ARCHITECTURAL INTELLIGENCE:
HARNESSING AI FOR DESIGN INTENT

LUKE KRATSIOS

HOW CAN THE REAL-TIME MASKING AND LAYERING OF AI-GENERATED IMAGERY BE USED TO EFFECTIVELY REALIZE AND REFINE DESIGN INTENT?

ADVISORS
CHRISTOPHER BATTAGLIA
PATRICK KASTNER
DONALD GREENBERG

TECHNICAL CONSULTANTS
JOHN WOLFORD
LEUL TESFAYE

CLASS
B.ARCH.

How can the real-time masking and layering of AI-generated imagery be used to effectively realize and refine design intent? This thesis investigates and develops state-of-the-art technologies in the field of generative content and examines how designers can leverage inherent training biases and input data to swiftly generate textures and imagery for use within the design process.

Ultimately, the developed workflows aid in formulating a cohesive project vision from early stages. By combining proceduralization, simplified modeling techniques, and intelligent masking, the design process accelerates. This not only enables the creation of high-quality renders but more importantly, encourages a nuanced approach to design thinking.

ACKNOWLEDGMENTS

I would first and foremost give a massive thank you to my advisors Chris Battaglia and Patrick Kastner. You both provided unique perspectives that were critical in the development of my project. Working with you was a complete pleasure and you both went above and beyond to ensure that I not only had a successful thesis, but grew, learned, and enjoyed doing so.

Don, thank you for all of your support and advice over the last three years. I cant overstate the impact you have had on me, my life, and my career until now and in the future.

I would like to also thank Chris Morse. Chris advised me through my pre-thesis research, which ultimately led me to the work I proudly present today.

To my two close friends and technical consultants, John Wolford and Leul Tesfaye, thank you for all your help, and in pushing me to fall in love with and become a better programmer.

Lastly thank you to my family and friends for making these five years so memorable and enjoyable.

CONTENTS

THE PROBLEM

REPRESENTATION BY MIR.

REPRESENTATION BY MIR.

According to industry standards in design representation, that number is around 5,000 dollars. Beyond the prohibitively high cost for most firms, this massive expense speaks volumes to how these renders are being utilized in our industry. These are not made for or by designers, they are purely marketing tools used to sell a product.

The quality provided by affordable and fast arch-vis alternatives falls short of yielding meaningful design feedback or freedom. They result in distracting and uninspiring depictions.

Buildings are full of intricate detailing and nuanced materials. So rarely are surfaces as uniform as we represent them.

With the current tools available, these subtleties take an incredible amount of time and effort to capture and represent properly.

And so, often they are just omitted from representations, which is also indicative of decisions being omitted from the design process as well.

IMPERIAL KILN MUSEUM | SUDIO ZHU-PEI

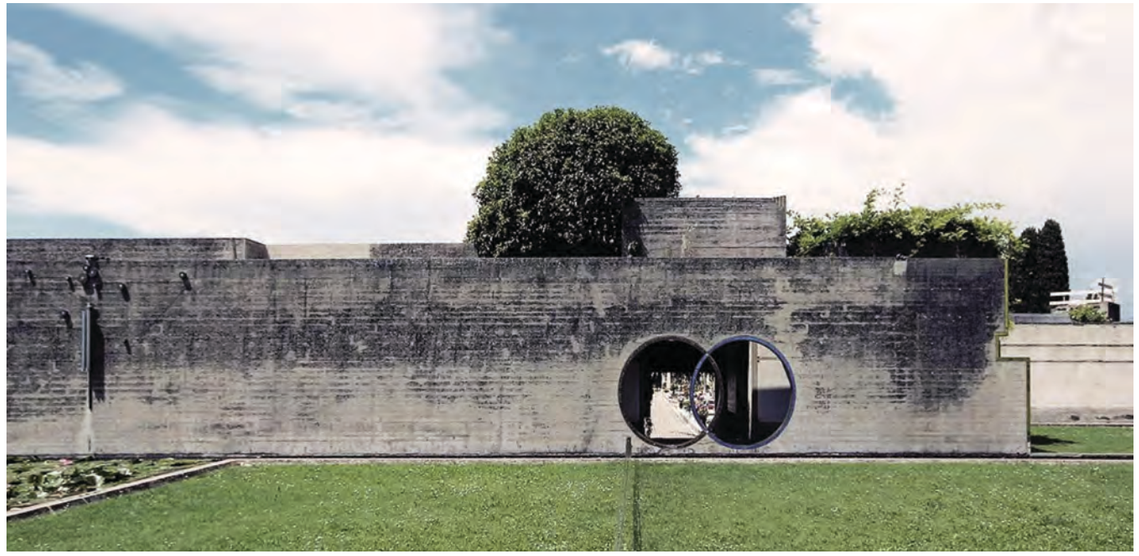
ɴK'MIP DESERT CULTURAL CENTRE | HOTSON BAKKER BONIFACE HADEN

NINGBO HISTORIC MUSEUM | WANG SHU

Moreover, buildings are not static. They age, they weather, collect dirt, soot, moss, They evolve over time.

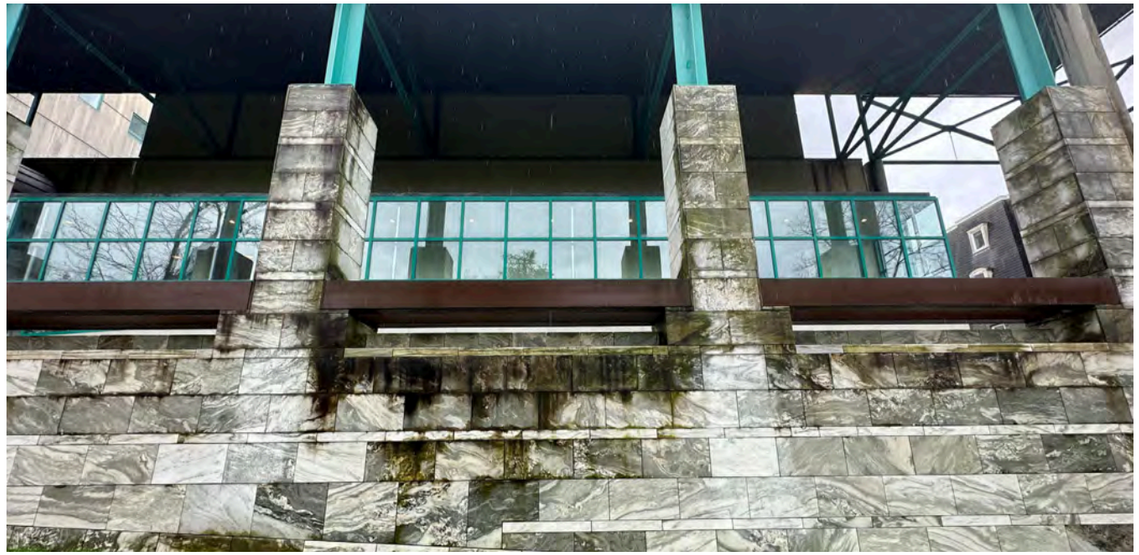
Buildings must situate themselves within their context, in the way they touch the ground and interact with their environment.

A building's materials will react differently based on weather conditions, type of use, and program.



TOMBA BRION | CARLO SCARPA

CHWARTZ CENTER FOR THE PERFORMING ARTS | JAMES STIRLING & MICHAEL WILFORD



JRIS HALL | GORDON BUNSCHAFT

And yet, so often are representations ultra pristine and embellished. Again, this is what the client expects, but also what we're basing our design thinking on.

In not capturing this intricacy we are really doing a disservice to ourselves. By not allowing us, as designers, to utilize this material information as an integral part of our decision making process.

MUSEUM BOIJMANS VAN BEUNINGEN | AD VAN DER STEUR

THE VESSEL| THOMAS HEATHERWICK

HIGH SOCIETY | GEOCON

THE PROBLEM

THE TOOLS AVAILABLE TO DESIGNERS ARE

ARTIFICIALLY LIMITING THE CREATIVE PROCESS

THE SOLUTION

EXPAND DESIGNER AGENCY BY DEVELOPING TOOLS

THAT EXPLOIT NEW GENERATIVE AI AND INTEGRATE

MATERIAL DECISIONS INTO THE DESIGN PROCESS

As a baseline for this to be effective we need to achieve competitive:

QUALITY

COST

SPEED

On top of this, I believe the tool must also provide:

CONTROL

FAMILIARITY

CUSTOMIZATION

Quality, cost and speed are quite straight forward. The other three may require more explanation.

Control refers to the ability to produce deterministic results with adequate user input. That input must also be feasible and easily manipulated.

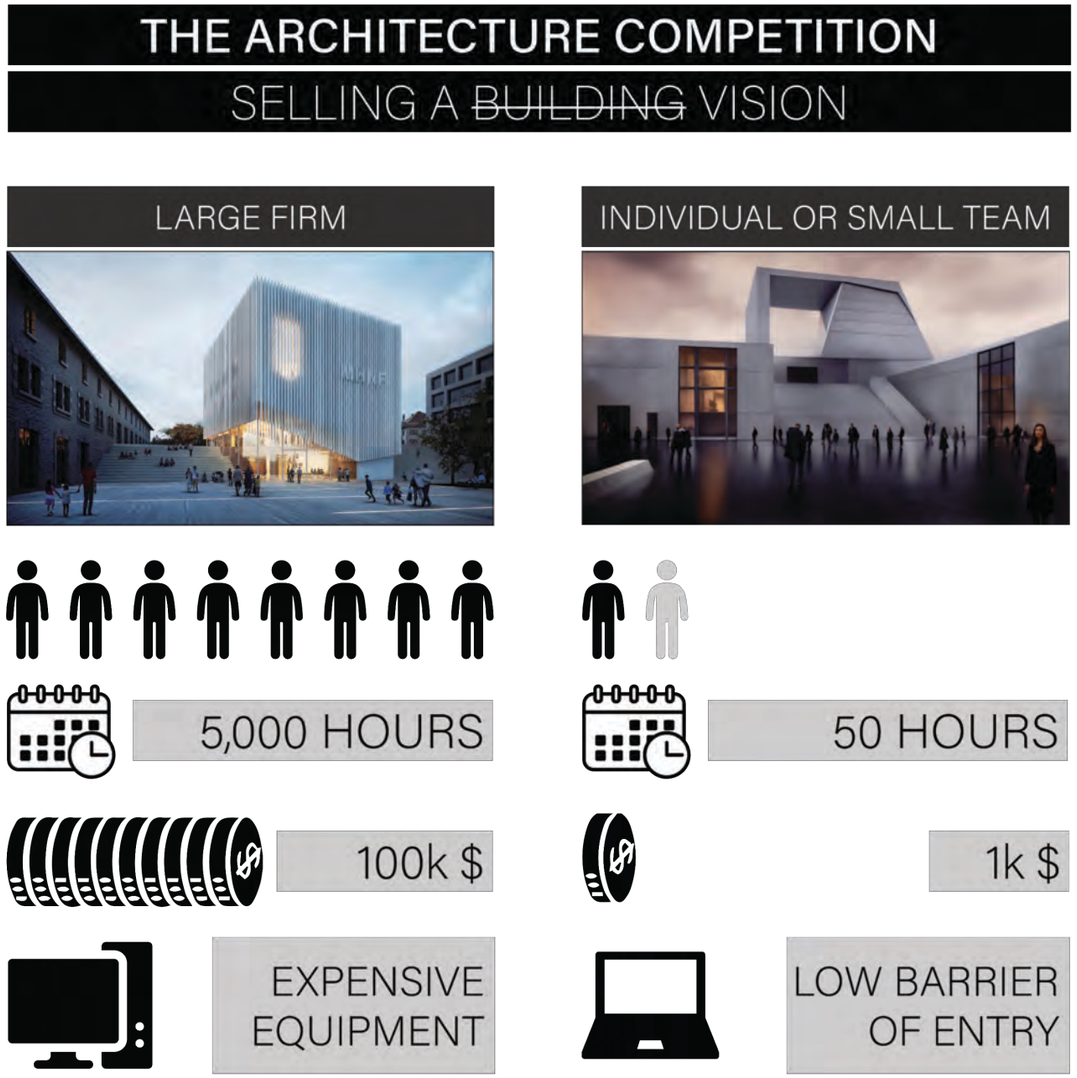
Which leads into familiarity. The tools, workflows and processes should be ones well established in the design world. Users should be able to use processes they are already comfortable with. Otherwise the software is not accessible.

This again leads to the next and final point: customization. There is not a single workflow that works for all users, and the tool should be flexible enough to allow designers with different skill sets and preferences to take advantage of it.

Let's use the architecture competition as an example of why these points are so crucial:

Large firms will spend thousands of hours, and hundreds of thousands of dollars on a large design competition. This makes competing in such a space nearly impossible for smaller design teams.

However, with what I propose today, I hope to prove that a strategic utilization of the rapid democratization of our tools will allow even individuals the opportunity to not only compete but excel.



HOW GENERATIVE AI WORKS

HOW DOES THIS TECHNOLOGY WORK?

Training begins with billions of images. Consider this sample (a) as representative of all our data. However, this grid does not contain all possible images that exist.

In reality, those billion images make up just a tiny fraction. The vast majority are actually random noise. The training data being black dots on the image (b).

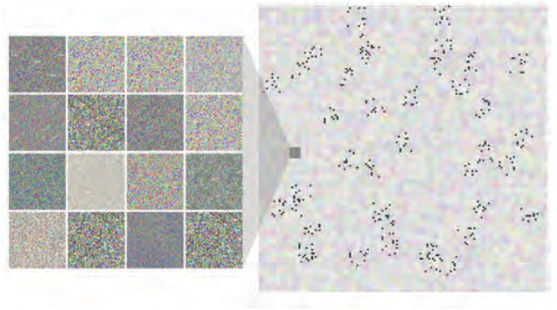
By categorizing our data within this massive image space we can then define clusters based on keywords. Take for example “dancer” or “Van Gogh” (c). These are two disparate ideas, but by drawing connections between these nodes, we can conceptually define a greater space of all normal looking images.

If we then choose a spot on this grid that is not defined, and as such, random noise, and then provide a prompt that connects these terms (d), we can progressively move from that spot towards this connection between nodes, we can arrive at a new image that is not part of our original data set (e).

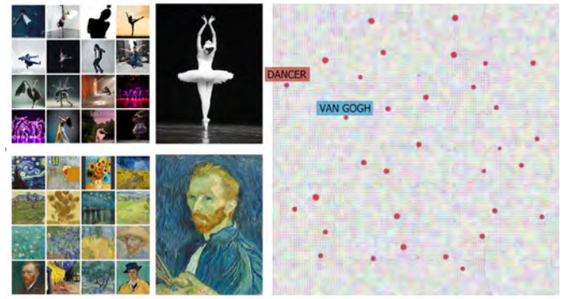
Even though this is an abstraction of the process, one thing that it demonstrates particularly well is the ambiguity that would come with finding a point along a curve in this way. We can get more control by being more specific with our prompts, however we are still essentially throwing blind darts.

LINK TO FULL ANIMATED VIDEO:

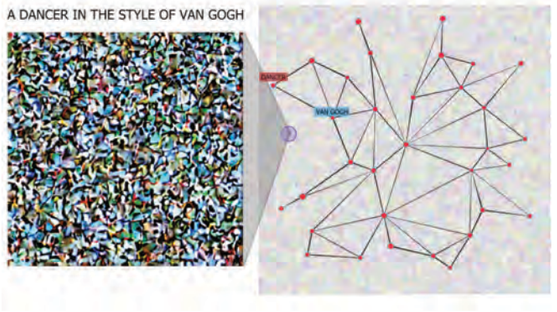
(a)



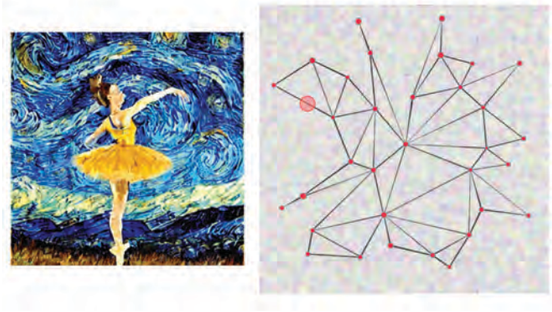
(b)



(c)

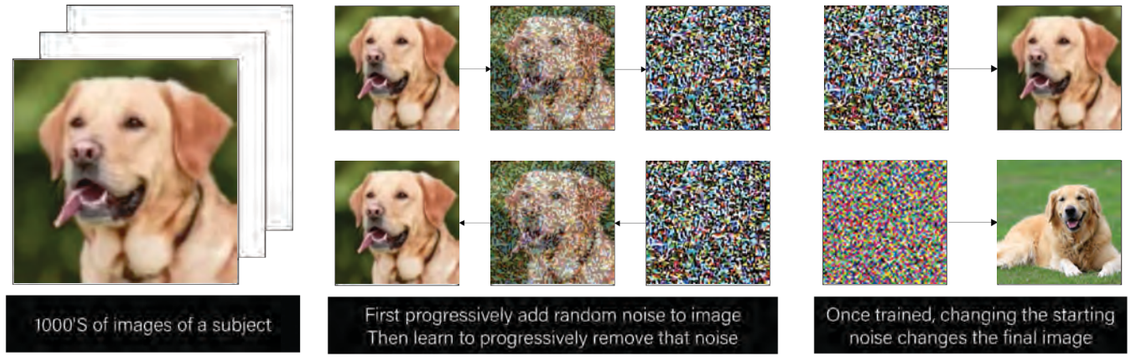


(d)

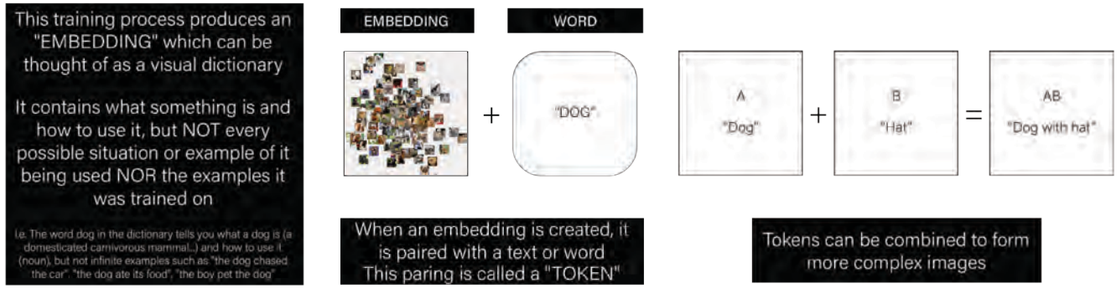


(e)

A MORE TECHNICAL BREAKDOWN OF STABLE DIFFUSION TRAINING:



(a) collection (b) training (c) generation



(d) embedding (e) pairing (f) combining

(a) Collection: The collection of 1000'sof images of a single subject. Ideally, this is a comprehensive, general set representative of that subject in its entirety.

(b) Training: During training, random noise is progressively added to each of the images. The algorithm then attempts to remove noise step-by-step to gradually arrive at its initial input.

(c) Generation: Once trained, using different deterministic starting noise as a base, i.e. a random seed, the final image changes.

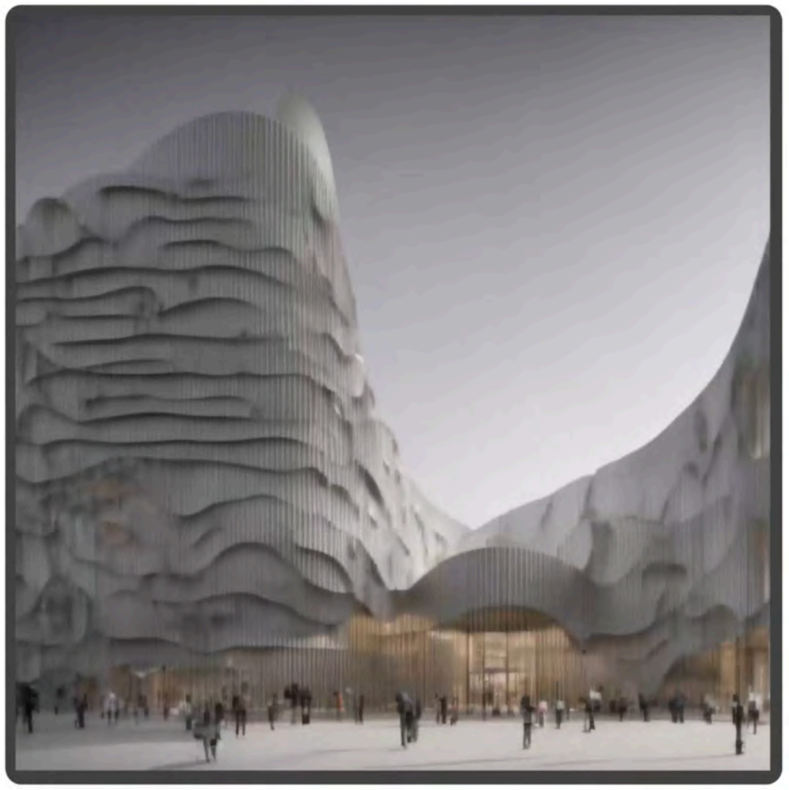
(d) Embedding: A trained subject is refereed to as an embedding. Containing knowledge of what that subject is.

(e) Pairing: From an embedding and a word token that matches the embedding, a connection can be made between human speech and image.

(f) Combining: Token-embedding parings can be combined to form more complex instructions.

(g) Natural Language: With this we can achieve natural language to image generation.

(g) natural language



The images on these pages were created by simply providing the given prompt and receiving a result. As compelling as some the produced images are, this process leaves almost no control to the designer.

A METALLIC FACADE MUSEUM WITH CURVED SURFACES

My goal has always been, to make a design tool, and so finer control is absolutely necessary. This means that additional input is required.

With the plethora of generative AI available ...

DALL·E 2

MIDJOURNEY AI

Why did I chose to use Stable Diffusion?

Stable Diffusion

OPEN SOURCE

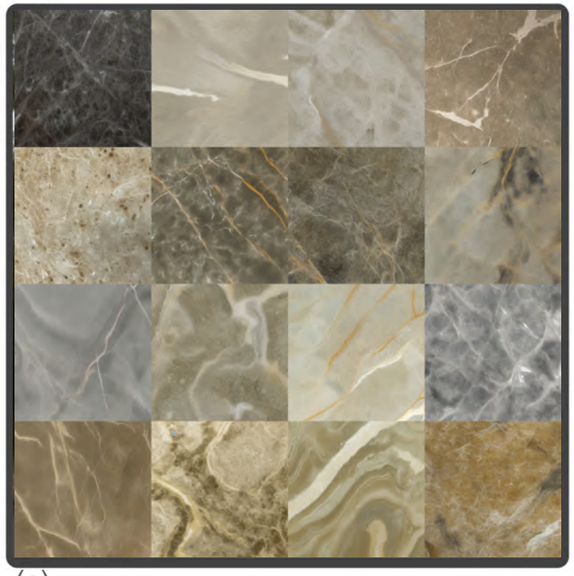
LOCALLY RUNNABLE

CUSTOM MODELS

These three points stand out as particularly important in the context of customization. Especially creating custom models from self-collected data.

Take for example Pentelic Marble. It happens to be the marble The Acropolis in Athens is built out of. Like all marble, this specific variety has unique colors and striations that give it a recognizable appearance.

Unfortunately, on the left (a) is what Stable Diffusion thinks Pentelic Marble is. It does clearly understand what marble is, but it lacks information on the specificity. This is a case where Stable Diffusion was not trained on a particular token.



(a)

(b)

Fortunately, new data (b) can be collected and either a new model can be made or the data can be added to the existing knowledge base.

This principle holds true with any type of data, such as a learning a specific site condition for a historical intervention. Equally valid would be a firm training a model on their existing rendering style to produce more representations in their own style.

CARITAS IN MELLE | JAN DEVYLDER

These examples show architecture that must yield to its given site condition with extreme precision. The ability to feasibly document and recreate the existing condition could provide an enormous value to a designer.

HEDMARK MUSEUM | SVERRE FEHN

PRODUCTION OF TEXTURES

GENERATING TEXTURES:

The first step in the production of useful material information with Stable Diffusion, is in the production of textures. Textures is the smallest scale explored in this research. Both my research and this book tackle control at multiple scales and the way in which we can build up layers of finer design input through these multiple scales.

If we ask for a texture of a brick wall, what would you expect? We likely have an image in mind. However, in reality there are ten's of thousands of distinct unique brick textures. (see right)

This opens up a broader discussion of bias that I will dive into deeper in my closing thoughts. Everyone has preconceived notions of what a material (or other subject is), however Stable Diffusion has its own biases. The set of images that were chosen to be part of the data is a bias, so are the captions and tokens given to images during training. Moreover even a "random collection" of images taken from the Internet on a given subject is bias. Those images were uploaded by people, and are being filtered by Google or other companies. The point being, bias is a integral piece of the training of an AI model like this. We can nott hope to remove bias from the models, and so instead we can leverage it.

The images on the right encapsulate the difficulty in trying to generate meaningful textures with such a simple prompt.

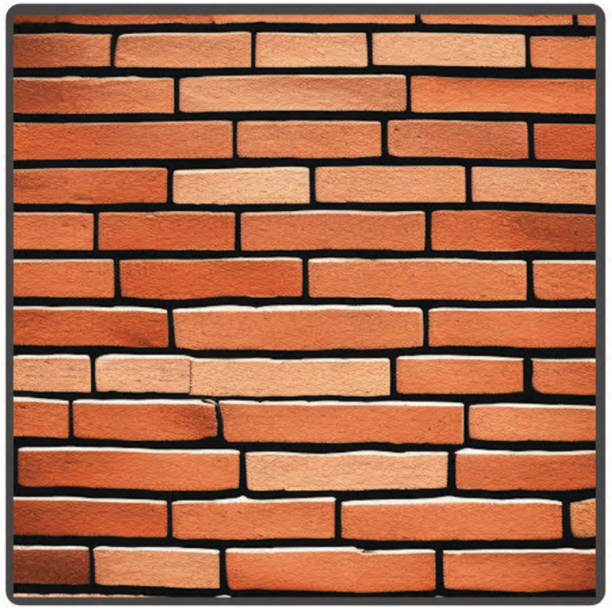
A TEXTURE OF A BRICK WALL

If we ask Stable Diffusion the same question, “a Texture of a brick wall”, looking at a single generated image, just as with our own preconceptions, does not tell the whole story. A large sample of images created from this prompt reveals two things.

1.Reafierms that preconceived notions of material is absent in the generations created.
2. These materials are objectively not good.

In order to combat this second point, we can introduce further specificity in three areas: Detail, Structure, and Lighting.

A TEXTURE OF A BRICK WALL



DETAIL

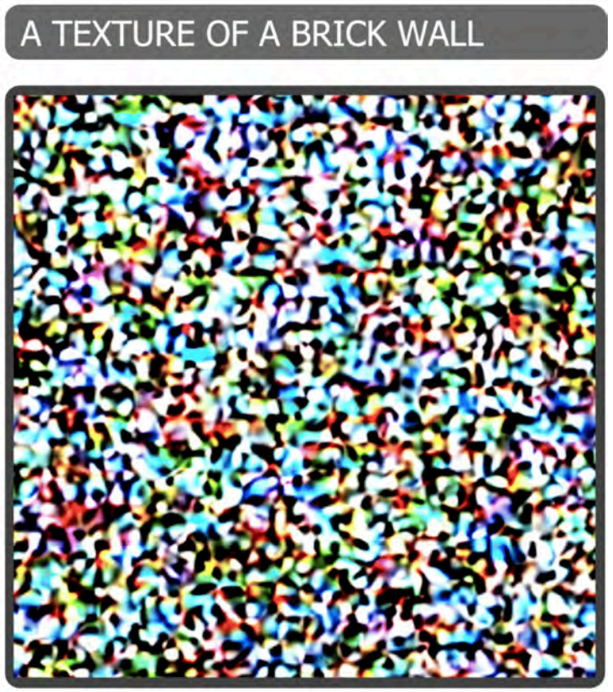
Detail is the simplest in that we can just be more specific in our prompting. Asking for the bricks to be blue, gives us results that are by and large more consistent.

Consistency here is a good thing, we should expect that as a prompt becomes more specific, our results converge towards an expected result. The user can decide how much specificity is acceptable for their own needs.

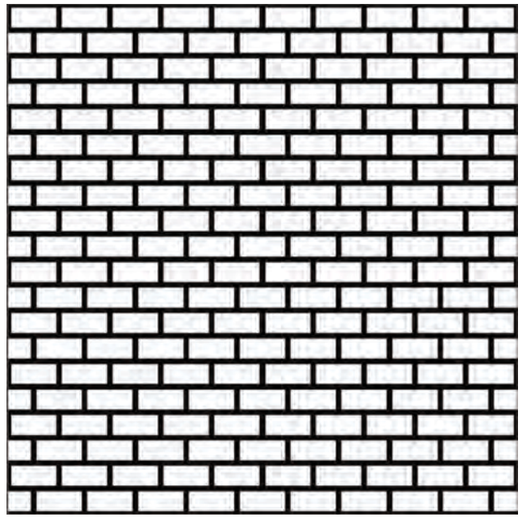
STRUCTURE

Our second issue to tackle is structure. By that I mean the physical composition and construction of the surface.

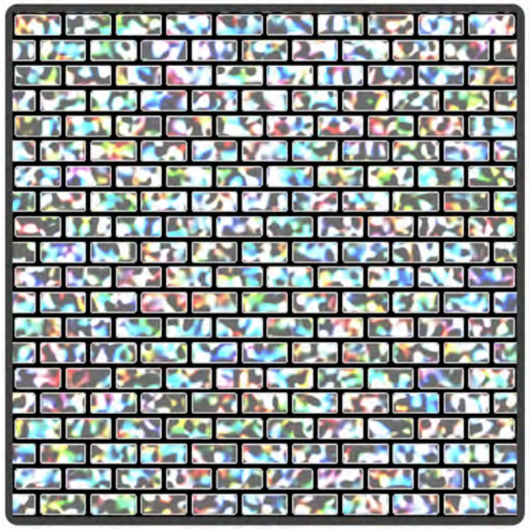
If once again we ask for our texture, but before generating, inject a new layer of data into our process such as this tiled brick pattern (b), we can use this extra information to guide the generation process. This uses a method called Control Net, specially trained to allow for the use of multiple forms of input such as edges, sketches, depth maps, and normals.



(a)

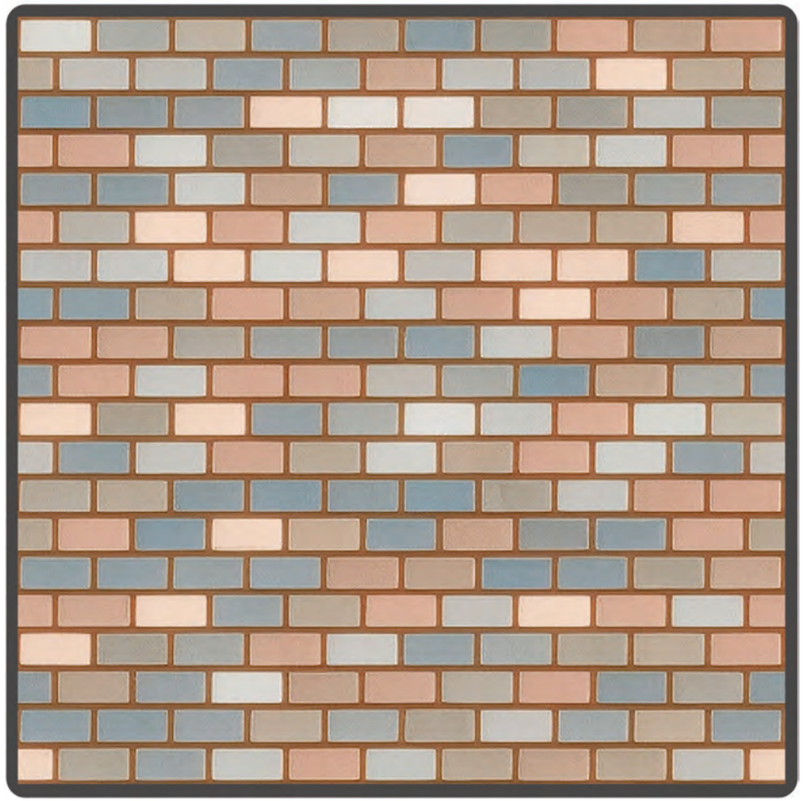


(b)



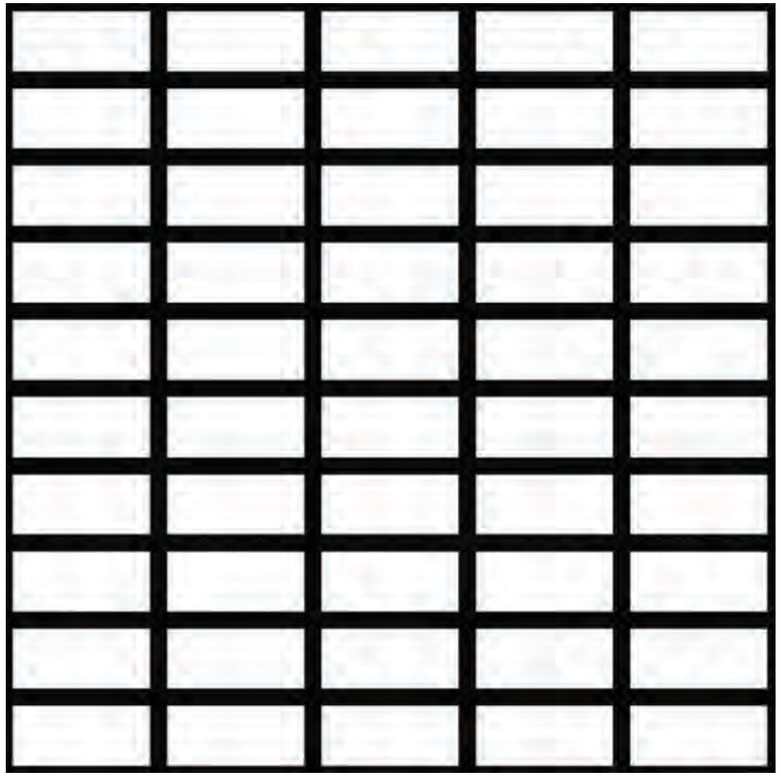
(c)

The consistency of results using this method is tremendous.



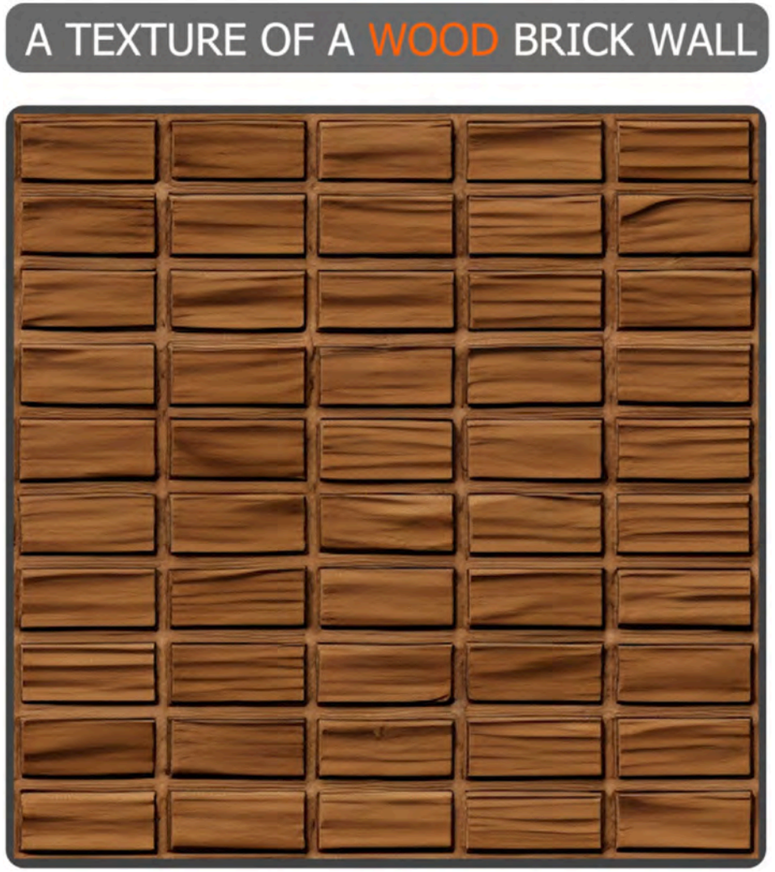
The use of this extra layer of information is independent of the detail we discussed.

Changing our input structure



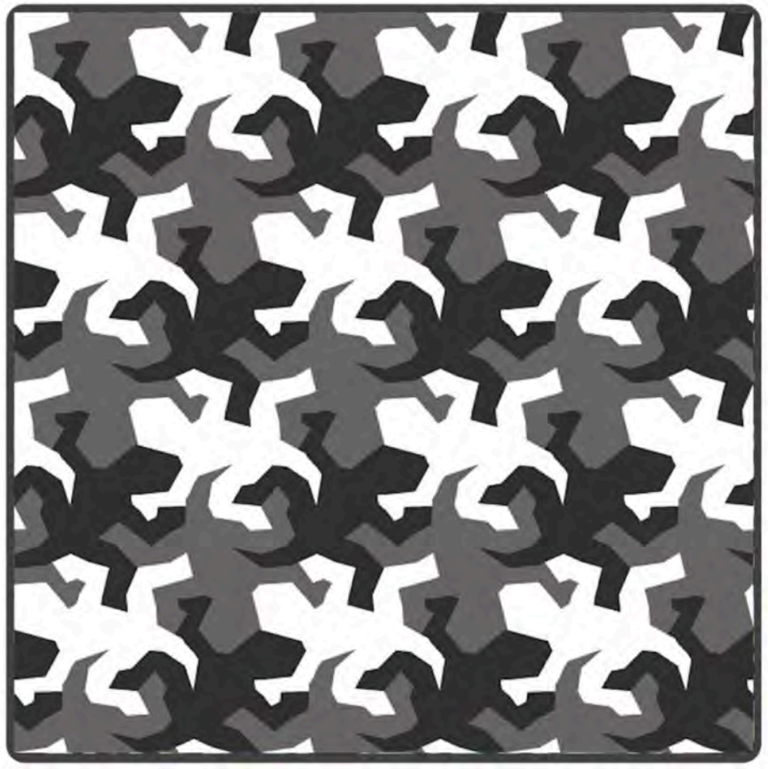
Gives a different result

Where this is most powerful however, is in decoupling the detail of a texture from its structure. We can use it to ask for something more creative. Here, a wood brick, along with other examples.



In separating these two elements of a material in this way, we can begin producing results that go beyond what you may find in a digital library.

A non-orthogonal structure

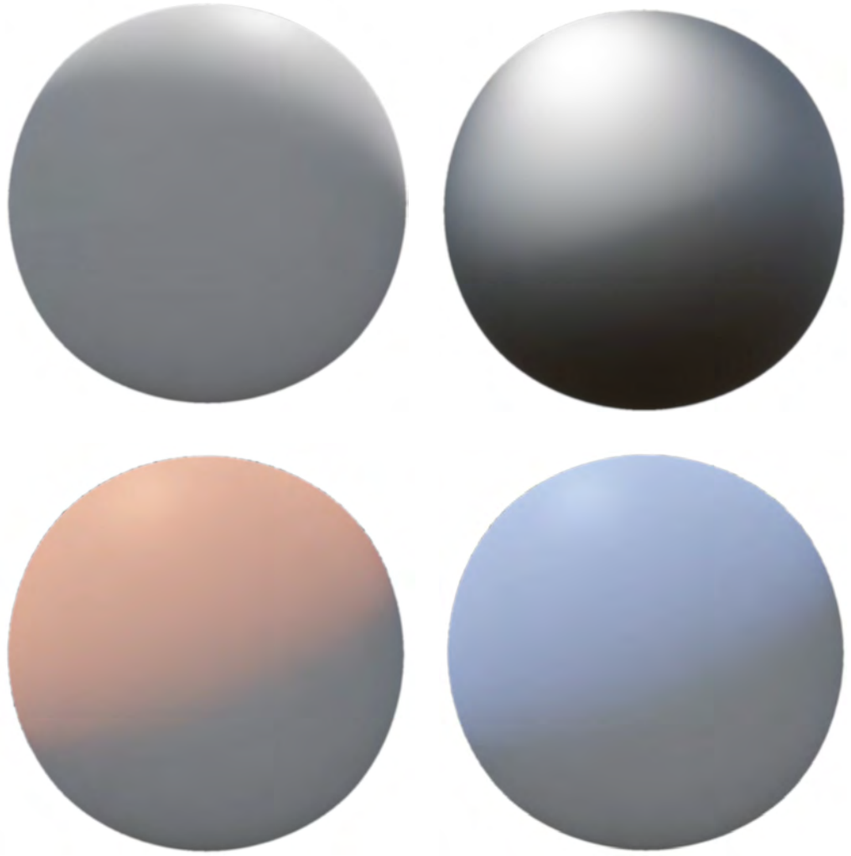


A TEXTURE OF A WOOD FLOORING

LIGHTING

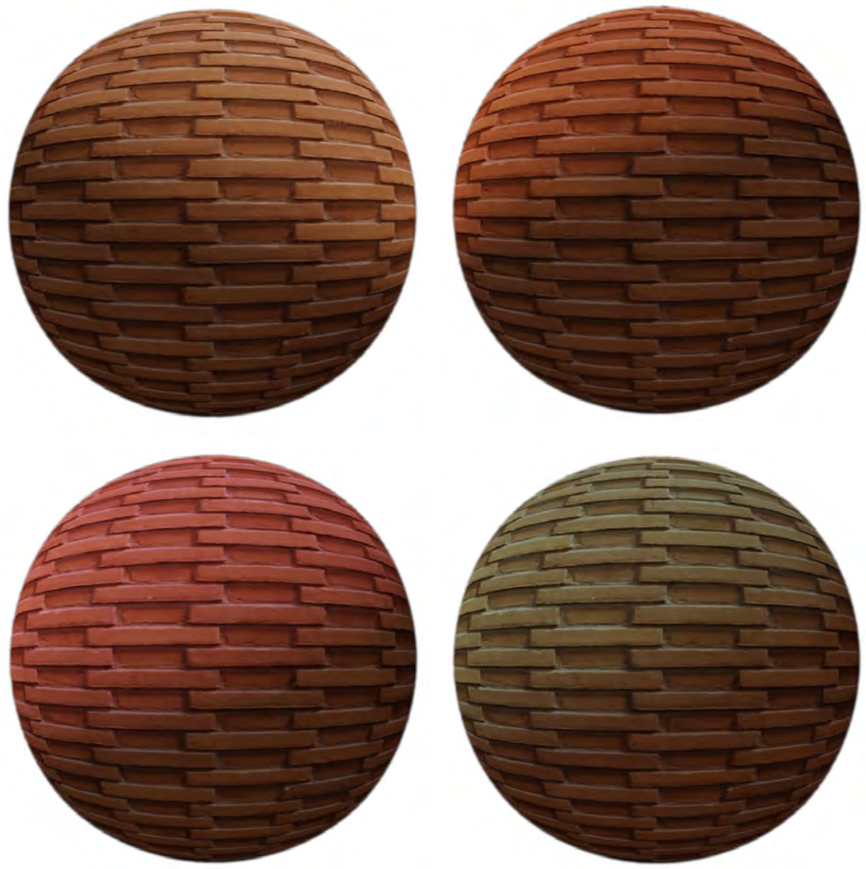
And third we have light.

Our perception of materials is not static. They react to environment changes such as lighting, color and reflection. Altering properties of a object such as metalness and roughness, or conditions of the environment such as color and direction of light(s), will produce drastically different results in a 3D simulation.



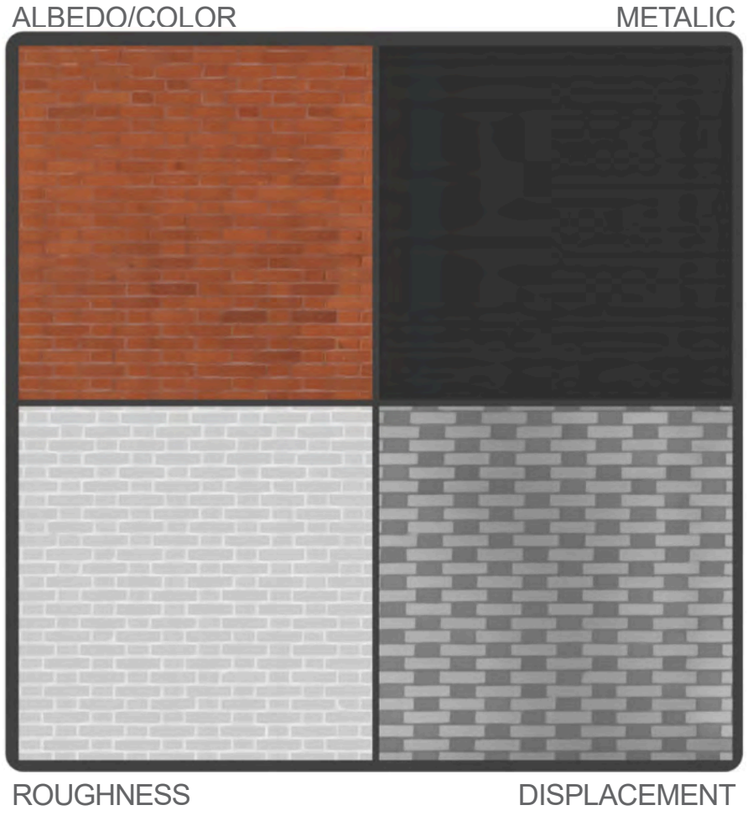
However when we generate a texture with Stable Diffusion, we are lacking necessary information for simulating light properly . That is because Stable Diffusion is not giving us a texture, but rather an image.

IMAGE

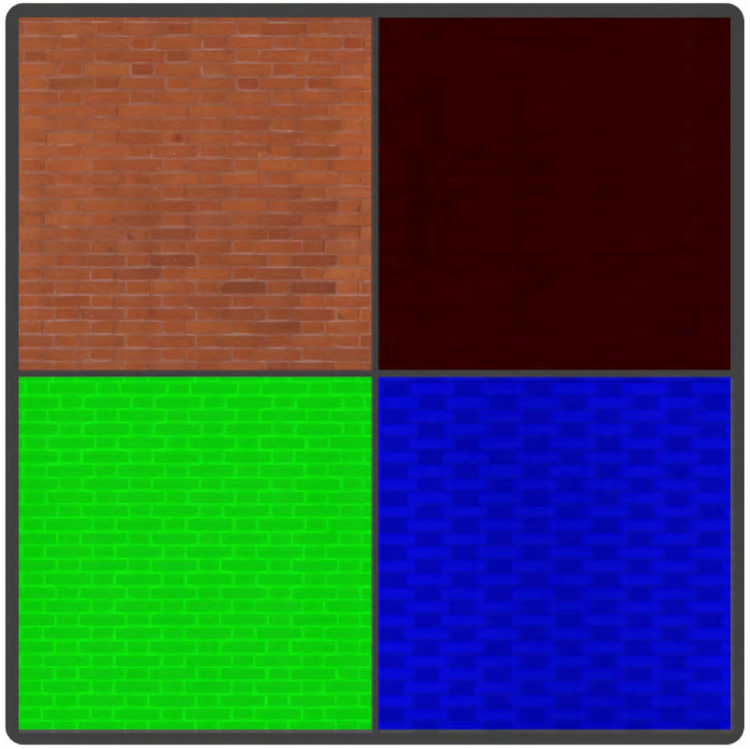


A texture is really made up of all these different maps that each provide information on properties of a material other than just color.

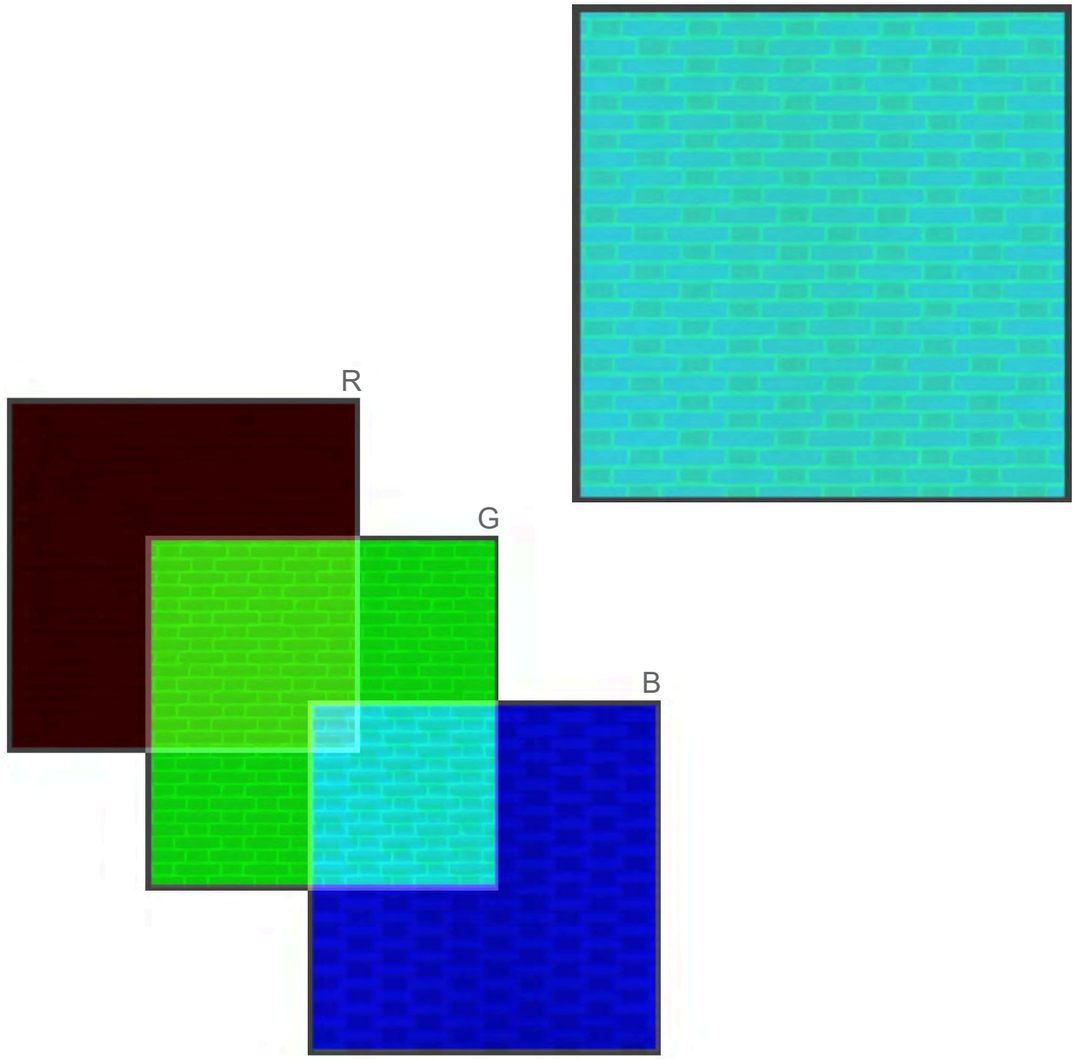
To reduce complexity, if we select just the most critical ones. We are left with color, metallic, roughness, and displacement. These maps are adequate for simulating light in a 3D environment, but for efficiency sake, we can go further.



Black and white maps (metallic, roughness, and displacement) can be combined into one single image, each making up one of the red, green, and blue channels. Taking our total down to just 2 maps for adequately representing our texture.



This is known as a channel packed map



Our texture can now be represented in just two maps.

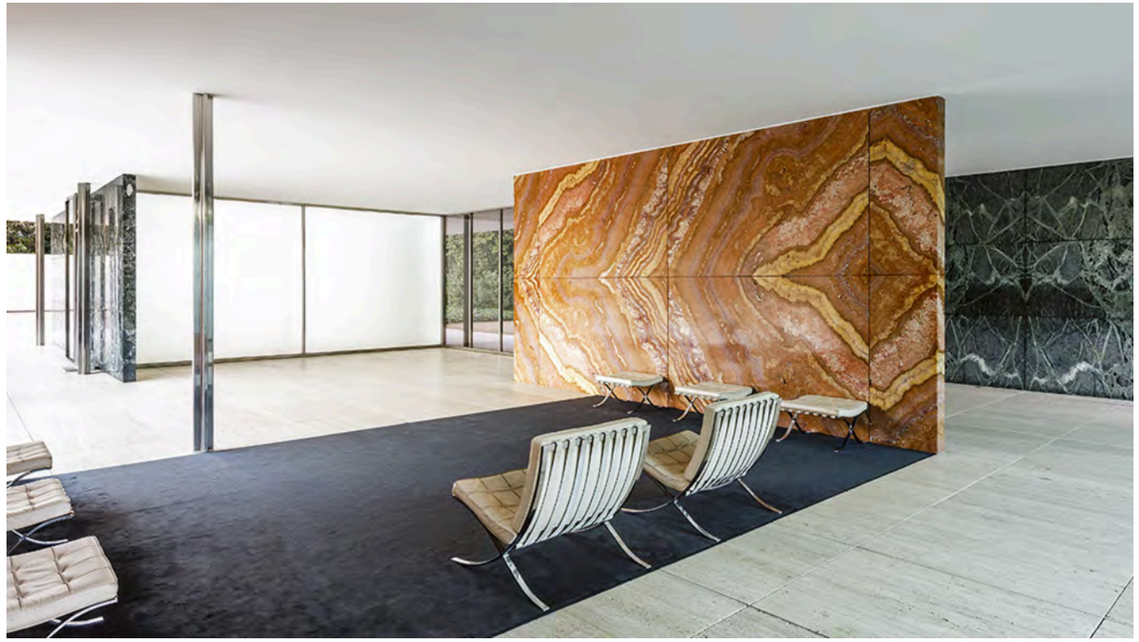
By training and generating these two maps in tandem, we can finally produce materials that react to changes in lighting and environment.

HOW MAY THIS BE USEFUL?

Take the Barcelona pavilion. Instantly recognizable by its unique book-matched marble contrasted by slim chrome columns.

BARCELONA PAVILION | MIES VAN DER ROHE

What can we do to reproduce this image (1)bellow? Using a library of materials, we could produce something to the extent of the right image (2). But of course this looses the most important aspect in the room: the wall. So focusing on the texture of the wall, in the lower image (3), even though it does not fully match that of the real life counterpart, It adequately captures the important aspects. Moreover, it captures the aspects of the original that it was given. How was the wall in the image made?

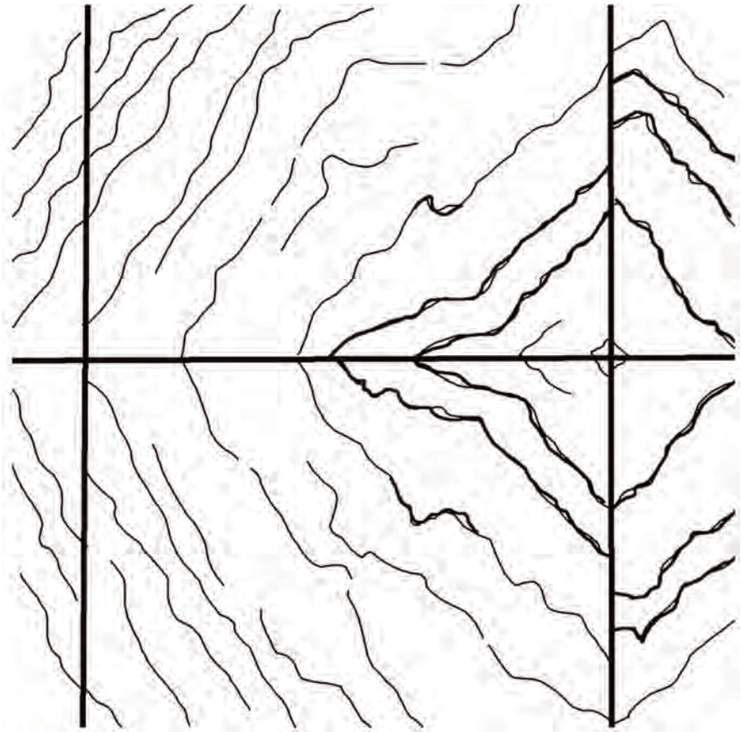


(1)

(2)

(3)

A TEXTURE OF A BOOK-MATCHED MARBLE, MULTICOLORED, BROWN, ORANGE, AND PINK WITH GOLDEN STREAKS



Begin with a quick sketch of the marble pattern. Combined with a specific prompt we can rapidly produce extremely convincing results. The focus on the input side here was the structure, and inconsistencies with color can be attributed to a lack of specificity in input with relation to that property of the material.

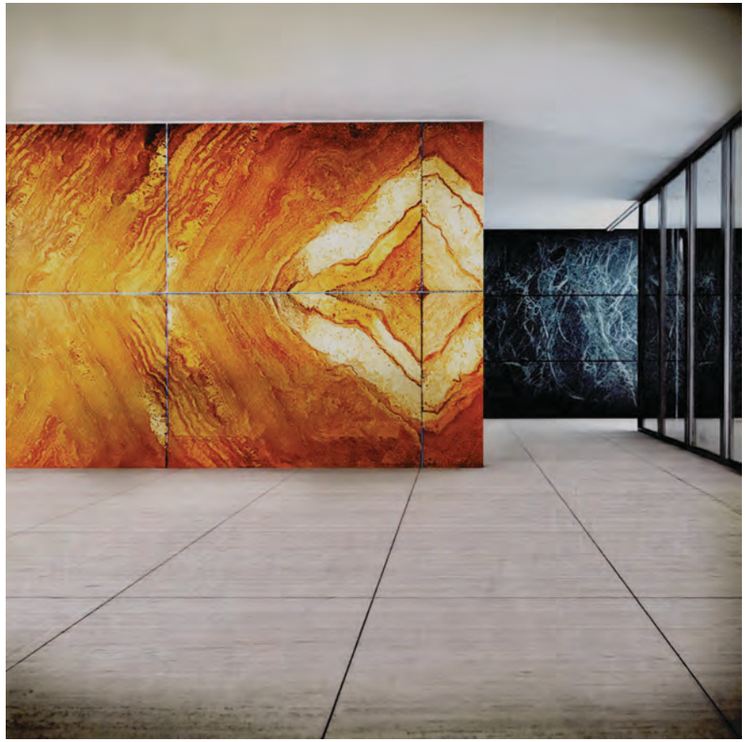
A confession: the image from earlier is not a picture of the Barcelona Pavilion, but rather a rendered reconstruction. It is worth noting that the majority of this not produced with AI, but rather only the textures for the marble walls. Other than the furniture, which for time sake and not being the focus of the study (which was taken from the original), it is a simple textured 3D model with a bit of Photoshop for post processing.

Render of Barcelona Pavilion

Using this as a tool, does not mean it replaces existing workflows, but rather enhances them. For me a big realization that I had through conducting this research is that the processes in place currently are often faster for certain things than the AI alone. Particularly when aiming for a designed result, a strategic utilization of AI within existing rendering workflows was most efficient and effective.

Photo of Barcelona Pavilion

Bare 3D Model



Textured 3D Model

MASKING AND LAYERING

MASKING AND LAYERING:

Utilizing this knowledge of generated textures, we can move into masking and layering.

The core concept behind this is that - through breaking down an image/object/process into smaller pieces, then tackling those pieces individually, a high level of control can be achieved.

We will tackle this problem at 3 scales:
Texture, Object, and View

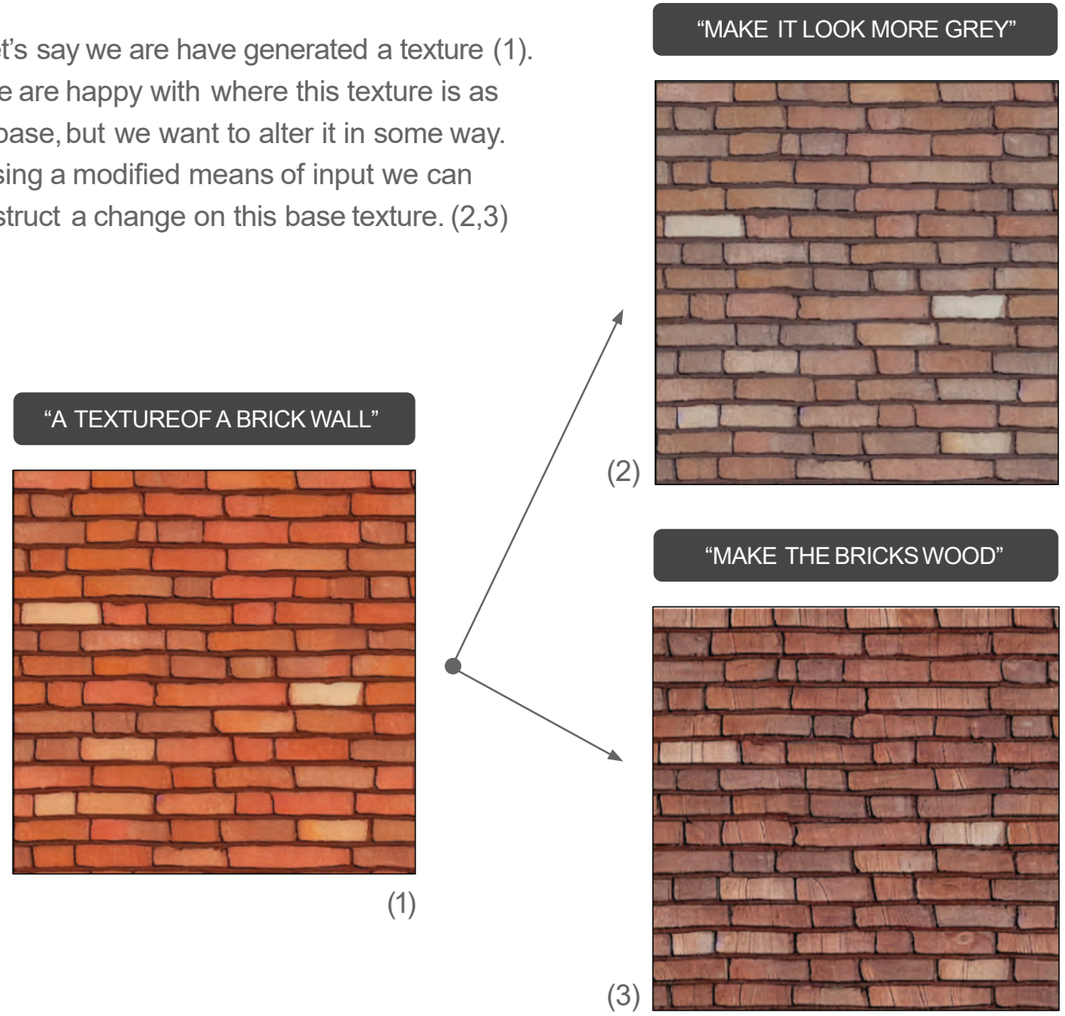
TEXTURE SCALE

OBJECT SCALE

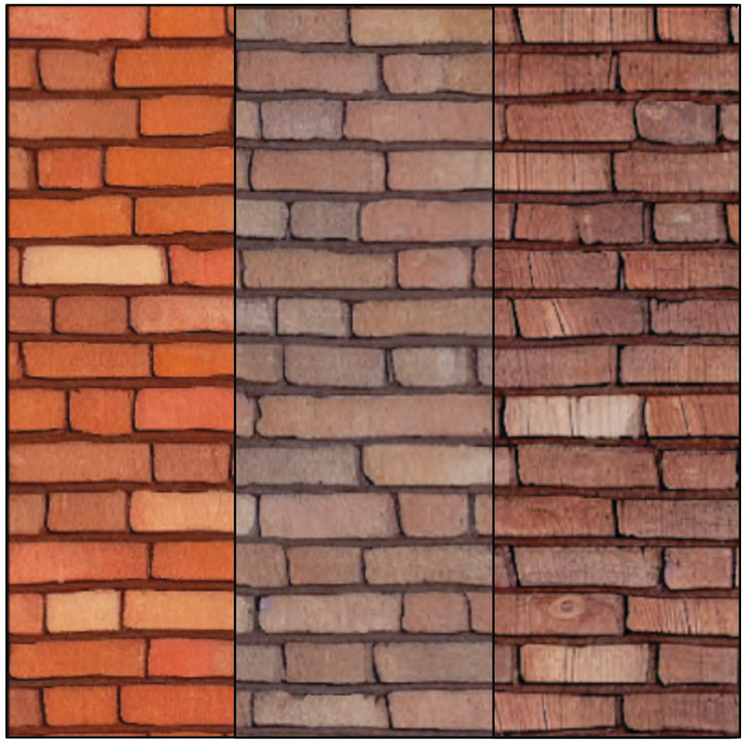
VIEW SCALE

INSTRUCTED PROMPTING

Let's say we are have generated a texture (1). We are happy with where this texture is as a base, but we want to alter it in some way. Using a modified means of input we can instruct a change on this base texture. (2,3)



Importantly, this method also provides extremely consistent results in structure.



Shared Structure Between Images

TEXTURE MASKS

Using the example, “make the wall very mossy”, we can introduce a new layer of customization. By injecting a mask into the generation process we can instruct Stable Diffusion to only modify certain portions of the image.

In this case, a generated edge mask has blocked out only the mortar of our bricks to be replaced. This can be thought of as overlaying the normal and mossy bricks together with the mas. In reality it is using the mask to only denoise certain parts of the image, and generating the new portions in the context of what is masked off.

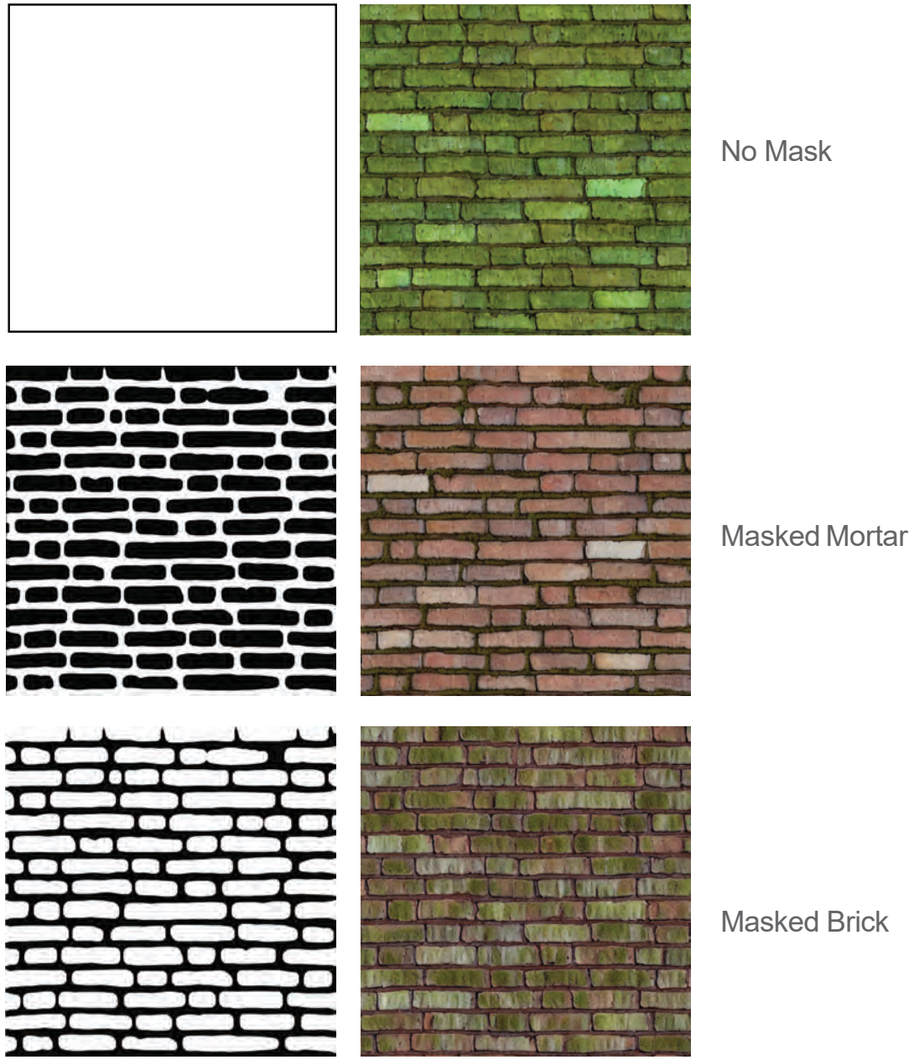
By inverting our mask we can produce the opposite effect.

The mask in this example is procedurally created using canny edge detection and then dilating and smoothing the edges. This is done through the OpenCV library in Python.

This is another case in which the use of non-AI technologies proved to be an effective means of supplementing this workflow. Although Stable Diffusion is technically a deterministic algorithm (give a specific input, seed, prompt, and receive the same result) the complexity of its inner workings, make is so that it is not feasibly predictable in the way a traditional algorithm is.

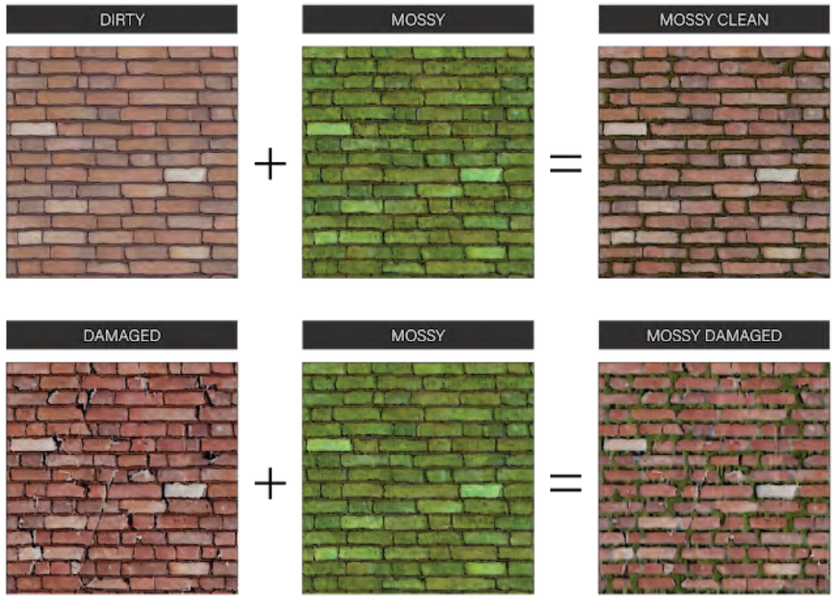
In this case performing an image processing, like edge detection, we can rapidly produce layers of information that are both understandable by the user and the AI simultaneously. These forms of input, where the user has a meaningful understanding of what is provided and a feasible means of altering that input, allow for designed results. With enough layers of low level input, a high level result can be achieved.

There are certainly other ways to produce masks, both procedurally or manually. A mask could be painted for instance, or in the way this process is functioning, the mask could be outlined in sketch and then dilated to an appropriate amount.



INTERESTING COMBINATIONS

Using a combination of instructed prompting and texture masking, we can begin to make a set of interesting permutations of our base texture.

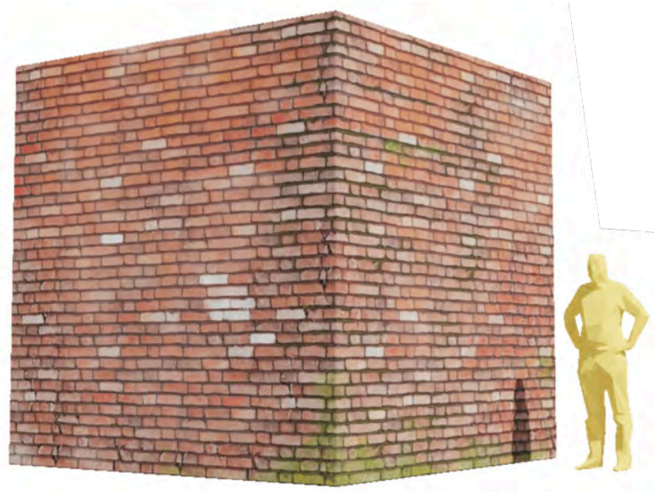


LAYERED OBJECT MASKS

This texture set can then be applied to our next scale. The object scale.
Here, the base cube was textured with our original brick.

In the second step, we generate a black to white gradient mask, based on the cubes proximity to the ground, and use that mask to mix our base brick with a gray brick.

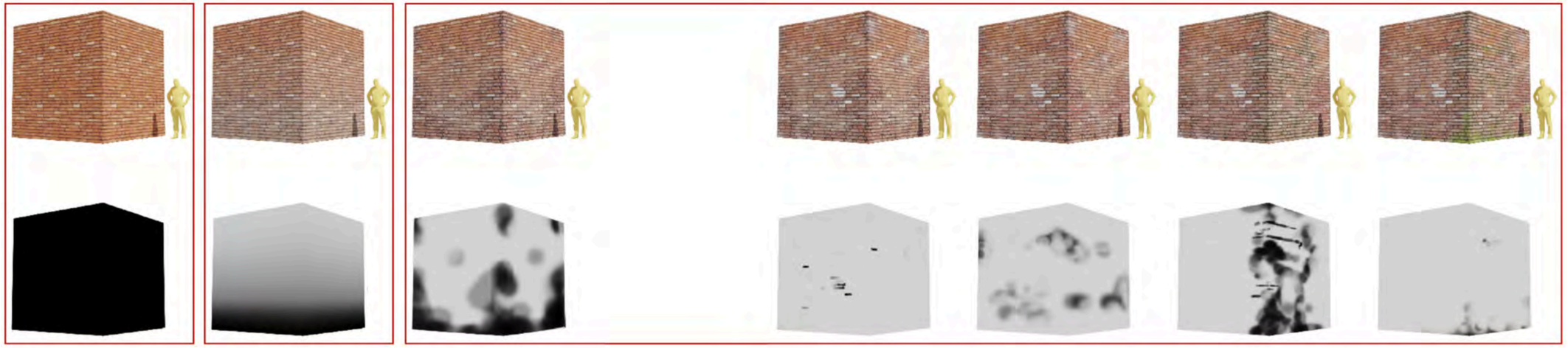
In the following steps, the mask is painted by hand, in 3D on the object. Each layer introduces a new texture and added detail. This process is just like a 3D version of layers in Photoshop.



(a) base texture

Generated Mask:
(b) Gray brick

Painted 3D Mask:
(c) Damaged brick
(d) Painted brick
(e) Red Brick
(f) Mossy Mortar
(g) Mossy Brick



(a) (b) (c) (d) (e) (f) (g)

APPLICATION

Let’s return once again to Mies.

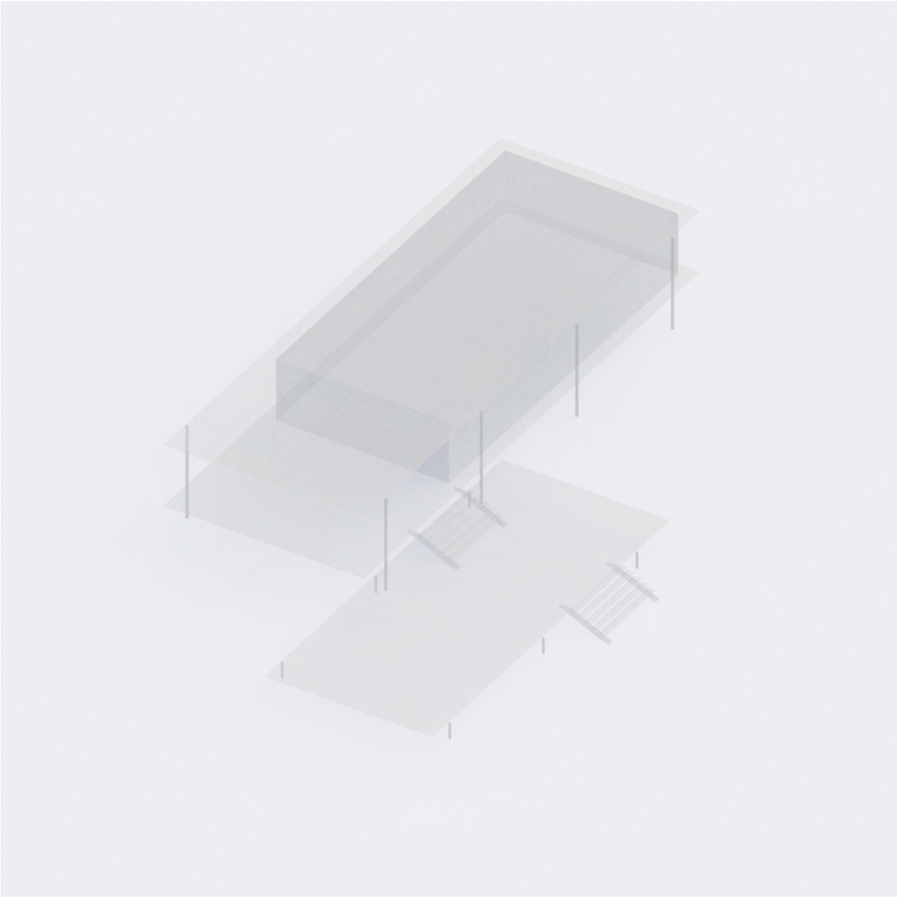
I promise this image is real this time.

We will use the Farnsworth House to demonstrate the object scale, and then expand that into our final scale of masking: the view scale.

EDITH FARNSWORTH HOUSE | MIES VAN DER ROHE

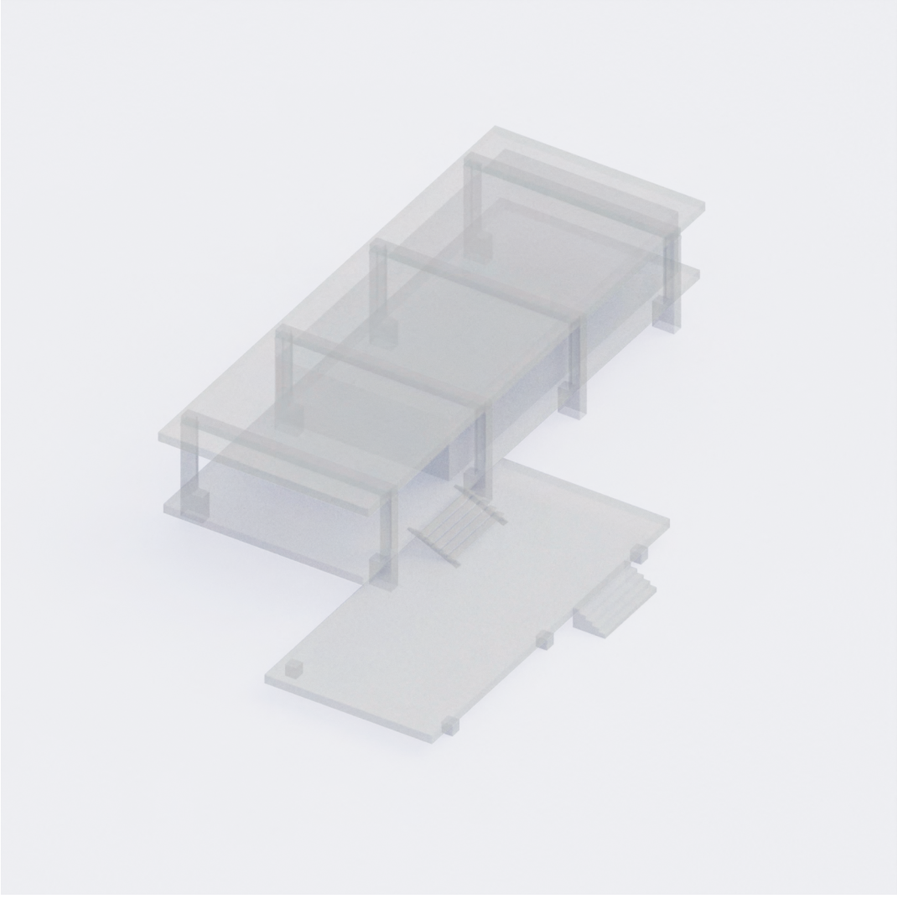
Say we want to reimagine this project in a new material(s), the one goal being topological consistency. How can we rapidly iterate to arrive at this image on the right.

We can begin with a simple surface model.



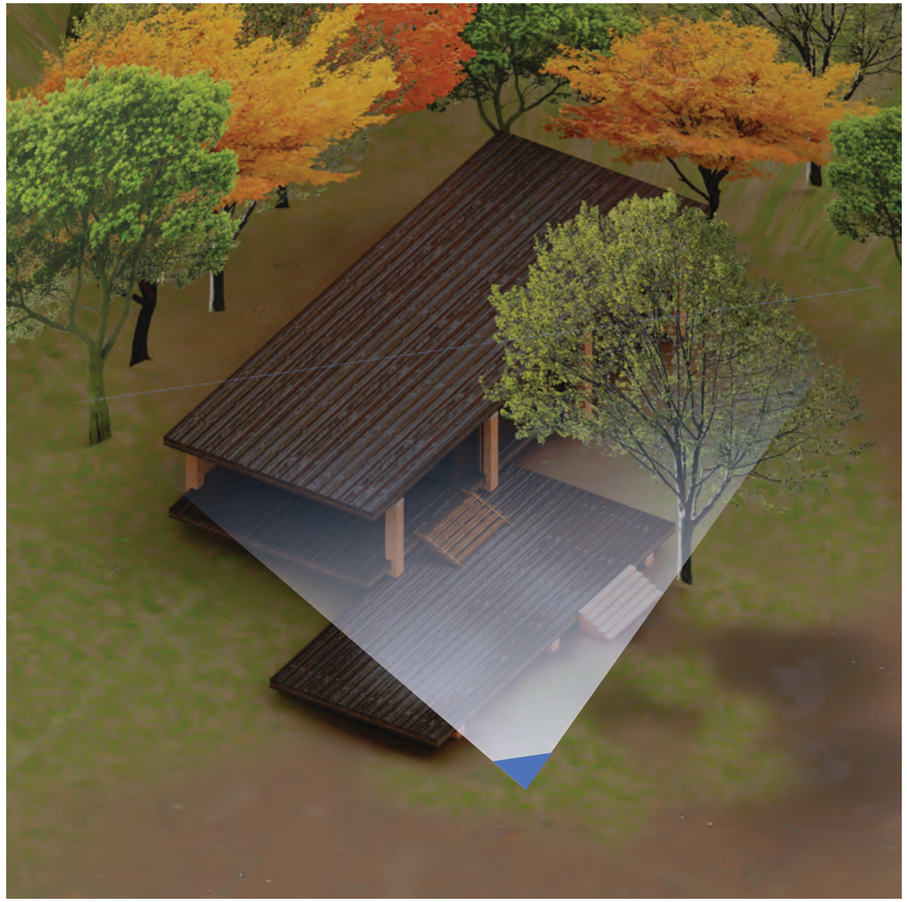
From there, we can generate textures for specific objects as well as paint layers of those textures to create variation across the surface

However we need to create thickness, so we embed the generated textures with the appropriate data to perform procedural extrusions on our geometry.



And the generated textures respond appropriately.

Simple flat cutout trees are then added to our scene.
Then all orient to face our camera position.



The view from our 3D camera.

To get from here to our final image, we can move to the third scale of masking: The view scale.

We can utilize a series of masks and layers from our 3D scene such as edges, depth, and object masks in a very similar way as we did at the texture scale.

(a) Object mask of tree collection

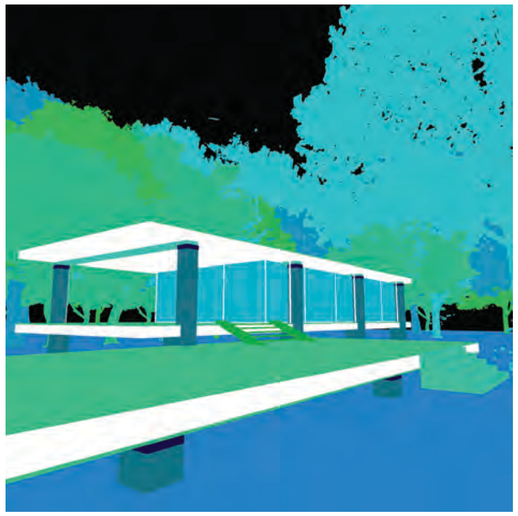
(b) Material cryptomatte, separating objects based on materials

(c) Depth map, calculated based on distance from camera

(d) Canny edge mask, generated in same way as for texture masks

(a)

(b)



(c)

(d)

A MASK OF THE TREES

ORIGINAL TREES ISOLATED

GENERATED IMAGE OF TREES

OVERLAYING BACK ONTO ORIGINAL

In continuing this process of masking, generating, and re-layering, the final image can be built up with a high level of precision.

I think it is worth mentioning that our three scales do not operate in the same dimensions. The texture and view scale operate in 2D, while the object masks operate in 3D.

2D IMAGE MASKS

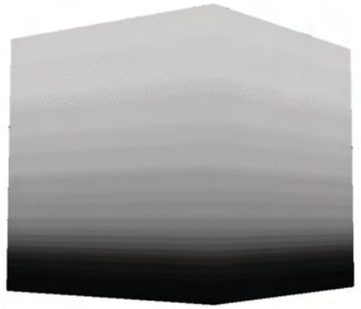
TEXTURE SCALE

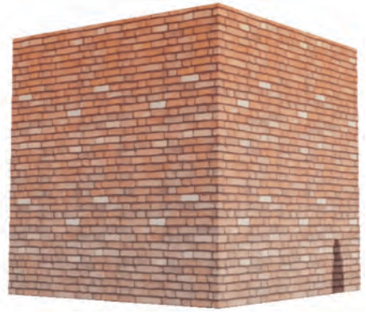
VIEW SCALE

This back and forth between these two dimensions is pivotal in getting the most from this tool. Arriving at a designed result is about utilizing the benefits that each dimension offers.

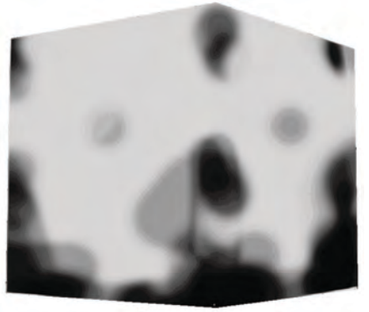
3D OBJECT MASKS

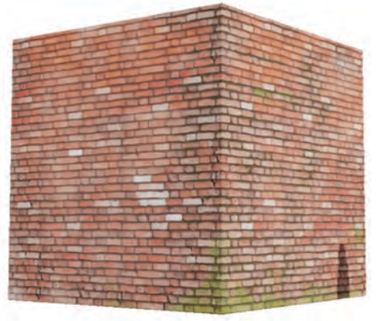
PROXIMITY





PAINTED





DESIGN STUDY

This design study reconsiders the Icelandic Turf Hut in a contemporary context.

I'll note that these are homes that are still built today often in a very similar manner as their original construction. I chose this typology a base point due to its clear connection to its environment both formally and through its materiality.

This study focuses on the diversity of workflows available while utilizing Stable Diffusion and is demonstrative of the versatility of the tool.

COLLECTION OF TURF HOUSES | ICELAND

Stable Diffusion understands what a turf hut is to some extent, however the images produced when asking for one were not good. Here, similar to the Pentelic Marble example from much earlier, we see the specifics are off.

From things as minute as the color of the grass, to larger issues with proportions of architectural elements, the existing model is an unstable foundation for building up imagery of the house that is going to be designed in this study.

To combat this, a model was trained on dozens of collected images of turf houses in Iceland. To be critical, these are images I curated and selected, however they are still images that were pulled from the Internet, and so there are biases in the model I trained that are out of my control. This is obviously not ideal, and best would be if I could make it to Iceland myself to properly curate a set of images for training.

That being said, the results of the model are still fantastic, and can be seen on the following pages.

Control image generated prior to specific training

Collected training data on Icelandic Turf House

Generated image after training using same prompt and seed as control image

I proceeded with picking a site to work on. This seemed like a nice enough spot.

I liked the rocky ledge protruding from the hillside, it could serve a natural shelf for the house to site upon.

Lets say that I, the designer of this project, have no experience with complex 3D softwares and digital tools. Maybe I do know those tools, but prefer to design with physical models in a tangible way. In any case, I want to model quickly and freely with less concern for craft and more of an emphasis on rapidly capturing the design ideas in my head.

However, I would still like to make full use of this technology and material information as feedback towards the design decisions I'm making. How can this specific scenario or others like it (such as sketching) be accommodated for?

Here is a beautiful reconstruction of the chosen site. It uses fancy materials like rocks, dirt, and a cardboard box.

An initial iteration is then carefully put together over the course of a few minutes, with stunning hand-cut chipboard, hot-glue, and more dirt. A pristine and fully accurate background is put in place behind the model.

By overlaying a photo captured on my phone from a perspective close to that of the site image, then keying out the yellow background, the model can be placed in context.

From there, a pair of images can be generated: a canny edge detection and an AI-generated depth map. This set of information, in tandem with the original color data, is all the information necessary to produce an image.

The image on right was procedurally generated with only the input of the model photo, the site image, and a prompt. “A DSLR photo of a contemporary Turf House in Iceland, dark wooden walls, grass roof, buried on one side.”

I was impressed by the AI’s ability to pick up on small details in the model, particularly the thin slit window along the left wall of the house.

Here a single simple change was made to the model, diagonal score marks were added to the front wall. This change was well integrated by Stable Diffusion. Note, that from this point forward, quickly painted masks were used to in-paint only specific areas of the image where a change was desired.

I am getting near immediate high resolution and highly polished feedback in order to make rapid design decisions.

Here, a pitched roof was added. Once again, by masking off the areas for change, the rendering is updated accordingly. With more drastic changes like this, updating the prompt to better reflect the design is important. In this case adding “pitched roof” to the prompt helped.

A stair and mullions. I will say, the stairwell is out of proportion, in both the model and the render. This is a good thing. Decisions like this are not necessarily a mistake, so the AI should take in the information at face value, unless in some capacity instructed otherwise.

Here the roof is shifted back to reintroduce a live roof to the house.

After this set of iterations on the physical model, we can continue to get some additional information out of the AI. By instructing "make the house look weathered", we can receive useful visual feedback. This is by no means an accurate simulation of weathering (although with proper training it could be), but rather a reasonable interpretation that is close enough to reasonably situate the house in its environment.

Here, the roof was made grass, by painting a mask in that area and reintroducing significant noise to that portion of the image.








At this point in the design study I wanted to work on building up the interior of the space.

In order to demonstrate the tools and workflow my research focused on most, this portion switches over to a fully digital workflow.

I somehow have not mentioned yet that I chose to develop software inside of Blender. Blender is an open source 3D modeling, animating, and rendering platform that is largely absent from the architectural office.

Personally, I have used Blender along side Rhino for the last five years, and I have found it fill in large gaps in my workflow that Rhino does not provide alone. Its absence in the discipline at large is due to a multitude of reasons, likely in large part to its steep learning curve, but nonetheless I believe that it provides a platform for a much more interdisciplinary approach to design than something like Rhino.

Similar to Rhino, Blender has a set of growing node system tools that, particularly recently, have allowed for significant developments of procedural workflows. Beyond that, it provides built in support for texture painting and much better control over UV editing.

The open source nature of the software also lends itself well to development. Especially with Stable Diffusion being open source as well, there is a significant overlap between these two communities of users.

To the right is an overview of the process I used here.

IMPORT PHOTOSCAN OF PHYSICAL

CONSTRUCT DIGITAL BASE MODEL

SETUP CUSTOM NODE SYSTEM

GENERATE TEXTURES AND PAINT

REALTIME AI RENDERING

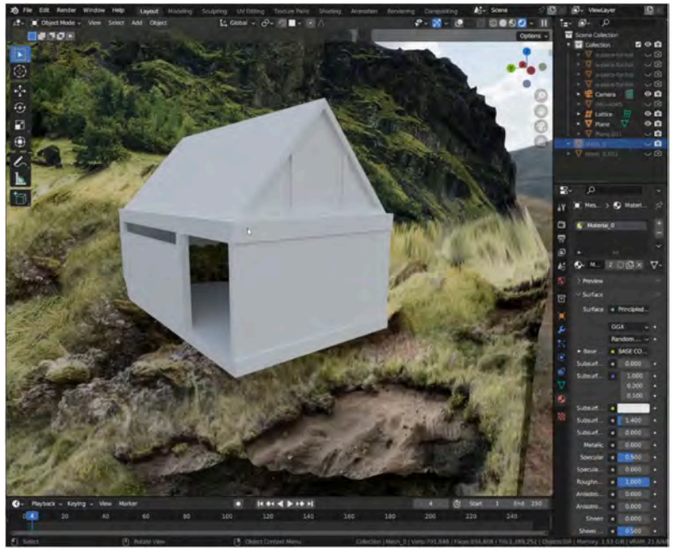
ITERATION

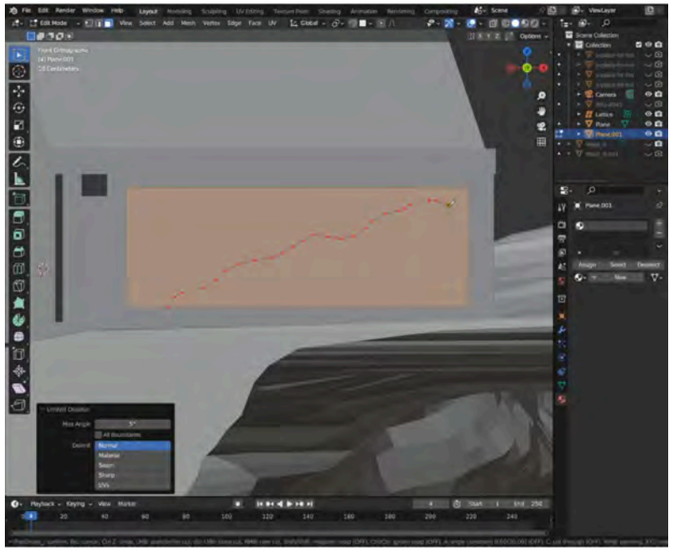
I began by generating a depth map of the site image, and used a tapered displacement to extrude the geometry in 3D space. This in conjunction with a HDRI provided a solid environment foundation.

This next step was not necessary for such a small project, but I photoscanned the physical model I made and placed It in the 3D scene. This was done to create a base in which I could model out a simpler and cleaner model on top.

I think more realistically, photoscanning could be used at the site scale to capture the environment more accurately. If I had access to the site, this is probably what I would have done, however the displaced site image proved quite effective. On the other hand, the method I used here, limits the 3D camera view significantly as compared to having a detailed environment scan.

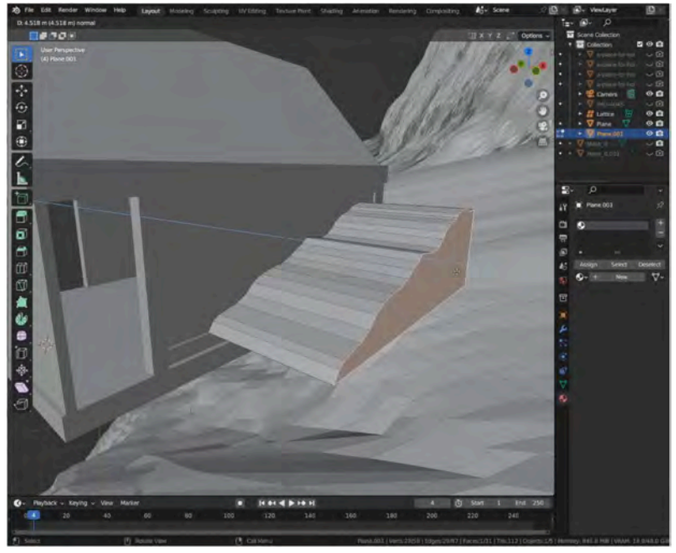
From here, I reconstructed the physical model. Even though I am using Blender for this, there is no reason any other modeling software would not be equally viable for this portion.





It was not visible in the photographs of my physical model, but i desired to have the far side wall be glass with dirt partly piled up against it. Here I am cutting out the shape of the earth mound.

I then chose a key view from the interior to focus in on. This is important because it can reduce the amount of unnecessary work required. The model only needs to be built up in detail in the visible areas.

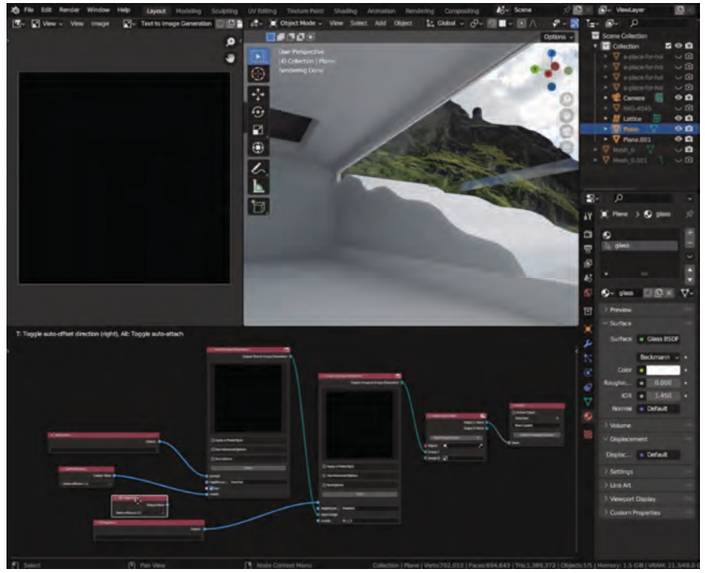


The mound is then extruded, and the modeling portion is complete.

CUSTOM BLENDER STABLE DIFFUSION NODES

Over the course of this research, I decided that the best interface for me to work with Stable Diffusion was through a node system. For those familiar with Grasshopper for Rhino, this is the same idea. This allows for rapid iteration through proceduralization of much of the workflow.

Blender has the functionality in its API to create a custom node system, however a thing that I sought to add to my nodes that is not easily doable in Blender natively, is a preview above each relevant node to show generated images. This was to provide better user feedback and make sure that long node networks didn't quickly become a black-box of information in the middle.

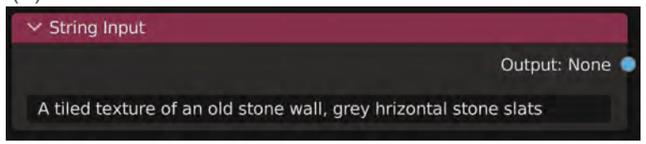


Input Nodes
(a) String Input (prompt input)
(b) Stable Diffusion Model Input
(c) Image input
(d) Object Input

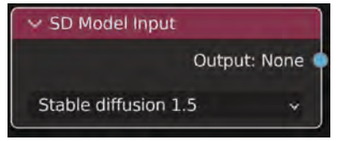
Generation Nodes
(e) Edge Detection Mask
(f) Text to Image Generation
(g) Image to Image Generation

Output Nodes
(h) Setup Object Material
(j) Result Node

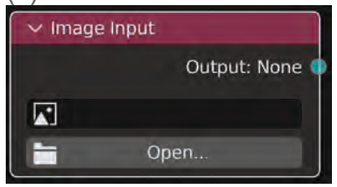
(a)



(b)

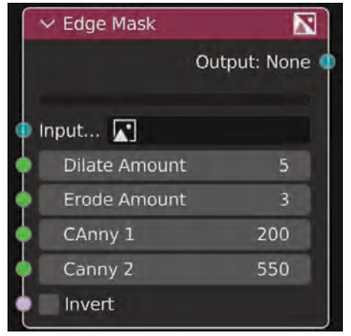


(c)

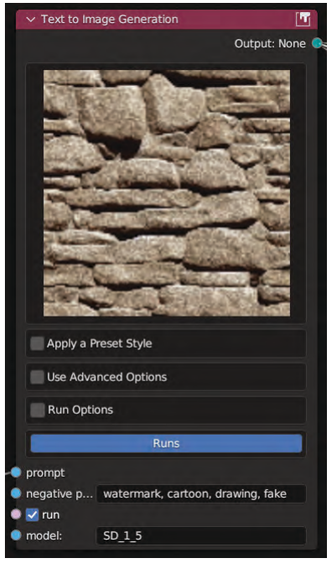


(d)

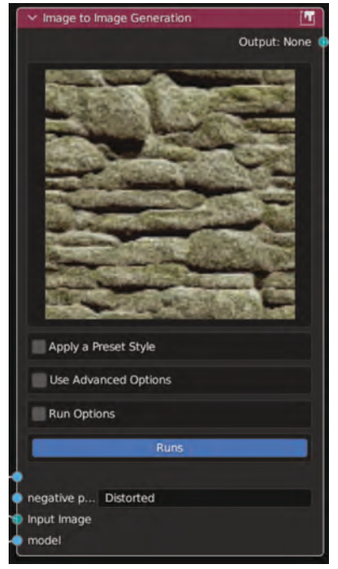
(e)



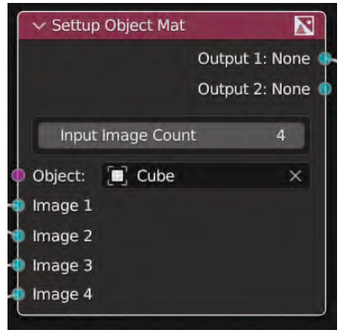
(f)



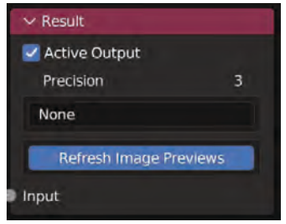
(g)



(h)



(j)



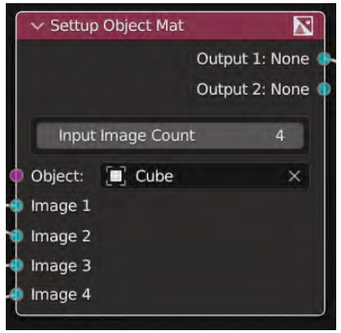
I would like to focus in on one node in particular: Setup Object Material Node.

This node takes in up to 10 images as input in addition to a mesh object.

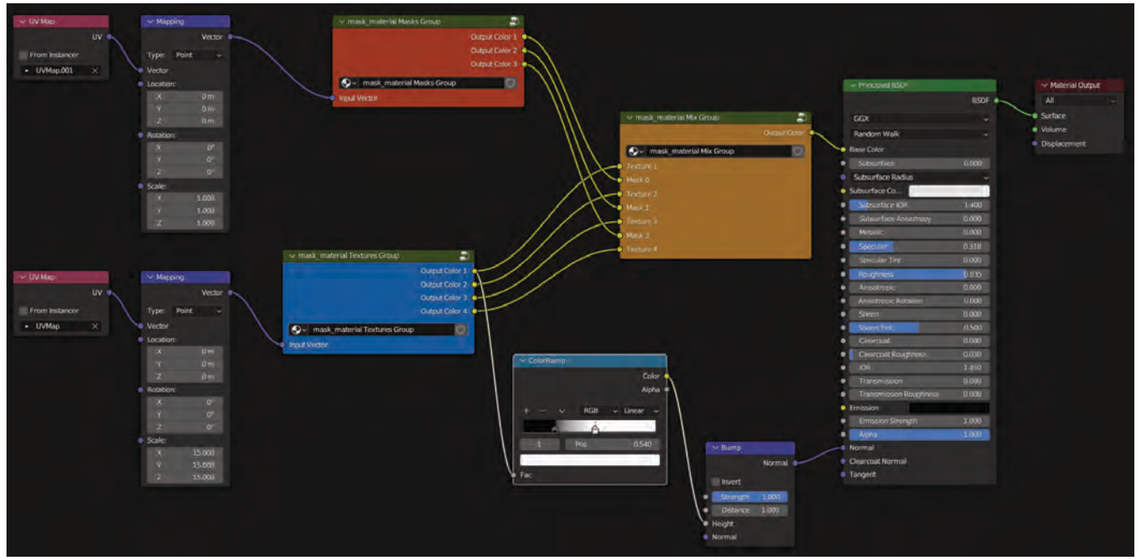
It then will automatically setup a Blender node material network for the user that arranges the input images as layers that are subsequently mixed together. It will also produce a mask layer for each image, which is setup ready to be painted on.

This is done in an attempt (I believe successfully) to remove as much grunt work from the designer and automate the tedious parts of the workflow.

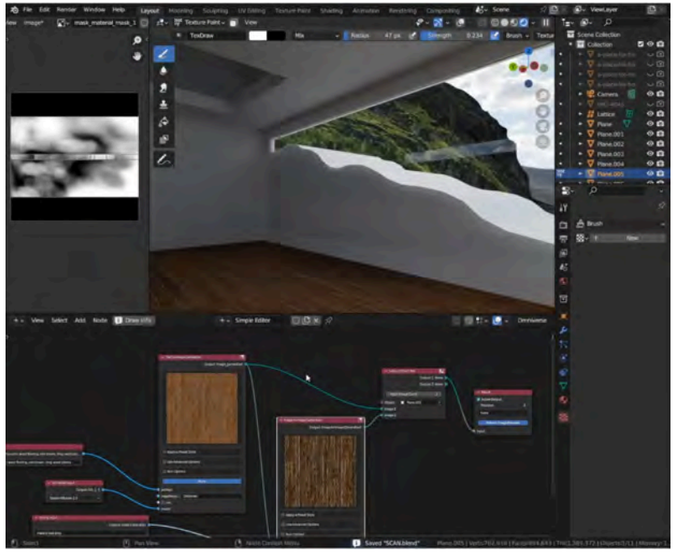
There are certainly areas of improvement and features laking in this current iteration of the node system, but with more time I hope to get it to a point in which the designer is able to, as freely and naturally as possible, generate complex and compelling materials.



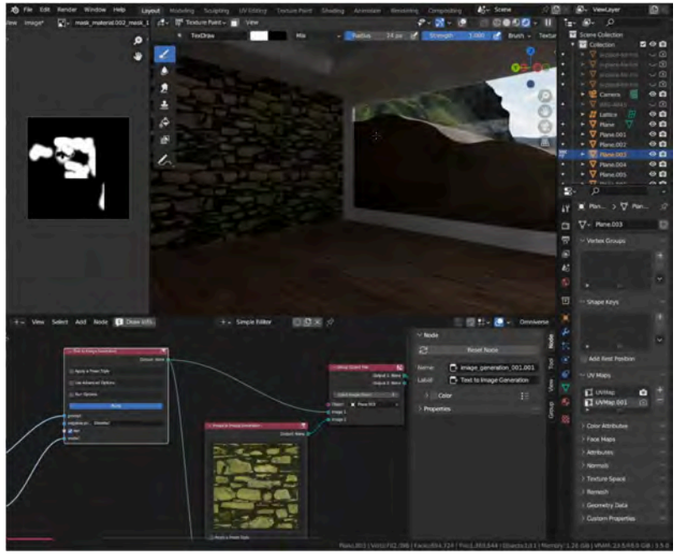
Generated material node system

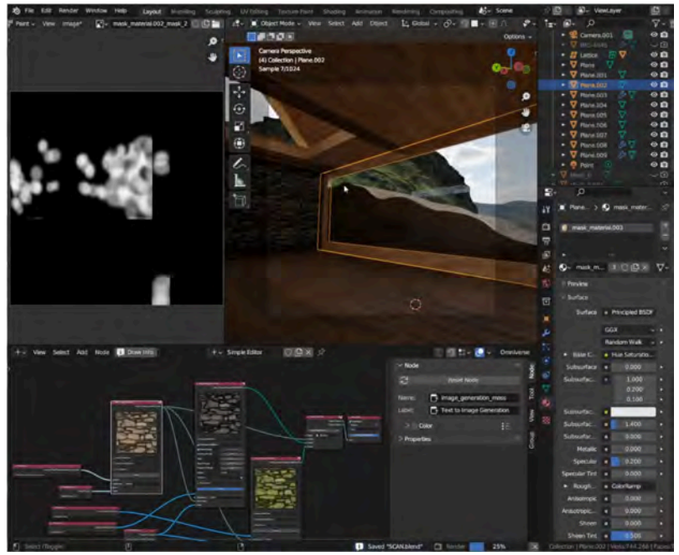


With a node network set up, values can be filled in to texture the different portions of the house. Here I have a prompt, in that is guiding a texture generation. That texture is then used in conjunction with an instructional prompt to produce a pair of textures. In this case a clean and worn wood flooring. These two textures are then applied to the floor with a Setup Object Material Node.

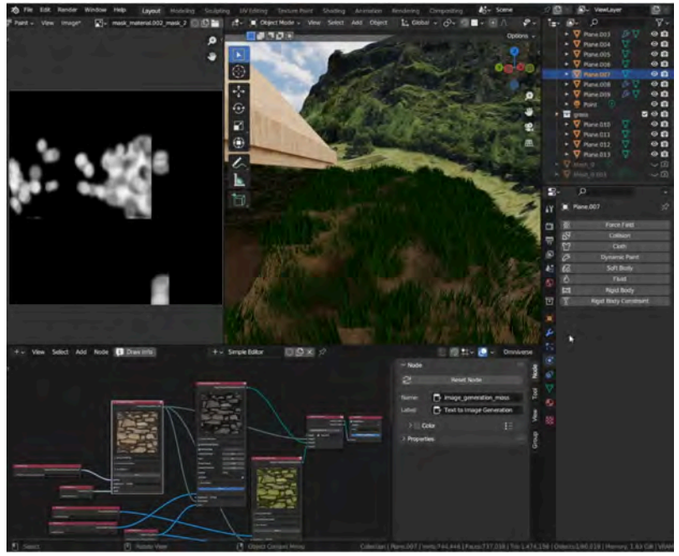


The same process can be used on the back wall and earth. Each object has paint masks auto generated and are being used to add variation to the floor and avoid a uniform application. The textures have embedded displacement which particularly on the stone wall makes a big visual difference.





The rest of the objects have materials applied, with the only changes needed to be made are the prompt and some quick UV remapping.



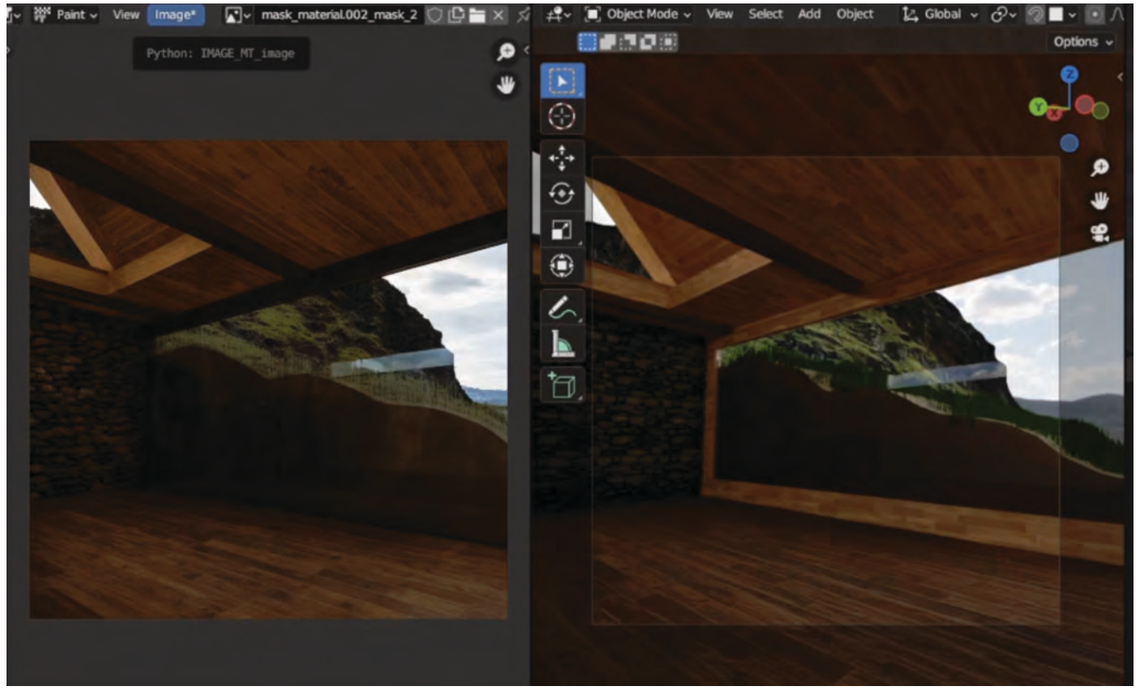
I decided to add a grass system to the mound of dirt. This is very low quality, but is useful in producing a proper silhouette for generating an edge map. For this purpose the same effect could likely be achieved with a few cutout grass png images.

REALTIME AI RENDERING

The other main software piece that I hoped to contribute to was Stable Diffusion viewport rendering. My solution to this for the time being, while under a tight time constraint, was to hijack the Blender render call and send updated information to a local Stable Diffusion server to produce and image in near realtime. In reality the updates take around 5 seconds on my machine, but I found this more than fast enough to consider this “realtime”.

Other implementations of Stable Diffusion in Blender could achieve something similar. But I believe both removing the need to press an update button and not locking the interface during generation, is actually quite significant in making the use of the tool not steal too much of the designers focus. The goal shouldn’t be to make changes and then wait for the response from the AI, “allowing” you to proceed with more changes. It should feel much more integrated and fluid.

To the right we see the Blender Cycles Viewport (far right) and next to it the realtime output from Stable Diffusion, making use of depth, edge, and color information. The output achieves a level of cohesiveness and realism in texture application that would take a significant amount of time and effort to achieve otherwise. It is filling in details that, at least at this point in the design process, are too specific for the designer to spend time on, yet they add a significant layer of dimensionality and weight to the decisions being made.

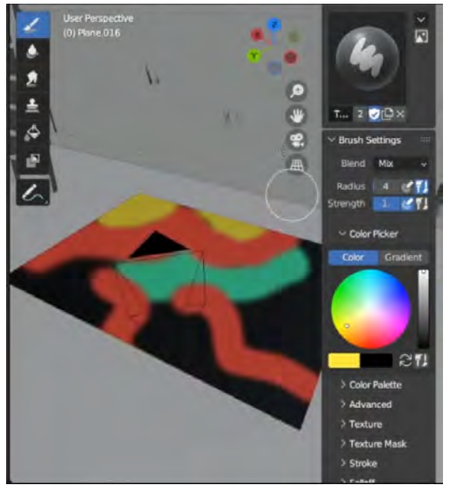


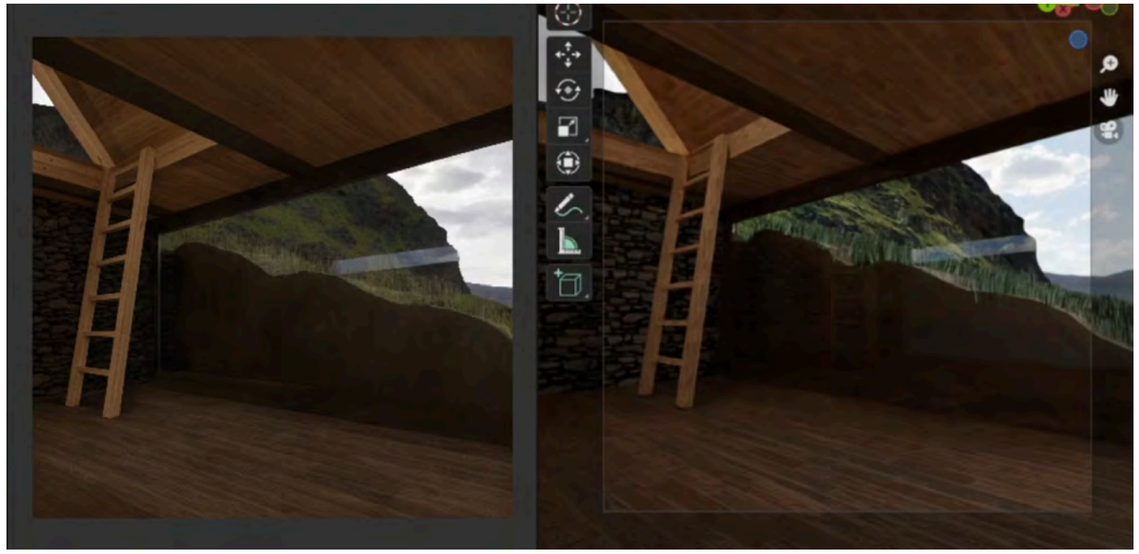
Here I begin to gradually add additional details to the scene, as well as make material adjustments to portions of the model.

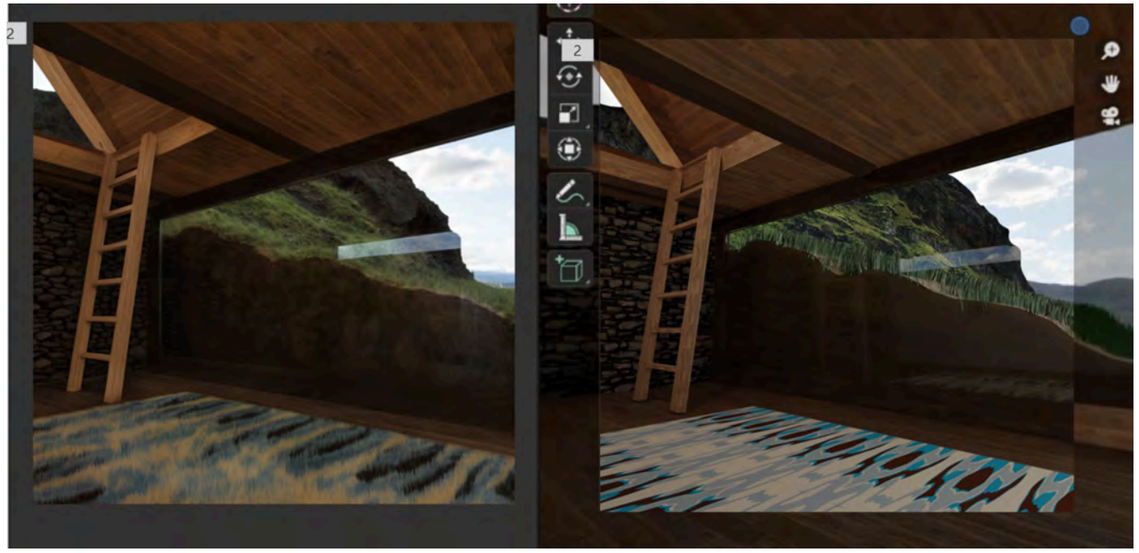
There is no masking happening here between generations. The consistency of results across the images can be attributed to the material work done in setting up the model. Here, every bit of additional information, results in more consistency.

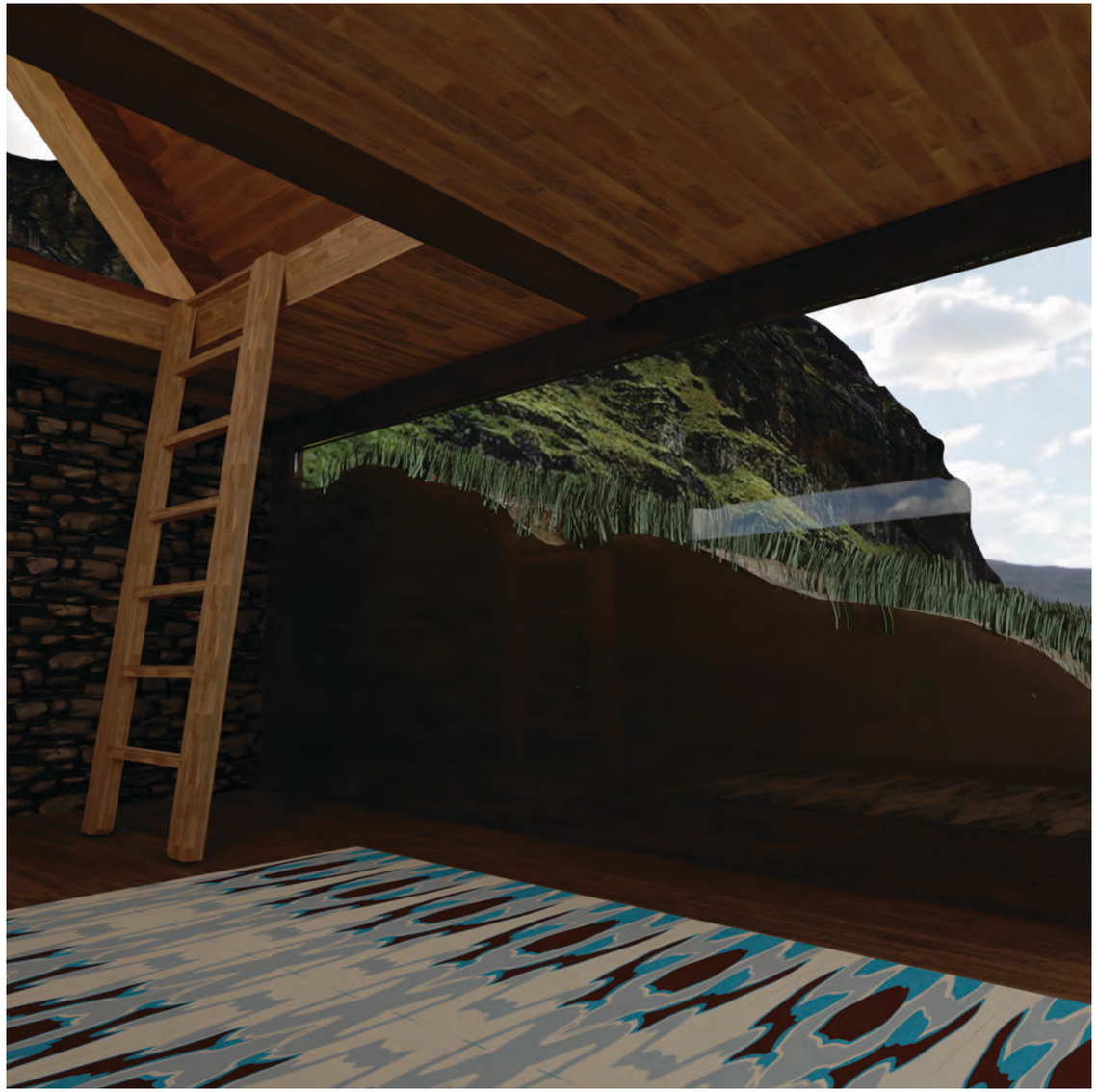
The carpet was an interesting and successful experiment. I began by painted colors on a flat plane. Then did some simple UV scaling to create the patterned look. Lastly, some hue adjustment to make it more palatable. Stable Diffusion did the rest.

The images before generation are well textured, yet the texturing distracts from the architecture in a way the AI pass does not. I find these details do a lot for understanding the space in context







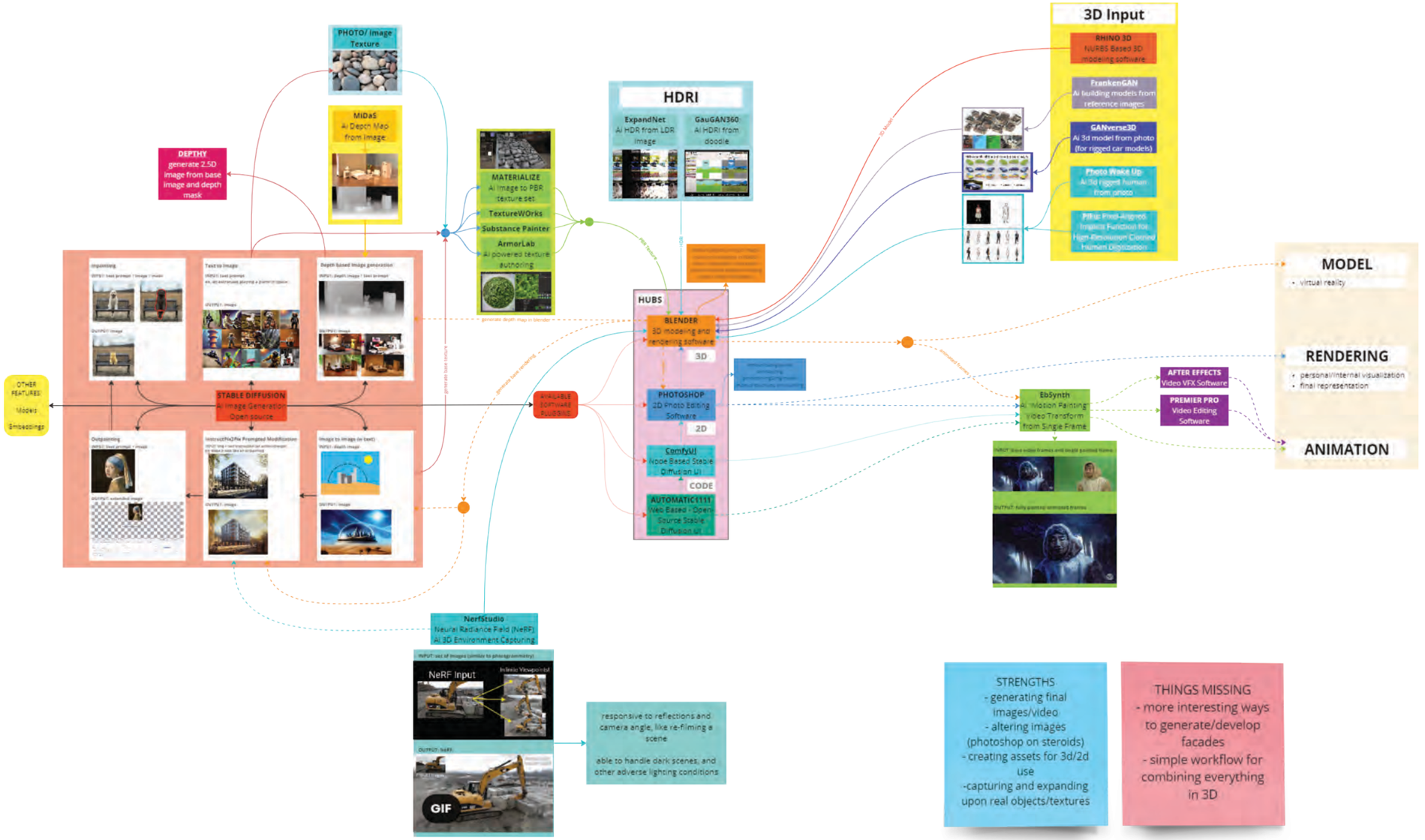


ADDITIONAL RESEARCH

Throughout the course of this research, which includes both the spring semester of 2023 and the one prior, in which I was doing pre-thesis research, the landscape of this technology has changed immensely. My research began really at the precipice of this technology's explosion in September of 2022.

In that time, I have explored a number of avenues to varying amounts of detail. Much of which was either too off topic to include in my final presentation, or not yet well enough developed.

Here I would like to discuss some of those explorations for potential further research and consideration. Additionally, I would like to go into further detail about my process leading up to the research just presented.



Here is a map of laying out the image generation landscape in January of 2023. The map focuses on the connectedness of different tools available, and how they may slot into a design workflow.

It was here in which I narrowed in on a series of software hubs that seemed most plausible for usability in a design workflow. This was based on the development being done at the time with the softwares, and the potential I saw for them to expand into.

There is a lot on this map, but the most important pieces are Blender as a 3D hub, Photoshop as a 2D hub, and Automatic1111 as the Stable Diffusion interface and API of choice.

This is also where I identified a set of holes in the current tools available.

An area of research that I did a bit of work towards, before determining it was too much in the context of the semester, was ways in which the noise field could be manipulated to produce better results.

Consider that the noise Stable Diffusion uses as a base may seem random to the human eye. However, the computer is picking up on indiscernible patterns in the starting noise that result in the gradual “diffusing” that is taking place.

We can see such a pattern when mirroring our starting noise.

Here both sets of images used the same prompt (“a rock”) and same starting seed, however the bottom set mirrors the noise pattern along the vertical and horizontal during the generation process.

In doing so the resulting image is significantly more symmetrical.

I believe that if one was to consider a similar process when working from a 3D model, in which the noise applied was reflected by specular surfaces like glass, it may result in more accurate images.

Essentially, the question being: can traditional ray tracing operations be used to enhance the quality of AI generations coming from a 3D scene?

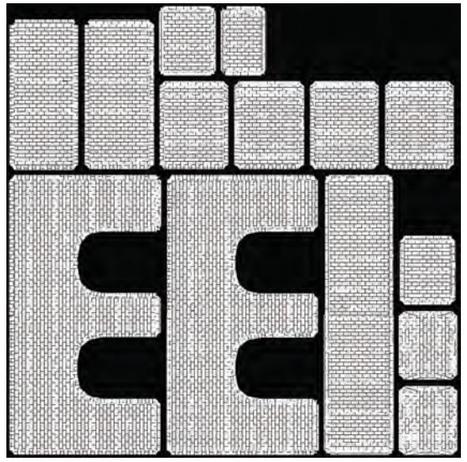
To some degree other released developments since, namely Control Net, may make this obsolete, or potentially the effect of such a technique may be indiscernible. We won’t know however until we try.

ALGORITHMIC ASSEMBLIES

An area of research that I did put a decent amount of time towards is Algorithmic Assemblies. This was research done in the fall of 2022 that revolved around the idea of not using tiled textures for material application.

Essentially utilize Stable Diffusion to generate full object textures. The method used procedurally generated brick patterns that would respond better than a UV map projection to topological conditions and uniquenesses in geometric form. Consider the specificity to which bricks are arranged on an arch or a doubly curved surface.

UV map of unrolled archway with stencil



UV map of AI texture from calculated stencil

Layers of algorithmic assembly logic on doubly curved surface

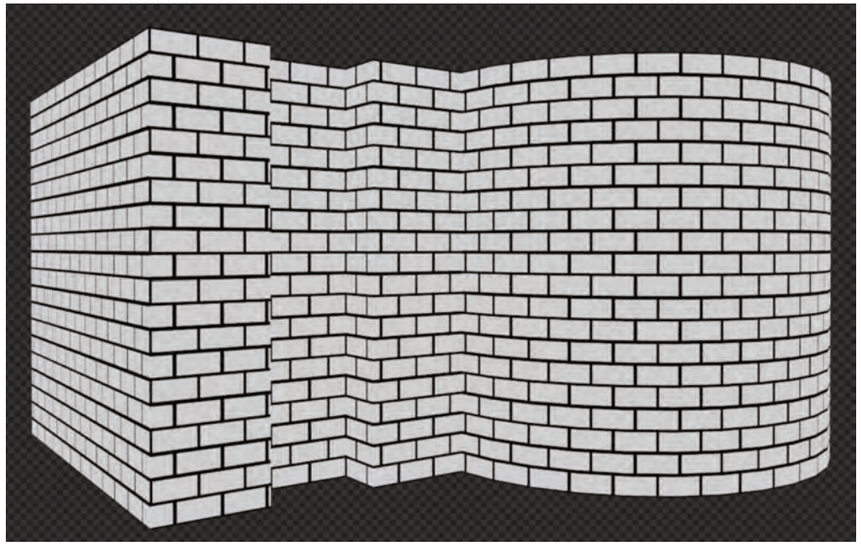
The algorithm would place vertexes and edges on a surface. Proximity from edges and the original mesh surface would then be used to paint a texture onto the object. This texture would resemble an outline of the desired final result. The unrolled outline would then be given to Stable Diffusion as input to produce a final texture.

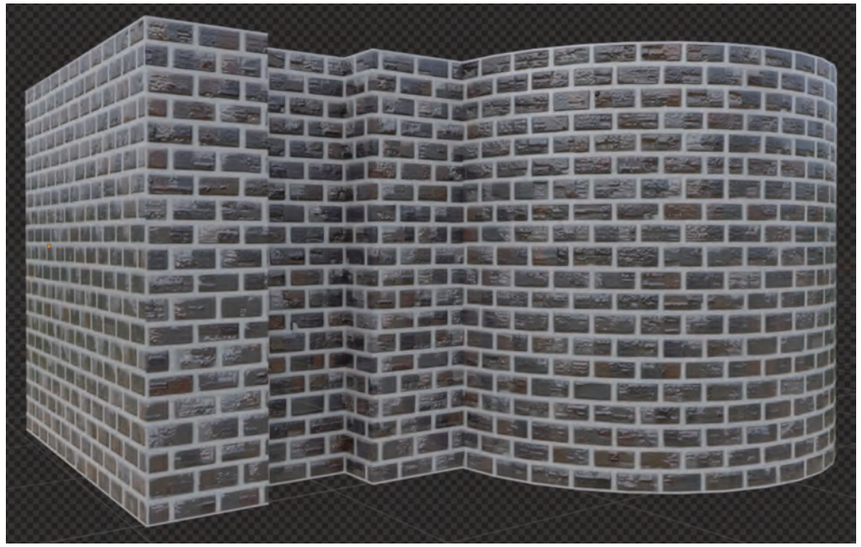
This method was decently successful, however it had some significant limitations in terms of complexity and size of geometry. Size in particular was a problem, because the resolution of the texture on the surface was a direct correlation. As the object got bigger, so did the size of the texture to allow for it to maintain resolution.

The other issue was, at the time, the method for turning the black and white outline into a colored result was not reliable, it became very difficult to maintain proper orientation of bricks on the mapping.

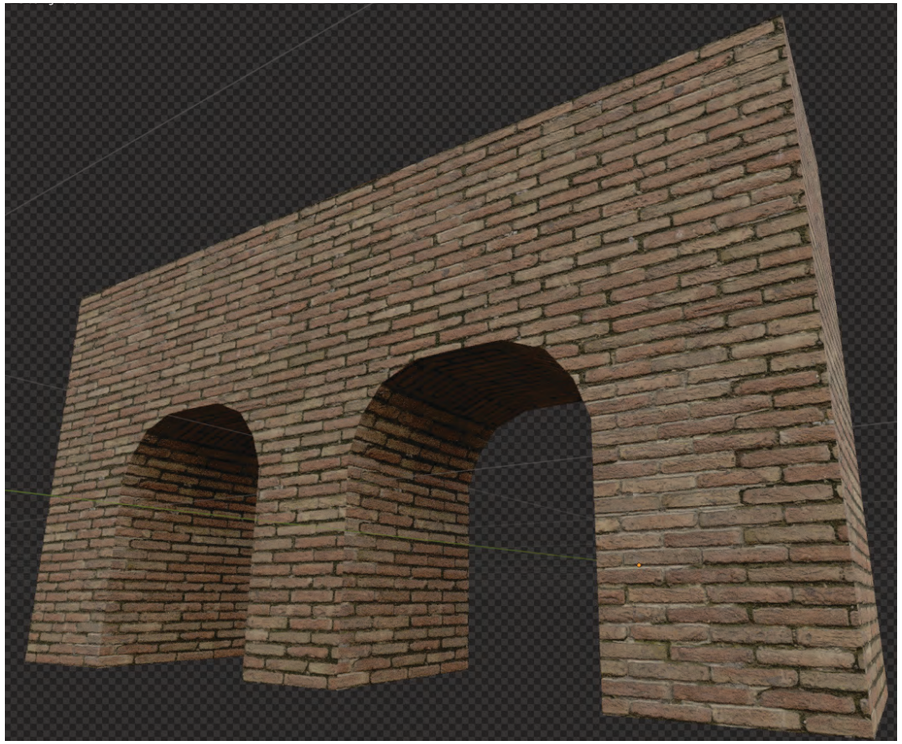
What would excite me to return to this is the advent Control Net. This would theoretically significantly improve the conversion between stencil and texture.

```
__author__="LabUser"
__version__="2022.10.05"

importRhino
fromRhino.Geometryimport*
fromRhino.InputimportRhinoGet
fromRhino.CommandsimportResult
fromRhino.DocObjectsimportObjectType
importrhinoscriptsyntaxasrs
fromscriptcontextimportdoc

fromoperatorimportitemgetter
importmath

defRunCommand():
  rc,objRef=RhinoGet.GetMultipleObjects("Selectobjectstobrick",True,ObjectType.Surface|
ObjectType.PolysrfFilter)
  ifrc!=Result.Success:
    return rc
  brick_height=rs.GetReal("BrickHeight",1)
  brick_width=rs.GetReal("BrickWidth",2)

  forobjinobjRef:
    #CreateMeshfrombrep
    brep=obj.Brep()
    ifNone==brep:
      returnResult.Failure

    jaggedAndFaster=MeshingParameters.Coarse
    smoothAndSlower=MeshingParameters.Smooth
    defaultMeshParams=MeshingParameters.Default
    minimal=MeshingParameters.Minimal

    meshes=Mesh.CreateFromBrep(brep,smoothAndSlower)
    ifmeshes==Noneormeshes.Length==0:
      returnResult.Failure

    brepMesh=Mesh()
    formeshinmeshes:
      brepMesh.Append(mesh)
    box=rs.BoundingBox(brepMesh)

    #NotRHINOspecific
    box_min=min(box,key=itemgetter(2))[2]
    box_max=max(box,key=itemgetter(2))[2]
    box_height=box_max-box_min

    brick_count_v=round(box_height/brick_height)
    brick_height=box_height/brick_count_v
    print(box_height,brick_width,brick_height,brick_count_v,brick_height)

    cut_planes=[]
    foriinrange(int(brick_count_v)+1):
      plane_z=x*brick_height+box_min
      cut_planes.append(rs.PlaneFromNormal((0,0,plane_z),(0,0,1)))
    contours=Rhino.Geometry.Intersect.Intersection.MeshPlane(brepMesh,cut_planes)
    contours_gu=[]
    forlineincontours:
      contours_gu.append(doc.Objects.AddPolyline(line))
    guide=rs.CurveStartPoint(contours_gu[0])

    count=0
    brick_count=0
    points_list=[]
    moved_points=[]
    translation=[item*int(brick_height)foritemin(0,0,1)]
    forlineincontours_gu:
      ifrs.IsCurveClosed(line):
        t=rs.CurveClosestPoint(line,guide)
        rs.CurveSeam(line,t)
      points,brick_count=divideCurve(line,count,brick_count,brick_width)
      points_list.append(points)
      points_m=[]

      forpointinpoints:
        points_m.append([point[0],point[1],point[2]+brick_height])
      moved_points.append(points_m)

      count+=1

    points_list.pop()
    moved_points.pop()
    contours_gu.pop(0)
    forpoints,points_moved,curveinzip(points_list,moved_points,contours_gu):
      forpoint,point_movedinzip(points,points_moved):
        param=rs.CurveClosestPoint(curve,point_moved)
        point_2=rs.EvaluateCurve(curve,param)

  doc.Views.Redraw()

defdivideCurve(line,count,prev_count,brick_width):
  length=rs.CurveLength(line)
  multiple=1
  brick_count=multiple*round((length/brick_width*2)/multiple)
  points=rs.DivideCurve(line,brick_count)
  patern=filterPatern(count,brick_count,prev_count)
  points=pointFilter(points,patern)

  returnpoints,brick_count

deffilterPatern(count,brick_count,prev_count):
  if(count%2):
    len_dif=abs(brick_count-prev_count)
    iflen_dif:
      repeat=prev_count/len_dif/4

      times=math.ceil(brick_count)/repeat/2.0

      patern1=[0,1]*int(repeat)
      patern2=patern1*2
      patern1.append(1)
      patern2.append(1)

      patern=patern1+(patern2*int(math.ceil(times)))
    else:
      patern=[0,1]*int(brick_count)
  else:
    patern=[1,0]*int(brick_count)

  returnpatern

defpointFilter(points,patern):
  keep_points=[]
  count=0
  forpointinpoints:
    ifpatern[count%len(patern)]:
      keep_points.append(point)
    count+=1
  returnkeep_points

if__name__=="__main__":
  RunCommand()
```





Automatic Projection



Topologically Arranged Alignment Passed Through AI

ABSTRACTION

An area of study that I felt the research I had done so far would yield a much more interesting response with a more in-depth exploration is abstraction.

This term can mean a lot of things especially in the context of Artificial Intelligence.

In its simplest form it may be stylization; Taking an image and altering its appearance to fit a particular aesthetic.

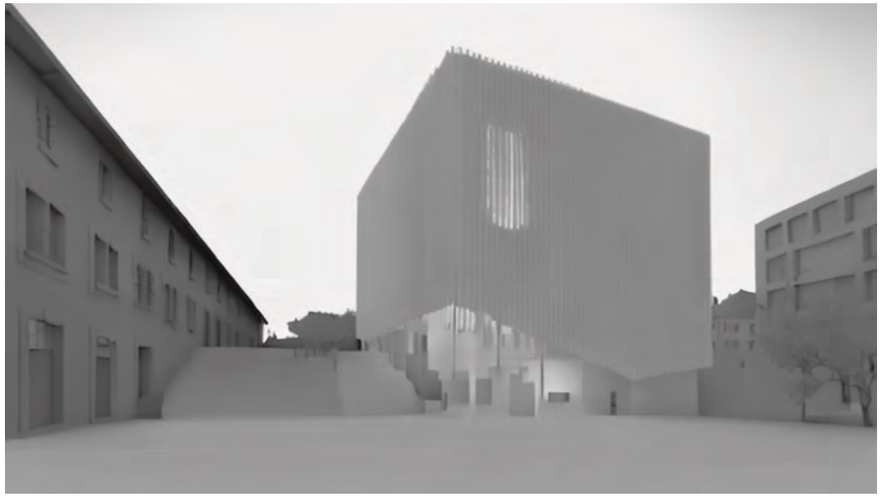
Here, the initial render is made to look like a painting, nighttime, a sketch and line drawing.

To take stylization a step further, we can take a render and change its season with instructed prompted such as “make it look like winter”.

Where this holds more design potential however is in removing layers of information. This could be like the line art produced on the prior page, or it could be instructing to “make it look like a digital model”.In doing so you are stripping the building of its material information and allowing the focus to be solely on the form and structure.

Say you are working within the context of an existing site condition. This could be very valuable in decoupling materiality from form to examine the broader operations and moves being performed.

MHNF Proposal | Localarchitecture



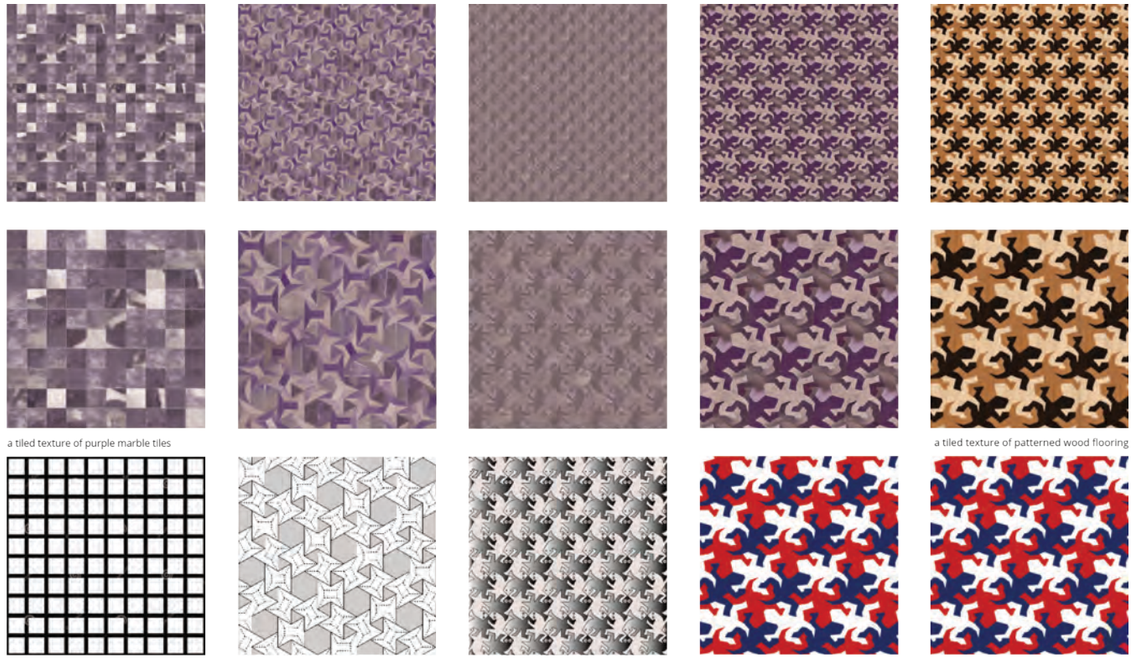
One level deeper, and abstraction begins to become quite fascinating. Here a hallway of the Martin House in Buffalo, NY. If we were designing this space, in early stages of design in particular, there may be a desire to avoid definitive material application.

This set of studies explore using other descriptive words for space to form an image. Here the particular materiality is not the focus, but rather capturing the feeling of being in a space that is “dark” or “ compressed” or “bright” or “reflective”.

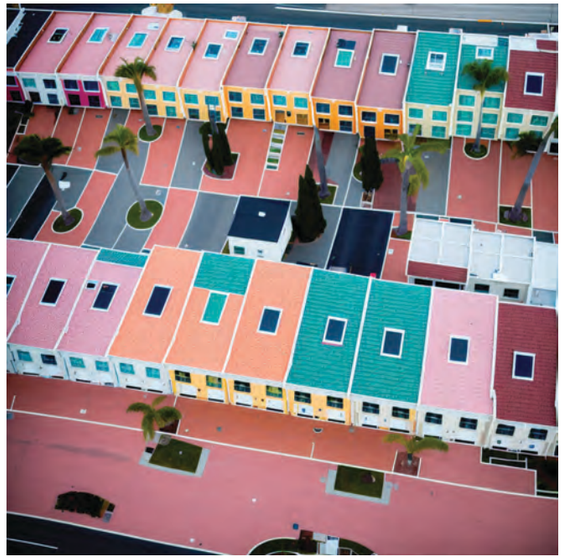
Further intricacy emerges from manipulating depth maps, collaging them, and utilizing physical model photographs in tandem with complex prompts, like quotes or theoretical texts, while also describing materials and desired spatial qualities. This crucial design stage allows for the establishment of a concept or direction, with a deliberate embrace of controlled ambiguity.

From initial sketches to detailed construction sets, we aim to explore the strengths and weaknesses of AI in architectural design.

Additional tests of the use of patterns for texture generation



Various additional imagery



FINAL THOUGHTS

Over the past eight months, my engagement with the AI community has illuminated a vast desire for versatile tools and personalized workflows. An increasing number of individuals, many working with open-source frameworks, share a common pursuit for design freedom, striving for innovative means to achieve this shared objective.

As my project and associated technologies evolved, I found myself harnessing my dual expertise as a designer and programmer to delve into overlooked aspects of the technology. This journey has been a testament to the potential of emerging tools – both mine and others' – for maximizing efficiency and effectiveness in design and development.

A critical understanding I wish to underscore is the inherent bias of these technologies, as they are trained on data they seek to mirror. Conventionally, we aspire to eradicate bias from our tools. However, the crux of my research has been to embrace and leverage this bias as an advantage.

It interests me greatly to consider where to take this research next. How about inserting a location and then a database can provide the necessary information to produce imagery accurate to culture and climate of that area. Or, running simulations to generate masks on surfaces of the way that dirt and moisture accumulate over time, taking into account sun exposure and climate.

In contemplating the future of these technologies, the direction we choose to steer them in will reflect our own ambitions and visions. By effectively employing these tools, I am hopeful we can enhance our capacity for expressing our unique designs and personal creativity.

Of course these technologies and workflows are not limited to the architecture discipline. I'm well aware that advancements in this tech will largely come from developments in other industries. Whether that be virtual world building for film, games, or VR, or other industries such as product design, interior design, or fashion. Fortunately, the flexibility of the software means advancements in one industry will likely correlate to advancements in all areas.

This again speaks to the democratization of information and technologies. It is important this progress does not happen behind closed doors.

Over time, my journey into the realm of digital rendering has cultivated an obsession with the minutiae and an appreciation for the beauty in imperfect images. This digital obsession has transpired into my everyday life, leading me to observe the material nuances and distinct details present in the architectural facades on my campus.

The character and vitality of architecture, I believe, stem from its unique materiality, which is in danger of being overshadowed by mass production. This has resulted in a monotonous echo in our creative designs, mirroring our standardized tools and workflows. However, the advent and breakthroughs in 3D printing and robotic assembly open up promising avenues for more personalized construction, challenging the constraints on our creativity imposed by the prevalent design assembly line.

Yet, this potential for personalization is often not mirrored in our design tools, imposing an artificial cap on our creativity. However, a paradigm shift occurred eight months ago with the release of DALL-E 2, triggering a revolutionary wave of generative AI, sending ripples of both excitement and fear across society.

Indeed, these powerful tools come with great responsibility. Thus, the question is, what is our duty in wielding this power, and what do I aim to achieve? I perceive the transition towards mass production and the advent of AI as an opportunity for intervention. My goals will continue to include enhancing the quality and speed of production, ensuring fine control over results, and maintaining a familiarity with processes and tools to keep the software usable. And hopefully, soon enough I can polish and release the tools I've been developing, and in that way contribute my part.

The AI-driven tools and workflows we are developing have the potential to redefine the boundaries of architectural design, delivering a degree of control and personalization that has been sorely lacking. However, this revolution in design methodology doesn't eliminate the need for human creativity and vision; rather, it augments and elevates it.

It is through our collective engagement with the AI community and our shared enthusiasm for control over outputs and design freedom, that we can continue to innovate and explore the untapped niches of technology. Utilizing these technologies, we have the potential to design more, instead of less, and to retain agency in design - this is our ultimate responsibility and ambition.

AUTOMATIC1111. "AUTOMATIC1111/Stable-Diffusion-Webui: Stable Diffusion Web UI." GitHub. Accessed May 24, 2023. https://github.com/AUTOMATIC1111/stable-diffusion-webui.

Bernstein, Phil. Machine learning: Architecture in the age of Artificial Intelligence. RIBA Publishing, 2022.

Documentation – arm developer. Accessed May 24, 2023. https://developer.arm.com/documentation/102696/0100/Texture-channel-packing.

Klanten, Robert (ed), Sven Ehmann, and Floyd Schulze. Visual storytelling: Inspiring a new visual language. Berlin: Gestalten, 2012.

Klanten, Robert, and Johannes Schardt. Data flow 2: Visualising information in graphic design. Berlin: Gestalten, 2010.

Leach, Neil. Architecture in the age of artificial intelligence: An introduction to AI for architects. London: Bloomsbury Visual Arts, 2022.

Lllyasviel. "Lllyasviel/Controlnet: Let Us Control Diffusion Models!" GitHub. Accessed May 24, 2023. https://github.com/lllyasviel/ControlNet.

McCandless, David. Knowledge is beautiful. London: William Collins, 2014.

"Tutorial: Architectural Rendering with Depth-Guided Diffusion Models." YouTube, January 19, 2023. https://www.youtube.com/watch?v=CHfCT2lqNdo&ab_channel=ArielNoyman.