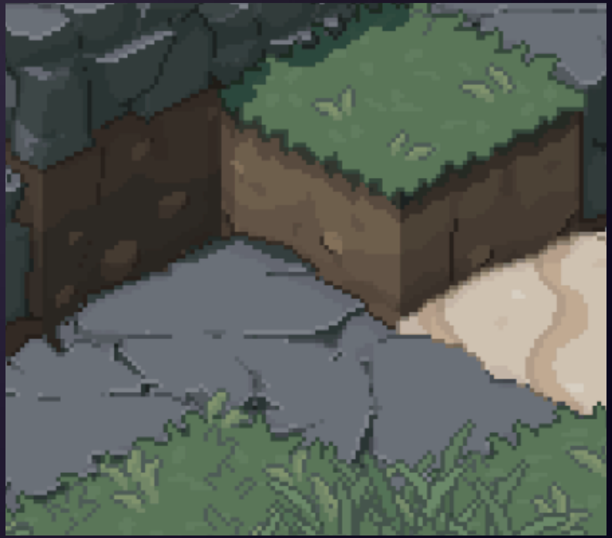
PIXEL SNAPPING: CAMERA POSITION

- Rotate camera position into screen orientation
- Snap the camera position to screen pixels
- Offset by half pixel for odd-sized viewports
  - (Renders same image for all viewport sizes)
- Reconstruct world position

OFF

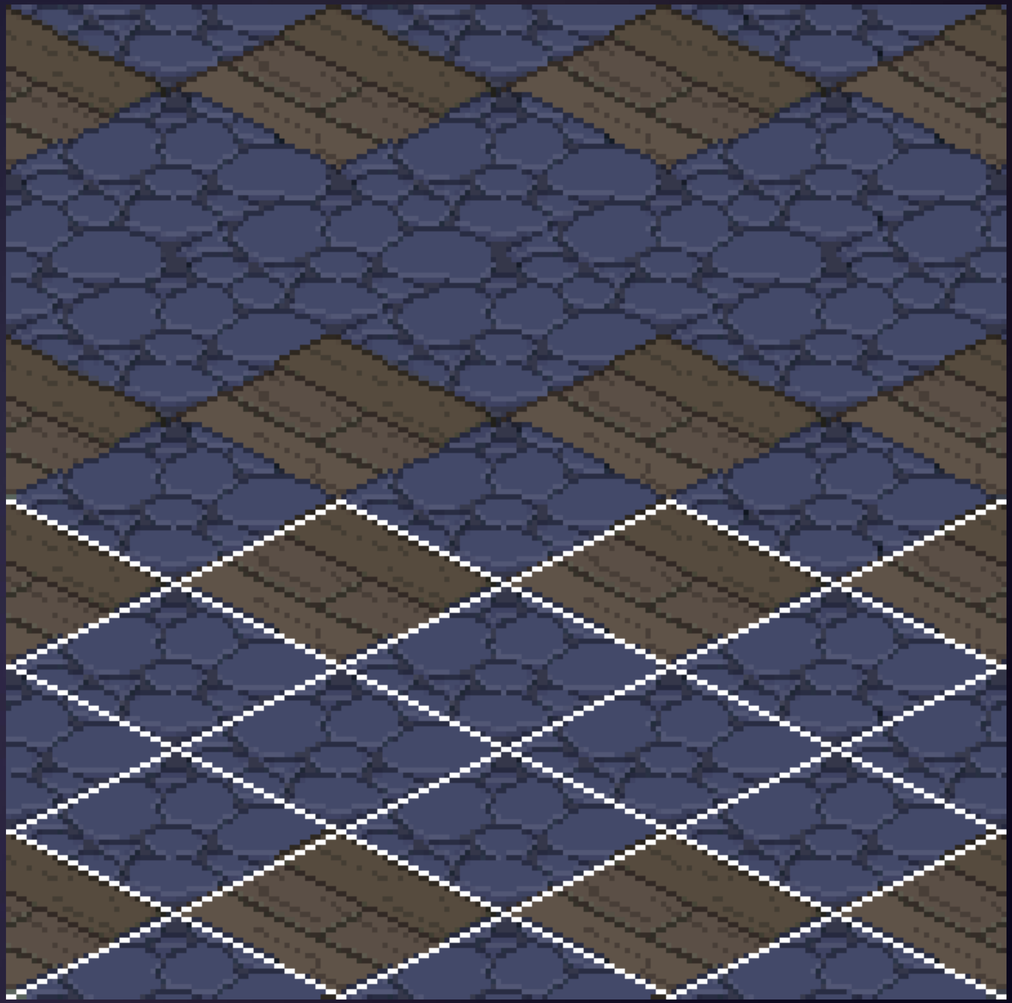


ON



PIXEL SNAPPING: CAMERA ZOOM

- Each 1x1 block needs an exact pixel size to tile properly
- Define the camera zoom based on desired pixel width of a block



PIXEL SNAPPING: CAMERA PITCH

- Camera pitch determines vertical spacing
- Compute pixel height of a block at desired pitch
- Snap the pixel height and recompute pitch

PIXEL SNAPPING: MODELS

- All instances of the same model render to the exact same silhouette.
- Snap root position to screen pixels
- Almost always disabled when attached (vs snap as part of parent “sprite”)
- Can disable for models that prioritize seamless connections (e.g. walls)

PIXEL SNAPPING: ANIMATION

- Joint translation snapped to screen pixels in model space
- Optionally inherit parent joint's snapped offset
- Keep feet on floor by disabling parent snap inheritance

OFF

ON

PIXEL SNAPPING: PARTICLES

- Snap particle position
- Snap particle size
- Particles specify min pixel width
- Odd sizes need to shift position by half pixel

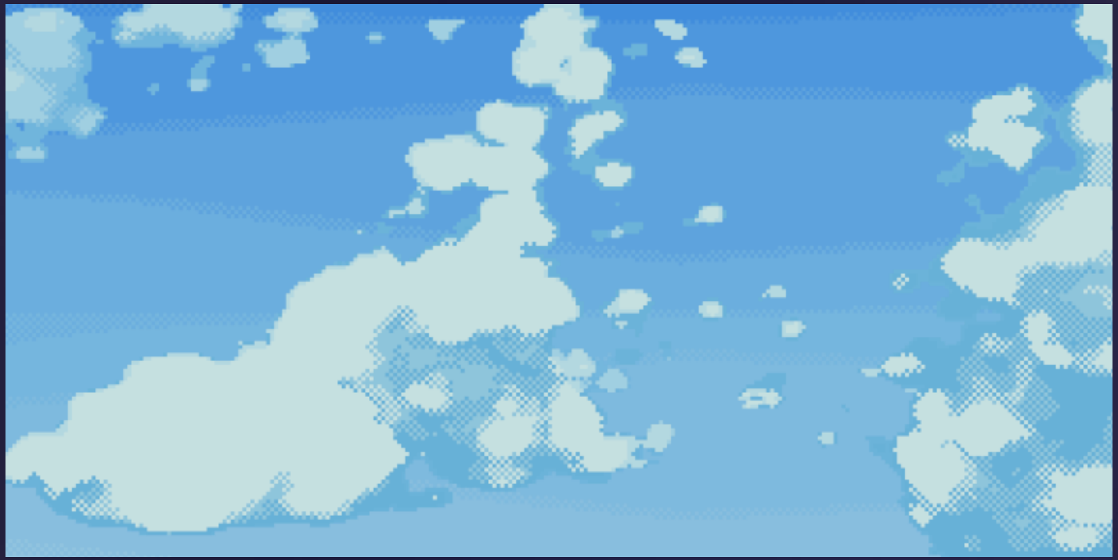
OFF

ON

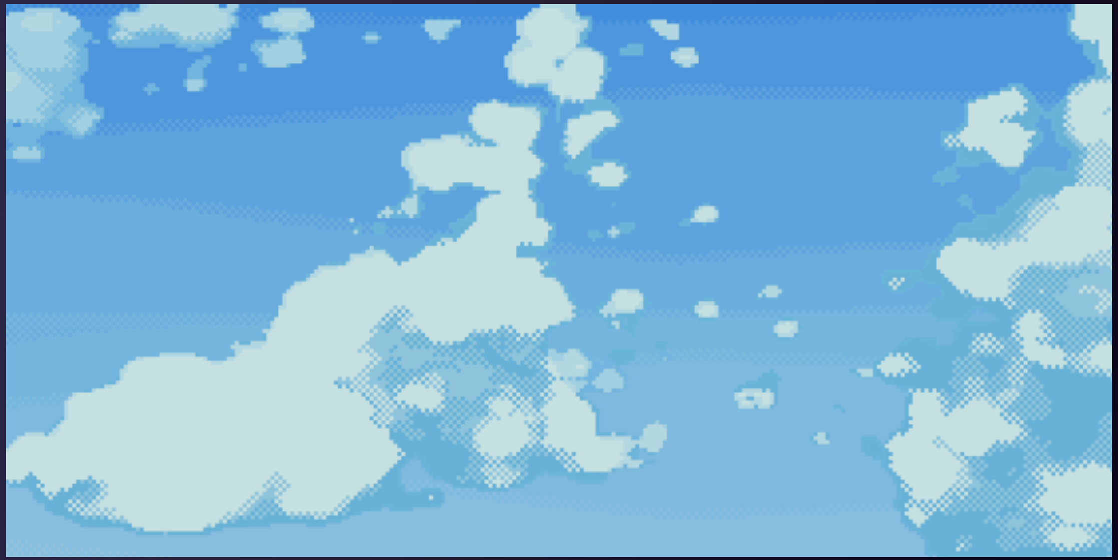
PIXEL SNAPPING: CLOUDS

- Cloud noise and dither scroll snaps to screen pixels (but depth complicates).
- Shape animation synchronized to the frames that horizontal scroll actually moves a pixel.

OFF

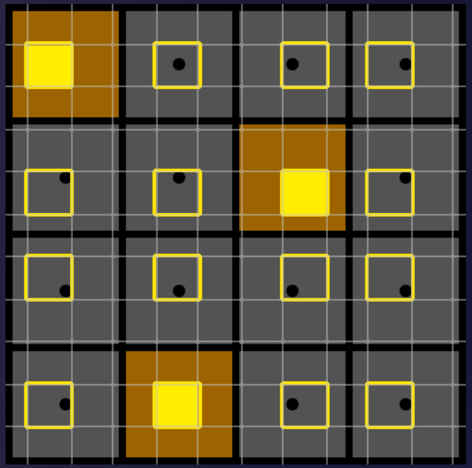


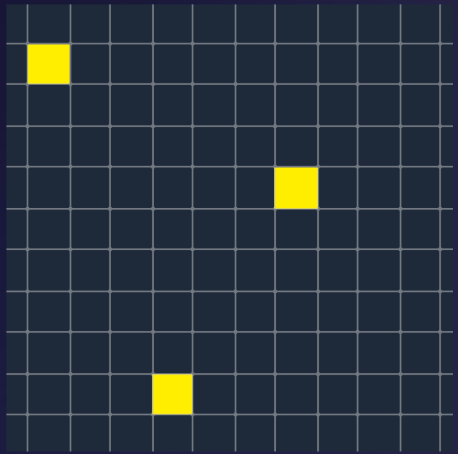
ON

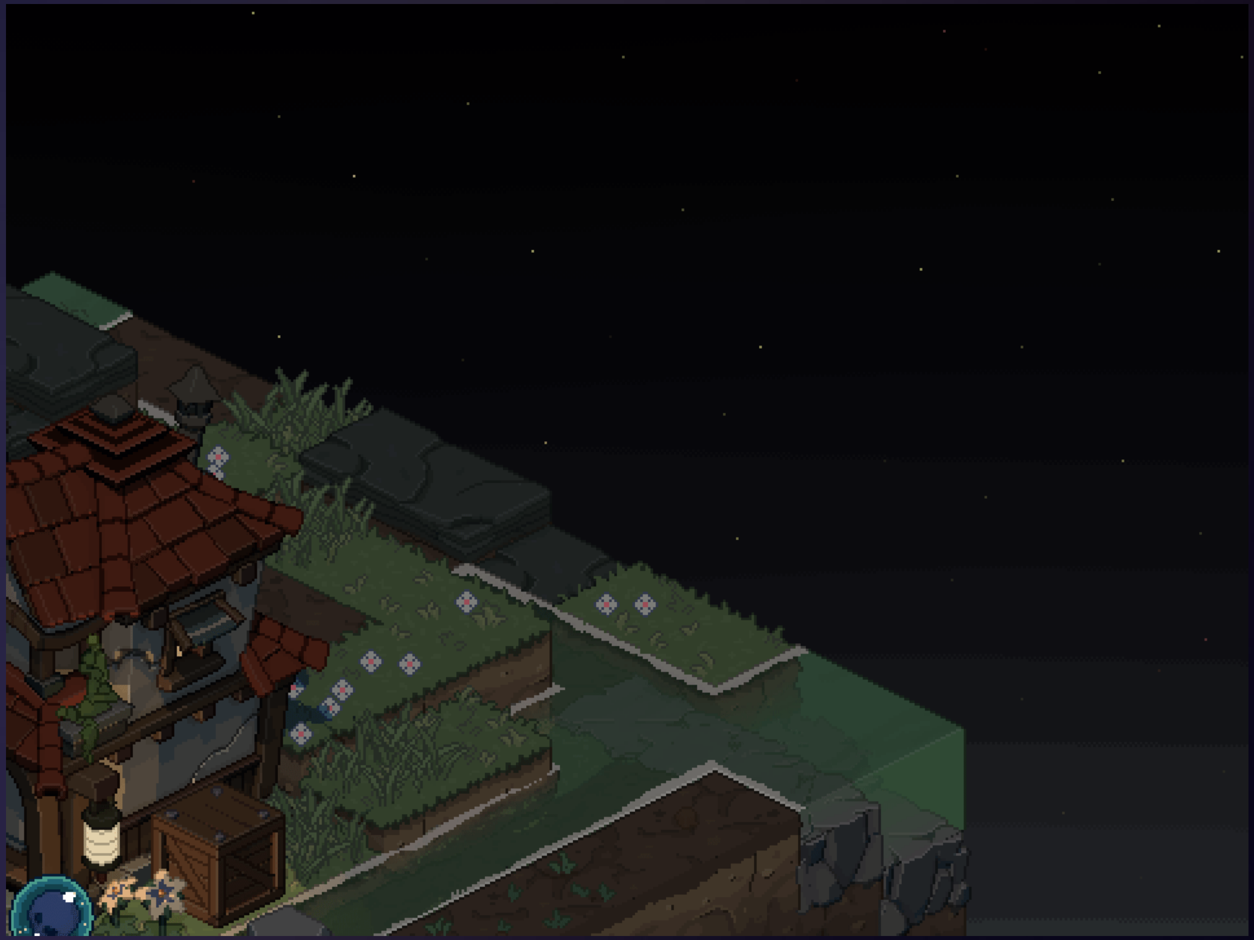


PIXEL SNAPPING: STARS

- Custom texture sample logic ensures stars are a single-pixel from any zoom







SPRITE SORTING

SPRITE SORTING: COMPARISON

- Sprites should layer over each other as 2D planes.
- Prevent objects from intersecting with the environment.

OFF ON

OFF ON

SPRITE SORTING: COMPARISON

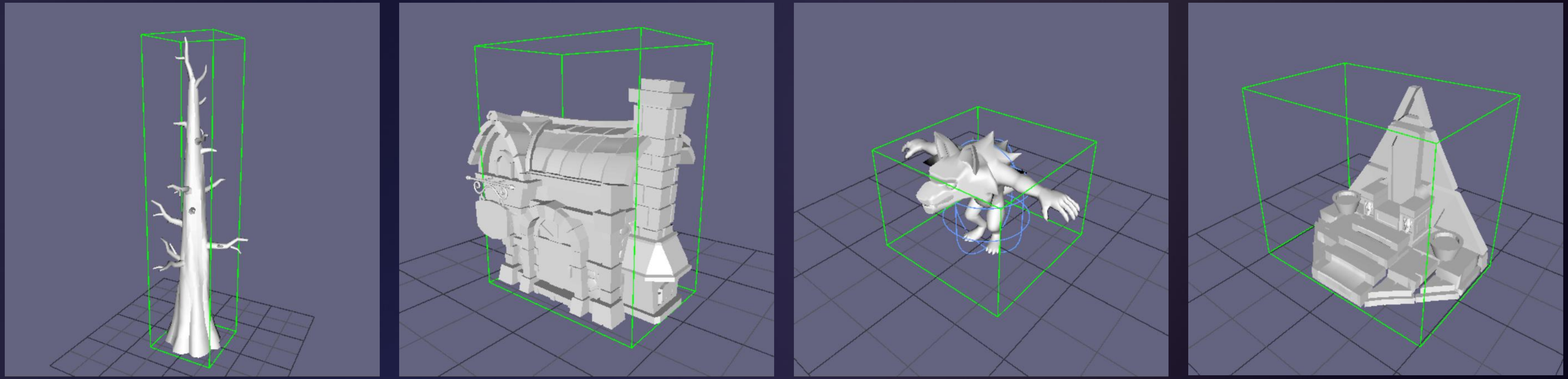
- Prevent objects from intersecting with each other.
- Prevent the environment from intersecting with itself.

OFF ON

OFF ON

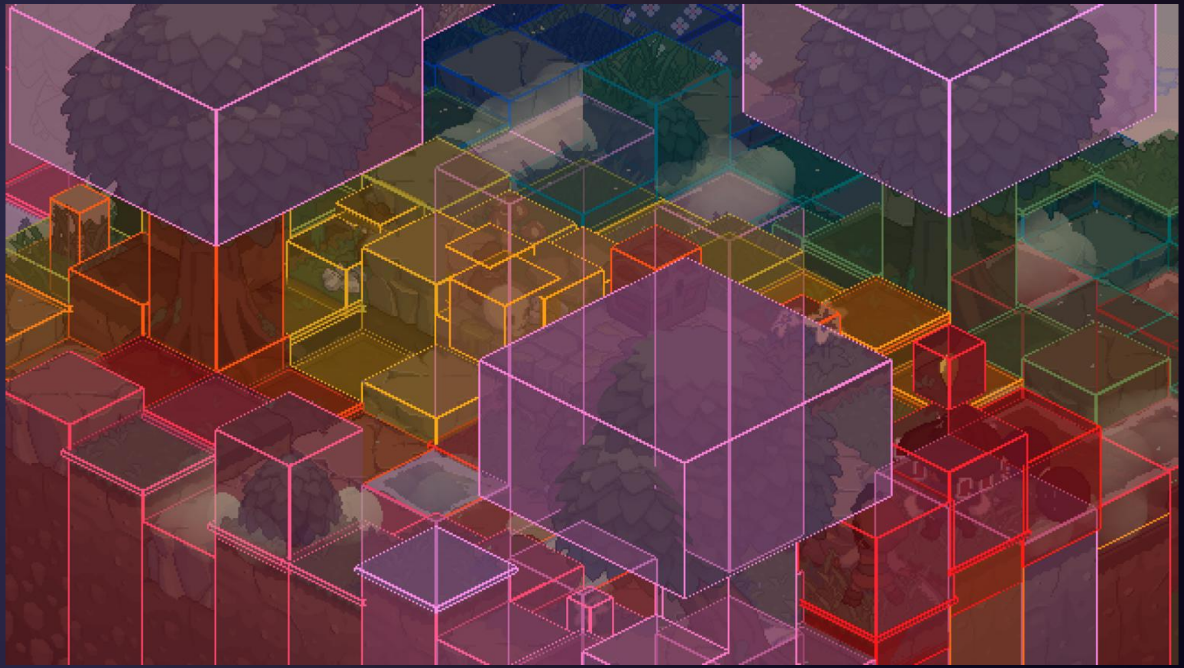
SPRITE SORTING: SORT VOLUMES

- Define axis aligned box for each object representing its volume in space.
- Normally covers entire volume, but choosing optimal size is case-by-case.



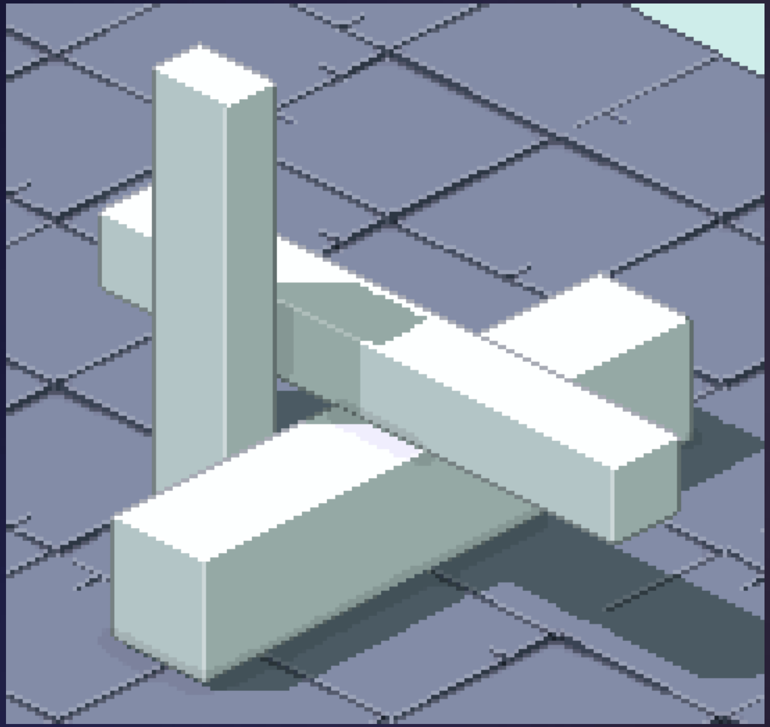
SPRITE SORTING: SORT VOLUMES

- Best if volumes do not overlap, but dynamic objects and variety of shapes makes that impossible.



SPRITE SORTING: CYCLES

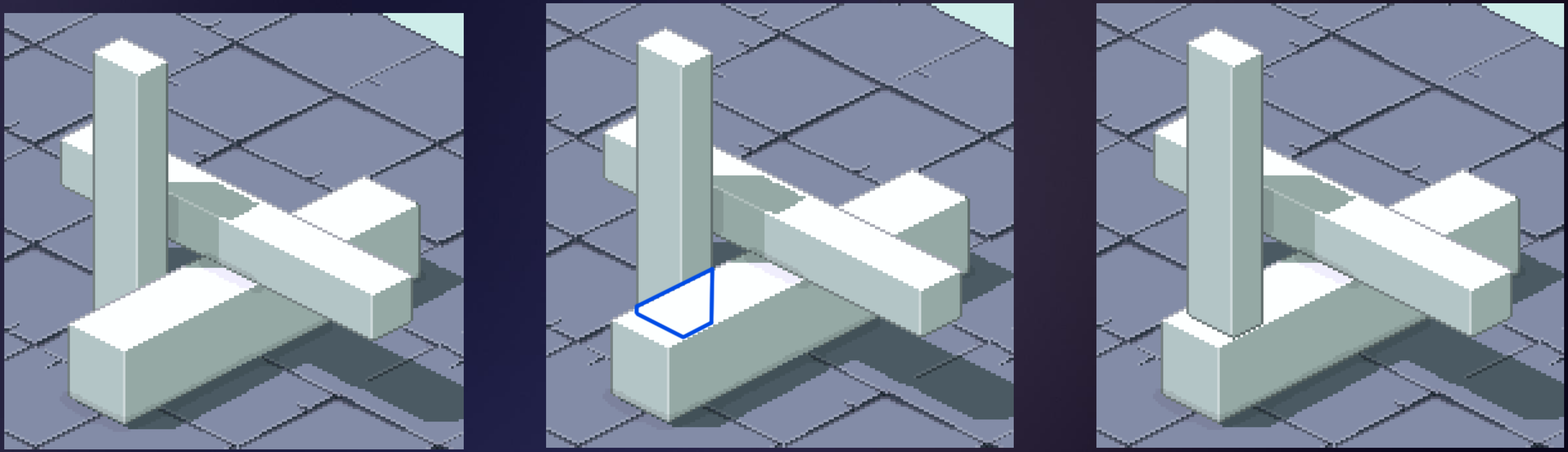
- Can get cycles of A in front of B in front of C in front of A
- No correct solution



SPRITE SORTING: BREAKING CYCLES

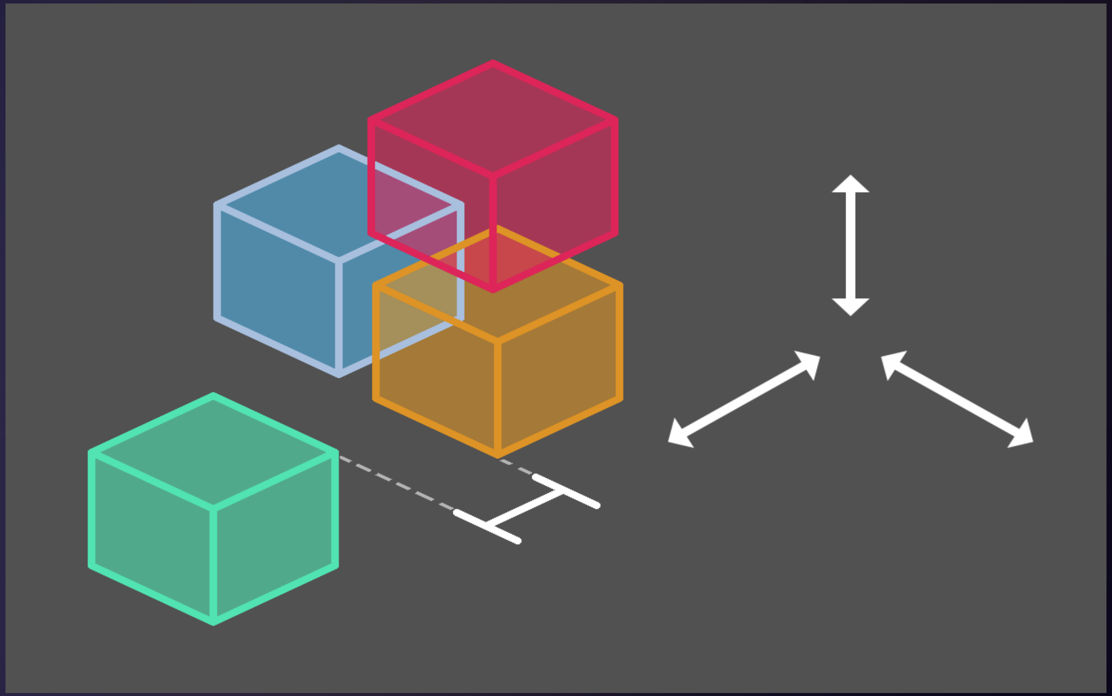
- Choose order that minimizes incorrectly sorted area

OFF **ON**



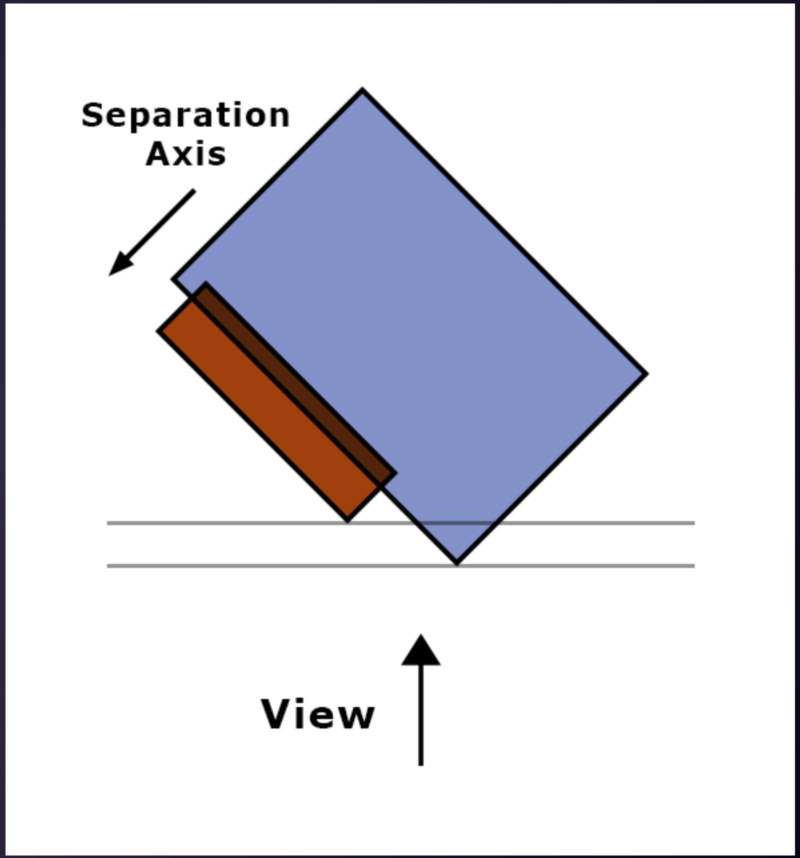
SPRITE SORTING: DETECTING OVERLAPS

- Project AABBs onto screen
- Track min / max along screen-projected world axes
- Perform separating axis tests to find overlapping pairs



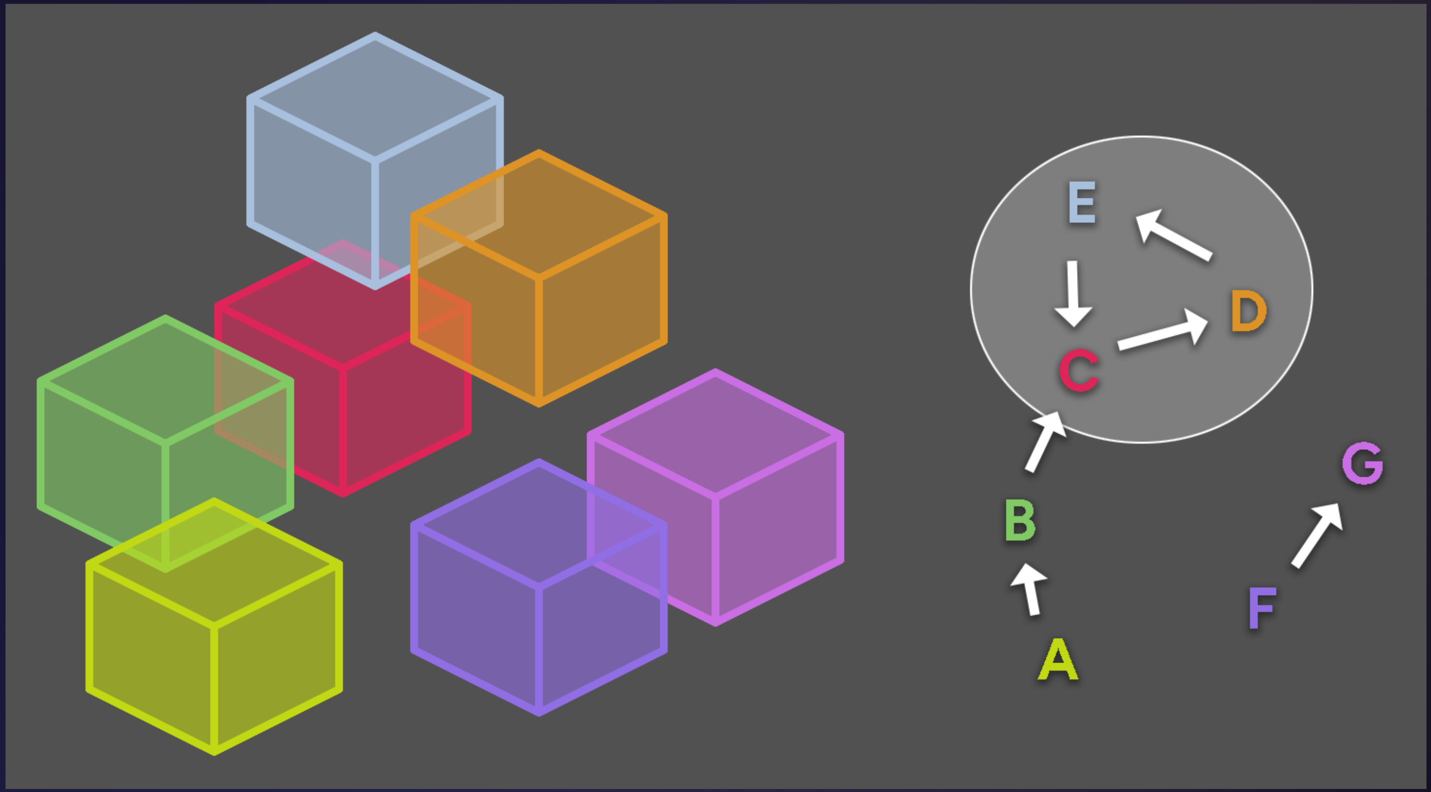
SPRITE SORTING: SORTING OVERLAPS

- Decide which box is “in front” for each overlap
  - Closest point in depth is not a good sort metric
  - Find minimum separating axis
  - Compare separation direction with view direction to decide which box is in front



SPRITE SORTING: FINDING CYCLES

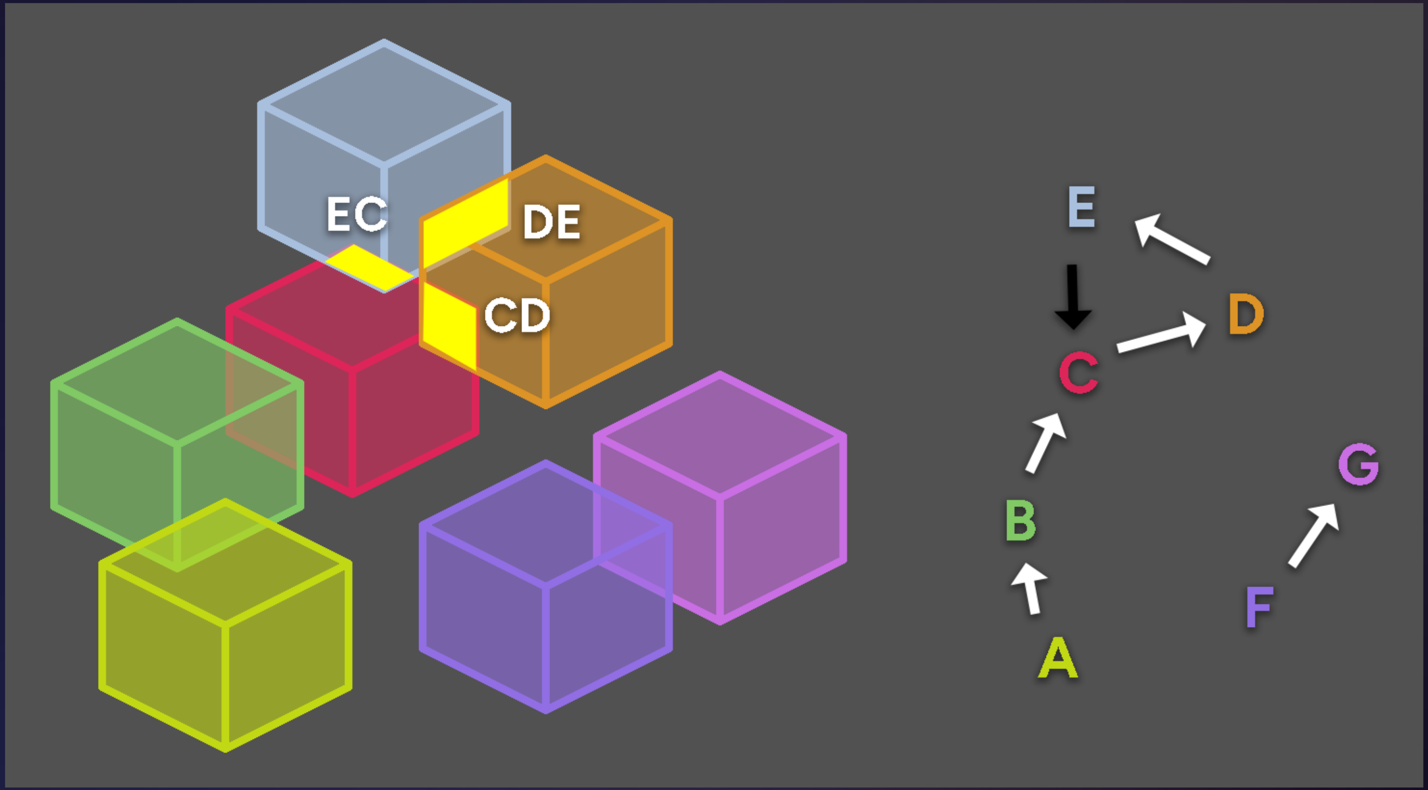
- Evaluate graph with Tarjan's strongly connected components (SCC) algorithm



SCC detected between C, D and E.

SPRITE SORTING: BREAKING CYCLES

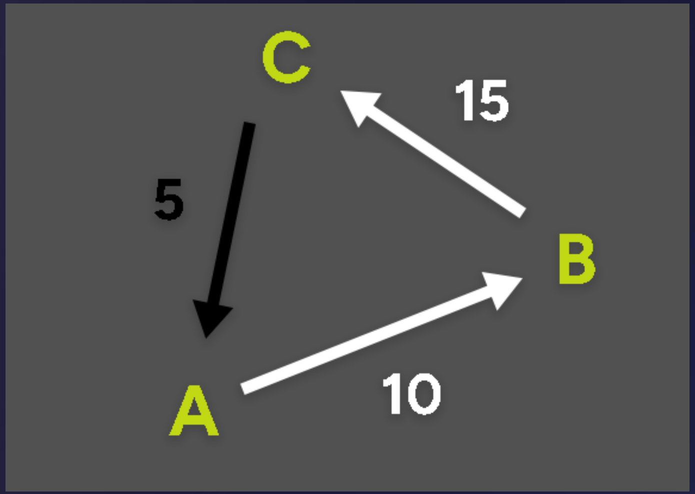
- Find edge of SCC with smallest screen space overlap.
- Break link, and resort the SCC.
- Continue until graph is acyclic.



SPRITE SORTING: STATIC SORT STABILITY

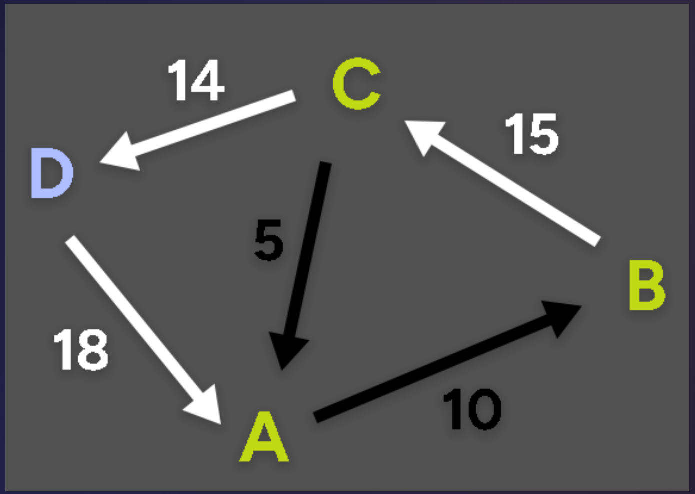
- Static environment sort needs to remain stable between frames.
- Dynamic objects moving through scene can change edge breaking results.
- Solve with multi pass solution

Static Only



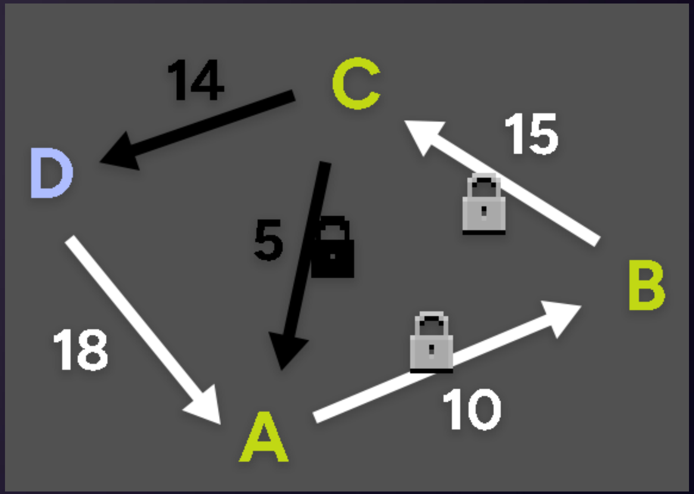
Sorted: A B C

Naïve Static + Dynamic



Sorted: B C D A

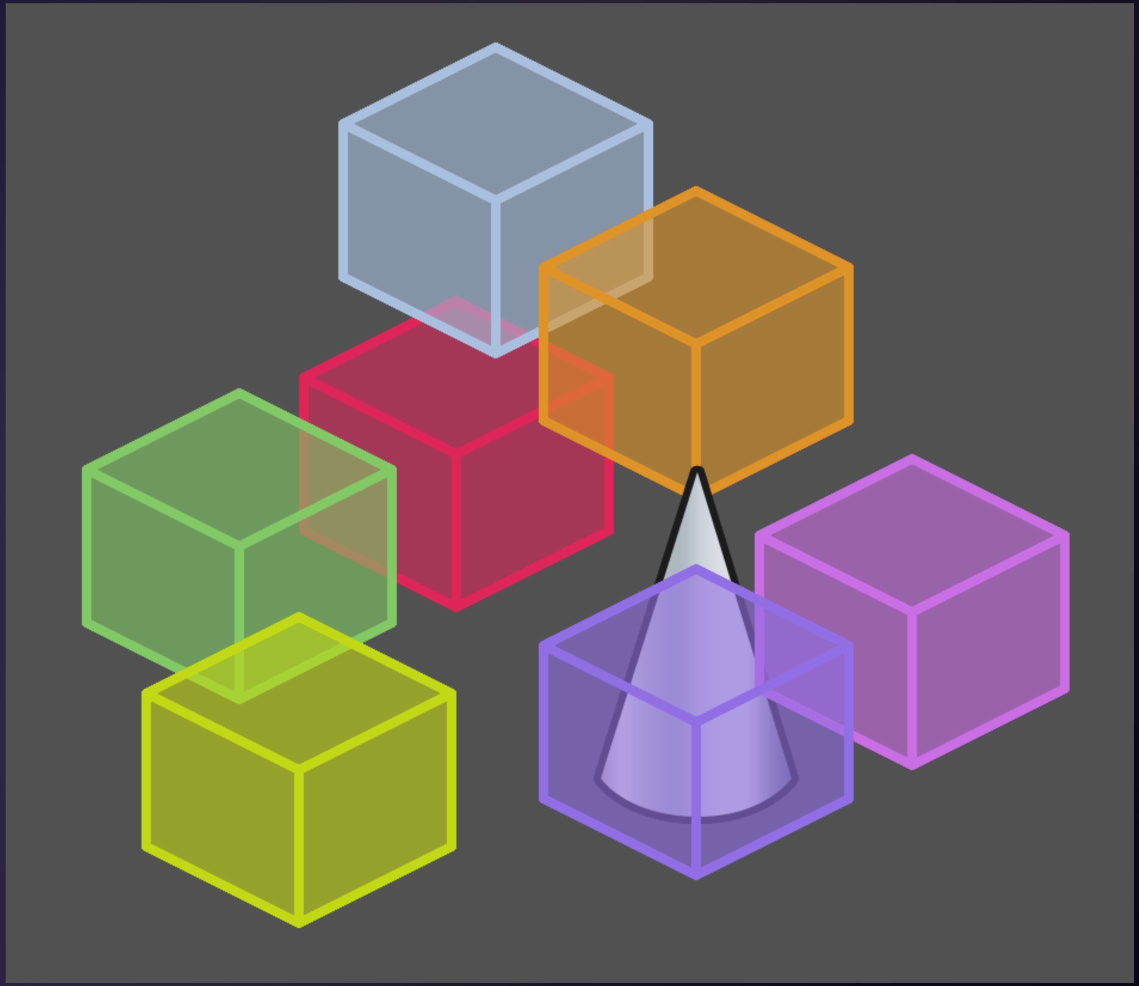
Static Locked, Then Dynamic



Sorted: D A B C

SPRITE SORTING: HEURISTIC PASS

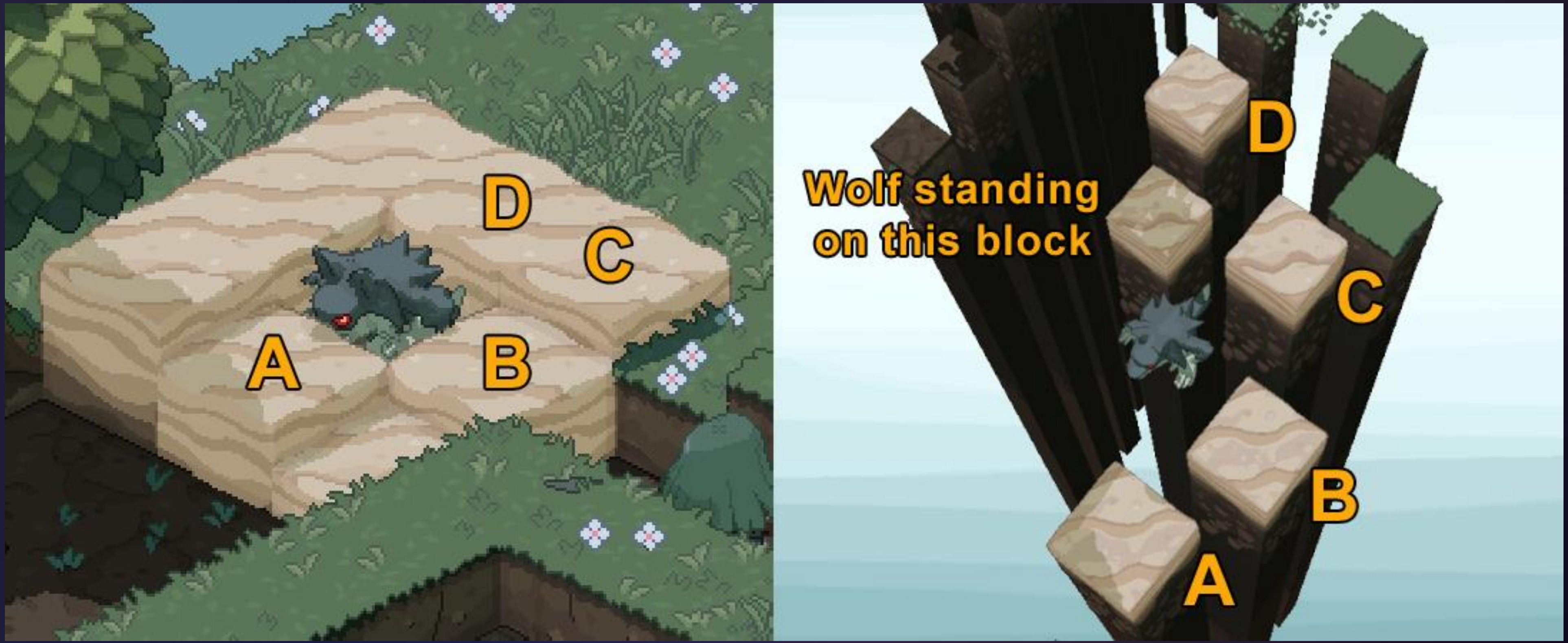
- Separately sorted groups with no logical overlap may still visually overlap.
- Interleave separately sorted groups with a heuristic
  - Volume with lower top (3D) goes in front
  - Otherwise, nearest goes in front
  - Otherwise, leftmost goes in front



SPRITE SORTING: RENDERING

- Offset models in depth according to sprite order.
- Cannot see offset from orthographic perspective.

SPRITE SORTING: RENDERING



Top down view of depth offsets.

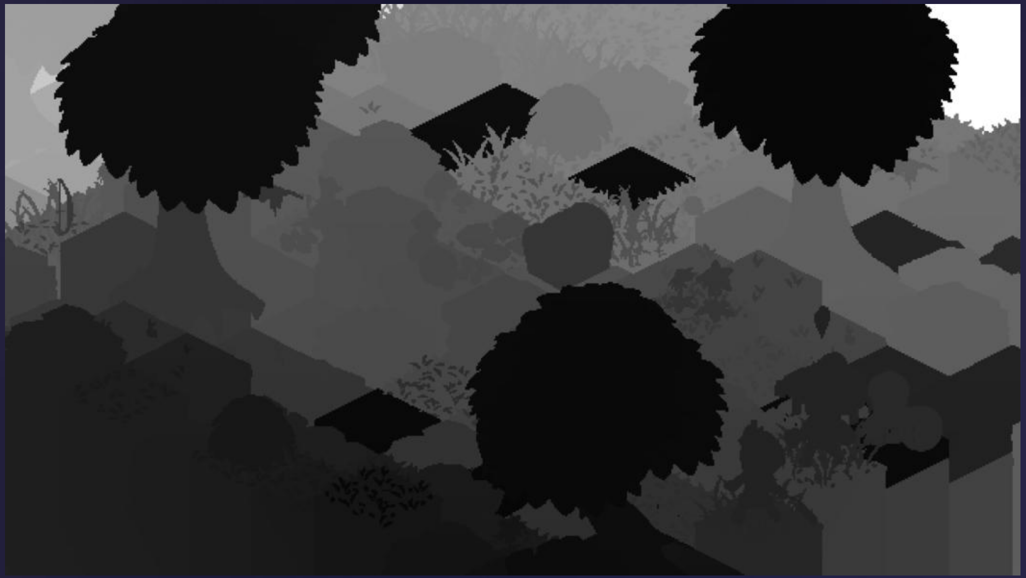
SPRITE SORTING: SHADOW IMPERFECTIONS

- Shadows still sample from within the intersecting geometry
- Can look odd, but often just perceived as a drop shadow

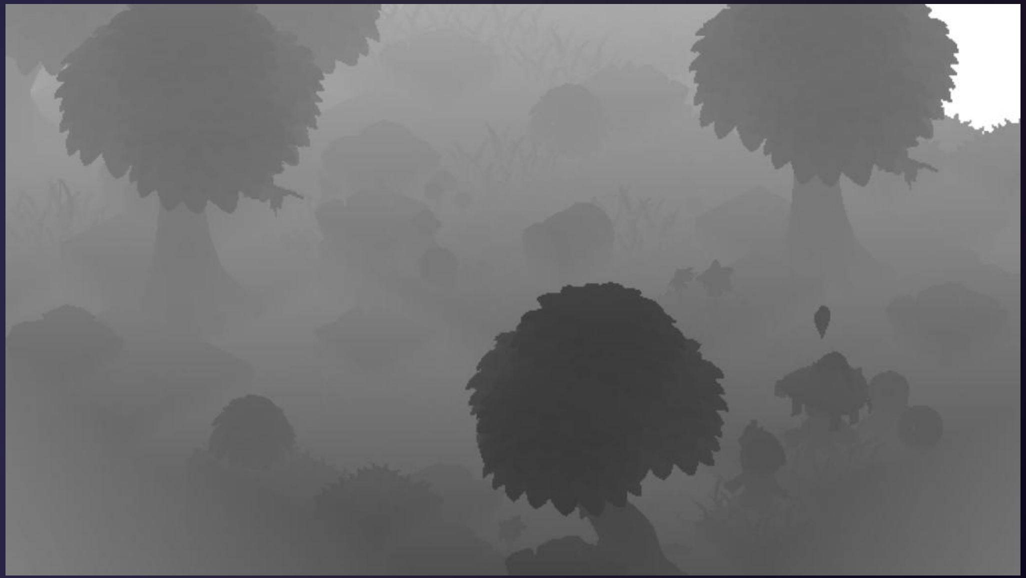
SPRITE SORTING: DUAL DEPTH RENDER

- Some effects need a standard depth buffer.
- Shader writes out both.

Sprite Depth



Standard Depth



Left depth not actually flat per sprite. Just hard to see compressed detail.

Black terrain blocks are puddles which sort last.

SPRITE SORTING: DUAL DEPTH RENDER

- Water and fog sample standard depth buffer.
- 3D intersections mixed with sprite sorted opaque models.

ANIMATION

ANIMATION: ROTATION SNAPPING

- Simulate sprite sheet with quantized model rotations.
  - Ensure important angles are represented
  - N values per axis of Euler vector depending on object type
- Only world space (camera rotation is smooth)

OFF

ON

ANIMATION: CURVES

- Hand placed step keys (no blending)
- Joint rotation is quantized in model space (fights accumulated float error)

ANIMATION: PARTICLES

- Particle systems can simulate and render at a separate frame rates.

OFF

ON

FINAL THOUGHTS

- Let tech limitations drive the art style.
- Extensive collaboration between engineering and art.
- Large attention to detail from artists and animators.
- Need 3D artists passionate about 2D pixel art.

QUESTIONS

Mar 10 Tuesday

10:30am - 11:30am

Technical Artists Roundtable (20th Year)!!! Day 1

Speaker: Jeff Hanna (NVIDIA Corporation)

Location: Room 314, South Hall

11:50am - 12:20pm

Shader Like a Painter: Building Character Shaders with an Artistic Approach at Naughty Dog

Speaker: Rosy (Rong) Fu (Naughty Dog)

Location: Room 2001, West Hall

1:50pm - 2:50pm

Shaders 501: Compute for Tech Artists

Speaker: Matt Oztalay (Epic Games)

Location: Room 3002, West Hall

3:10pm - 4:10pm

How We Draw a 3D Sprite World: The Stylized Art of 'Never's End'

Speaker: Ryan Juckett (Hypersect)

Location: Room 2001, West Hall

4:30pm - 5:00pm

Mastering Massive Worldbuilding: Production and Optimization of Ultra-Large Scale Maps

Speaker: Wenya He (Amazing Seasun Games)

Location: Room 2001, West Hall

Mar 11 Wednesday

1:50pm - 2:50pm

Technical Artists Roundtable (20th Year)!!! Day 2

Speaker: Jeff Hanna (NVIDIA Corporation)

Location: Room 314, South Hall

Mar 13 Friday

9:30am - 10:30am
Tech Art Panel Discussion
Speakers: Ben Cloward (Independent), Nicholas Burkhard (Meta), Matt Lake (Independent), Sen Lei Deng (EA Criterion), Camille Kay (Epic Games)
Location: Room 2014, West Hall

11:40am - 12:10pm
Demistifying the Clouds of 'Borderlands 4': Adapting Volumetric Cloud Technologies to a UE5 Open-World Game
Speaker: Spencer Luebbert (Disbelief)
Location: Room 2006, West Hall

10:50am - 11:20am
The Dynamic Weather Systems of 'Borderlands 4': How They Can Apply to Games Without Dynamic Weather
Speaker: Brian McNett (Gearbox Software)
Location: Room 2006, West Hall

1:30pm - 2:30pm
Technical Artists Roundtable (20th Year)!!! Day 3
Speaker: Jeff Hanna (NVIDIA Corporation)
Location: Room 312, South Hall

Please Remember to:

- Fill out the survey
- Thank our lovely CAs