STELLAR JOCKEYS SEPTEMBER NEWSLETTER

Welcome to the Stellar Jockeys newsletter for September. Due to this edition's length, you may need to click "View entire message" at the foot to see everything.

Benjamin, Stellar Jockeys

Explaining Brigador's Art Pipeline

Last month we talked about the game engine's origins; this month we're going to expand on that to talk about two main things:

1. What it is you're actually seeing on your screen when you play Brigador and why
2. What we're changing about that for Brigador Killers which we will show off here with a couple of screenshots

[To best follow this entire section, please refer to the glossary of computer graphics on Wikipedia in particular for words highlighted in bold.]

Oftentimes we get feedback about our first game from players that wish the camera could be moved to see different angles of the various models of the vehicles or buildings seen in the environment. We don't wish to disappoint those players but within the game engine this isn't possible because what you are seeing rendered in the game is *not* a 3D model that can be rotated along any axis. Instead what you are seeing are sprites (more specifically, you are looking at 2D quadrilateral shapes, and what you are *also* looking at is a flat, two-dimensional screen upon which is a moving image that is creating the illusion of depth through particular techniques).

Ironically, how we even make these sprites is initially by creating 3D models, typically through the process of kitbashing (this part of the process we won't go into, and we'll ignore animation rigging too, but feel free to check out this timelapse video for the Pantry Boy vehicle from several years ago that you may not have seen before) usually in 3DS Max. The part we're concerned with comes *after* a 3D model has been

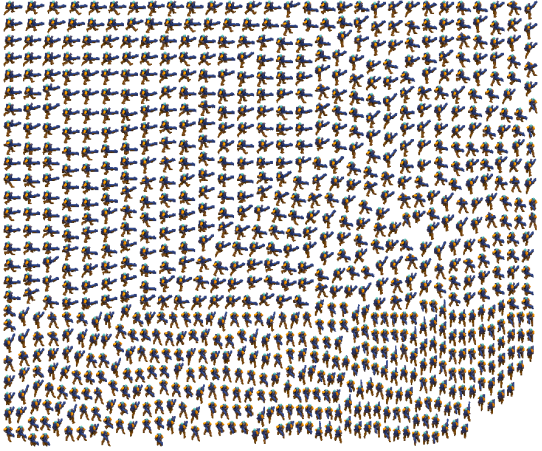
decimated, which is a process that reduces the size of the polygon mesh. Once this happens we can give the mesh a texture map, which is the point where the model starts to more closely resemble the final product.

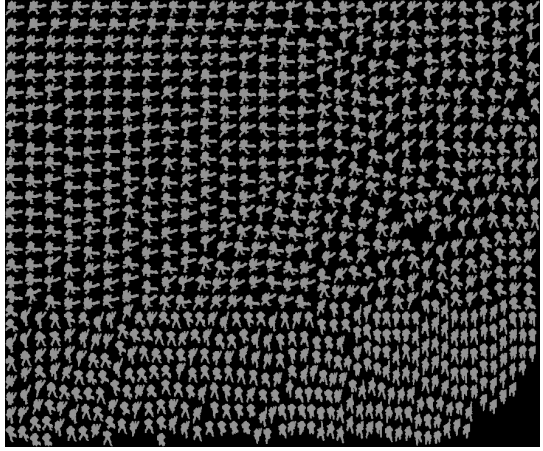
Before we do that we'll need UV maps first. A quick way to explain UV mapping is to imagine an animal that has been skinned: our 3D model is the animal and the skin that has been removed from it and can be laid flat is our UV map. The program we'll use to do this to our 3D model is called Houdini.

To give our UV maps texture information what we then do is take them from Houdini along with the 3D model that was made in 3DS Max into another powerful program called Substance Painter that allows us to detail the materials of the model (for example, how rough a stone looks or how glossy a metallic surface is). We don't have footage of us working on Substance Painter but you can get a good idea of what it's capable of just by looking at this short official video that touches on a lot of what we've just written.

At this point we have crafted the shape of our model, peeled off its skin, given it materials to make it resemble the final object, and now we have to take it into a *fourth* program: Blender. Why we take the textured model into Blender is to do three things. The first is lighting, which we only do a little of. Blender allows us to influence the light-matter interaction, or how 3D models are illuminated - a process that is often referred to as baking. The second thing we do in addition to this is framing the 3D model from an angle of our choosing - in other words we recreate the same view frustrum that the player will see when playing Brigador. The third and final major thing in the Blender step is we also get depth buffer information, which tells us how far away the model is with respect to the camera's perspective.

However, we're nowhere near done. We said at the top that in the game you were technically looking at sprites, not 3D models. The basic reason for that is because the game engine is told to display sprites, which in turn spoofs the appearance of 3D models in an apparent isometric perspective. How we get from 3D models to sprites is via open source software - a version of which comes included with the Brigador Modkit & Map Editor called SJSpritePacker. What this does is takes the original 3D model and (depending on the model's level of detail) captures up to 64 rotations of all that model's positions and animations at a particular resolution and creates not one but *two* sheets of sprites along with the XML data for the sheets. Below you will see the sprite sheets for the loyalist infantry NPC (AKA loy_foot_01), which if you own a copy of the game you can find in the folder Brigador\assets\units\loyalists\foot.





This is where things start to get complicated now that the models are getting to the point where they will appear in Brigador's engine as sprites. In addition, the purpose of the second sprite sheet is it provides z depth information (more on that later). The XML data that's outputted for *both* of these sprite sheets by SJSpritePacker is "pointer data" - this is information that tells the sprites to face the correct direction but before it can do so, it's fed into a .json file, which is the file type the game engine reads for the majority of the game's data.

...Wait hold on - you might be asking - *why* did we go to all the trouble of putting together 3D models, give them detailed materials for their textures, bake in some lighting and *not just use those models in the game* instead? This question was asked many years ago and was finally answered around the winter of 2012. While Brigador does have a 3D *look*, it is *not* 3D. Games that are *true* 3D are extremely complicated because once you go 3D, now you really are in a situation where the player can view a model from every conceivable angle in an environment. This is a considerable undertaking for any studio's engineers and technical artists to deal with, who essentially have to figure out a way for models that are exported in a particular format to be understood by their game engine *and* also be optimal (i.e. not grind to a crawl and run at single digit FPS). In other words, everything we've talked about so far took five different programs alone: 3DS Max > Houdini > Substance Painter > Blender > SJSpritePacker - and that was only for 2D sprites. So, knowing this, and with respect to the amount of time, people, money and energy it would take to make a *true* 3D Brigador game, hopefully it's clearer now what some of the reasoning was for Brigador's look.

So Brigador's not true 3D and yet visually the game still looks impressive - so what else is going on? Do you recall that second sprite sheet from earlier, the one that looks black and white? Let's look at another one that's only ever seen at one rotation - the orbital cannon (AKA battery_01 in the game data).

The variant of the orbital cannon on the left you can think of as the diffuse version, while the variant on the right is the Z depth information visualized as a grayscale image (where the brightness can be considered as an indication of how close the model is to the camera). The purpose of the latter is to inform the lighting of the Brigador engine against the sprite, which you can see in the final version of the game itself.

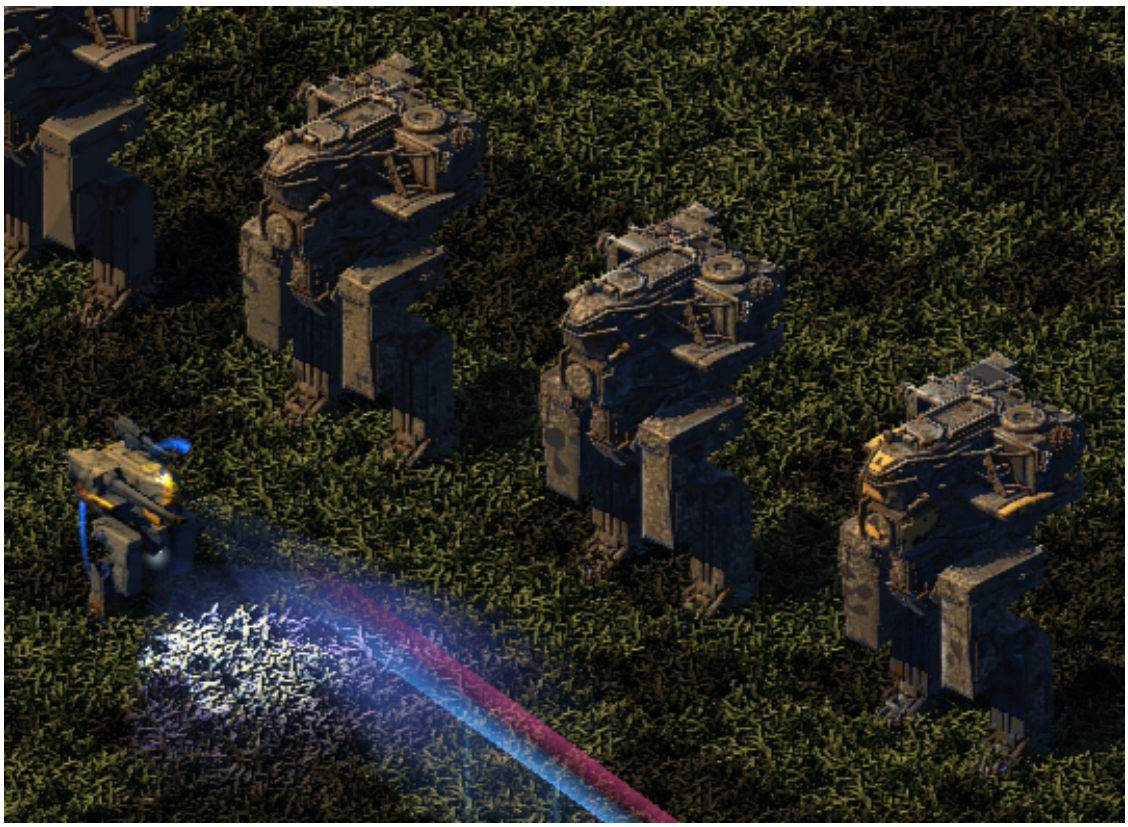
Or to see it in motion, look at this gif from early 2016. What should be apparent from the above image and the gif is there is additional lighting being displayed on the sprite itself. As it turns out, the Brigador engine is doing two specific things: deferred shading and with it dynamic lighting and is what we've been building up to this entire newsletter.

We encourage you to read up a little on both to save space on this email, though we will point out here how even for the mid-2010s, applying deferred shading and dynamic lighting to 3D models was a very expensive thing to do in terms of hardware... except we are not using 3D models in the engine! Remember: the sprites are fundamentally just two-dimensional quadrilateral shapes - not polygons where every facet would have to be lit properly and because we are able to decouple a scene's geometry from its lighting, and because we have the depth information from the Z-depth sprite sheet in the XML data, this effectively allows us to put in a bunch of lights into the levels without causing major performance hits *and* allows the engine to apply dynamic lighting to those sprites. Or,

via a quick visual representantion, watch how the lights of this Touro from an early BK demo illuminates the surrounding building and pavement as it walks past.

What you've read so far is an abridged version of what goes into what you actually see on your screen when you play Brigador and why it looks the way it does. What we are *changing* about the visuals for our next game is we are doubling the output resolution of all sprites from their 3D models. In the next two images you will see the game's masthead Touro Loyalist mech. The image on the left is from Brigador with the camera set to a 3x zoom, while the image on the right is the Touro at this new 2x resolution at 1x zoom.

...And together in the same scene within a dev build of BK.



There's a lot to say about the "new" Touro (NB: both images are still based

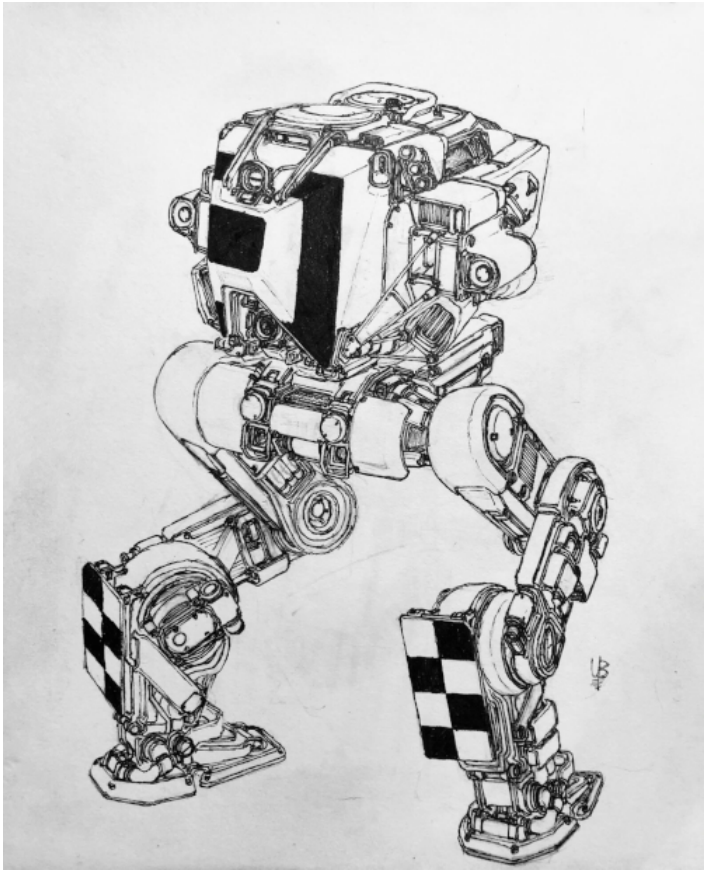
on the same original Touro 3D model, it's just that one of them is rendered at a greater resolution than the other) but for now the most obvious thing to point out is that at face value the fidelity of the image is vastly more detailed and the filigree details and greebles of the mech now stand out much clearer than before.

We realize upon writing this out that this may not end requests from people asking for different camera angles, but at the very least with this development to our content pipeline we hope players will be able to better enjoy a new level of detail that goes into each art asset in our sequel.

Community Spotlight

A short few entries this month. Mexquier made this out of brass while they were machining...

...Unwelcome Brigador graced us with an excellent Doric...



...and MrUnimport has written a lore entry in the style of Cephei Chatfield concerning the illusive "Man From Twitter".

The Man from Twitter

As intimate as I have become with this misbegotten server named Stellar Jockeys, it is sometimes easy to forget about the wider internet. Man from Twitter is a SNC recruit from Twitter, another social media platform and the site of one of the early-model 'micro-blogging' experiments. A harsh and ruthless place, it is not unlike the forums and imageboards from which we spawned, save for its crudity and damnably character-constrained nature.

He once had a name, but the esoteric rituals in the warrior death clique to which he belongs stripped him of any identity beyond violence. His entire being is the posts he makes; to his people, a followerless man is nobody, without a face, and the replies of the fallen enemy are collected as trophies. High-powered weaponry and desperate struggle over daily items of discourse are the only things the population of Twitter know.

Despite the brevity of his guttural speech, I cannot help but feel a certain affinity for this Man from Twitter. There is an immediacy to his irony that I find refreshing. Born of necessity, it remains unadorned by the garish flourishes my compatriots are so fond of. It's a pity they won't appreciate his artistry tonight.

-m.p.C.C

Want more? Go to the #becks_best channel on our Discord server.

Join Our Discord Server

Next Month

In October we hope to announce the winners of the Grave to the Rave mapping competition, which will be made part of Brigador's freelance mode in a Halloween update.

Stellar Jockeys
112 N Neil St Apt D, Champaign, IL
United States

Newsletter Archive Privacy Policy

You received this email because you signed up on our website or made a purchase from us.

Unsubscribe

mailerlite