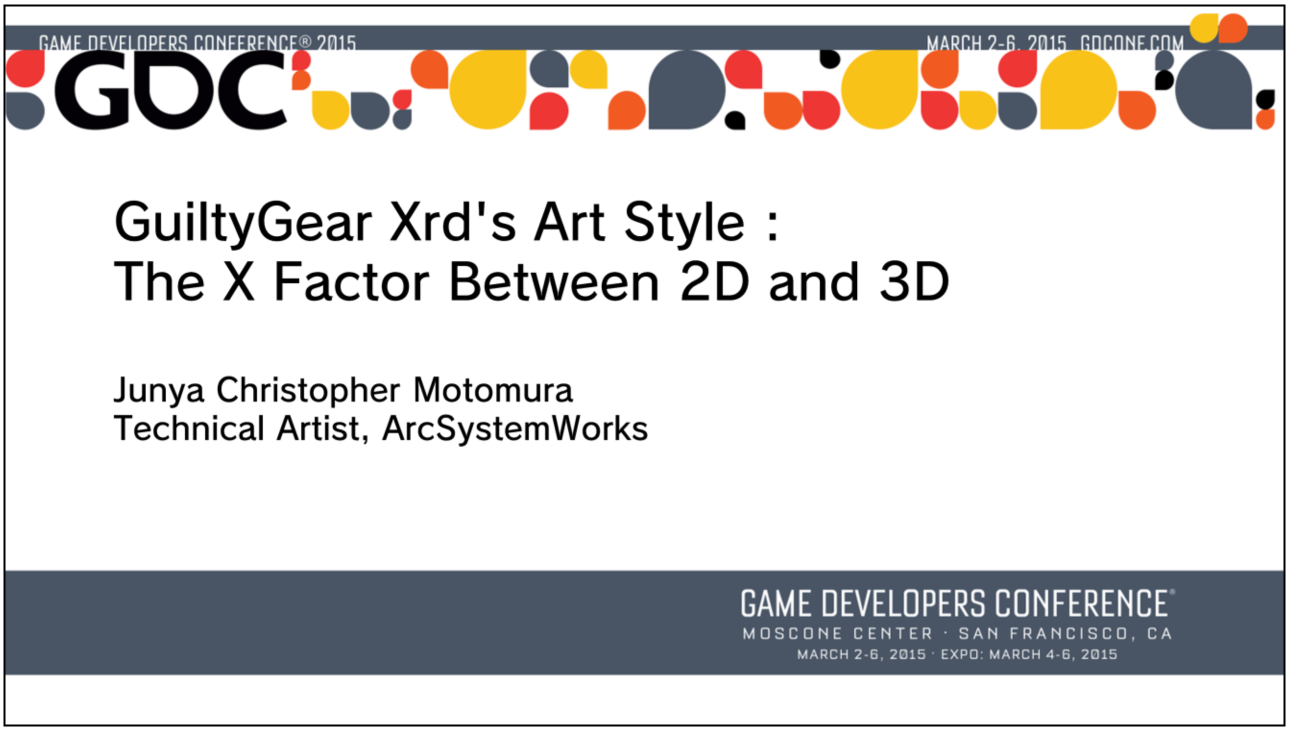
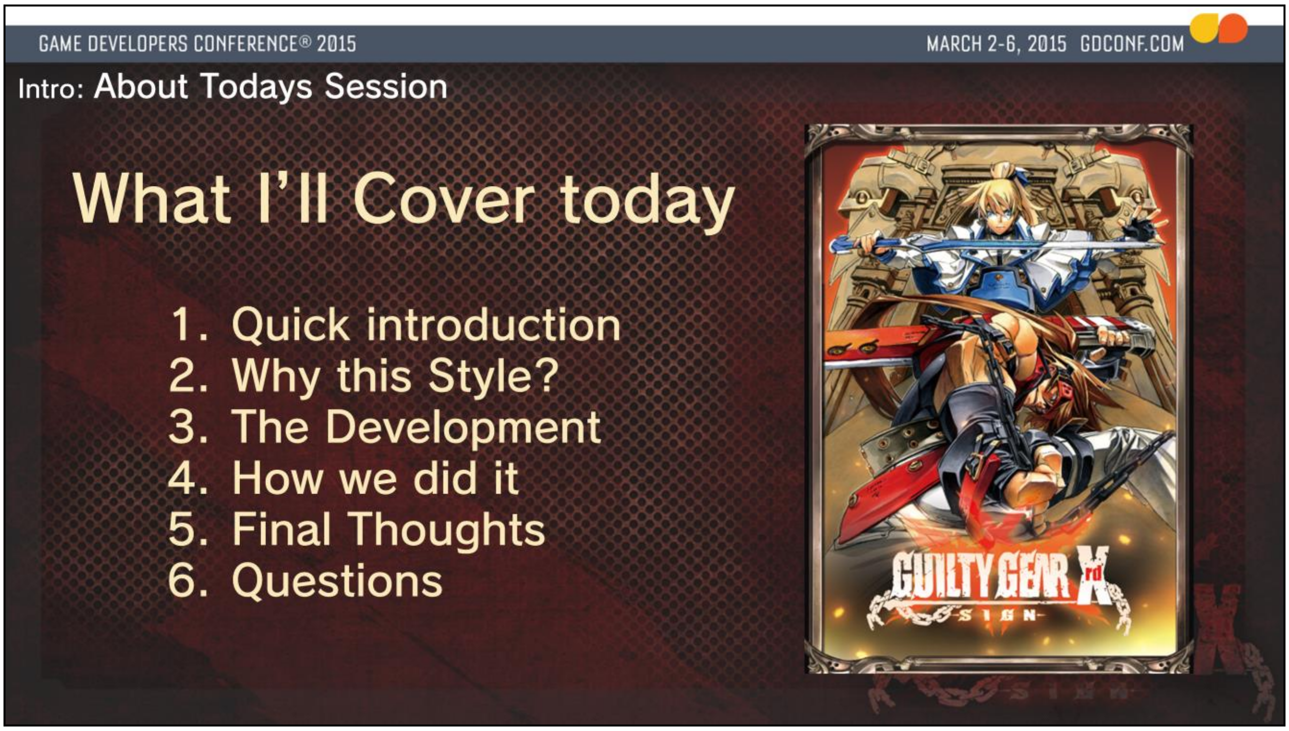


GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

GuiltyGear Xrd's Art Style : The X Factor Between 2D and 3D

Junya Christopher Motomura
Technical Artist, ArcSystemWorks

GAME DEVELOPERS CONFERENCE®
MOSCONE CENTER · SAN FRANCISCO, CA
MARCH 2-6, 2015 · EXPO: MARCH 4-6, 2015



Intro: About Todays Session

What I'll Cover today

(Trailer)
Today, I would like to talk about the art style of our latest title GuiltyGear Xrd.
Why did we choose this art style? What made it so hard to fake 2d in 3d?
How did we accomplish it? What made the difference?

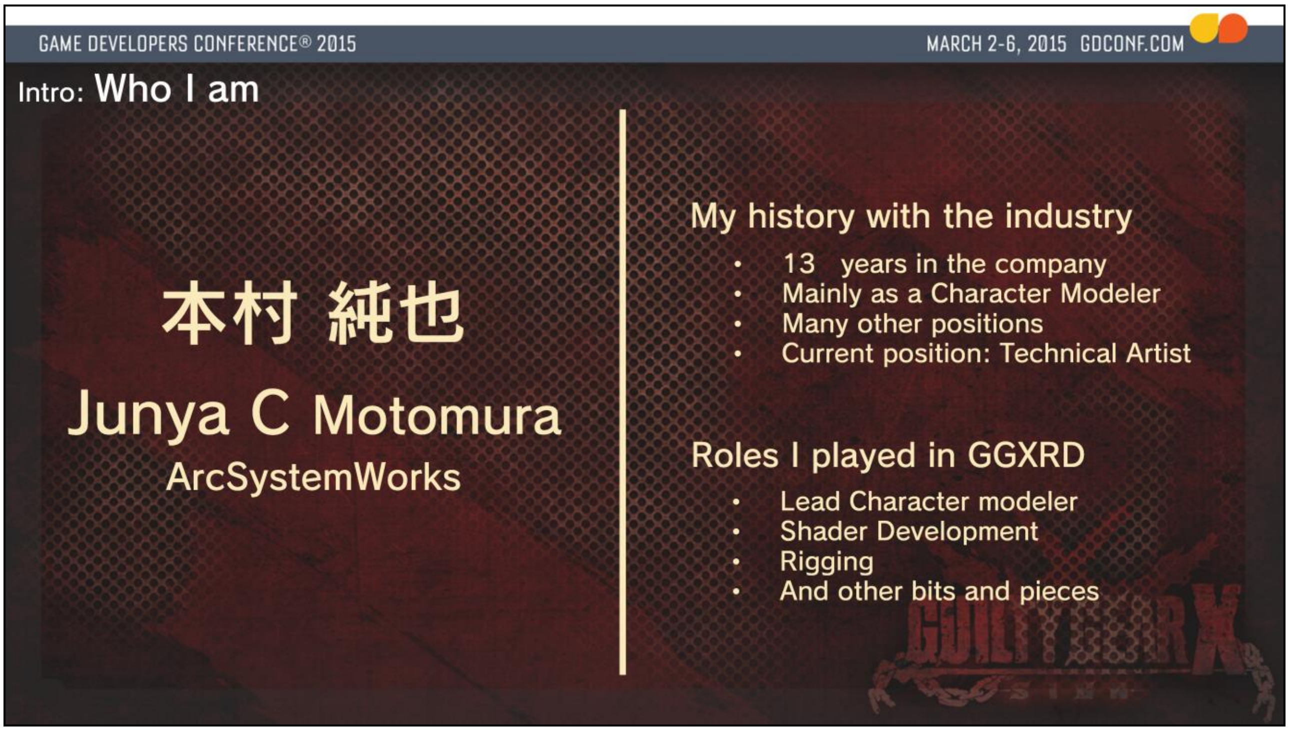
We only have a limited time frame, and I won't be able to cover each and every aspect of it,
So it will mostly consist on the 3D Character Art and the Animation.

GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

Quick Introduction

GUILTY GEAR Xrd SIGN

First, let me do a super fast introduction,


GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

Intro: Who I am

本村 純也

Junya C Motomura

ArcSystemWorks

My history with the industry

- 13 years in the company
- Mainly as a Character Modeler
- Many other positions
- Current position: Technical Artist

Roles I played in GGXRD

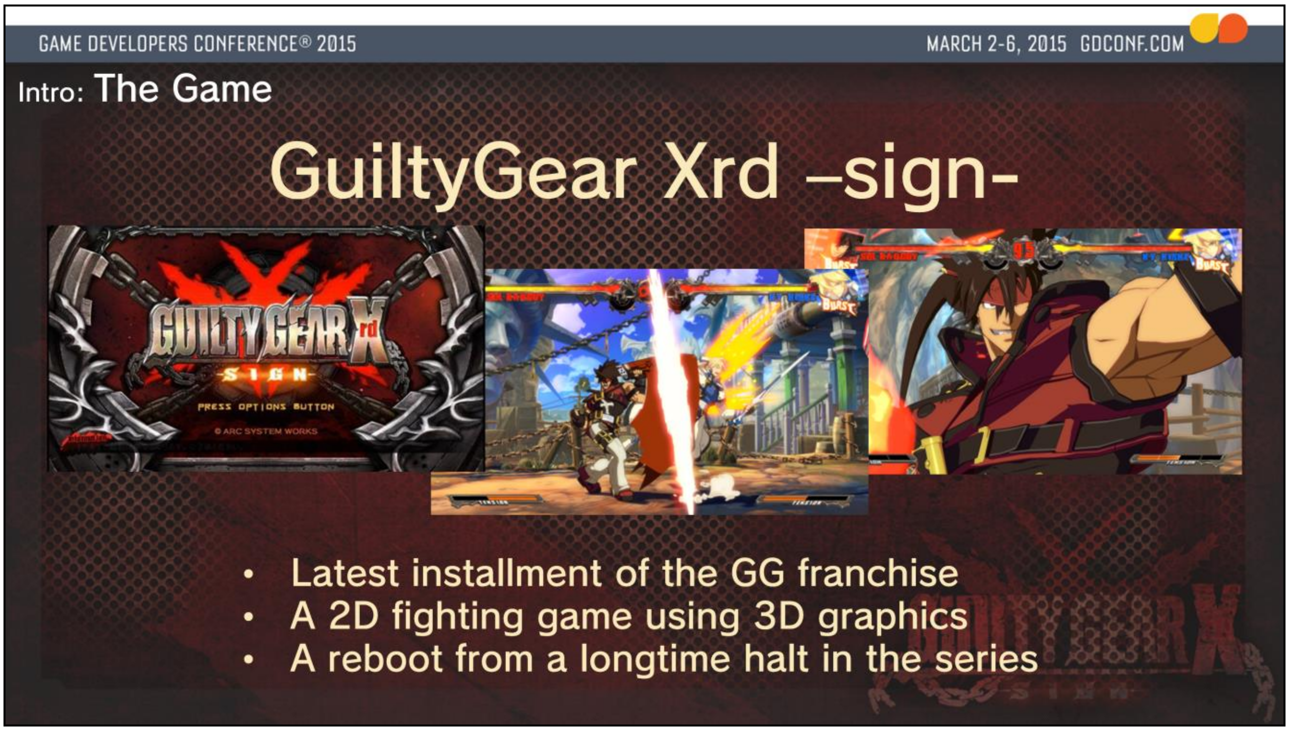
- Lead Character modeler
- Shader Development
- Rigging
- And other bits and pieces

I am Junya Christopher Motomura, a Technical Artist and Character Modeler working at ArcSystemWorks.

I've been Working at the company for 13 years, mostly as a 3D artist, but in many other positions, such as game designer, game director , story script, localization, voice acting, and even some singing.

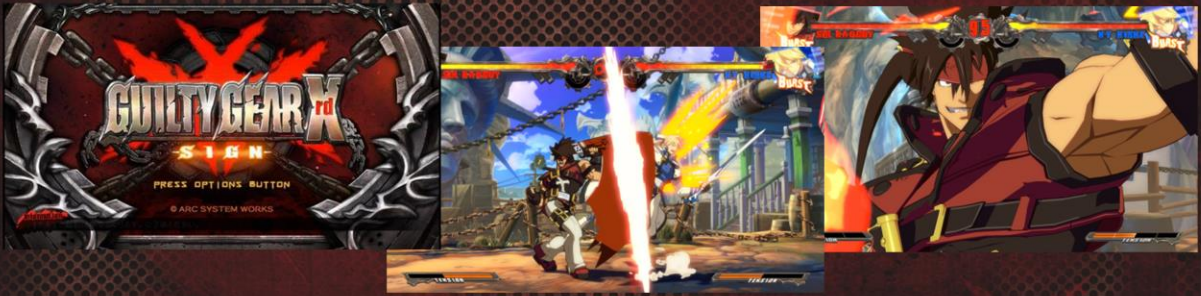
My role in GuiltyGearXrd were
Lead Character modeler , Character look development, including Writing Shaders , Rigging, and other bits n pieces

So, basically I'm the guy behind how the 3D characters look on the screen.



Intro: The Game

GuiltyGear Xrd –sign-



- Latest installment of the GG franchise
- A 2D fighting game using 3D graphics
- A reboot from a longtime halt in the series

GuiltyGearXrd is the latest installment of the GuiltyGear franchise, a long running series of 2D fighting games.

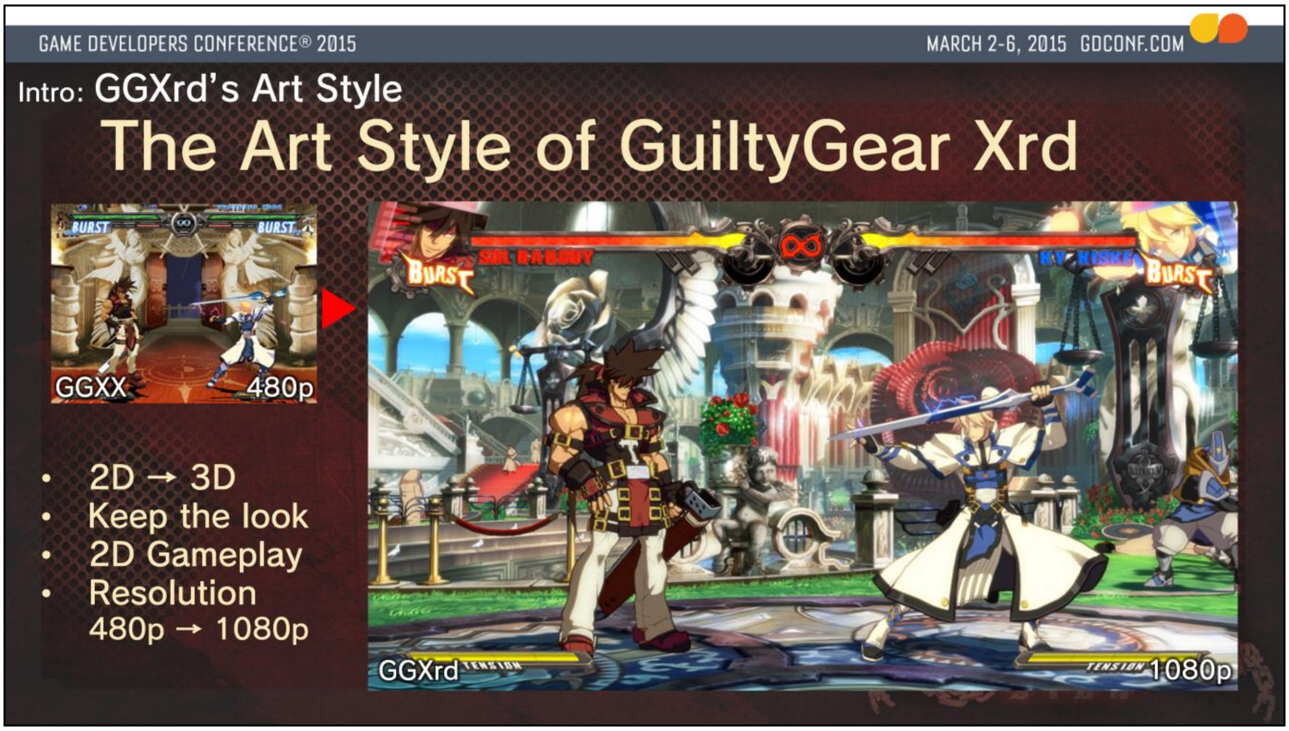
Previously, most games in the series used 2D sprite based graphics.

This title is a reboot from a long time halt on the series, and has some drastic change in art style.

We chose to do it in 3D, instead of 2D sprites, while maintaining the charm the sprites had.

This was a great leap of faith, and we are thankful it paid out.

(Game Play Video)



GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

Intro: GGXrd's Art Style

The Art Style of GuiltyGear Xrd

- 2D → 3D
- Keep the look
- 2D Gameplay
- Resolution
  480p → 1080p

In GuiltygearXrd, our goal was to keep the look and feel of the old titles, and represent them in a new form, 3D.

The visual's needed to resemble those of the sprites we have used for so many years, and look even better with higher resolution in mind.

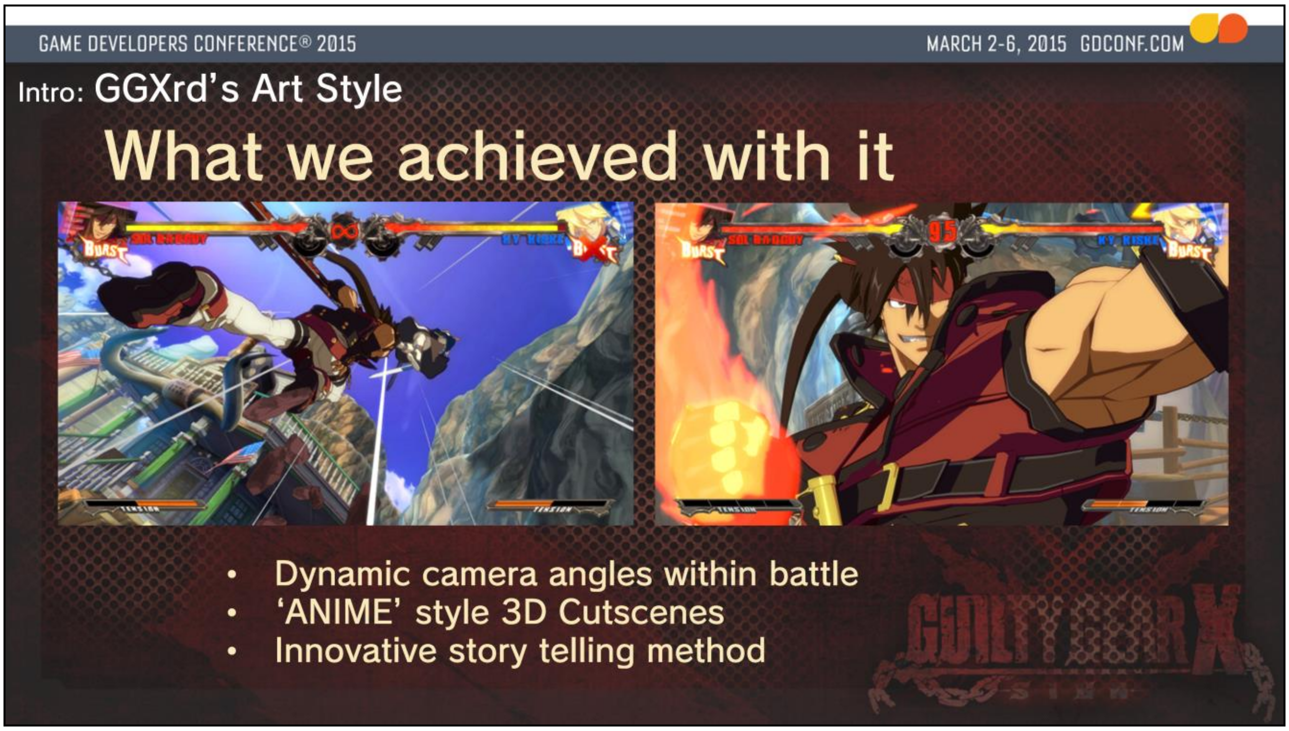
Although it utilizes 3D a lot, the game it self is still played in a 2D plane, being true to it's nature.

As you may see, the art style is strictly based on Japanese Anime, which is also a tradition brought over from the 2D sprites back in the days.

This was done by combining custom Cel-shaders and 3D models made especially for those shaders.

Turning 3D, meant that we were no longer limited by resolution.

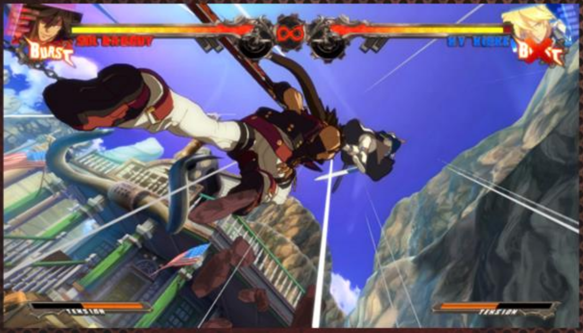
The screen is now 1080p for PS4, 720p for PS3, so that was a great boost in resolution from the old 480p sprites.

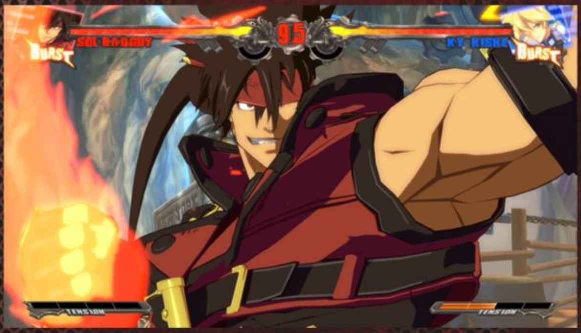


GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

Intro: GGXrd's Art Style

What we achieved with it





- Dynamic camera angles within battle
- 'ANIME' style 3D Cutscenes
- Innovative story telling method

The shift to 3D graphics gave us much more benefit than just higher resolution.

It's the freedom we achieved on the camera.

The Cel-shaded 3D graphics work well in the battle screen, but the real shine is when the camera swings.

Being able to move the camera at will, dynamic camera angles are now viable in the game.

For special attacks and finish scenes, the camera moves around the 3D space and shows the action from a more dramatic perspective.

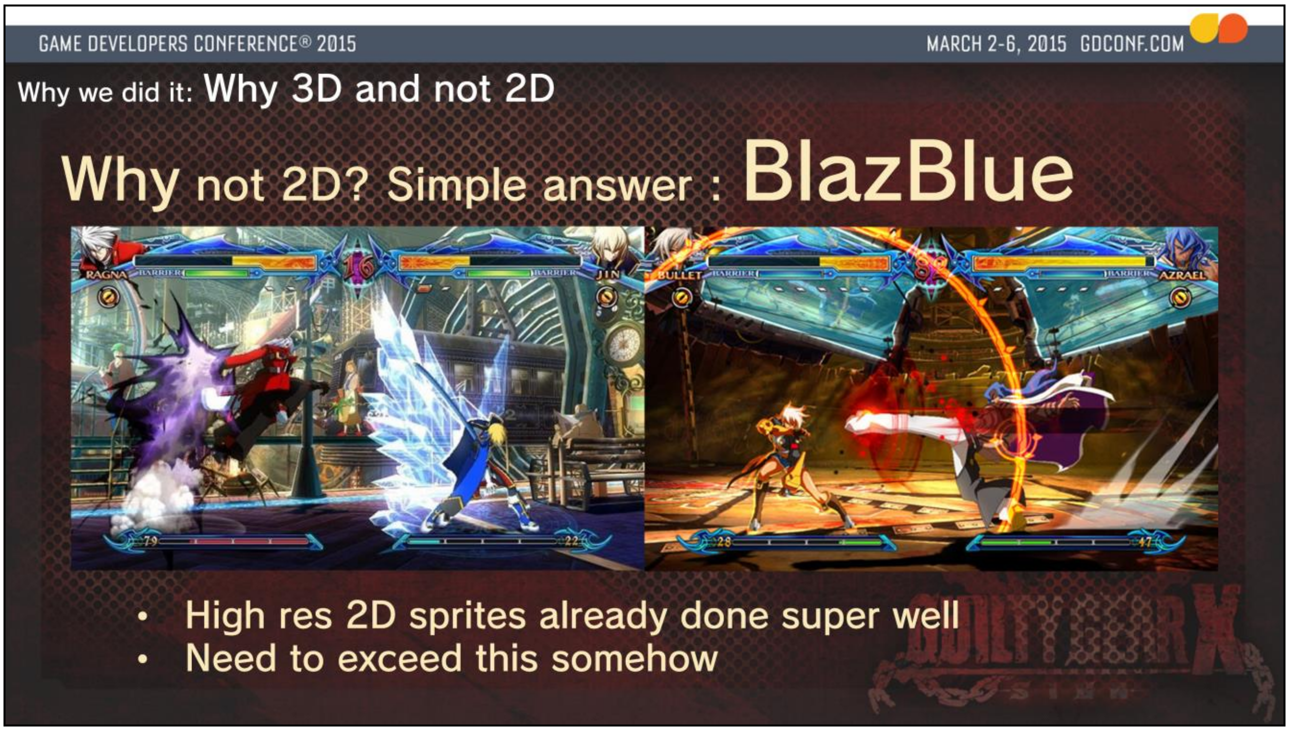
GAME DEVELOPERS CONFERENCE® 2015

MARCH 2-6, 2015 GDCONF.COM

Why this Style?

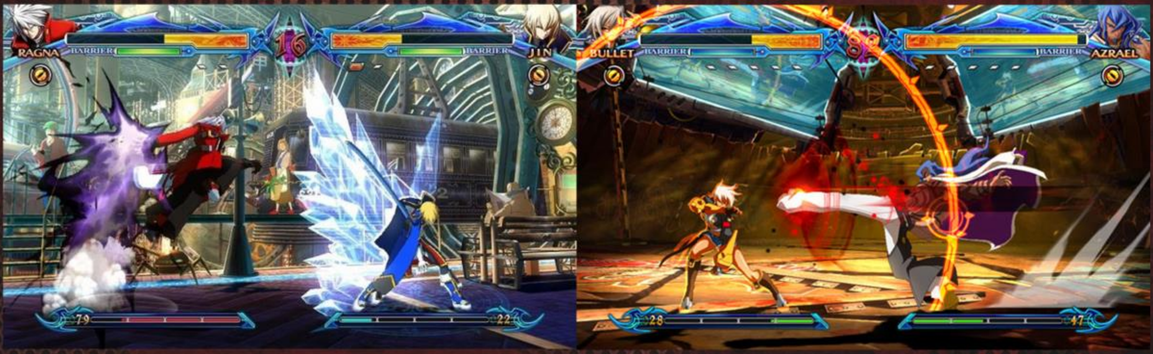
GUILTY GEAR Xrd SIGN

So we got a new art style, but where did the idea come from?
Why did we chose this path?
Why didn't we stick to 2D sprites?
Well, there is a simple answer..



Why we did it: Why 3D and not 2D

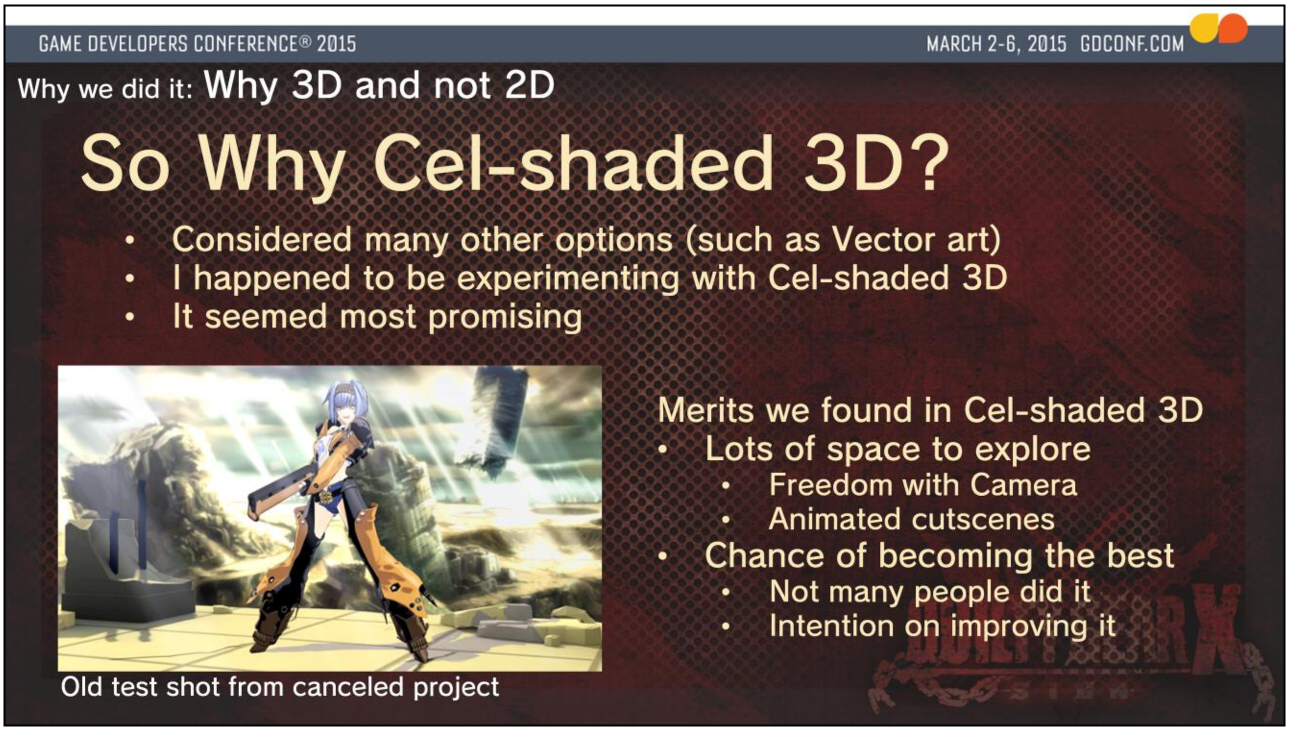
Why not 2D? Simple answer : BlazBlue



- High res 2D sprites already done super well
- Need to exceed this somehow

We already had Blazblue.

BlazBlue is our other 2D fighting game franchise that utilizes high res sprites.
And it does it so well, we didn't see much room for improvement.
We didn’t want to compete with our selves, so for GuiltyGearXrd, we needed something different"

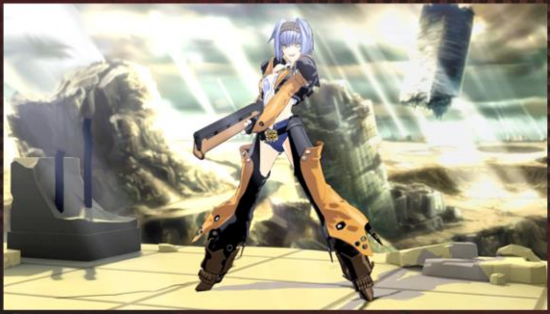


GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

Why we did it: Why 3D and not 2D

So Why Cel-shaded 3D?

- Considered many other options (such as Vector art)
- I happened to be experimenting with Cel-shaded 3D
- It seemed most promising



Old test shot from canceled project

Merits we found in Cel-shaded 3D

- Lots of space to explore
  - Freedom with Camera
  - Animated cutscenes
- Chance of becoming the best
  - Not many people did it
  - Intention on improving it

And that's when we decided to use Cel-shaded 3D.

Well, We considered many other options such as vector art, ultra high res sprites, and others.

But what emerged most promising was Cel-shaded 3D, which I , at that time, happened to be experimenting with.

This here is a piece from a canceled project that I was working on.
The project didn't work out, but the art style showed a lot of potential.

So when GuiltyGearXrd finally started to take form, we decided to take this art style even further.
If done right , we could maintain our 'Anime' style while gaining the various benefits of 3D.
Dynamic camera angles and Animated Cut scene were things we were craving for back in the days

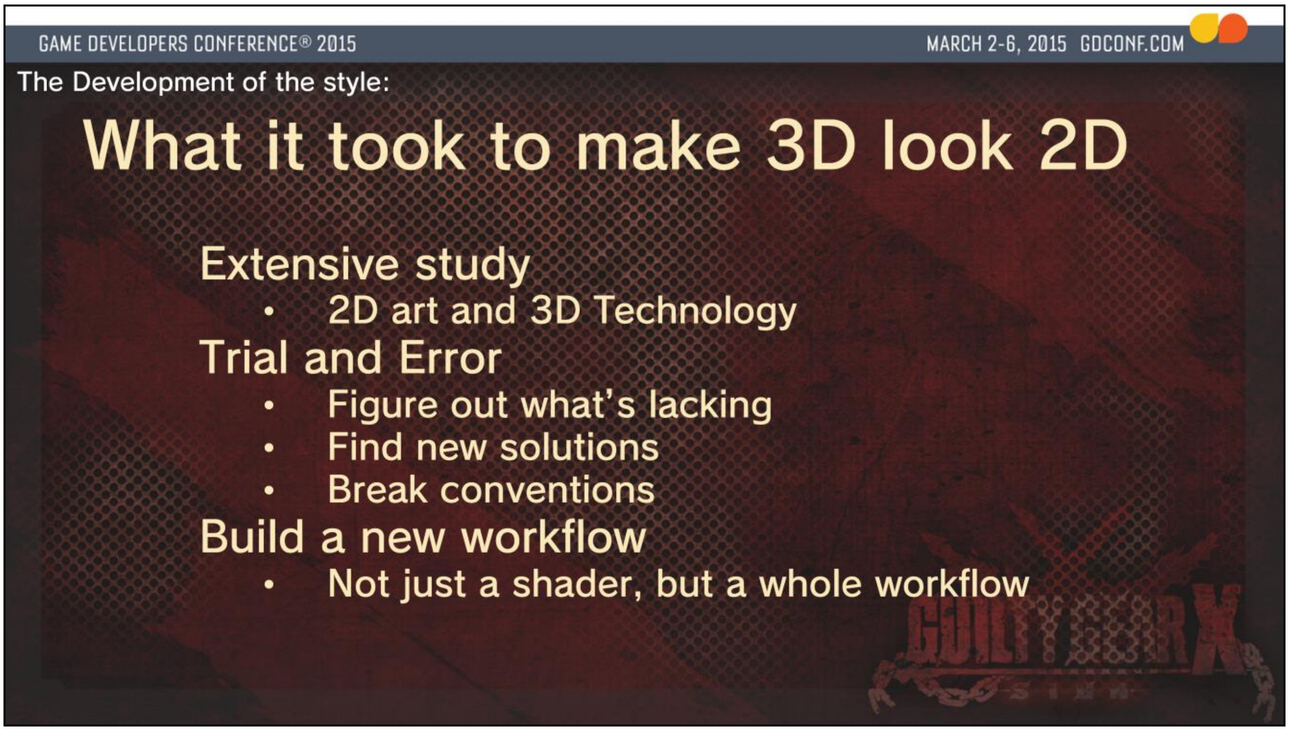
We also found potential in it, because the technique was not well explored, and had lots of room for improvement.
By advancing the state of the art, there was a great chance to stand out in a crowded market.

GAME DEVELOPERS CONFERENCE® 2015

MARCH 2-6, 2015 GDCONF.COM

The Development

So we decided which way to go, now we needed to actually start moving.
Lets have a look at how we faced the challenge.



GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

The Development of the style:

What it took to make 3D look 2D

Extensive study
- 2D art and 3D Technology

Trial and Error
- Figure out what's lacking
- Find new solutions
- Break conventions

Build a new workflow
- Not just a shader, but a whole workflow

Getting 3D to look like 2D was going to be a tough challenge, and we knew it.
It was seldom accomplished, so we had to find our own way, and pave our own path.

A lot of study went in to both the 2Dart and 3Dtechnology.
Because knowing both sides was essential to tackle this task.

Elements that were not translated well enough from 2D to 3D were picked out, and solved one by one.
We broke a lot of conventions on the way, because they just didn't Suit this style

As a result, we came up with a new unique workflow, mostly built from scratch

GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

The Development of the style:

Great Tools we used

Unreal engine 3

- Need for a multi platform engine
- Fixed deadline and limited human resource
- Ease of use for artists
- Stable, at the end of its lifecycle
- Deeply customizable with full source code

Autodesk Softimage

- Highly optimized for game dev
- Great modeling and animation tools
- Built in RT shader editor
- Unmatched flexibility letting us continually improve our assets throughout the development

At the same time,
we introduced our selves to new tools in order to face this huge challenge.

First, Unreal engine 3.
We considered multiple engines, and Unreal Engine 3 was chosen because of multiple reasons.

We needed a new engine that could handle multiple platforms,
Our deadline was strict, and building an engine from scratch was not an option.
We had a team mostly consisting of artists, and the artist friendly design of Unreal3 matched our needs.
The fact that Unreal4 was on it’s way, and Unreal3 was around the end of it’s lifecycle, also benefitted us.
The engine was well debugged and stable, and we had a lot of freedom customizing it with the full source code.

The other tool that helped us a lot was Autodesk Softimage.
The flexibility in modeling and animation let us focus on quality.
A built in shader editor made possible to experiment with the Cel-shading a lot.
Because of it’s non destructive workflow, we were able to improve our assets continually throughout the whole production, right till the last minute.
With out this tool, I believe it just wouldn’t have happened.

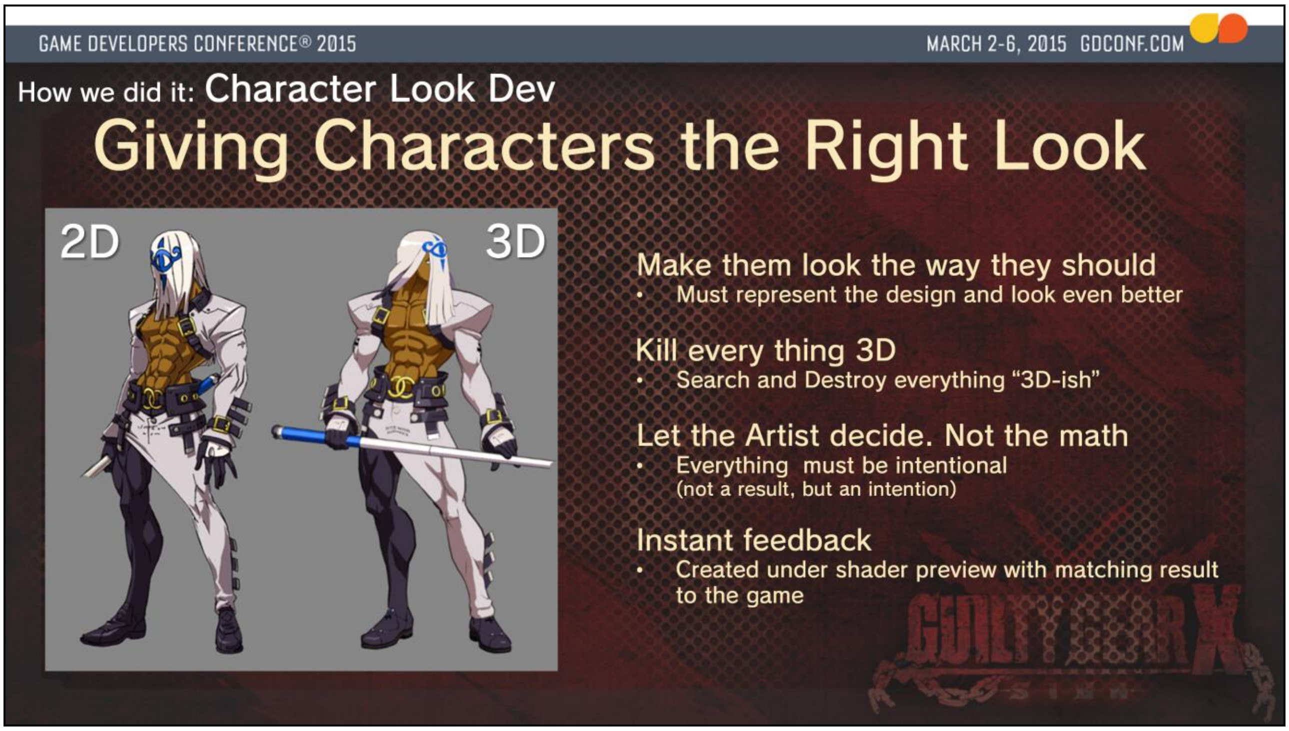
GAME DEVELOPERS CONFERENCE® 2015

MARCH 2-6, 2015 GDCONF.COM

HOW we did it :
Character Look Dev

GUILTY GEAR Xrd SIGN

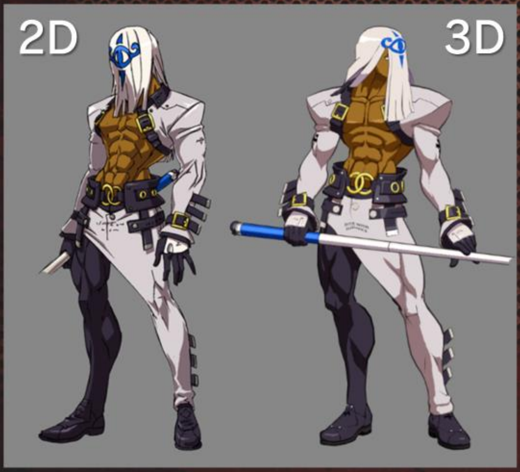
Now, lets dive in to the details of "How" we did it.
Starting with the look of the characters



GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

How we did it: Character Look Dev

Giving Characters the Right Look



Make them look the way they should
- Must represent the design and look even better

Kill every thing 3D
- Search and Destroy everything "3D-ish"

Let the Artist decide. Not the math
- Everything must be intentional
  (not a result, but an intention)

Instant feedback
- Created under shader preview with matching result to the game

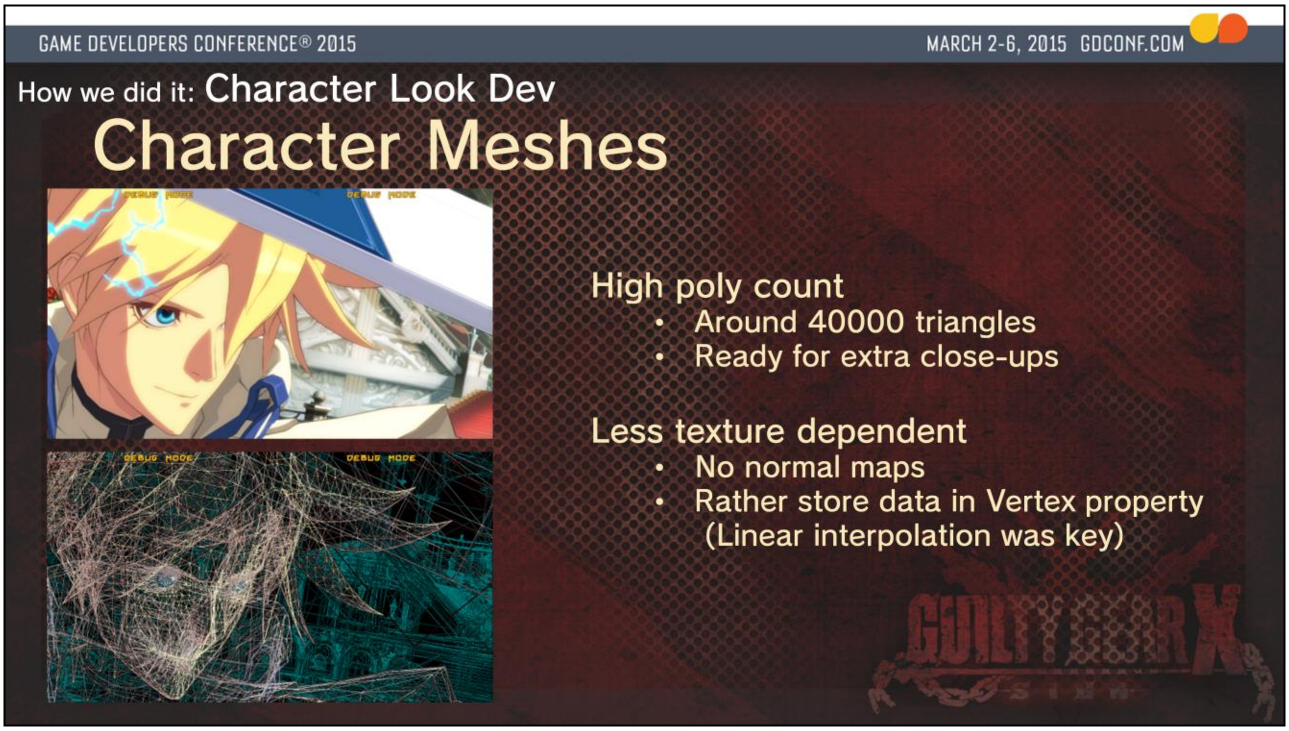
The most important assets of our game were the characters.
Without getting them right, there was no way we were going to get the game right.

First, the characters needed to represent their 2D design well enough to appeal to the audience.
The models had to be as good as the designs, or even better.

And they had to look 2D as much as possible.
To do this, our principle was simple.
Kill everything 3D. If you find something that looks 3D, you just have to find a way to avoid it.

This often boiled down to hand crafting by the artists,
Because while the math within the shader is always "correct", "correct" is just not good enough.
To get a convincing 2D look, every thing on screen has to be an intentional choice, not just a result of a calculation.

To achieve all of this, instant feed back was key.
We were fortunate to have a real time shader preview on our modeling software,
that gave us a trust worthy representation of the actual game.
(LIVE DEMO)

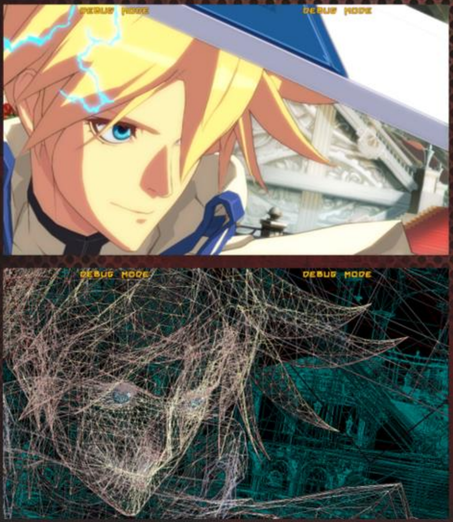


GAME DEVELOPERS CONFERENCE® 2015

MARCH 2-6, 2015 GDCONF.COM

How we did it: Character Look Dev

Character Meshes



High poly count
- Around 40000 triangles
- Ready for extra close-ups

Less texture dependent
- No normal maps
- Rather store data in Vertex property (Linear interpolation was key)

The models have a fairly high poly count, around 40000 triangles in average.

We knew we were going to do extra close-ups in cut scenes, so the models had to hold up even at an extremely close distance.

Details were directly modeled-in with geometry.

What may be special about the character models in GGXrd, are that they are very texture independent.

For example, unlike average 3D games, we do not use textures such as normal maps for our characters.

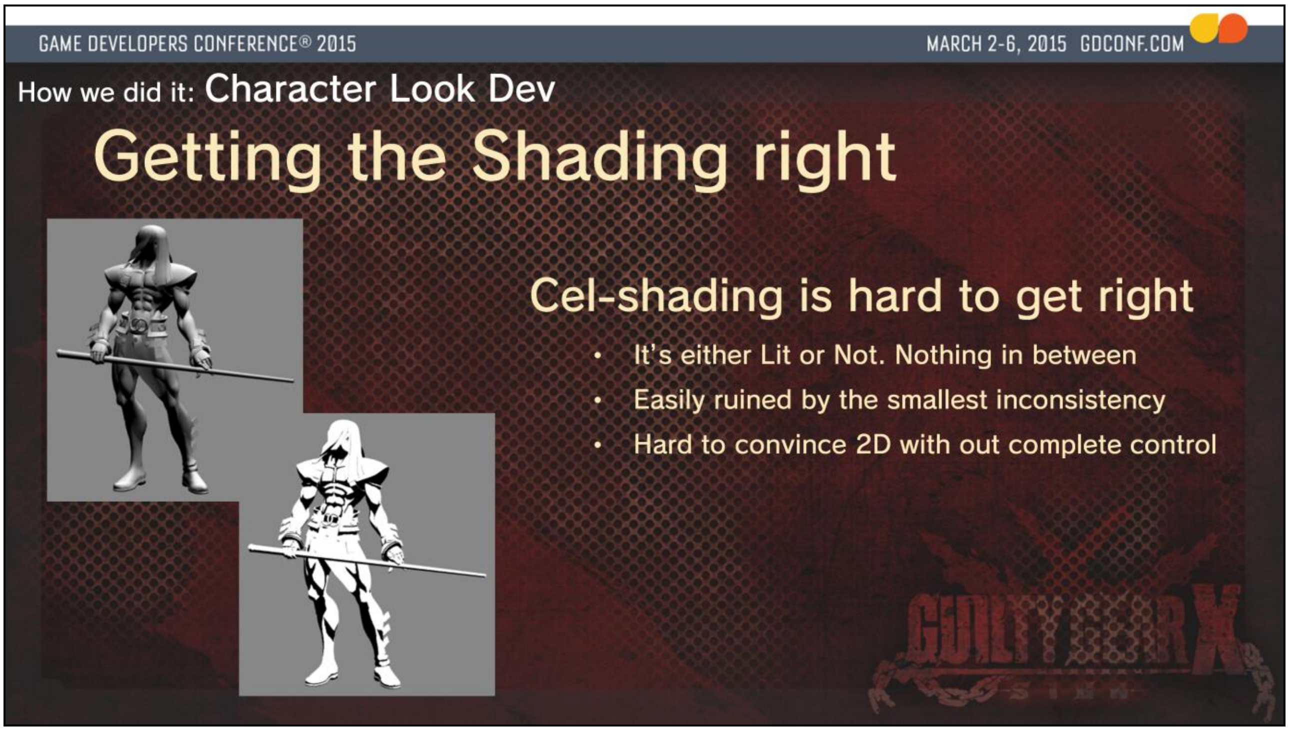
Instead, we use vertex properties such as Vertex Normals, Vertex Colors, and UV coordinates a lot to store data in.

The reason for this is, again, the extra close-ups.

The problems with texture data is that they are very resolution dependent.

Pixel data easily can get jaggy at super close-ups, and that can not be helped unless you have a ultra high resolution.

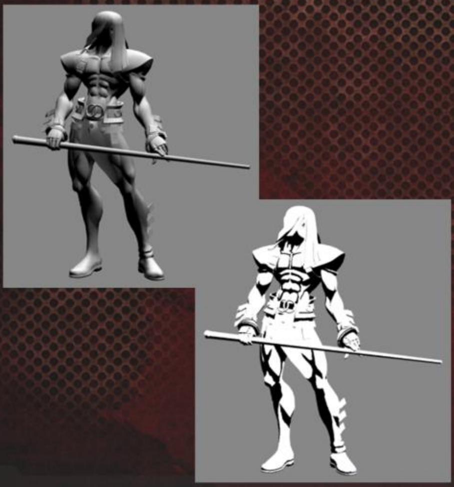
On the other hand, vertex properties are linearly interpolated between vertices, which is completely resolution independent.



GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

How we did it: Character Look Dev

Getting the Shading right



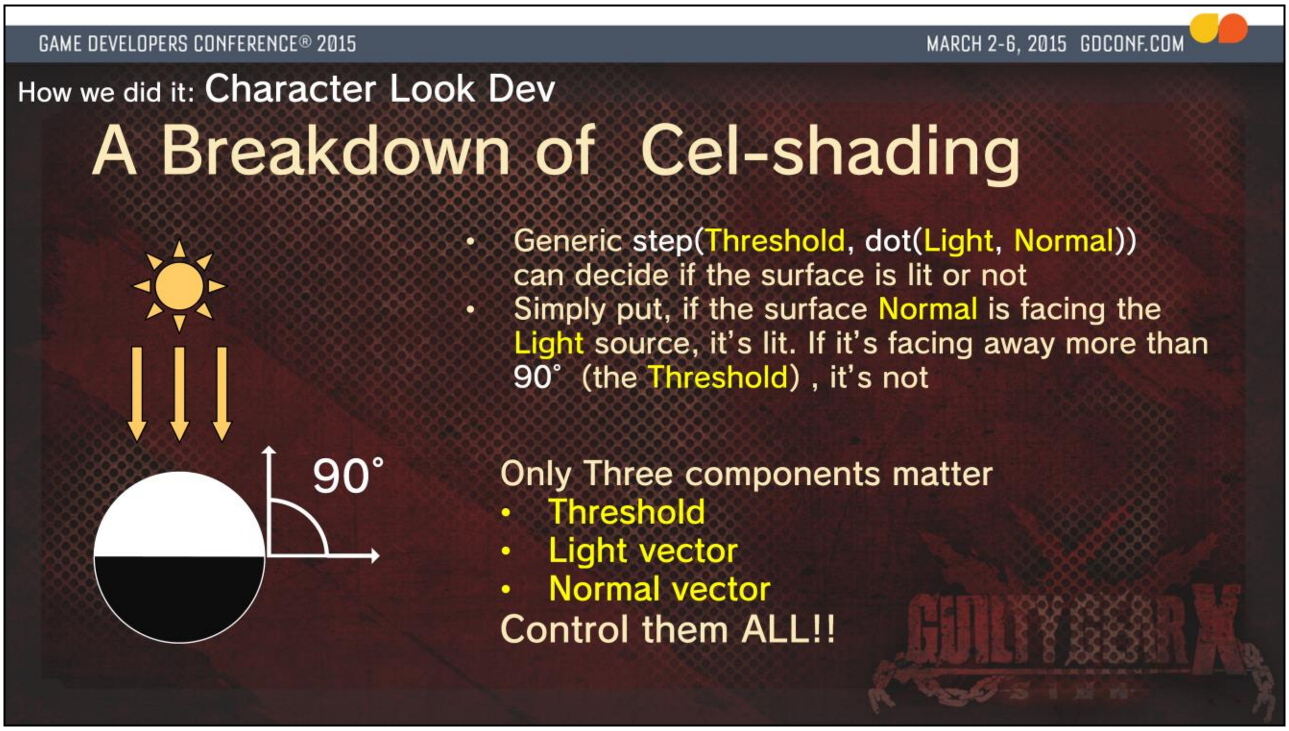
Cel-shading is hard to get right

- It's either Lit or Not. Nothing in between
- Easily ruined by the smallest inconsistency
- Hard to convince 2D with out complete control

Now, let's go in to the actual shading.

In Cel-shading, the surface is either lit, or not, and no value in between.
This is one of the reasons Cel-shading is so hard to get right.
In a drawing, the artist will choose the most convincing distribution of light and darkness.
But in a shader, it's all math, where unforgiving thresholds, mercilessly splits between light and dark.
In Cel-shading, every little noise on the surface will become extremely distracting.
The slightest difference in the surface normal may end up as a huge blotch .

A convincing 2D look can only be achieved with precise control over the distribution of shades.



How we did it: Character Look Dev

A Breakdown of Cel-shading

- Generic step(Threshold, dot(Light, Normal)) can decide if the surface is lit or not
- Simply put, if the surface Normal is facing the Light source, it's lit. If it's facing away more than 90° (the Threshold) , it's not

Only Three components matter
- Threshold
- Light vector
- Normal vector

Control them ALL!!

The code for our Cel-shader is actually pretty simple.

In the main part of the calculation, a generic "step" function is used to decide if that pixel is lit or not.

To put it simple for artists, this code works like this.

If the surface normal is facing the light source, it's lit.

If it's facing more than 90 degrees away from the light source, it's not.

90 degrees, is the threshold here, but it could be any other value.

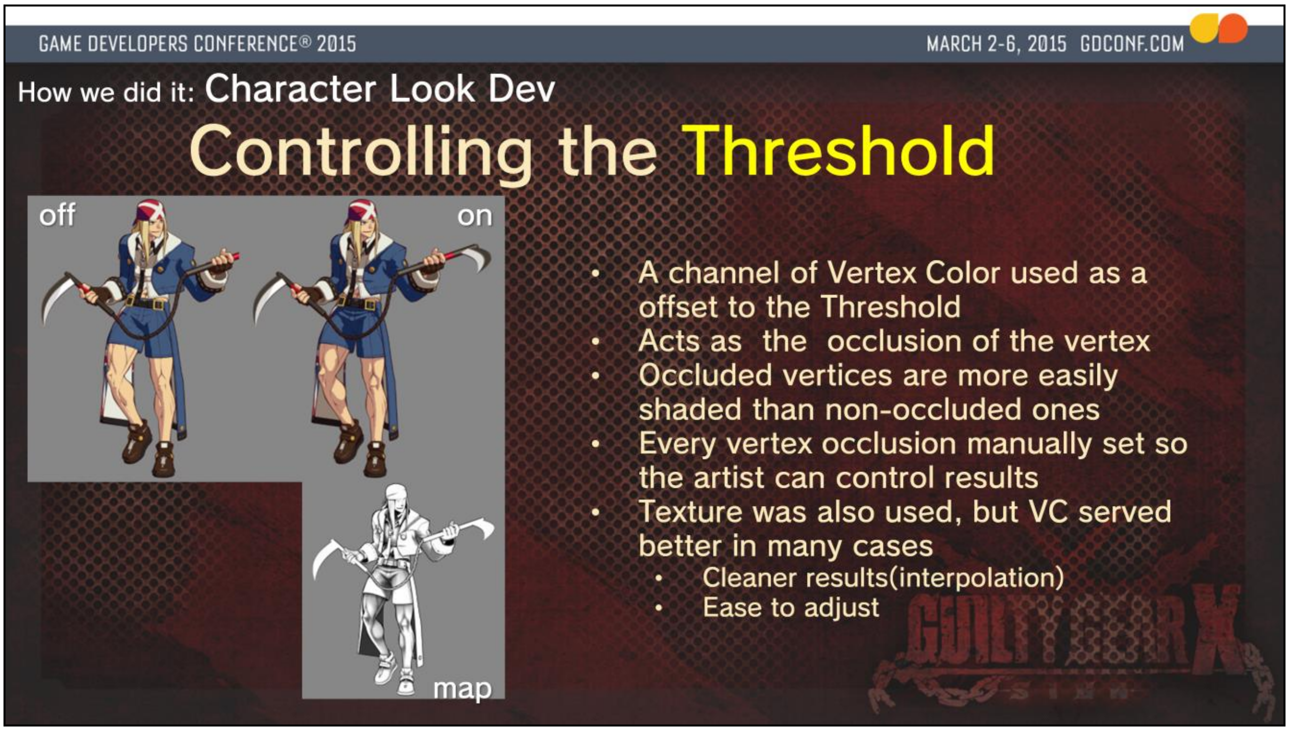
What matters here are the following.

The Threshold, the Light Vector, and the Normal Vector.

And that's all. Only these 3 components matter in THIS shading code,

So what we have to do is simple.

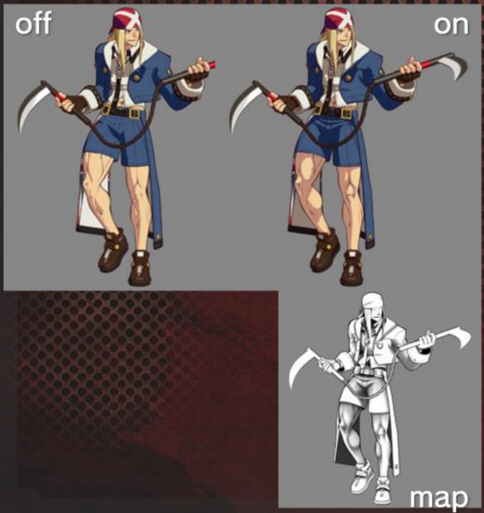
Take full Control over those three.



GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

How we did it: Character Look Dev

Controlling the Threshold



- A channel of Vertex Color used as a offset to the Threshold
- Acts as the occlusion of the vertex
- Occluded vertices are more easily shaded than non-occluded ones
- Every vertex occlusion manually set so the artist can control results
- Texture was also used, but VC served better in many cases
  - Cleaner results(interpolation)
  - Ease to adjust

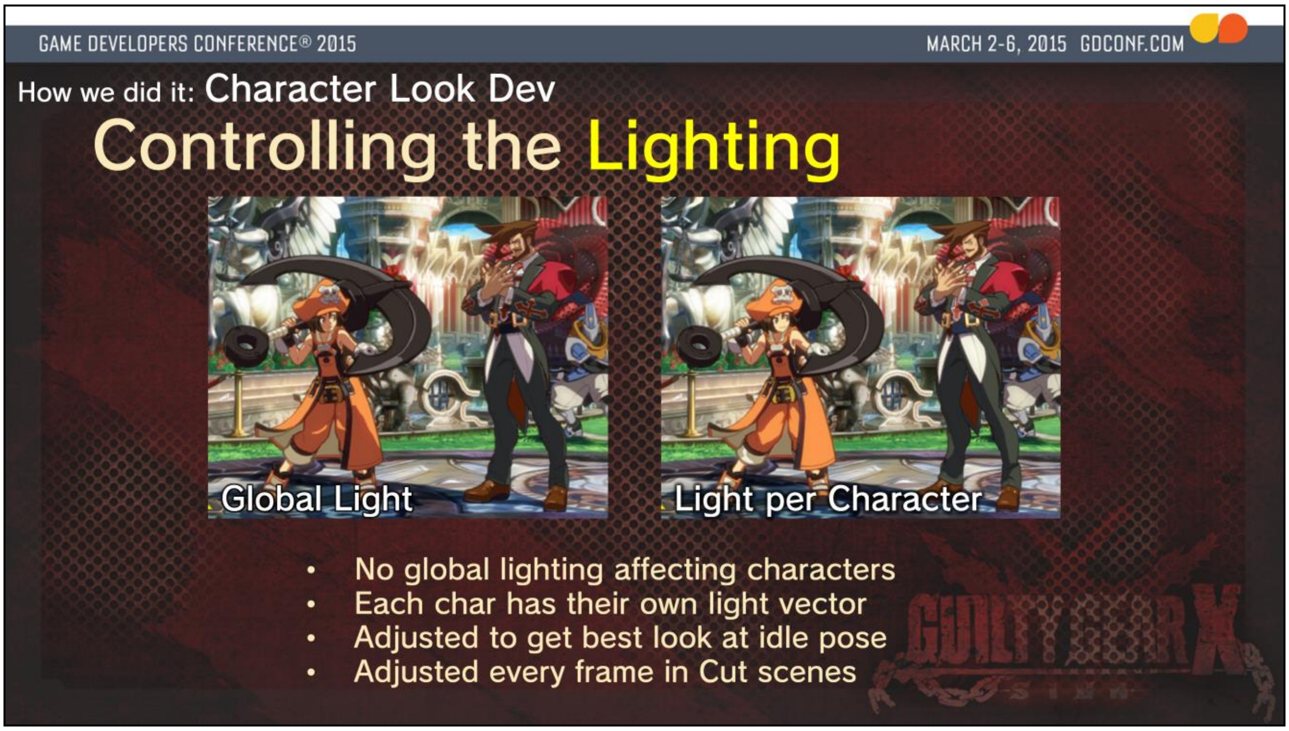
First lets look at the Threshold.

We used a channel from the Vertex Colors as an offset on the Threshold.
This made it possible for the artist to make certain areas on the mesh get darker more easily.

This could be used to represent the occlusion of that vertex.
Areas more occluded from the light, have a less chance to be lit.
The artist will go around the mesh setting values, defining the areas more likely to be shaded.
The artist may even set the value to Zero, making the area always shaded no matter what.

We used texture data in the same way in places,
but Vertex colors served us better in many cases.
Again, texture is dependent on resolution, but Vertex Properties are not.
The linear interpolation between vertices gave us a very clean result with out any pixelation.

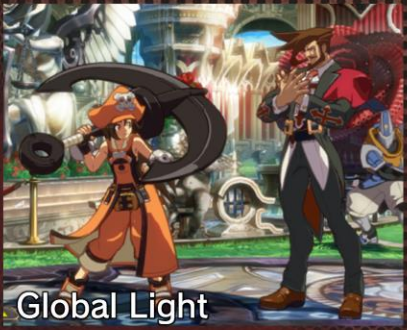
Adding to that, it was much easier to adjust with Vertex colors, because you get instant feedback from the preview, even while fiddling with the values

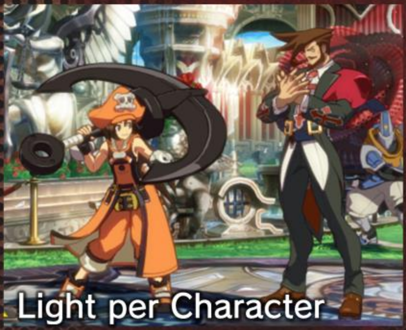


GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

How we did it: Character Look Dev

Controlling the Lighting





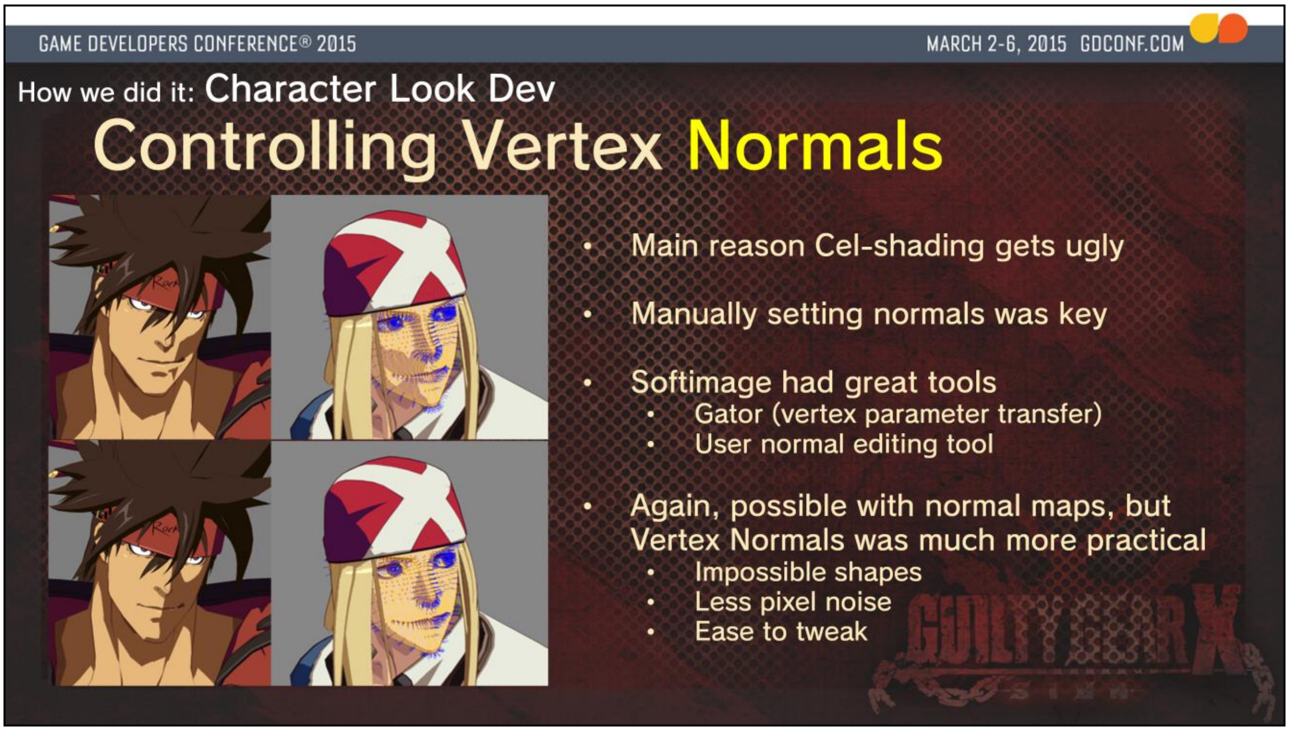
- No global lighting affecting characters
- Each char has their own light vector
- Adjusted to get best look at idle pose
- Adjusted every frame in Cut scenes

Next, to Lighting.

Unlike normal 3D games, GGxrd does not have a global lighting system affecting the characters.

Each character has his or her own dedicated light vector that lights their idle pose in the best way.

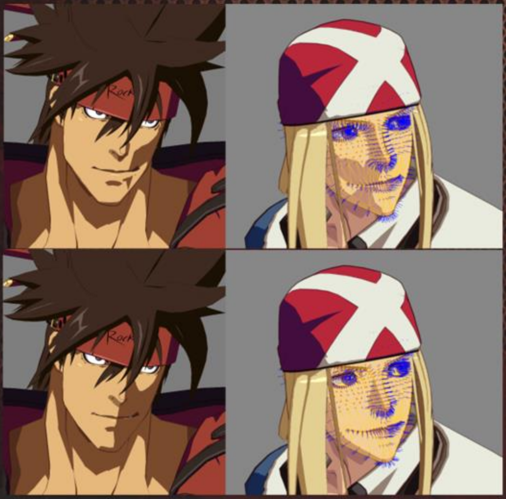
This lighting doesn't change frame by frame in the battle scene, but in cut scenes, it animated along with the character to get the best result each and every frame.



GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

How we did it: Character Look Dev

Controlling Vertex Normals



- Main reason Cel-shading gets ugly
- Manually setting normals was key
- Softimage had great tools
  - Gator (vertex parameter transfer)
  - User normal editing tool
- Again, possible with normal maps, but Vertex Normals was much more practical
  - Impossible shapes
  - Less pixel noise
  - Ease to tweak

And finally the Normals.

Surface normals are one of the main reasons Cel-shading is so hard to get right.

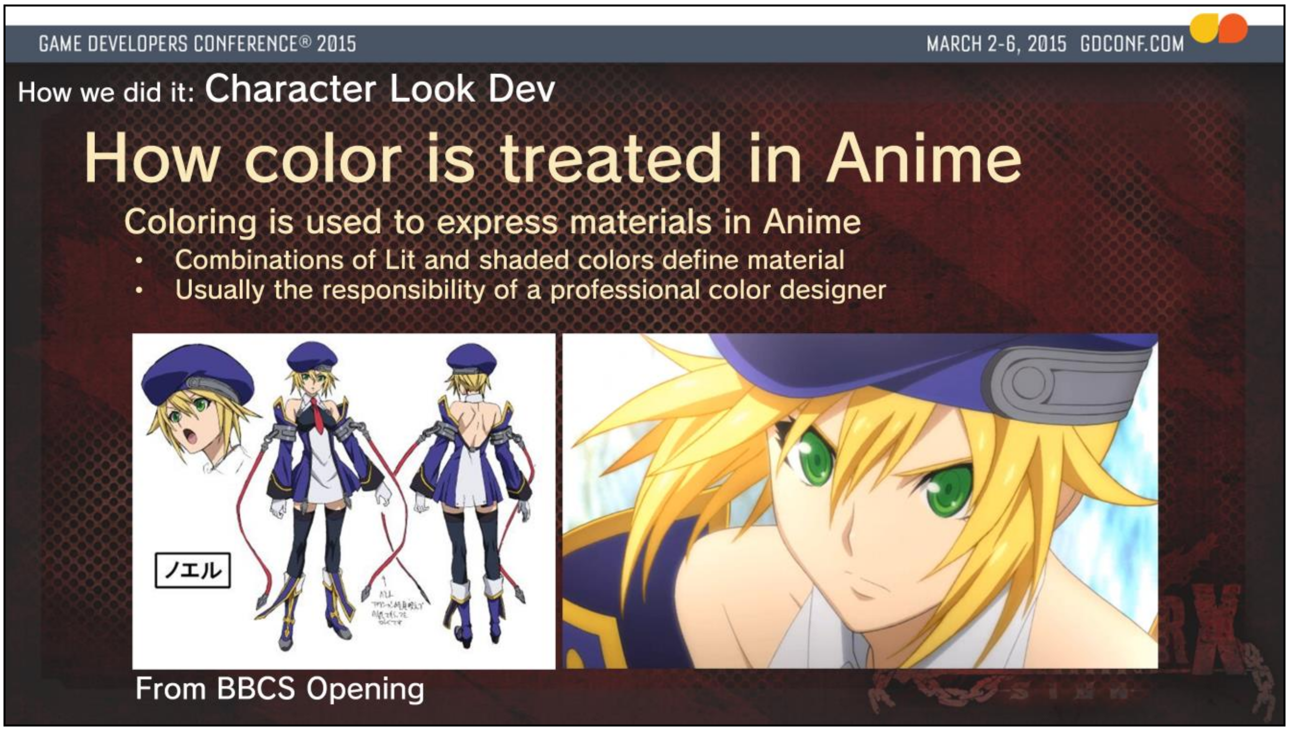
The smallest inconsistency in normals result in a super evident artifact under Cel-shading.

The problem with normals are that they are automatically calculated. And the result of this calculation, is often not what the Artist intended. On the other hand, shading in 2Dart is very intentional.

To close this gap, there was only one thing we had to do. Control the Normals with intention. Fortunately, Softimage had powerful tools built in, for this purpose.

Although editing normals is nothing new to game graphics, we took it even further, To get rid of unintentional shading, we modified the normals on every major feature of the model. The faces of the Characters especially needed to be hand crafted to get a clean anime look.

(LIVE DEMO)



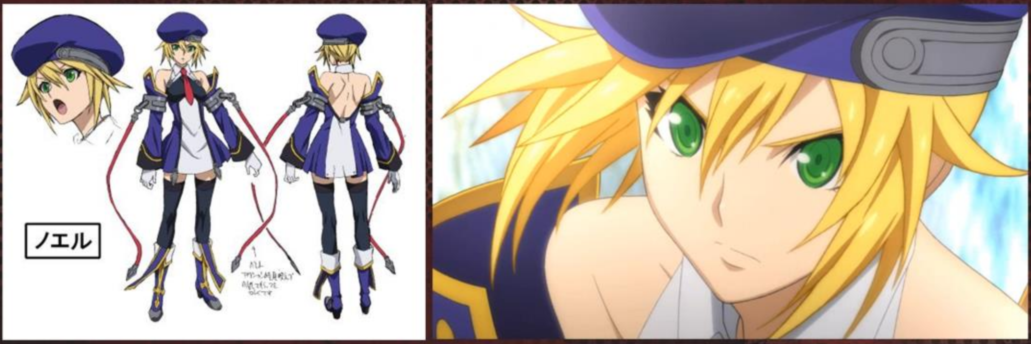
GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

How we did it: Character Look Dev

How color is treated in Anime

Coloring is used to express materials in Anime

- Combinations of Lit and shaded colors define material
- Usually the responsibility of a professional color designer



From BBCS Opening

Now, we sorted the lighting out, we have to think what color to paint the Lit and Shaded areas.

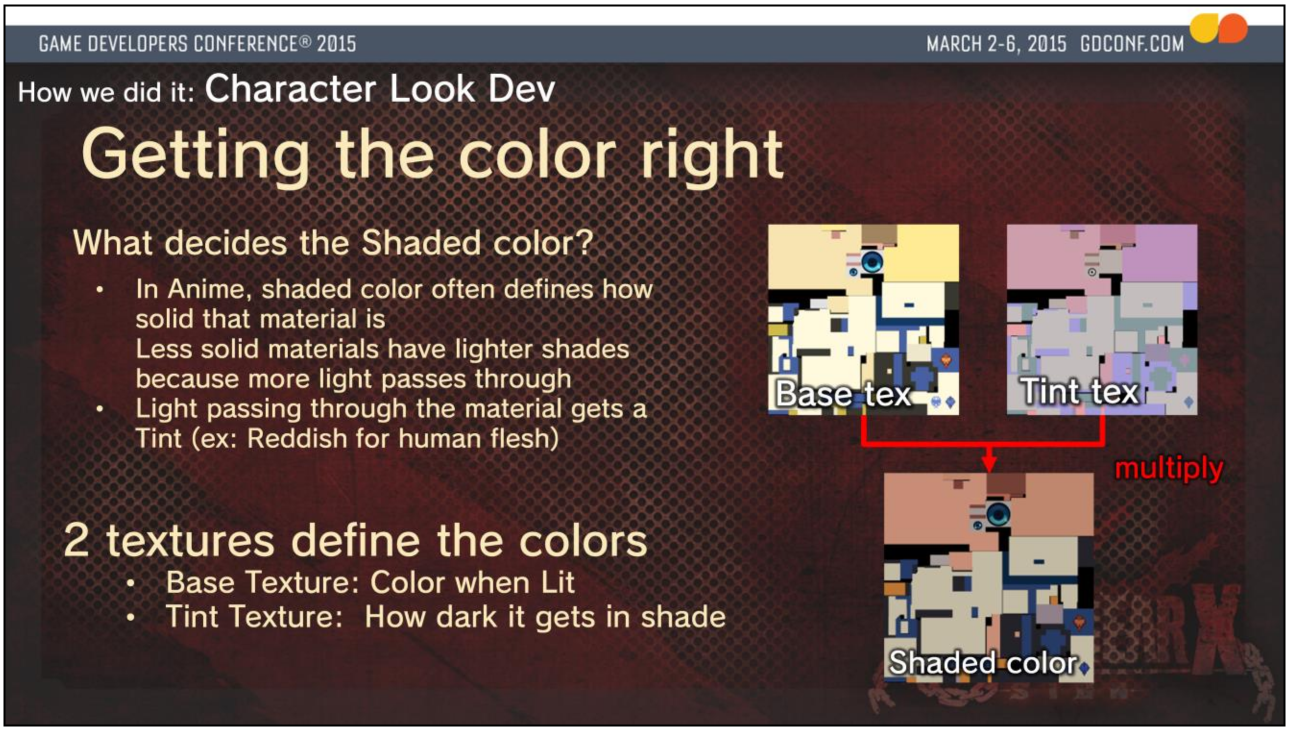
Color selection in anime is vital. Every color is carefully chosen to express not just the material, but the character it self. It is not as simple as just multiplying the shaded area with an ambient color.

The coloring style in Anime can vary from title to title, but one thing is in common. This too, is very intentional.

The combination of the lit and shaded colors can express a vast amount of characteristics. Not just the material , but the atmosphere the character wields.

It's art in it's own sense, and is a task , usually given to a professional color designer.

To get the Anime look right in 3D, we took this by heart , and worked on a way to replicate this process.


GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

How we did it: Character Look Dev

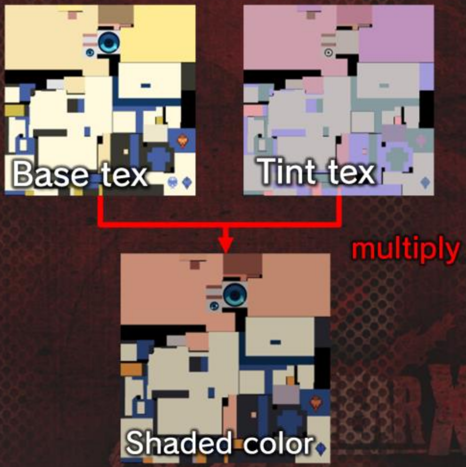
Getting the color right

What decides the Shaded color?

- In Anime, shaded color often defines how solid that material is
  Less solid materials have lighter shades because more light passes through
- Light passing through the material gets a Tint (ex: Reddish for human flesh)

2 textures define the colors

- Base Texture: Color when Lit
- Tint Texture: How dark it gets in shade



By studying anime art style extensively, we found that the shaded color often expresses how solid that material is.

A less solid material will pass more light through it , getting a lighter color for the shades.

The light passing through the material will get Tinted depending on the materials composition.

For example, shades on human skin get a red tint because of the flesh under it.

In order to implement this in the shader, we used two textures. The base texture defines the color of the surface when it's lit.

While the Tint texture defines how dark it gets when shaded. Multiplying the two textures, we get the shaded color.

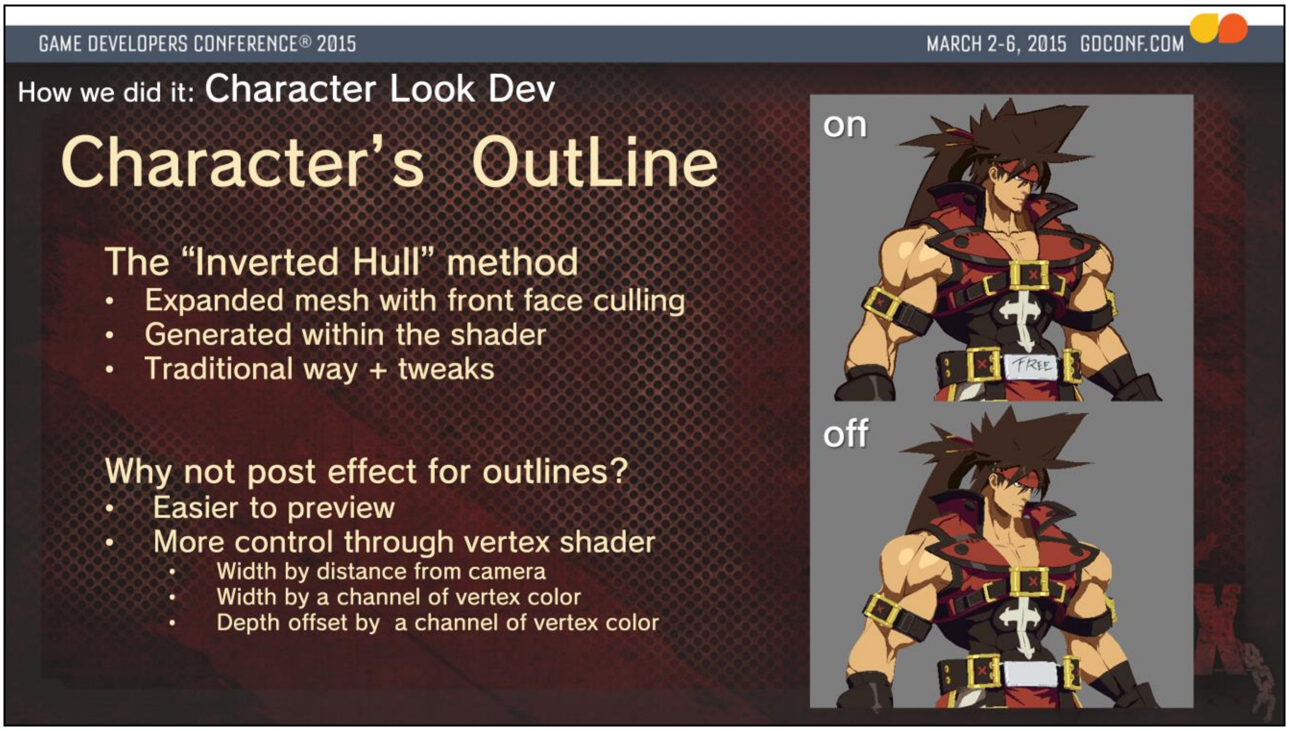
This way, the choice of color is completely up to the artist. The artist can choose what ever color he or she may see fit, for the lit area and the shaded area of that material.

On a side note.

As you can see, both of these textures are just square areas consisting of single colors.

This is because we simply use them as color lookups, and do not draw-in any details to the textures as images.

As stated before, most meaningful information is stored in the mesh it self, rather than in textures.



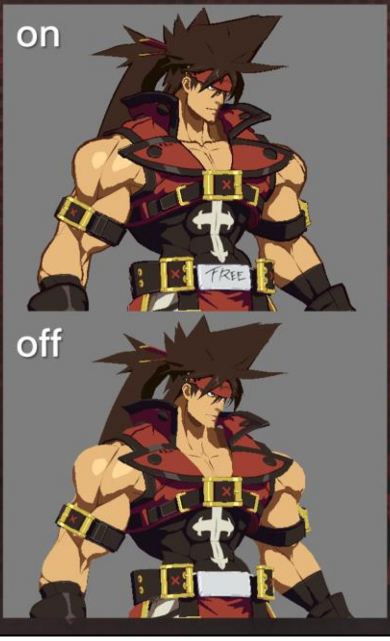
GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

How we did it: Character Look Dev

Character's OutLine

The "Inverted Hull" method
- Expanded mesh with front face culling
- Generated within the shader
- Traditional way + tweaks

Why not post effect for outlines?
- Easier to preview
- More control through vertex shader
    - Width by distance from camera
    - Width by a channel of vertex color
    - Depth offset by a channel of vertex color



Along with colors and shading, Lines play a huge role in cel-shading.

In fact, if you think in black and white manga style, the lines would be the most important aspect of the visual.

In GGXrd, we use a traditional "Inverted Hull" method as outlines for the characters.

A second set of darker polygons are generated in the shader and are expanded in the normals direction to create the hull which shapes the outline.

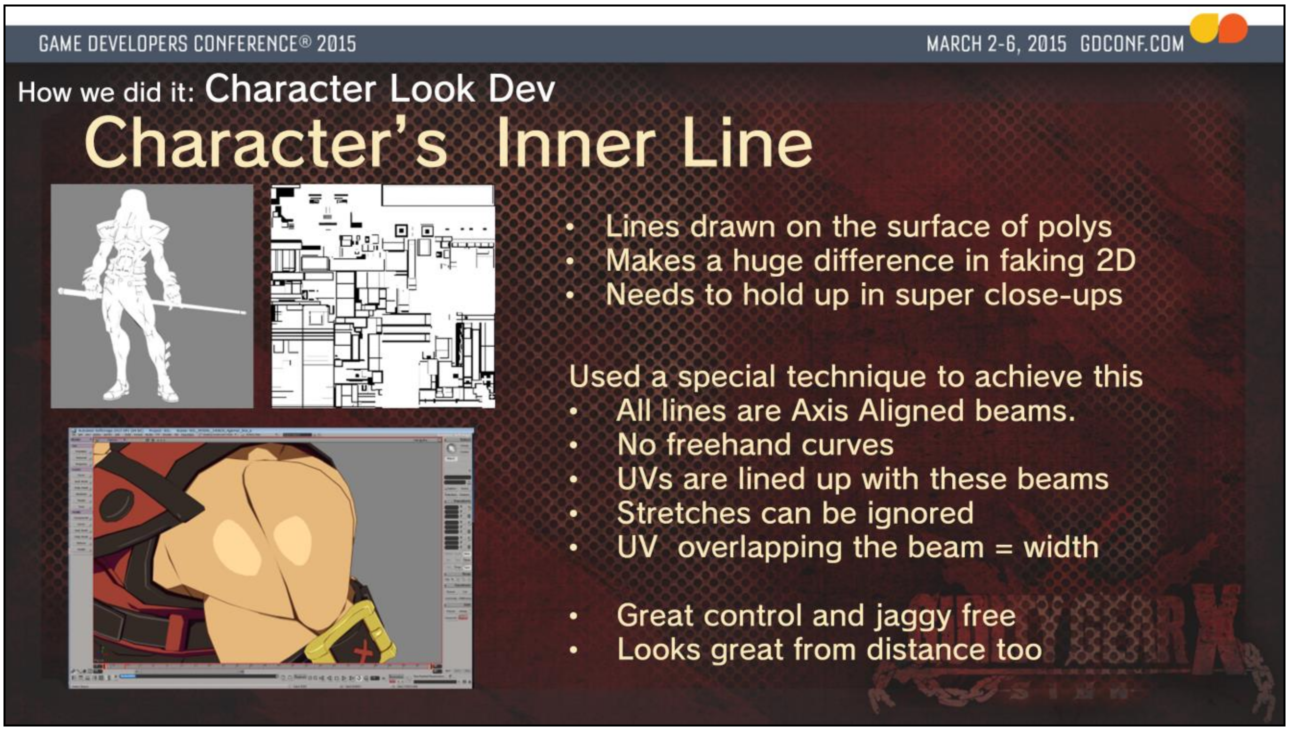
We added some new tricks to this long established method to achieve even more control over the results.

It's common these days to use a post effect for the outlines, but we found THIS method suited our needs better.

First of all, we could preview the result right in the modeling viewport, letting us know exactly how it's going to look in game, we had more control over the lines, through the vertex shader, giving precise control over the whole model.

With this , we were able give the outline variable widths in places we saw fit, even erasing it completely at times.

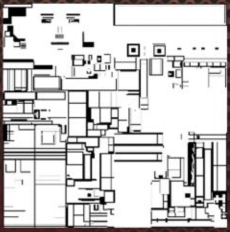
Again, Vertex color was used a lot to control these lines properties.

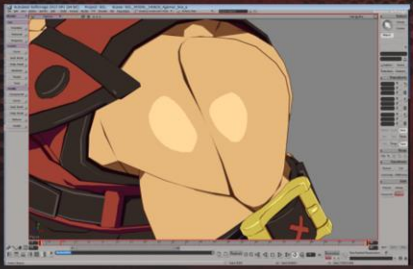


GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

How we did it: Character Look Dev

Character's Inner Line





- Lines drawn on the surface of polys
- Makes a huge difference in faking 2D
- Needs to hold up in super close-ups

Used a special technique to achieve this

- All lines are Axis Aligned beams.
- No freehand curves
- UVs are lined up with these beams
- Stretches can be ignored
- UV overlapping the beam = width

- Great control and jaggy free
- Looks great from distance too

We got our outline solved, but there is another set of lines that need to be discussed.

The inner lines, or the lines on the surface.

We needed a different approach for the inner lines, because the "Inverted hull" method used for the outlines do not work for these kinds of lines.

These lines play a huge role in 2D art, conducting where one surface ends and another starts.

In order to get a 2D look, we just needed to find a way to get this right.

The easiest way would've been to simply draw the lines on the texture, but we knew there was a major limitation to this.

The challenge here was , again , resolution.

We needed the characters to be able to hold up at extra close-ups with out any jaggies or pixelation to be found. To avoid pixelation, you would need an infinitely high resolution for the textures, which is simply impractical.

So we had to find a better way.

We came up with a technique using special UV assignments, to get jaggie free super clean lines even at close-ups.

Here is how the technique goes.

All the lines are drawn as Axis Aligned beams on the texture, and the UV is lined up alongside them.

How much the UV overlaps these beams, defines how thick that line becomes.

As long as the pixels are Aligned to an Axis they don't get any jaggie, so we figured we could use that to our advantage.

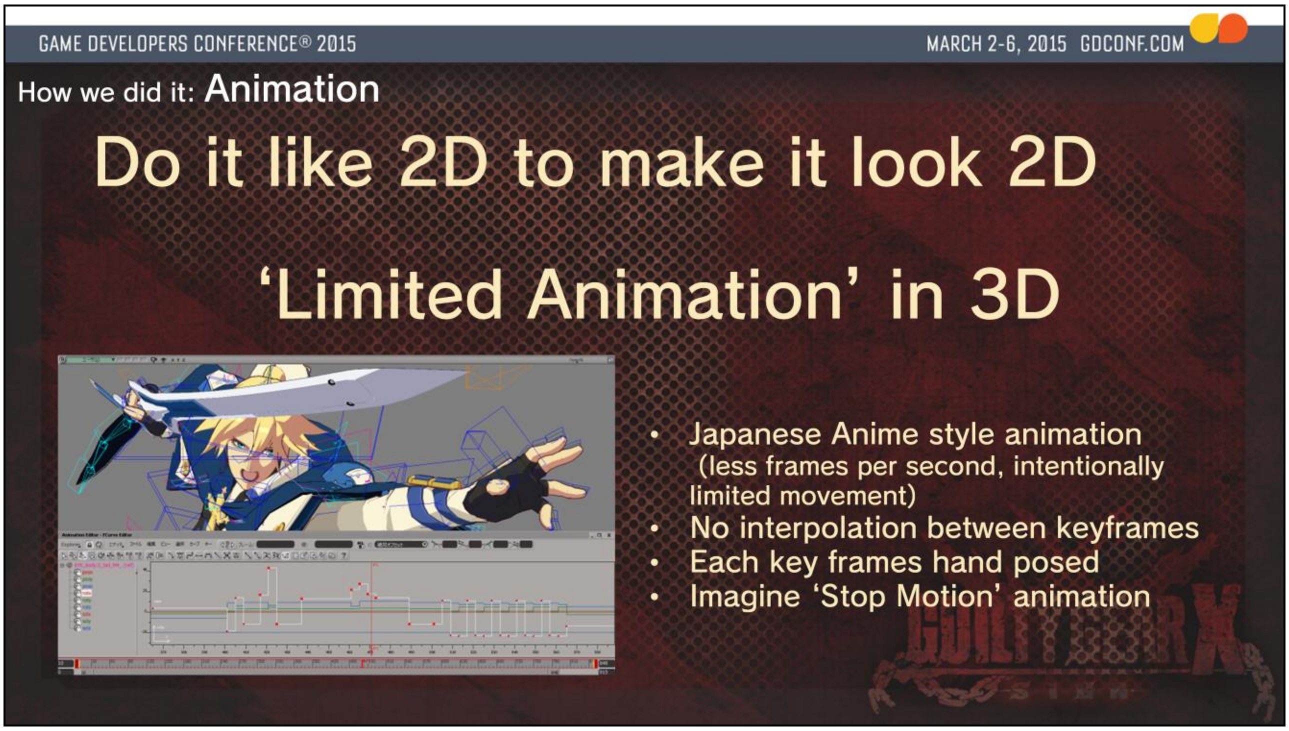
The down side is that you get a very distorted UV for the surface, but that was not a problem for us because we do not put any details on the texture.

(LIVE DEMO)

GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

HOW we did it :
Animation

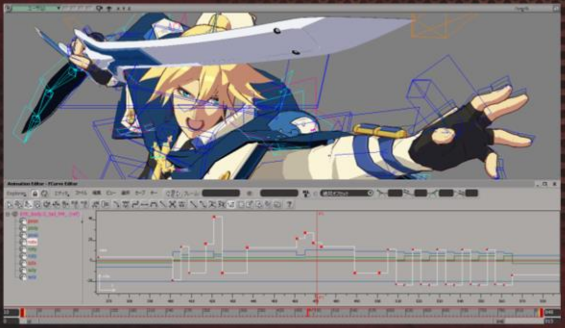
With that, we have a pretty good model with many features representing a 2D feel.
Now Lets move on to animation.



How we did it: Animation

Do it like 2D to make it look 2D

'Limited Animation' in 3D



- Japanese Anime style animation (less frames per second, intentionally limited movement)
- No interpolation between keyframes
- Each key frames hand posed
- Imagine 'Stop Motion' animation

Animation is where the merits of the models and the shaders truly come to play.

As with the models and the shaders, our goal was to make it look as 2D as possible.

To do this, we took a style called "Limited Animation"

This is a term often used in the Japanese Animation industry, in comparison with "Full Animation"

Unlike western animations such a Disney, it focuses more on tricking the eyes with less frames .

This was the way the old Guilty Gear titles were animated, and to replicate that, we decided to use the same principle, but this time in 3D.

We did try the "full animation" path in early development, but it simply didn't convey the sense that it was 2D.

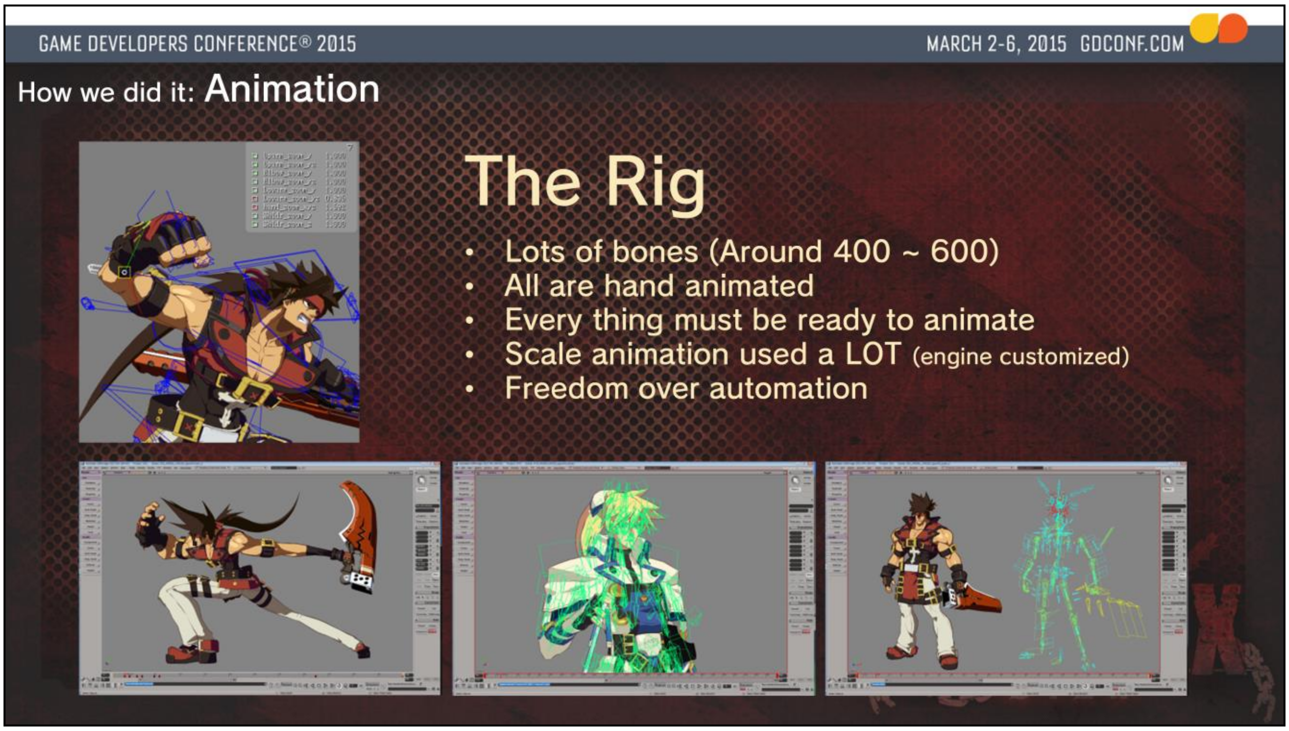
Having the poses interpolate between key frames, makes it look smoother, but at the same time, makes it look more 3D.

So we just stopped using interpolations between key frames.

Every frame now is a key frame, and the animator poses the character in the best way possible.

You could Imagine "Stop Motion" animations, where dolls are posed each frame. Basically, that is exactly what we are doing.

This looked more true to our old sprites, and also made the cut scenes look way more "2D-ish".



GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

How we did it: Animation

The Rig

- Lots of bones (Around 400 ~ 600)
- All are hand animated
- Every thing must be ready to animate
- Scale animation used a LOT (engine customized)
- Freedom over automation

In order to use limited animation as our style, we needed to build a rig to match this method.

Because we were going to have less frames per animation, we knew we had to add more information to each frame.

This was done by having a LOT of bones so the animator can move every feature of the model at a frame by frame basis.

The bone count is around 500 per character in average.

No simulation was used, because again, it just doesn't look 2D. There is just no way to capture the anime style movement with simulation.

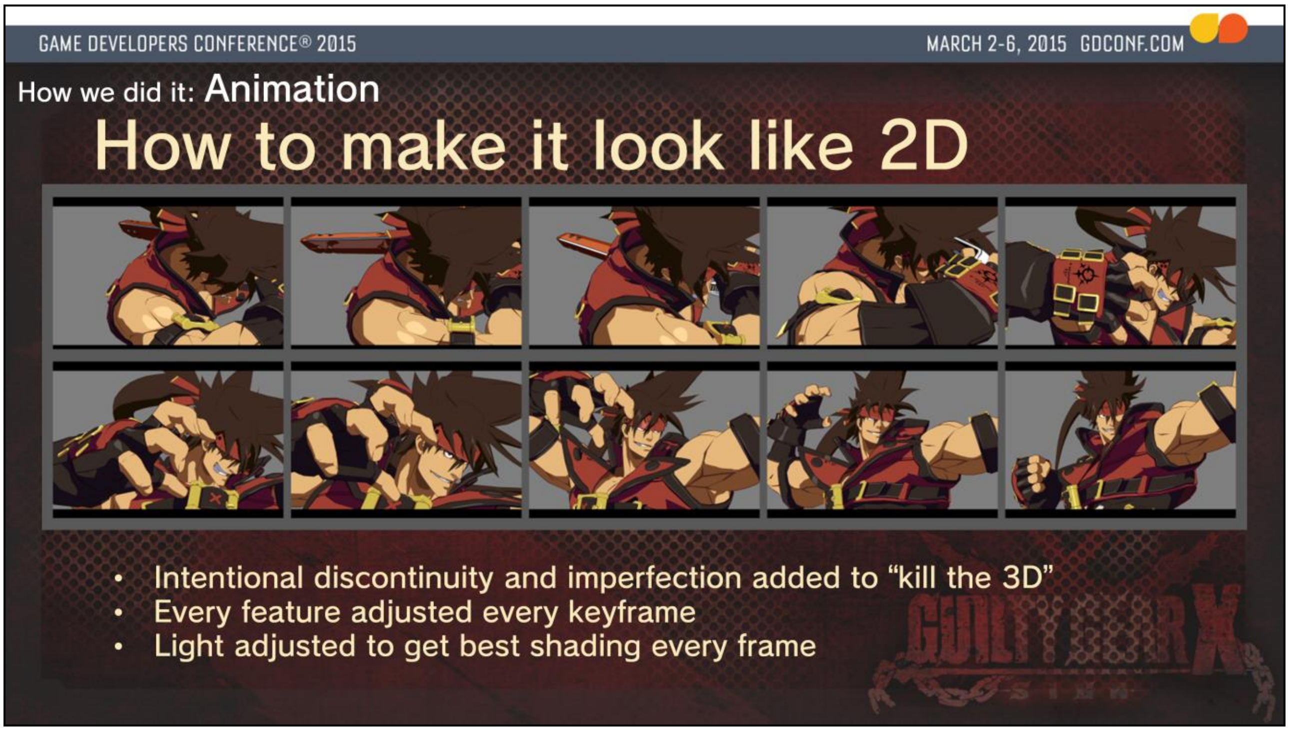
SCALE animation was used a LOT, because it let us do many sorts of tricks.

Exaggerations of actions, let things hide or appear, traditional squash and stretch, and more

We knew we needed it, but the engine didn't support it, so we had to implement the scaling system by ourselves.

It was difficult task, but thanks to our talented programmers, it was done.

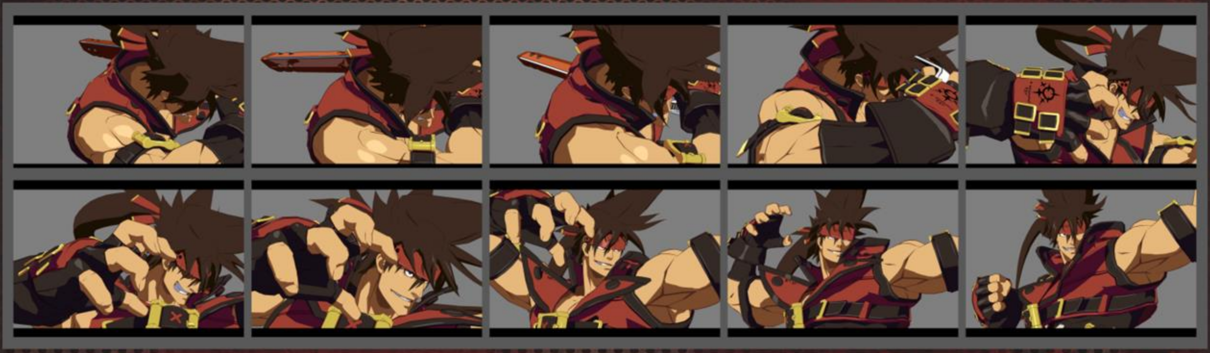
And it was totally worth it.



GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

How we did it: Animation

How to make it look like 2D



- Intentional discontinuity and imperfection added to “kill the 3D”
- Every feature adjusted every keyframe
- Light adjusted to get best shading every frame

Turning off interpolation between key frames is essential, but that alone, is not enough.
You need another trick to mimic 2D animation.

The secret, is to deform the mesh every key frame, to add Imperfection.

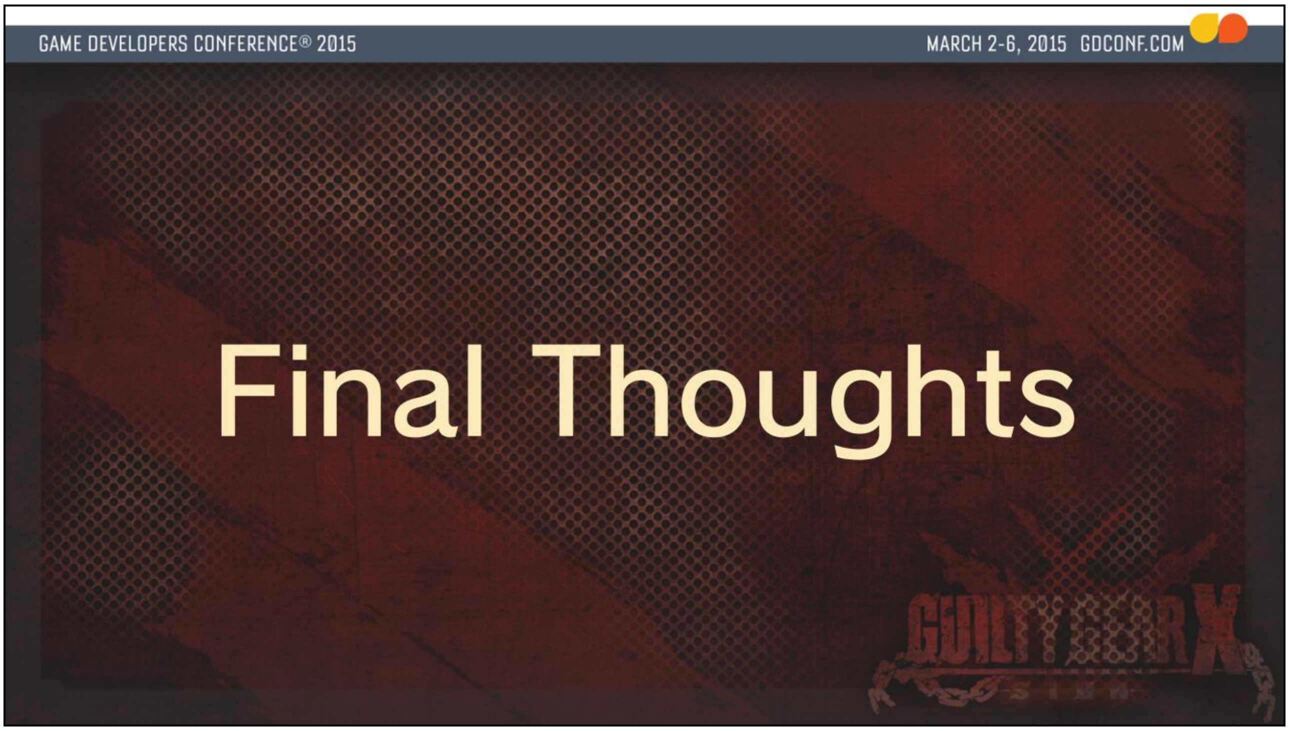
The human eyes and brains are very sensitive when it comes to perspectives.
If a part of a 3D model, moves through the perspective perfectly maintaining it’s shape, the human brain instantly recognizes it as a rigid 3D object.
To avoid this, adding imperfection was the only way.

Nature is imperfect, the artist is imperfect, therefor perfection looks ‘too artificial’.
(LIVE DEMO)

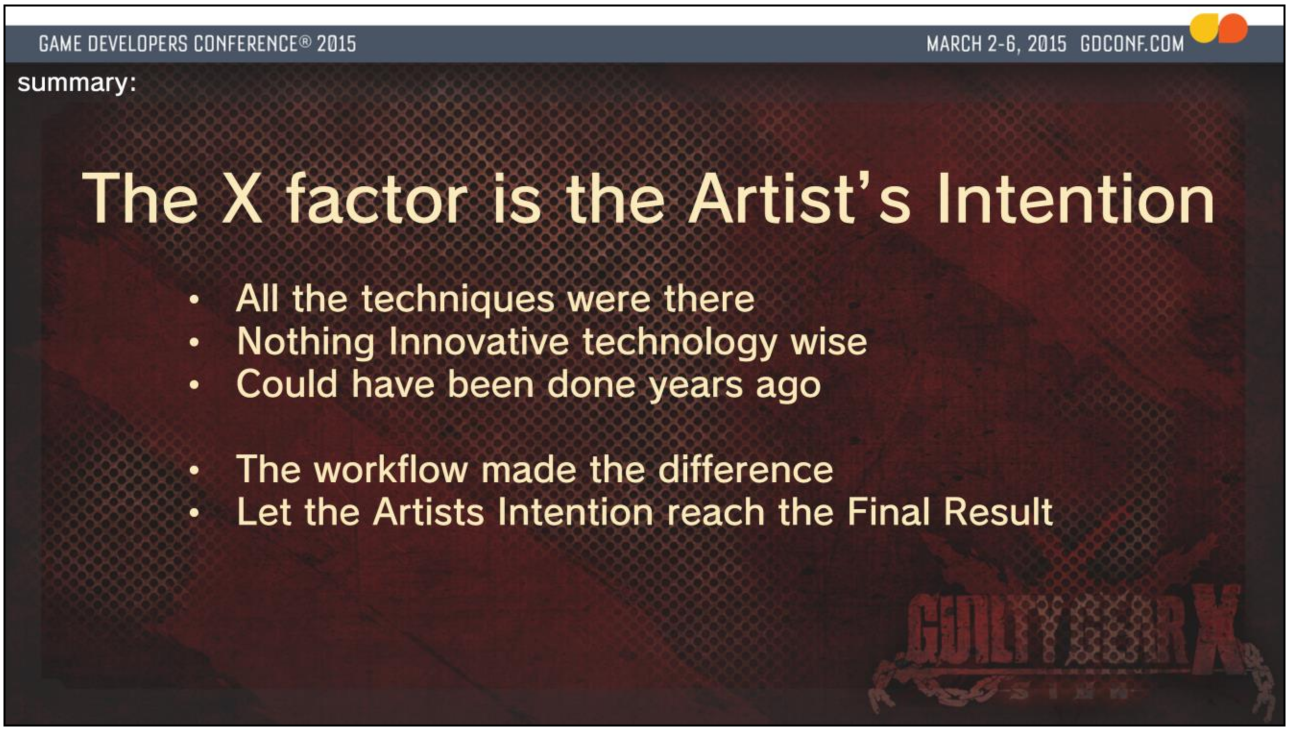
Every key frame, Every bone on screen is adjusted by the animator to break the perfect transition.
This makes each key frame more distinctive, adding even more information to a single frame.

“You need to think in 2D” is a word used a lot while animating in our team.
3D accuracy is not a priority in this workflow,
We often sacrifice it in order to get a more dynamic composition.
Limbs, hands, and feet get a lot of scale animation to exaggerate the perspective, and facial parts do not maintain natural positions.

This is exactly the same as 2D animation.
Expressiveness over accuracy.
And what we did was simply bring the same principle to 3D



Final Thoughts


GAME DEVELOPERS CONFERENCE® 2015

MARCH 2-6, 2015 GDCONF.COM

summary:

The X factor is the Artist’s Intention

- All the techniques were there
- Nothing Innovative technology wise
- Could have been done years ago

- The workflow made the difference
- Let the Artists Intention reach the Final Result

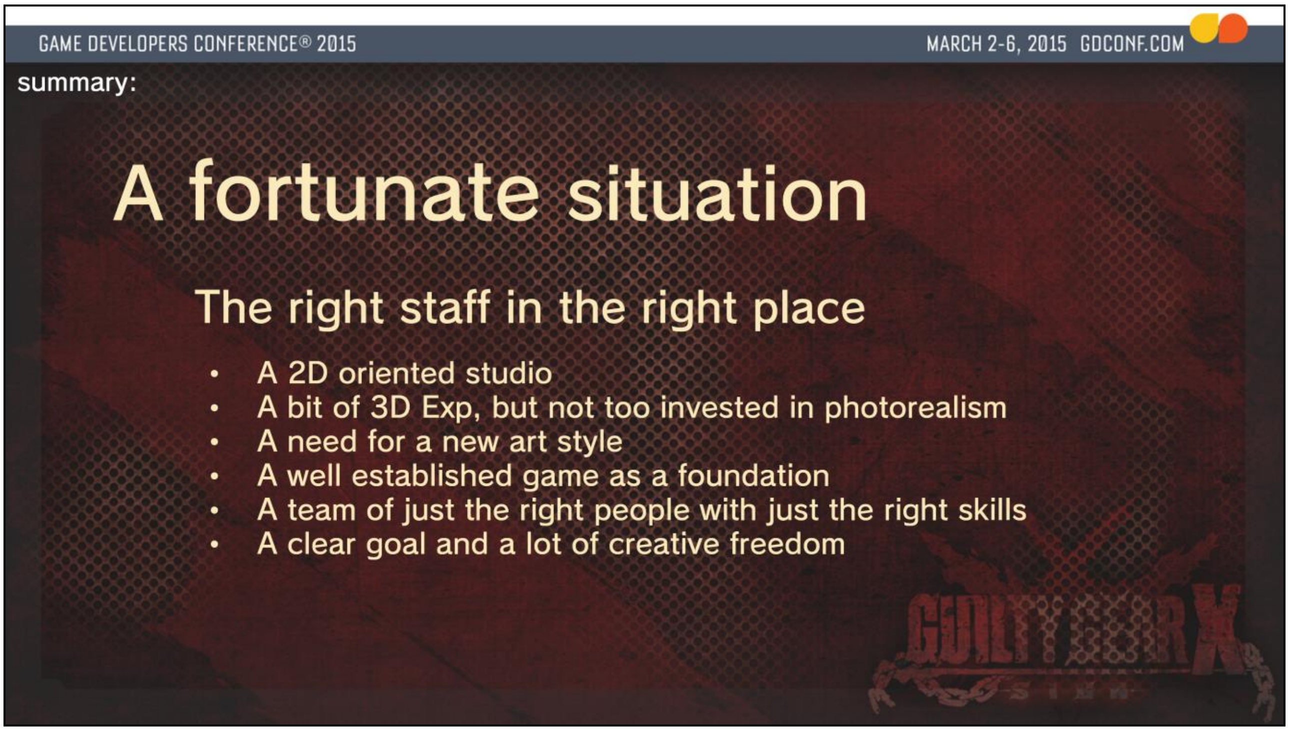
GUILTY GEAR Xrd SIGN

What made the difference?
In my opinion, it was figuring out this one thing.

The X factor is the Artist’s Intention.

All the techniques we used are actually nothing new.
No new hardware or software innovation was involved.
Technically, the art style could have been done Years ago.

What made the difference, was the notion that the Artist’s Intention was the X factor.
A workflow, which let the Artist’s intention carry through, Right to the Final Result, was the core of our accomplishment.


GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

summary:

A fortunate situation

The right staff in the right place

- A 2D oriented studio
- A bit of 3D Exp, but not too invested in photorealism
- A need for a new art style
- A well established game as a foundation
- A team of just the right people with just the right skills
- A clear goal and a lot of creative freedom

GUILTY GEAR X SIGN

But Why was it us, and not any one else?

One simple answer, would be that we were lucky.

Thinking about it now, we were in a very fortunate situation to get this done.

We were a 2D oriented studio that had a lot of 2D experience, many artists who knew 2D inside out.

We were not new to 3D, but not heavily invested in photo-realism either.

We were in a need for a new art style, to differentiate from the best of our own, BlazBlue.

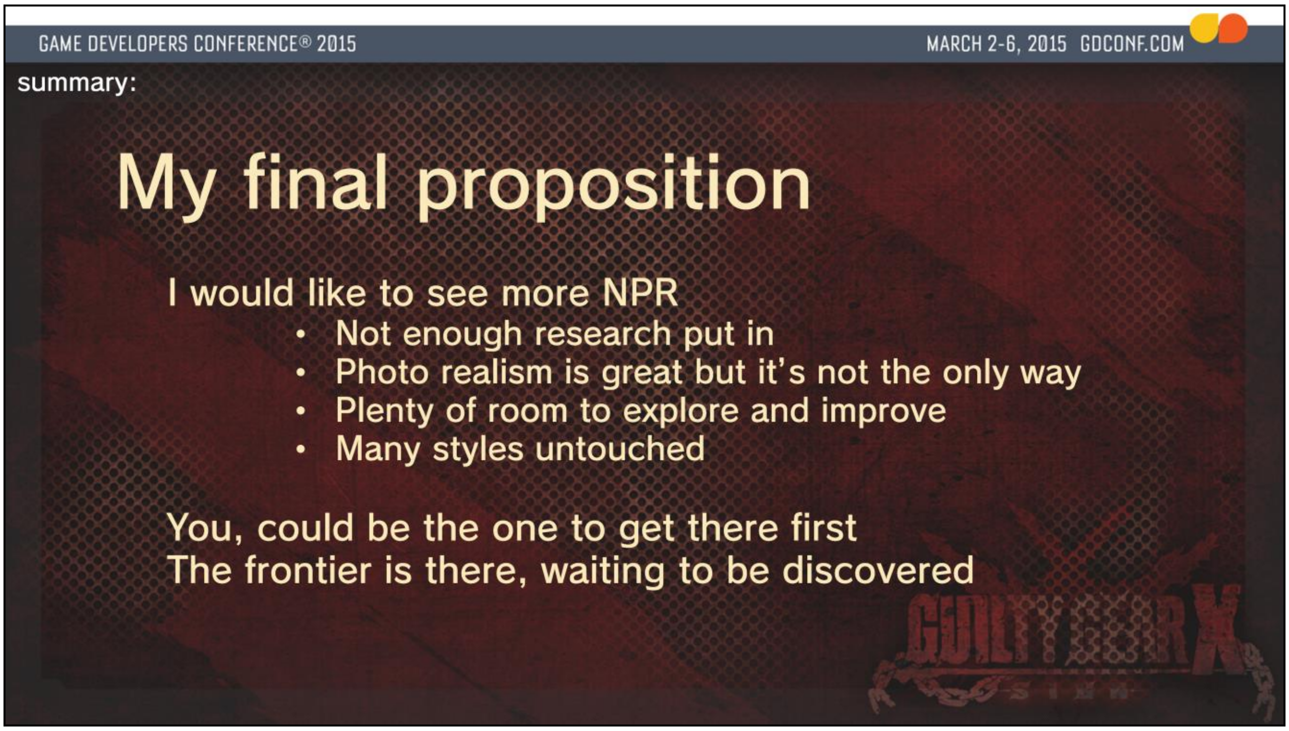
The project was a reboot of a very well established game, so we knew exactly what we were working on.

And the team. It was the dream team.

We had exactly the right people with the exact skillset needed for this challenge.

Off course, fortune wasn't all.

The dedication and the hard work of the people involved in the project, and the strong support from our fan base was what made all this possible.



GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

summary:

My final proposition

I would like to see more NPR
- Not enough research put in
- Photo realism is great but it's not the only way
- Plenty of room to explore and improve
- Many styles untouched

You, could be the one to get there first
The frontier is there, waiting to be discovered

GUILTY GEAR Xrd SIGN

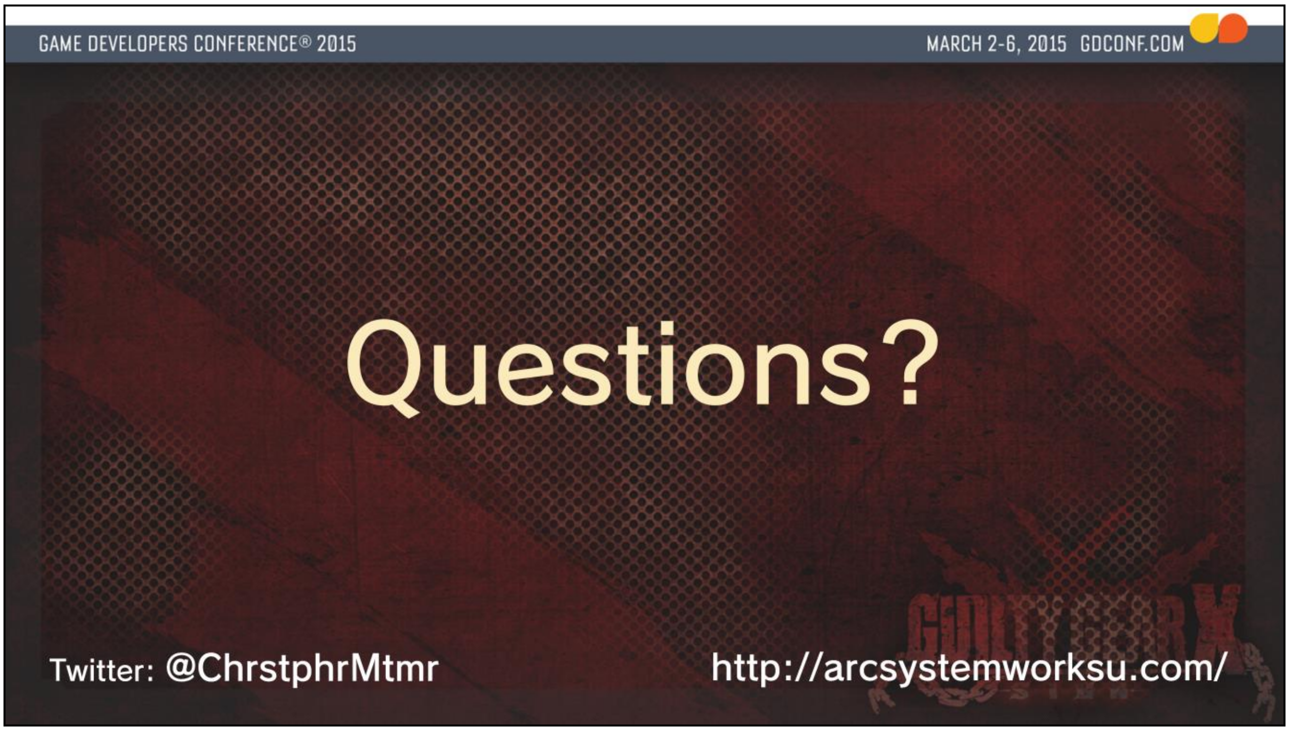
Now to wrap up, here is my final proposition.

I want to see more studios explore Non Photo-realistic Rendering.
I feel that not enough research has been put into this area, and there is much more room to explore and improve.

Photo realism is great, but it is not the only route, and it's a crowded one too.
If you Look in different directions, there is a whole new horizon.

You, could be the pioneer that gets there first.
The frontier is there waiting to be discovered.

Thank you.



GAME DEVELOPERS CONFERENCE® 2015 MARCH 2-6, 2015 GDCONF.COM

Questions?

Twitter: @ChrstphrMtmr

http://arcsystemworksu.com/