

# ***Level Design***

## ***In Pursuit of Better Levels***



# Table of Contents

◆ <b>Planning</b> ◆	<b>4</b>
❖ Concept process ❖	4
❖ Metrics ❖	6
❖ Prototyping & Blockout ❖	9
✂ Structural geometry ✂	11
✂ Form follows function ✂	11
✂ The Backdrop ✂	12
✂ Check the scale ✂	12
✂ Empty space ✂	13
✂ Testing the level ✂	14
✂ Documentation & important questions ✂	14
✂ Iteration & playtesting ✂	15
◆ <b>Navigation &amp; distinction</b> ◆	<b>15</b>
❖ Perception ❖	15
❖ Readability ❖	19
❖ Orientation ❖	26
✂ Landmarks ✂	26
✂ Signposting ✂	30
✂ Vantage point ✂	31
❖ Guidance ❖	32
✂ Overestimating guidance ✂	32
✂ Attention & saliency ✂	34
✂ Visual language ✂	37
✂ Nope zone ✂	40
✂ Guiding with AI ✂	41
✂ AI as visual & audio language ✂	42
✂ Breadcrumbs ✂	43
✂ Teasing ✂	46
✂ Prospect & refuge ✂	47
✂ Contrasting paths ✂	49
◆ <b>Pacing</b> ◆	<b>50</b>
❖ Pacing tools & tricks ❖	50
✂ Flow in Singleplayer ✂	50
✂ Gates & Valves ✂	51
✂ Loops ✂	52
✂ Recycling levels ✂	53
❖ Planning & structure ❖	54
✂ Intensity, variety & time ✂	54
✂ Beats, charts & timelines ✂	55

✂ Non-linear pacing ✂	57
◆ <b>Enhancing the experience</b> ◆	<b>59</b>
❖ Choices ❖	59
❖ Enhancing with audio & visuals ❖	62
✂ Color, lighting & mood ✂	64
❖ Narrative ❖	66
❖ Emotions ❖	68
❖ Spicing it up ❖	69
✂ Wow moments ✂	69
✂ Reactivity ✂	70
✂ Exploration ✂	71
◆ <b>Designing for combat</b> ◆	<b>72</b>
❖ Vantage point ❖	72
❖ Combat Spaces ❖	73
✂ Combat fronts ✂	76
✂ Flanking ✂	78
❖ Cover ❖	79
✂ Cover placement ✂	83
❖ Enemies & encounters ❖	85
✂ Enemies & cover ✂	85
✂ Spawning & tracking enemies ✂	86
✂ Encounter design ✂	90
✂ Allied NPCs ✂	92
❖ Line of Sight for SP & MP ❖	93
◆ <b>Multiplayer map design</b> ◆	<b>97</b>
❖ Overview ❖	97
✂ Readability ✂	99
❖ Level structure ❖	101
✂ The Circle ✂	101
✂ Figure 8 ✂	102
✂ Clash points & Double Clash ✂	103
✂ Tug of War ✂	106
✂ Defence ✂	107
❖ Balance ❖	109
✂ Symmetry & Asymmetry ✂	112
✂ Time ✂	114
✂ Spawn points ✂	114
✂ Potential Player Positions (PPP) ✂	115
❖ Flow ❖	117
◆ <b>Briefly on accessibility</b> ◆	<b>121</b>

## ◆ Foreword ◆

The goal of this write-up was to collect less commonly found knowledge from veteran game developers as well as other relevant information. Occasionally I will also provide my own suggestions and observations where it's relevant. But I don't want to take any credit for the knowledge contained in this document, I merely collected it.

While I stand behind everything in this document as good guidelines to follow, it's up to you to decide if and when to apply it or not.

If you feel anything contained in this document is wrong or you have any questions, feel free to contact me on twitter @TychoBolt

// Alex K



## ◆ Planning ◆

Good planning at the beginning (and during) a project can save both you and your team a lot of headache. In this section I will go over some tips that can help make the level creation process go more smoothly.

### ❖ Concept process ❖

The first thing you should do is to nail down the *Restrictions*, *Goals* and *Context* of your level.

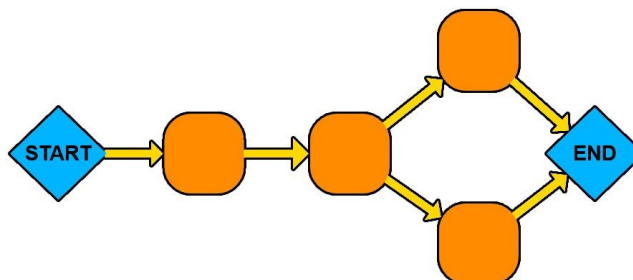
- **Restrictions:** Things you have to do. Often not decided by the level designer.
  - For example: *Introduce and teach a new mechanic (item). Can't take place on Earth. Needs to introduce and use a new enemy type throughout the level. Needs to meet up with friendly NPC at the end of the level.*
- **Goals:** Things you want to do. This could be a levels Theme, Challenges or the like.
  - For example: *Lots of verticality and should create a sense of acrophobia (fear of heights). Alternative routes. Platforming.*
- **Context:** Things you need to consider.
  - For example: *7th level in the game that roughly takes place during the midpoint. There are already X and Y type of levels planned, how will your level be unique.*

The reasons for this are:

- ★ It creates a starting point so you avoid the “blank canvas” problem.
- ★ You’ll reduce the risk of having to rework your level part-way to make your level work.
- ★ It helps you create structure and focus from the beginning.
- ★ It helps the rest of your team plan their work around your level more effectively.

Then it usually helps to create a flowchart or timeline so you can nail down when certain things should take place.

Some things will probably logically or have to take place in a certain order, while some things are more fluid. For example, if an NPC is supposed to give you a mission briefing it doesn't make sense for the player to meet them at the end of the mission.



After you've nailed this down, some good questions to ask yourself are:

- **Where does the level take place?**  
➤ *A forest? A ghost town? In France?*
- **When does the level take place?**  
➤ *Noon? At night? After a big battle?*
- **What are the mechanics of the game?**  
➤ *The level should facilitate the gameplay the game has, so make sure you understand and stay up to date on them. Making a level in a vacuum and then trying to shoehorn in the game's mechanics is generally not a good approach.*
- **Why will players remember your level?**  
➤ *What sets your level apart? What are the highlights that will leave an impact on the player?*
- **Does the location fit the gameplay?**  
➤ *If the level is focused on long-ranged sniper combat then having it take place indoors with tight corridors is probably not ideal.*
- **What is the "story" of the location?**  
➤ *What kind of place is it? What happened there? Who or what lives there? The things that makes your level feel more believable and alive.*
- **What will I need to communicate to my team?**  
➤ *Is there anything you would need to communicate with your programmers, artists or sound department, etc. about? What do you need to know so your artists can make the level pretty and what do they need to understand so the gameplay experience remains intact?*
- **Is your level possible?**  
➤ *Can the level be made in terms of the engine, time, asset creation, etc? Can you and your team actually make the level?*



### **Critical & Golden paths**

Critical path = The quickest and most direct route to beating the level.

Golden path = The designers 'preferred' route. It's the one you expect most players to take and the one which will offer the optimal experience.

Don't forget to predict and plan for these early so your level flows well and makes sense.

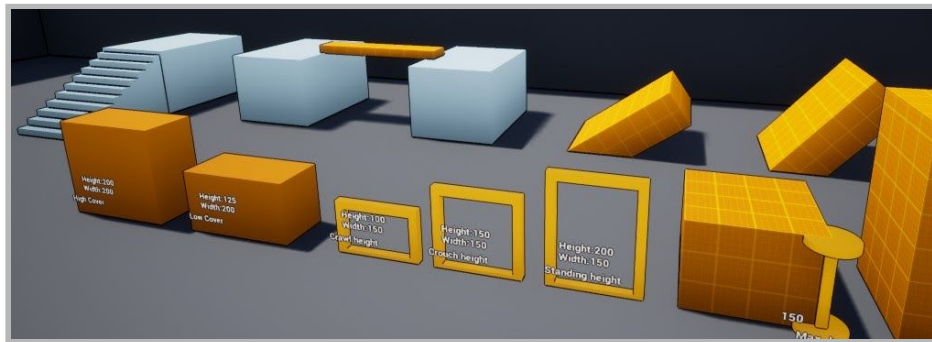
But also don't forget to test out your level by following *neither* of these paths.

## ❖ Metrics ❖

Nailing down all the relevant measurements and documenting them (and later updating them when necessary) is a very important step.

- It allows you to test and figure out the proper values in a quick and dirty setup.
- It helps you (and your fellow level designers) ensure that things like cover, platforming jumps and the like are always correct from the beginning. Meaning you won't for example accidentally create impossible jumps forcing you (and potentially artists) to re-do parts of the level.
- If for example an environment artist comes in to replace your placeholder assets it allows them to be better aware of what metrics are important for gameplay.

Usually it's a good idea to collect all of these metrics inside of a 'playground' level often called a "Gym" (see image below for example).







The Gym is where you place boxes and measure out these metrics so they can be tested and serve as reference during development.

While not necessary, it's a good idea to create either meshes with these fixed metrics or dynamic tools that can act as 'rulers' (way more flexible).

Said tools should be dynamically updated with their measurements and data to make the process of using them quick and easy for everyone on the team.

Below are example tools made in Unreal Engine 4.



	<p><b>Box/Frame</b> Useful for measuring windows, door frames, cover boxes, etc.</p>
	<p><b>Sphere</b> Useful to quickly measure nearest distances, AoE, etc.  Can be very handy to ensure your cover is spaced properly, ideal weapon ranges and so on.</p>
	<p><b>Notes</b> Notes to help you remember or to communicate important things to the rest of your team. <i>“Don’t move these boxes”,</i> <i>“Add explosion VFX here once it’s created by artist”,</i> etc.</p>
	<p><b>Spline/Path</b> Useful to measure total distance along less linear paths. Can be especially handy in Multiplayer level design to ensure that each teams side of the level are balanced.</p>

So what kind of metrics should you have? Obviously it heavily depends on what kind of game you’re making, but below are some examples:

**World**

- Hallway/Room width & height (roughly 2.7 meters to 3.6 meters can be good)
- Stair scale
- Window & Door sizes
- Various Cover sizes (low, high, etc.)
- Minimum distance between Cover (roughly 2-3 meters)
- Main (roughly 5 meters in width) and Side paths (roughly 3 meters in width)

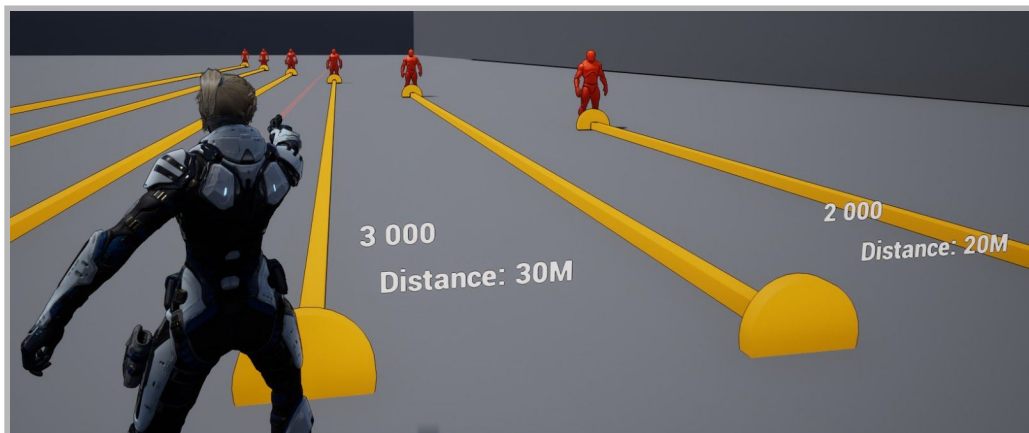
**Player, movement & other characters**

- Character dimensions (in different positions like standing, prone, crouching, etc.)
- Jump height and distance (when standing, running, etc.)
- Maximum step height (what characters can just walk across)

- Max slope degree (before the characters start to slide or can't walk on the surface)
- Max fall distance (before taking damage or dying)

### Items & abilities

- Weapon & ability (effective) ranges
- Combat engagement distance
- Grenade throw distance



*Example of how you can check combat engagement, weapon range, etc. using metric tools. This can allow you to get a feel for where combat should take place and the like.*

Also don't forget to document more unique cases that can still be relevant, like how far a player can throw a grenade if they use a jump pad and double jump, if it could for example affect the balance of one of your levels.

Likewise, don't forget that you might need to document various metrics for your AI characters (cover distances, navmesh, jump height, etc) so they can properly function in your level.

Additionally, something that is important to keep in mind is that you often can't apply real world proportions to common things like doors, windows or even ceilings. It can very easily look and feel cramped.

You also have to consider how the camera will function (3rd person games tend to require greater ceiling height) in the environment.



*Door on the right uses close to real world scale, while the door on the left is scaled up.*

## ❖ Prototyping & Blockout ❖

So now that you've got a general idea of what you want to do with the level, you've got (at least some) of the metrics down and you're a bit more familiar with the basic mechanics of the game it's time to start making our level.

The eagerness to just jump into the engine and start blocking out a level can often be felt at this stage. While tempting, it's typically a good idea to plan the level out a bit more and map it out beforehand.

Everyone's process is different, but the one I'm going to describe is one I feel should work for most people.

### 1. Create a bubble diagram or flowchart

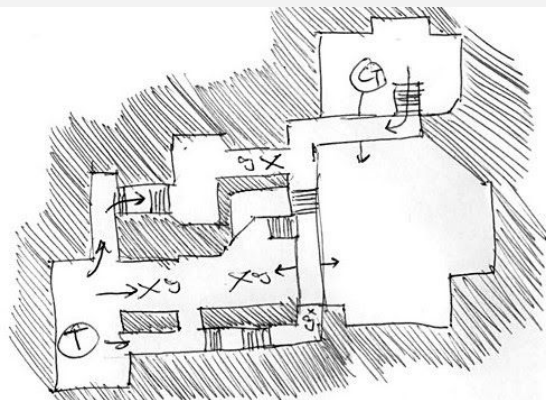


Basically, think of all the rooms/areas in your level, place them into 'bubbles' and connect them in the way they'll be connected in the level.

The goal here is to get a sense for what areas your level will have, how they connect and how the player will flow through the level.

Here you could also loosely plan events like when the player meets an NPC, fights enemies or where they can find a key.

### 2. Draw up a map (not final) that roughly covers your entire level

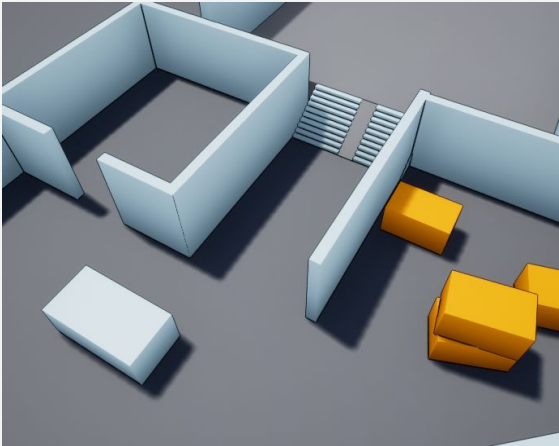


The idea is to create a more detailed version of the bubble diagram that gives a more accurate image of how the entire level will look and connect.

Planned combat encounters, platforming sections, etc. should be fleshed out a bit and you should indicate roughly how many enemies and the like each room will have. Here you can also write down some notes like *"The player will fight 1 wave of enemies and a turret, then 1 wave when the player has the turret"*.



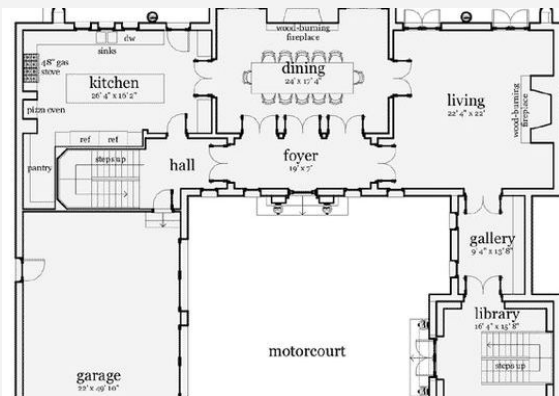
### 3. Jump into the engine and place big blocks and meshes to get a sense of space



The keywords here are “quick” and “dirty”. This is not supposed to be pretty, final or even something you will re-use. While it doesn't have to be exact, try and stick to the rough scale you had in your map as well as any metrics you already have.

This is to get a quick sense for how your level would roughly look in-engine so you can more easily visualize it, spot potential issues and get a sense for the scale of the level.

### 4. Go back to your map, make adjustments and make it more accurate



Make adjustments to the map based of the observations made in step 3. Try and make it more accurate and to scale.

It's important to not that this is not the “final” version of your map, just a solid starting point. This is also the point where you can more easily communicate and get feedback from your team with said map (and rough level from step 3).

### 5. Start blocking out the level using your map you drew up as a starting point



This is the point where the cycle of blocking out your level, testing it and iterating on it starts.

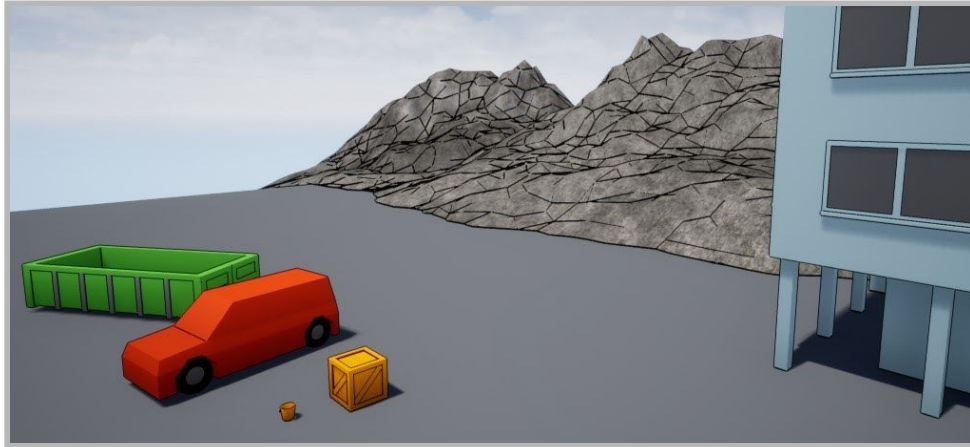
Also try and get feedback early and often on your level. Don't wait until you feel you've gotten it really 'polished' before receiving feedback, otherwise you might end up having to rework large portions of your level which could've been avoided with early feedback.

## ✂ Structural geometry ✂

So now that you've started blocking out your level, the first step would be to just place out any of the larger more rigid surfaces and structural geometry your level has.

This means any buildings, caves, mountains, pits, etc. and not things like cars, benches or other non-rigid objects (often called "props").

This is because at first you just want to get the backbone of your level created, the thing that is less likely to change or move around than something like a car or a "decal".

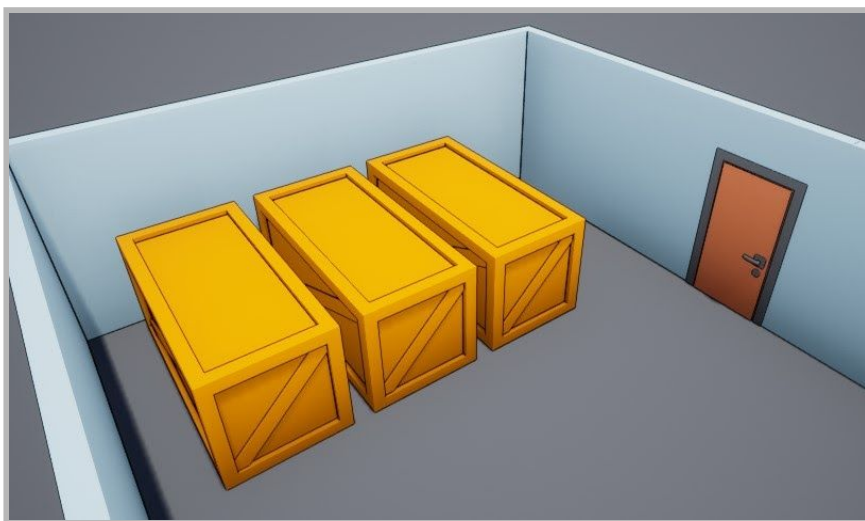


*Non-rigid "Props" on the left, rigid structural geometry on the right.*

## ✂ Form follows function ✂

While we'll go into this in more detail in a later chapter, it's important to also consider how "form follows function" at this stage. That things in your level contextually makes sense.

A diner obviously has a bathroom, pipe systems shouldn't connect to nowhere and most buildings don't consist of a bunch of hallways.



*How did these giant boxes get into this room with such a tiny door?*



## ✂ The Backdrop ✂

As you're blocking out your level, also consider what the backdrop (non-playable area) will look like.

While an intricate and beautiful backdrop can have great impact on the players experience, it will also be more 'expensive' in terms of performance and workload for your team.

So while indoor environments will generally be 'cheaper', things like windows or if the player travels outside of the building it would most likely demand extra work and performance.

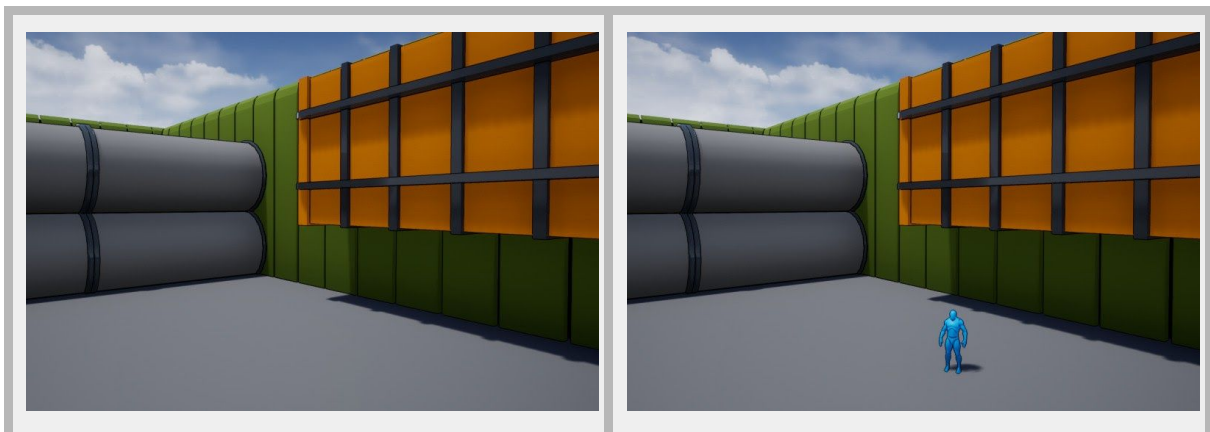


## ✂ Check the scale ✂

As you're blocking things out, remember to also keep an eye on the scale of your level.

Just looking at it or walking around in first person can be deceiving without objects to help us give a sense of the scale of the world.

You don't want to end up spending a lot of time creating an area that is either too small or too big for the requirements of your level. Improper scaling can look very weird (feeling like "something is off") and hamper the believability of your level.



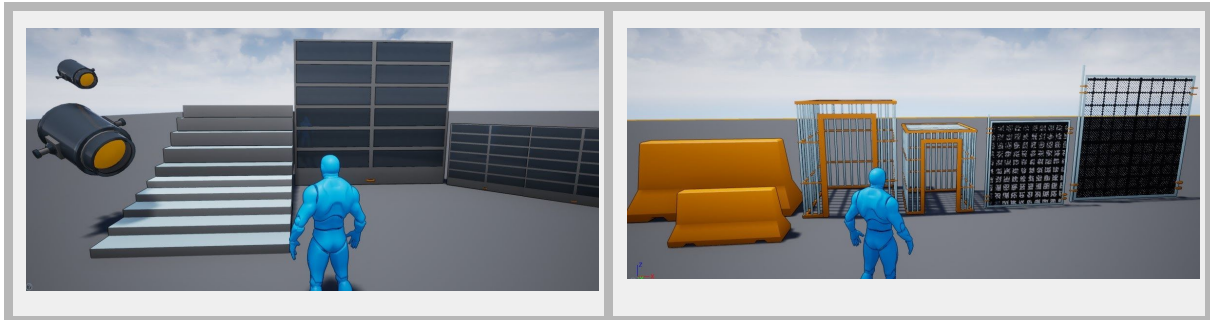
*Notice how in the image on the left we don't get a good understanding of the scale of the environment?*

*Meanwhile on the right when we place a character reference in the scene we notice it's quite large!*

*You might end up creating a space that might not work once you start placing tables, cover, etc. in them, resulting in you having to rework your level needlessly.*

This also applies to re-using assets, but scaling them differently. Especially in close proximity and if said assets are more distinct.

The player will get used to assets being a certain scale and when their perceived scale of an object is off, their mind will notice it.



*Some examples of how scale can make things look a bit strange.*



On the other hand, scale can intentionally be used to give a more surreal feeling. In the game *Dusk* the player enters a door and emerges from a hole finding themselves having been shrunk to the size of a mouse!

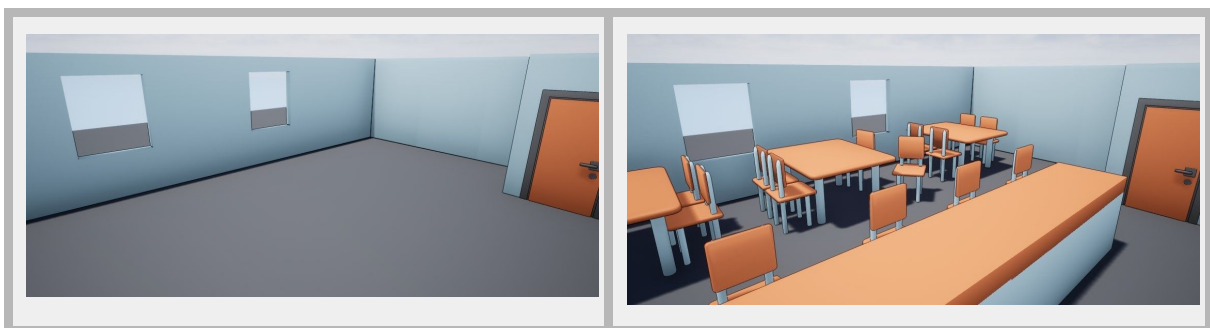
◀ *Dusk* (2018)

### ✦ Empty space ✦

Even if you don't have any particular design intention for a specific space, it's worthwhile to still place out some props just to check how it looks once it's "furnished".

For example, imagine you designed a cafeteria space that the player is just supposed to move through. Then imagine if the space isn't big enough for the environment artist on your team to create a properly looking cafeteria once you're done with your level.

Try and minimize the risk of either you or someone on your team having to re-do their work or having a tough time when it can be prevented.



*Just placing out some simple props can help you see if the space will work once it's "furnished".*

## 🔧 Testing the level 🔧

Once you get further along in your level there will eventually come a point where you need to test it with the new systems, AI, etc. that's created by your team.

For AI this could be how your AI partner moves in the environment and if they can perform actions like boosting the player over a ledge and the like.

For enemies this can involve checking so they use cover properly, if during a stealth segment they can detect the player properly (not hear or see them when they shouldn't), being able to open doors, etc.

Basically anything the AI would need to do in your level should be tested.

Testing so the camera behaves properly in your level is also a good idea, especially if it's a non-first-person camera.

Don't assume anything works 100%. This is not you 'expecting' your team to screw up or anything. It's to help them and the rest of your team's minds be put at ease, as well as avoiding someone having to fix tons of stuff later when they might be busy with other stuff.

Finally, try and set up a system so you can more easily test your level. If you have to replay a 20 minute section every time you want to check a 1 minute long segment or there isn't a way to easily test your level in a later build (with a level select or the like) you will lose a lot of time.

## 🔧 Documentation & important questions 🔧

Remember to document everything that's important about your level. This is so both you and your team is aware of and can read up on everything that's important about your level. This could be goals, interactable objects, events, a summary of a playthrough of the level, puzzles, secrets, where narrative will happen, etc.

Creating asset lists for materials, VFX, sounds, meshes, etc. as well as potential performance related information is also a good idea to document.

The documentation step is important since it allows you to scope your level properly and to reduce the risk of potential problems arising during development. So spend a little extra time on it to make sure it has the information that's required.

But also don't forget to ask your team important questions that can affect your level. Things like how will the checkpoint & save system work? How and when will control (if ever) be removed from the player? How will achievements be implemented? How will modular assets work? Etc.

## ✂ Iteration & playtesting ✂

Playtest early and often. It's easy for you to become blind to the problems in your level. Playtests should drive the iteration, not the other way around. It's easy to get into the mindset of "Just wanting to polish it a little bit more before a playtest". Playtesting is to learn what you need to change. The later and less frequently you playtest, the more expensive the needed changes could become.

Playtesting is a science in and of itself, but what is important to remember is that the negative feedback is by far the most important. Several playtesters might comment on the nice composition, but then two (or even just one) playtester comments on difficult jumps. Do not embrace the praise of the composition and the idea that "Well these other playtesters didn't complain about the jumps so it's not a problem". From the psychology (cognitive biases) of the playtesters (and yourself) to just the small sample size, there are tons of factors that can muddy your feedback. So any potential problems that arise needs to be examined and potentially addressed.

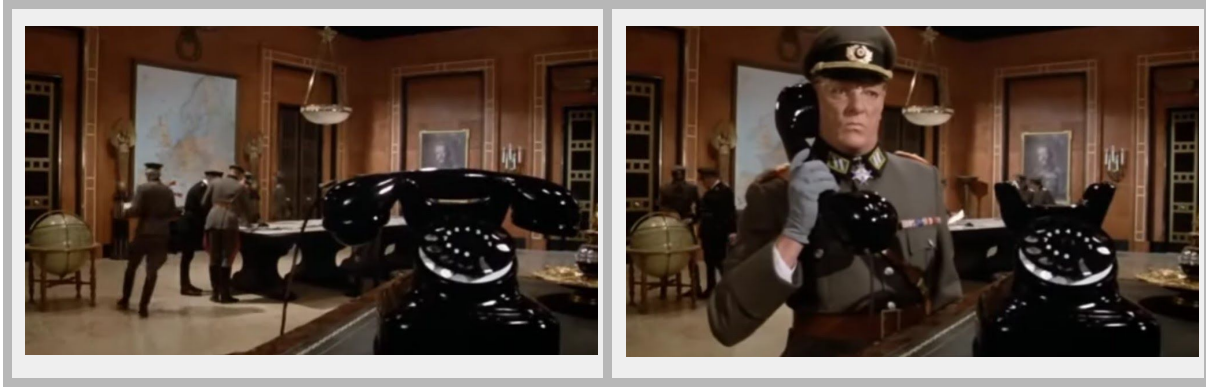
## ◆ Navigation & distinction ◆

This section focuses on ways we can guide the player and to make levels more readable. Since a lot of these topics are connected, some of it can end up overlapping. So while they should all be used together, they're broken up to make it more organized.

"Distinction" is basically how you make something noticeable to players. How to increase readability, player awareness and how to make players see what you want them to see.

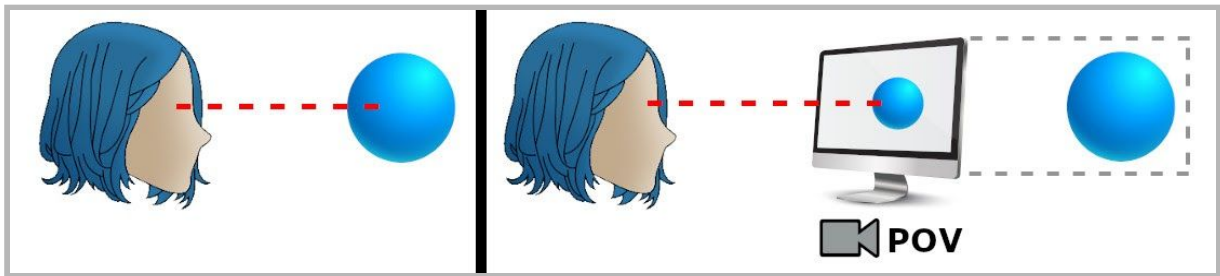
## ❖ Perception ❖

While humans have good depth perception and sense of scale, it doesn't directly translate 1:1 when it comes to games. There are a number of reasons for this, but for the sake of brevity I'll try and condense it (but I recommend reading up on it if this interests you). The reason why this is important to know is because if the player doesn't get a good sense for scale and depth it can result in missed jumps, confusion, etc.



*A gag from "Top Secret!" (1984) that showcases how perception and scale can play tricks on us.*

The easiest way I can think of to explain why this is harder to gauge in a movie or a game is with an image (see below).

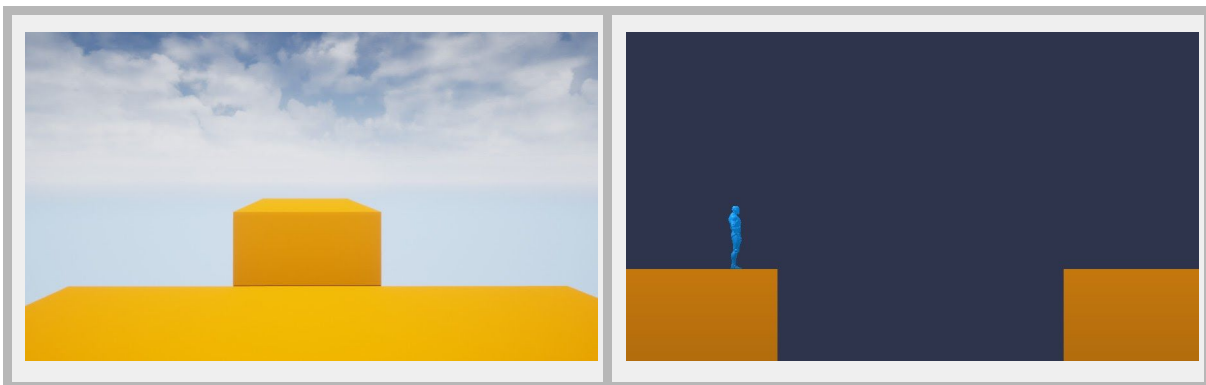


In the real world you're looking directly at an object with both eyes using your own position in the world as reference.

In a game you're looking at a screen that shows you its view of an object with the camera having its own position in the game world. This is in addition to the game possibly lacking cues otherwise present in the real world (because it's artificial).

If the game uses a camera detached from the player avatar (i.e. 3rd person) then that can make it even more difficult to gauge if for example your avatar can make a jump or not.

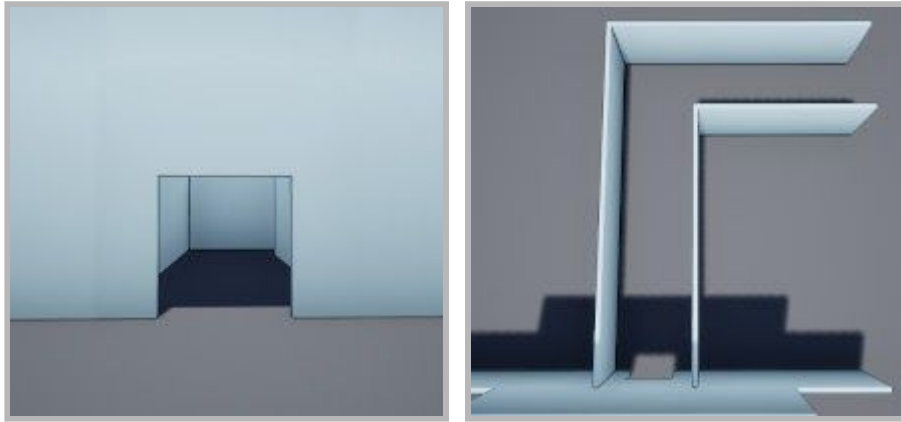
This is for example why it's easier to visually measure distance in a 2D platformer than in 3D. You're gauging horizontal distance, not depth, in a 2D platformer.



*Identical distances, but different perspectives. Notice how it's easier to gauge the distance in the 2D view because you're not measuring depth.*



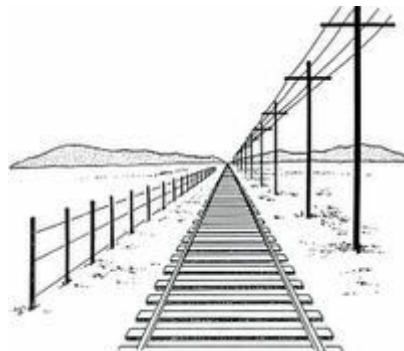
Another issue players can encounter is when it comes to hallways/paths that turn.



*Notice how in the image on the left it's difficult to tell that the hallway turns to the right at the end.*

So how can we minimize these issues?

Giving the player depth cues and reference so they can more easily calculate the distance and notice openings is a good approach.



Placing objects the player knows the size of (like the football in the image) and then having it placed further away can help give them a rough idea of the distance.

But placing something that stretches out more linearly (doesn't have to be in a straight line), preferably with consistent repeating elements, is a pretty effective way of helping the player gauge depth.



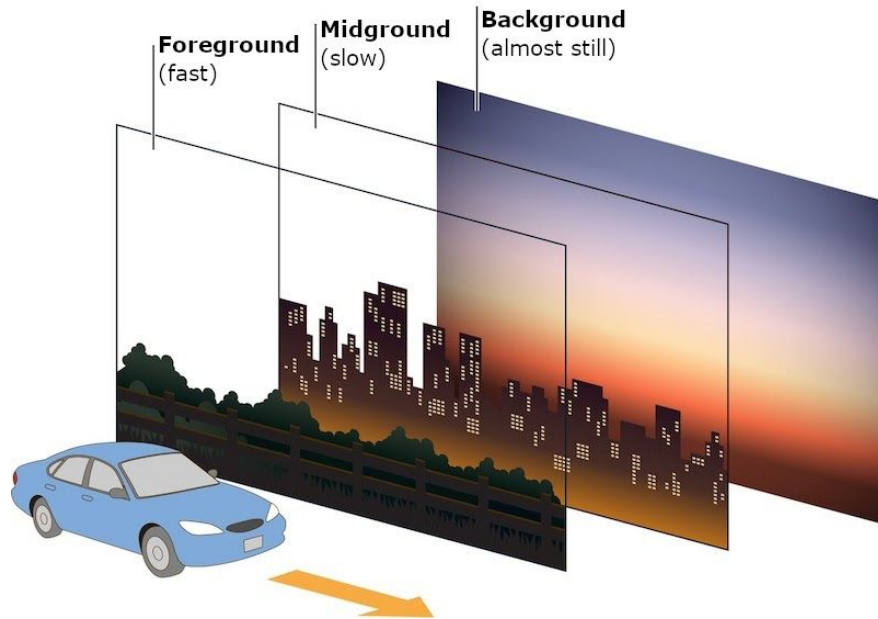
Just be careful when it comes to using objects that don't have fixed sizes in the player's mind.

Even though they might've *seen* it before, it can be hard to tell how large a tree is in the distance. Is it a small/young tree or is it a big tree just very far away?

This is especially true if you or your team also scale the trees differently for visual variation.

A football on the other hand has a known size for the player, so there's less risk of confusion for the player.

Another way is to make use of the parallax effect. This is when things close to you appear to be moving faster and things further away from you appear to move slower. You basically break down your environment in *foreground*, *midground* and *background* elements for the player when depth perception is more important (or you just want to communicate depth).



This can also be done by simply using things like pillars, railings, boxes, vehicles, etc. in a room. Try and space things out so there are identifiable foreground, midground and background elements.



*Here we can see the railing in the foreground, the wagon in the midground and the truck in the background. As the player would move these things would 'move' at different speeds. Here we can also see how we use the pillars on the left and the railing to also help with depth perception.*

But what about the hallway? What it boils down to is basically just giving hints of that the hallway continues (or any other path that initially doesn't indicate that it continues).

Some examples:

- An object the player knows isn't visually complete (like the upper half of a corpse).
- An element the player can visually see or assume continues (like a rug or a pipe).
- Revealing a small part of the area the turn in the path leads to (like the next room).
- A lightsource from around the corner, possibly even projecting a silhouette of something (like a person).



In addition, being aware of your established metrics can help as well.

If the player can jump a 3 meter distance, then placing a platform 3,15 meters away from them over a fatal pitfall probably isn't a good idea. In cases like these it's generally better to exaggerate the distances so it really looks like the player can't make that jump.

## ❖ Readability ❖

Readability is about making it easy for the player to discern (or "read") their environment. To not be visually overloading and providing sufficient contrast so the player can see what you want them to see. The best way to demonstrate how low readability can harm the experience is through some examples.



*Notice how everything blends together? The walls, the door (center) and even the character on the right?*





*In addition to most of the surroundings being green and brown, all the vegetation and their shadows end up creating a very visually 'noisy' and overwhelming environment.*

In many games it has even gotten to the point where in-game guiding elements (see image below for example) and UI elements (arrows and special vision modes) are almost mandatory since reading the environment can be difficult.



*Other examples of visual language can be in the form of white paint, the yellow markings in God of War (2018) and white scratches surrounding ledges you can grab in the Uncharted series.*



*Without guiding elements turned on, can you guess where gatherable resources are, the path forward is and where you can climb?*





*With the guiding elements turned on and the special vision mode activated we can now clearly see that which was very difficult (if at all) to see before.*

In addition to this, guiding elements like these can easily break the 4th wall since you're more directly talking to the player.

But despite all this, using these kind of guiding elements might not be a problem for your game. But if you want to move away from using them, what can be done?

Well the first step is to consider how visually complex your scene is.

Visual complexity can for example come from how detailed/noisy it is (see foliage image), a large number of objects to take in, a large number of saturated colors, a large number of possible points you can be attacked from, etc.



The more visually complex something is, the more extreme something has to be for us to notice it (like in the jungle image above with the guiding elements turned on).



Usually it's not even intentional or something people are aware of. The goal is often just to increase visual fidelity, make scenes feel less empty, etc. But the end result is still that it becomes harder to visually read.



So how can we reduce visual complexity and increase readability?

One approach is to go for a more minimalistic and desaturated approach. This is what Valve did with *Team Fortress 2*.

It's easy to initially think that it's because of the stylized visual approach, but it's more than that. They have lower saturation (more pale and greyish) in the environment and higher saturation on the player characters.



*Rainbow Six: Siege* does something similar, except they make their characters very dark to contrast against the lighter less saturated environment. This is also a high fidelity game and doesn't have the stylized visuals of *Team Fortress 2*.

In fact, a bunch of multiplayer games do this. This is because good readability in a competitive shooter is extremely important, especially in games where the TTK (time to kill) is very low. If players have to "fight" the level to make out where players are that can easily cause frustration.



This in turn can also create a more harmonious look due to color harmony. High vibrancy makes all colors scream for attention and makes an image more noisy. Meanwhile if we lower the saturation of the colors all the colors get along better. This in turn means that if we add a few colors with higher saturation they still fit in with the rest of the colors, but stand out more. So it's not an absence of color, but color harmony.



Games like Unreal Tournament 4 also does something else to aid readability, but still allow for more visual complexity.

This is their approach to keep the complexity low around areas where players are more likely to frequently traverse, such as floors and walls.

But in the ceilings, pillars and other such areas that's where more complexity comes in. This helps keep the playable area more readable

Yet another example of how multiplayer games makes their levels more readable is the wasteland level in CoD: Modern Warfare 2.



Here they use fog to create contrast (so distant objects in the fog create silhouettes).

Then they make sure the vegetation is both thin (see-through) and isn't above the waist in height.



So the key is to basically to try and reduce visual complexity while aiming to make things in the environment more distinct.

While reducing visual fidelity might not be an option, thinking about the gradient silhouette of your environment can be helpful.

This includes once you add people, animals and things the player needs to see into the mix.

Sticking to the color harmony approach from before we could for example have the browns and greens of the forest be less saturated, but the browns and greens of animals and people be more saturated.

The effect would then be having a more saturated environment with darker elements along the ground to help make things stand out.

Combining this with a gradient of visual complexity where the play space has less visual complexity and then as the space transitions to where players/enemies can no longer traverse (or where something visually important is) we increase it.



In addition, when it comes to vegetation what can easily cause a lot of noise is color variation amongst the same 'group' and when strong contrast is created by the shadows of other vegetation (like tree leaves and bushes).

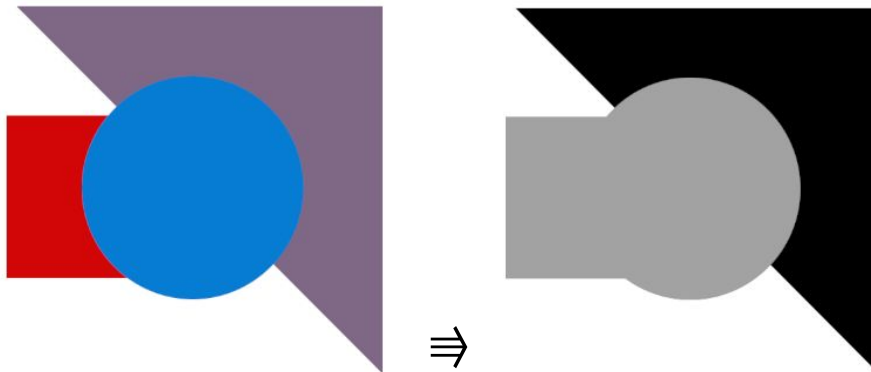






**Left:** Each strand of grass has both contrast and color variation, combining green, yellow and brown. While this can result in more realistic looking grass, it also adds a lot of noise.  
**Right:** Notice how the vegetation's shadow adds a lot of contrast and noise where there wasn't one.

What you basically want to do is reduce the amount of complex shadows in an already complex environment and try to blend shapes together, so they're perceived as one whole shape.



While you don't have to go to this extreme, at least keeping it in mind and potentially implementing it to a certain extent can help make your environment become more readable.

## ❖ Orientation ❖

Orientation is how we can design our levels so the player understands their surroundings better. This is so they know where they are in the world and how they can get to where they want to go.

In other words, how we can help the player create a mental map of their playspace.

The easier it is for the player to remember how the level is arranged and where they've gone, the better their experience will be.

If the level (or game as a whole) has backtracking then designing around proper mental mapping becomes even more important, otherwise they can get lost and easily become frustrated.

Though some elements that help with orientation can also be used to guide the player (like large landmarks) that isn't always their only (or even intended) function.

## ❖ Landmarks ❖



**Micro** = Small  
**Meso** = Medium  
**Macro** = Large

I will use these words to make it more clear when I'm talking about the scale of things.

For example: *Micro* Landmark (tree stump), *Meso* Landmark (small house) or *Macro* Landmark (massive tower).



Landmarks are things which stand out from their environment and serve as a reference point for where you are.

If you looked outside through a window and saw the Statue of Liberty you would know you were in New York. Even if you've never been to New York before, so long as you kept the statue in sight as you walked around town you would constantly be aware of where you are in relation to the statue, thus preventing you from truly getting lost.

Landmarks can also serve as a goal the player is trying to reach.

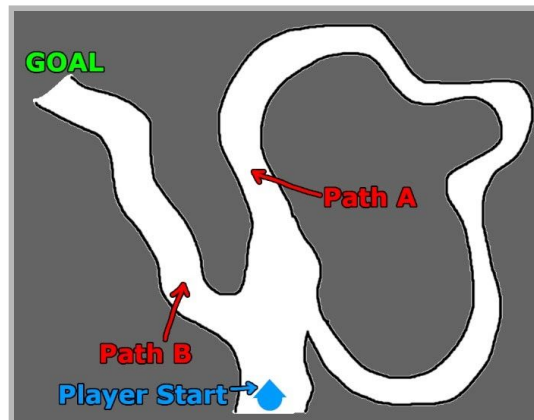
A variation of this are "Weenies" used at Disneyland, which are basically landmarks except they have a sense of mystery to them to serve as visual magnets.

So Cinderella's Castle is a weenie, a stone pillar is not.



What's important to remember however is that landmarks do not have to be big. In fact using *micro*, *meso* and *macro* landmarks is an effective way to boost player orientation.

A good example of the value of this is from various examples of players getting lost while playing various games. One of the more widely known ones is how Valve cut a looping part of a cave system in Half Life 2 because a playtester kept going around in circles for 30 minutes.



*In this simplified visual mockup of the Half Life 2 situation the player would head down towards Path A, come out again and then head into Path A again. Because the cave system was so similar the player couldn't easily tell that they were heading down the same path over and over.*

So if we used a smaller micro landmark in this situation we could effectively 'mark' this split so the player would instantly realize "Oh I'm back here again".

What would be important however is to make sure that the landmark isn't something that can be mistaken for something that's naturally occurring in the environment and that it stands out. So a bush wouldn't be a good landmark, but a skeleton pointing in the direction of one of the entrances would be. It should basically almost feel unique (at least in that area).

Then try and keep the landmarks varied. If you always use a body in similar situations you not only diminish their effect as landmarks, but can also take them out of the experience.



*Image on the left has no landmark. Image on the right has a landmark. In our example the landmark could be a decaying body or a skeleton.*

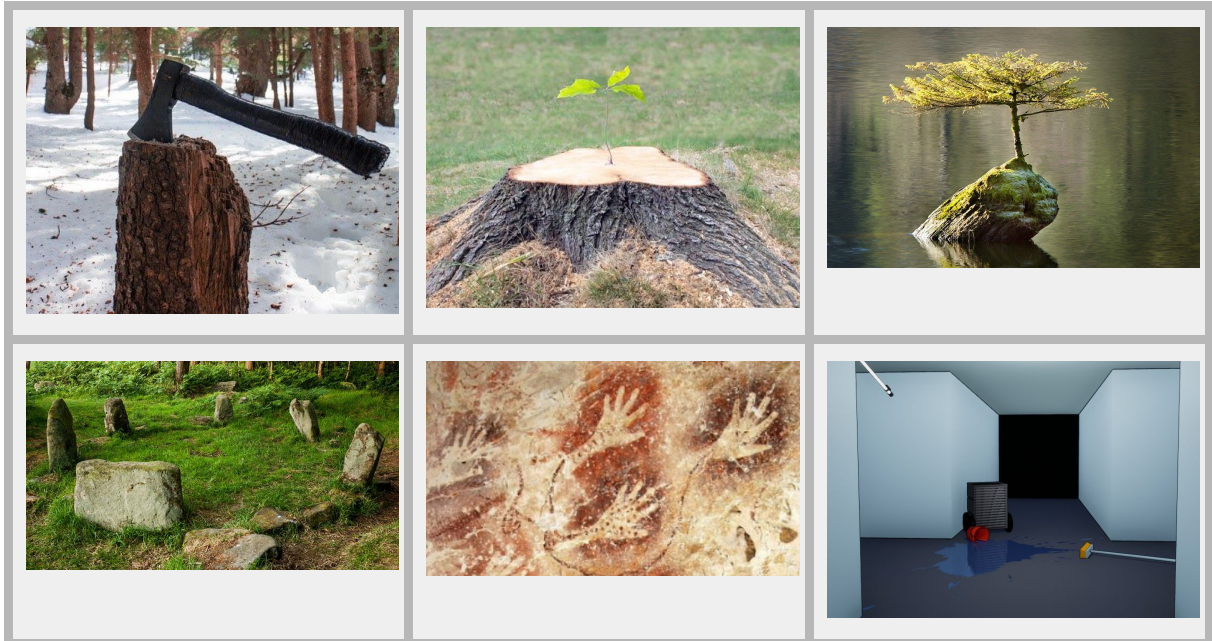
Micro and meso landmarks are particularly effective in spaces that tend to repeat themselves like caves, interiors, forests, etc.



Now let's look at some more examples of micro, meso and macro landmarks.

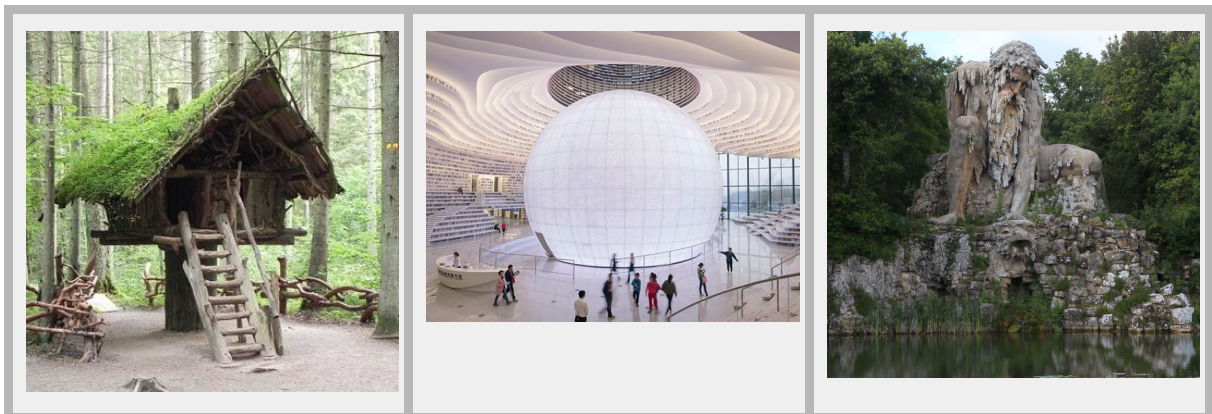
### Micro landmarks

Small in scale and effective to break up similar looking environments. Become more important if macro landmarks aren't visible (i.e. in a cave or dense forest). Examples below.



### Meso landmarks

Moderate in scale. Serves as a bridge between micro and macro landmarks that can act as unique elements in their area. Unlike macro landmarks they're typically not viewable from a great distance. Examples of meso landmarks could be a witch's hut, a police station, etc.



## Macro landmarks

Large landmarks viewable from a great distance. They generally function as a means to orient the player in large open worlds, player goals and as weenies.



## Area landmarking

Another way of landmarking is to create a theme or style around certain areas. This makes different areas look and feel more unique, which not only helps with orientation but also variety. This could be the usage of different colors, plants, graffiti, dropped floors, raised ceilings, different floor textures, etc.

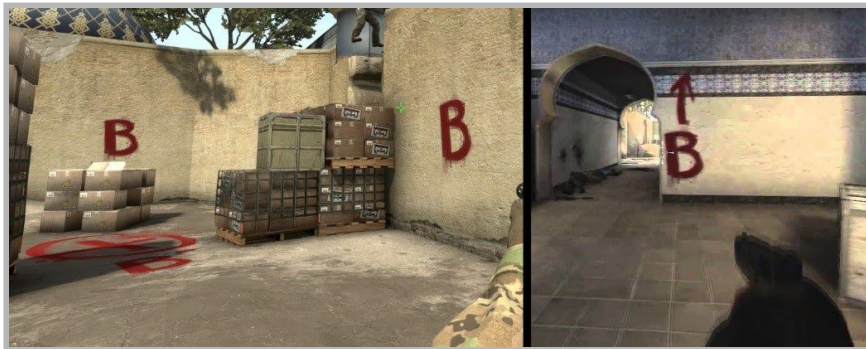




## ❖ Signposting ❖

A variation of the micro landmark is signposting. It's essentially a more direct form of landmarking that speaks more directly to the player and typically makes less sense in the context of the game world.

In for example Counter-Strike: Global Offensive the A and B sites are marked out with letters, arrows point the way to the sites and the place where you can plant the bomb is painted on the ground.



The more complex the signposting, the more time and attention it will require from the player, but it can still be useful in moderation to for example help with the world-building (if used properly).

In addition, in-world maps can be used to help players orient themselves, create a mental map of the area and how it connects to other areas.

These sort of maps can be used to tease the player of things yet to come or what they might've missed in the surrounding area.



You have to be careful though since overusing or mishandling signposting can cause a disconnect for the player. This is especially true for signposting that doesn't make in-world narrative sense.



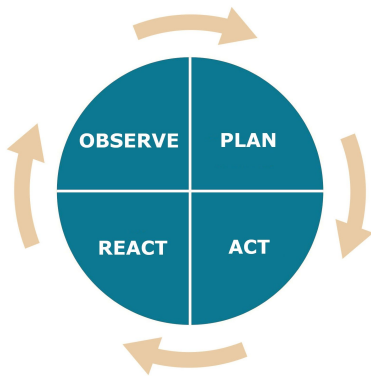
### Orientation design tip

Think of how you would explain to someone how to get somewhere or how a treasure map would be laid out:

*"Follow this road until you get to a fallen tree, then take a left turn..."*

Could you easily guide someone through your level in this way? If not, consider adding some landmarks, signposting, etc.

### ✂ Vantage point ✂



A vantage point is a place where we can survey the area, orient ourselves and plan our approach.

It's commonly found in open world games and games with stealth and/or combat, but can be found in all sorts of games.

While typically an elevated position, it doesn't need to be. It's just a place where we can observe our surroundings, so it could be a room with a glass window looking into a larger room.

In terms of orientation, the player can more easily pick out landmarks, understand how to best move through an area and start forming the mental map they'll use once they enter the new area.

We'll revisit the Vantage Point in the *Designing for Combat* section of this document where we dive deeper into how it applies to combat and stealth.



*In Horizon: Zero Dawn the player can get the lay of the land, orient themselves and plan their route. In the Batman: Arkham series the player can take in their surroundings so they understand how they can move around the level and take out their enemies.*

## ❖ Guidance ❖

Now that we've talked about both how to make a level more readable and easier to orient yourself in, let's talk about how it all comes together.

How we use this knowledge and more to effectively guide players.

There are basically 3 ways to guide the player:

- UI
- In-game world visuals
- Narrative

We will mainly focus on the two last ones, since they tie more directly into level design.

While UI is effective, it can easily become a band-aid solution for problems with the game's level design or visual readability if not used carefully. It can also make people stop taking in the world around them to focus on UI elements instead.

Said UI elements could be arrows that point exactly where the player needs to go, a detailed minimap or vision modes (that completely change how the game looks).

### ❖ Overestimating guidance ❖

Sometimes it can be easy for us as designers to overestimate how effective our methods for guiding the player are.

For example, let's look at leading lines. This is when shapes in the environment form lines that 'point' in the direction of where the player should go.

Not only are there tons of things in our daily lives that form lines that don't 'guide' us, but the same is true for games as well. Most things that form lines in games aren't intended as leading lines.

So even if leading lines would be very effective, it would create mixed messaging because of all of these other 'lines' that don't point where we should go.



*Does the player really go down this path because of the leading lines or because there is no other path forward and it's simply a hallway?*

We also need to consider our natural bias as designers. Because we're aware of the concept of leading lines we might look for them when there are none.

Is the person in the middle of the image below really walking down the street because of the 'lines' (yellow) in the environment?

What about the conflicting lines (blue) that point elsewhere?

We need to remember that correlation is not causation and if we're not careful we can easily deceive ourselves.



We also can't assume that the player knows what we know.

When we as designers play a game we can perceive to see things that either aren't there (like leading lines another designer didn't put in) or certain 'tricks of the trade'.

For example:

*"Where do I need to go? Aha, a vague light source is over there so my fellow designer probably wants me to go there!"*

Meanwhile the player could be completely unaware of the concept of guiding lights and not actively look for it, nor understand their meaning.

To further elaborate on light, people are not automatically drawn to light itself (like a moth). What usually happens is that light can create contrast (dark area vs brightly lit area), but this could've happened in an environment with flat lighting and just black and white colors.

You also have to consider that other things in the environment might be more interesting to the player (people, doors, things related to our goal, etc).

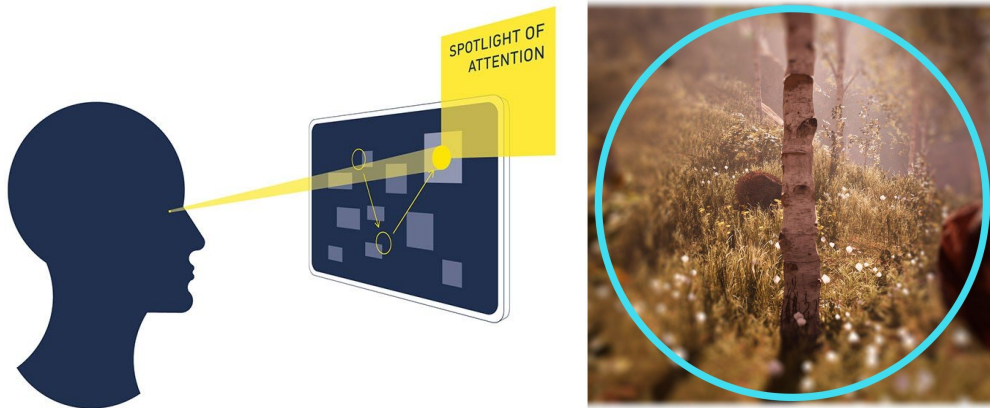
Also if light is sometimes a guiding element and sometimes not (decoration) this can further muddy the water and lower lights effectiveness, especially if both kinds of light look similar. That said, light can be effective to help guide attention to what's in the "spotlight" (so to speak), since what's in the light is more important than the light itself.

To conclude, I'm not saying that certain guiding elements have no effect and that we shouldn't use them. But that we should be aware of how our designer knowledge differs from our players and not overestimate the effectiveness of the guiding elements we put in. This is why playtesting with people that don't share our knowledge of guiding elements is important.



## ✂ Attention & saliency ✂

Our attention basically functions like a “spotlight” where a small area at the point we’re looking at is in focus, while the things surrounding it become blurred.



So how do we know where players are the most likely to aim this “spotlight” and how can we make players pay attention to what we want them to?

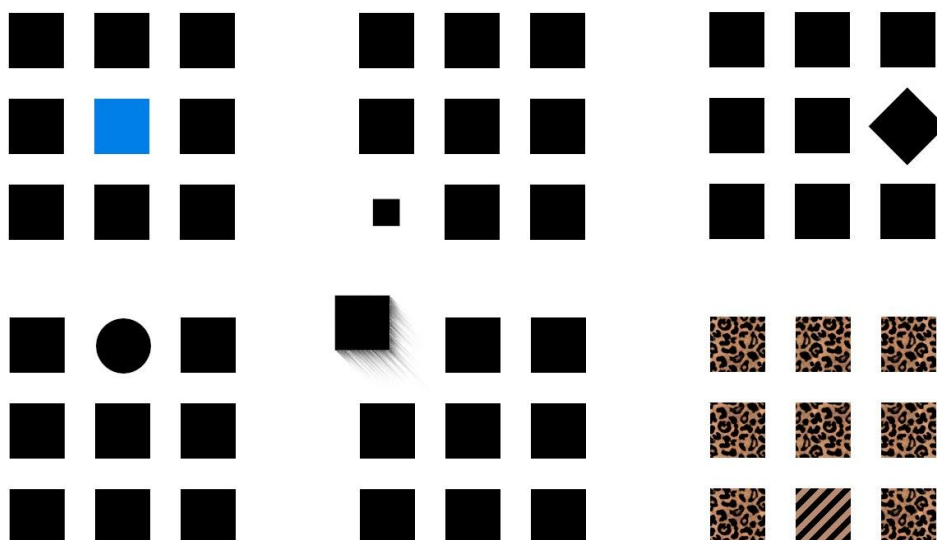
The answer to that lies in *saliency*. The more distinct and attention grabbing something is, the more salient it is.

There are two kinds of saliency: *Top-down* and *Bottom-up*.

### Bottom-up saliency

Bottom-up saliency is about contrast.

Contrast of color, size, orientation, shape, movement and texture.



## Top-down saliency

Top down saliency is about what's meaningful to us.

Human faces, people, things associated with our current task/goal (attentional goals).  
An attentional goal could be you trying to find a pencil in your bag or keeping your eyes on a ball during a ballgame.

An attentional goal could also be you meeting up with a friend with blond hair, green shirt and brown pants. You would then pay more attention to people that have blond hair, green shirts and brown pants (which can mean you become more 'blind' to other things).



The more we are in a task orientated 'mode' the more Top-down saliency will dominate.  
This means that Top down saliency is more relevant in games than in say movies.



### More on attentional goals

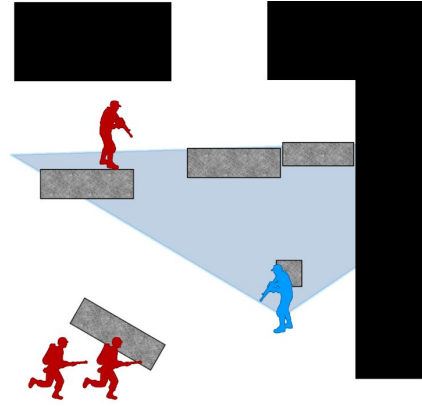
Not only can it help players notice and pay attention to things like health pickups or collectibles more easily, it can actually also result in problems with guidance.

If for example a player comes to an area with a locked gate they might assume the goal forward is to open the gate (even though the solution is to climb out a window).

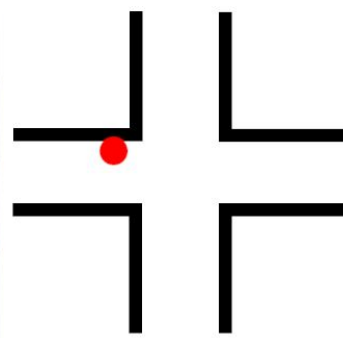
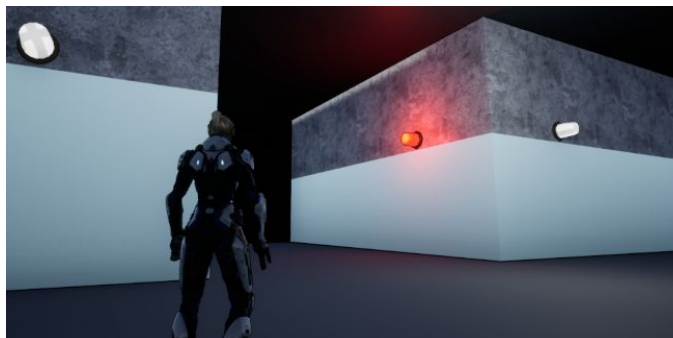
Because they assume the gate is the path forward they might try and open the gate, look for a switch, etc. and pay less attention to the guiding elements directing them to the window (because they're focused on things that will help them get the gate open).



It's also worth noting that if players are already focused on something (like being engaged in combat) this can make noticing other things harder (like a health bar flashing red). An example of this is when enemy reinforcements arrive mid-combat or if the area where enemies spawn during a wave based event changes.



*If enemies spawn from off-screen mid-combat and the player isn't notified in some way it can feel as if the enemies showed up from out of nowhere to the player.*



*If the player has to defend an area and enemies can spawn from all directions, having something that notifies players of where enemies will spawn can be helpful. This is especially true if you plan to have multiple enemy waves spawn at the same time from different hallways.*



To inform the player that enemies have spawned before or during combat you can have them come in via a dropship, blowing up a wall with an explosion, etc.

You can also present them in view to the player when you know they will be looking at a certain direction or let the player be in an advantageous situation (multiple enemies not directly looking at the player).

## ✂ Visual language ✂

Visual language is probably one of the more common ways to guide the player. It's basically a means of visually communicating information without using UI.

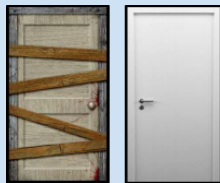
You'd rely on things like *affordance* (see below) and creating patterns (what certain shapes, color, etc. means) to allow players to better understand their environment.

When the readability of your environment is poor it generally increases the importance of an effective visual language.



### Affordance

Affordance is basically what someone perceives to be possible. Good usage of affordance can make your games more intuitive and friction free, while weak or false affordance can lead to confusion and frustration.



Positive affordance: Looks like it affords to be used and can be. Like a door the player can open.

Negative affordance: Looks it doesn't afford to be used and can't be. Like a barricaded door the player can't open.

False affordance: If it looks like it affords to be used but can't (or vice versa).

Like a button that looks like other buttons you've been able to push before, but now can't.

It can be worthwhile to consider if your level/game should for example have ladders if the player can't climb ladders (or only climb them on some levels).

When the player expects to be able to climb a ladder but they can't it creates a disconnect, which can break immersion and they can start to second guess their environment.

Some examples of visual language can be seen below:



The *Uncharted* series frequently uses white or contrast to highlight where players can climb, shimmy across, etc.



In *Dishonored* they use cables that connect to electrified gates (and other things that can be powered) and “whale oil” batteries (that can be put in/disconnected).



In *Bioshock: Infinite* they visually separate a searchable or empty/searched object by having it be opened or closed.





In *Fallout 4* they use “Railsigns” to show where one of the game’s factions have marked certain locations to indicate hidden loot, danger, etc. At first the player doesn’t know their meaning, but can start to piece it the meaning together on their own or eventually find out their meaning at the faction’s HQ.

Visual language is generally more effective when it makes contextual narrative sense in the game world and makes use of affordance.

This way you avoid taking the player out of the experience by it “not making sense” why there’s yellow cloth at every ledge in the wilderness. You also minimize the risk that it feels too hand-holdy when you more directly speak to the player.

Something you also need to remember is to always maintain visual language consistency. If you use a color, shape, etc. as a form of visual language, make sure you don’t re-use it in ways where it doesn’t apply.

Imagine if you used the color blue to indicate that certain objects are breakable. Now imagine if you also used that color for things which weren’t breakable, possibly even in the same space.

This would not only cause readability issues (breakable assets blend into the background) but also confusion (“what can I break?”).





The absence of visual language can also cause issues. If the player is used to seeing white paint where they can climb, but in now in your (or a fellow level designers) level there isn't any white paint they might no longer know where they can or cannot walk.

Also consider what your guiding elements look like from different angles and different points in your level. If you use a shape or color as a guiding element that only 'makes sense' to the player or guides them from one specific angle or point it might end up not guiding them.

In the image below if the player looks up as they approach and pay attention they might see that the wooden beam points to a wall they can grab on to.

But if they miss this (disregard the beam as just decoration, don't move their camera the 'right' way, don't notice the grabbable wall, etc.) they might end up getting lost and confused.

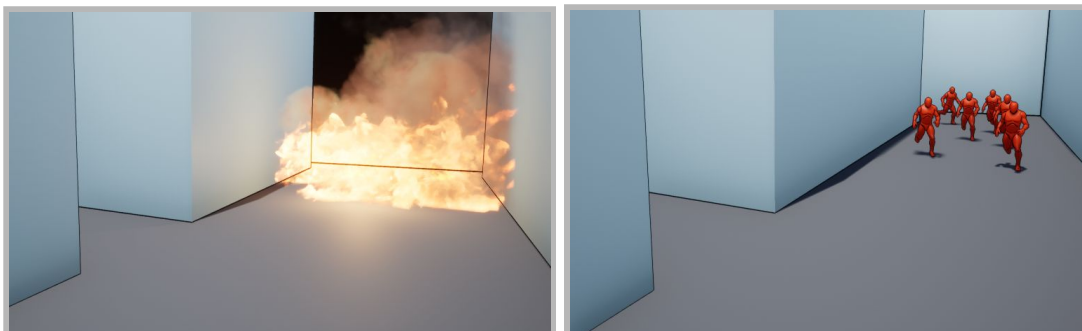


*In the right image it doesn't look like the wooden beam leads anywhere, since their view of the grabbable wall is blocked when they're that close.*

### ❌ Nope zone ❌

This is effectively just you guiding the player away from an area by making it seem dangerous or uninviting: "Nope, I can't/shouldn't go here".

It functions in a similar manner to visual language in that it uses (negative) affordance or other forms of visual communication the player understands or is taught.



One way to do this is with an obvious hazard or threat like fire, spikes, deadly pitfall, etc. This includes using enemies that are far superior in number in an area with no safety/cover, etc. Meaning it doesn't have to be physically inaccessible, but that they die/fail if they go there.

Another way is to effectively block off the area, making it physically inaccessible. This could be by using broken or blocked off stairs, guards (that disallow entry), barricades, etc.



But be careful of overusing the same type of *Nope zone*. Communicate with your team so you don't all use the same method(s), especially in close proximity to one another. If 90% of all stairs the player can't access is blocked by furniture it can feel strange ("Are all stairs blocked off by furniture?!") and possibly even 'lazy' ("Did the developers only know how to block off stairs with furniture?"). Variety is the spice of life, after all.

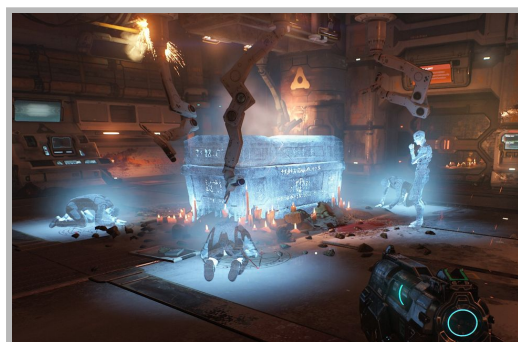
### ✂ Guiding with AI ✂

Guiding with NPCs like humans, animals and the like can be really effective.

This is because they generally have high saliency (attention grabbing) because they combine both Top-down and Bottom-up saliency.

This could for example be an allied NPC that runs over somewhere and says "Hey what's this?" or "I think this is the path forward".

For example, Elizabeth from *Bioshock: Infinite* constantly guides the player to various things of interest.



In *Doom (2016)* they not only use hologram of people and demons to guide the player to places of interest, they also use it as a tool to tell the narrative.



*In Uncharted 4 they used animals that not only shows the player the path forward, but also teases where there might be things worth investigating.  
So the dog could walk up to one path, look and bark, then start heading towards the path forward.*

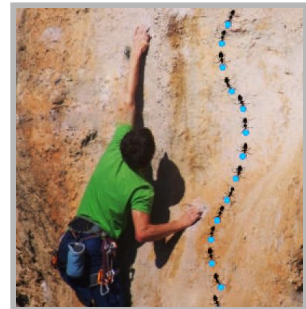
### ✂ AI as visual & audio language ✂

You can also combine AI with visual language if you want to create a more organic and believable environment, which can also allow for things like motion and sound. This can be a way to avoid having a yellow cloth on every climbable ledge or blue paint along surfaces you can climb (which while effective can create a disconnect for the player).



A basic example of this is how in the game *Evolve* they used endemic life to inform the player. In the game players control either a monster or hunters chasing down the monster. In the level there's an animal called a "Spotter" that will give a loud call if it sees the monster, informing hunters of roughly of where the monster is.

Imagine then if for example ants with glowing or colored bodies traveled along a path. This color could either just stand out or be your game's color for 'climbable surface'. The color could also vary to signify that some kind of resource is nearby as well and if you follow the ant trail you can see what they're harvesting. These ants wouldn't even need AI and could just be a set of particles or something you could draw along a path (with a tool) to place in your level.





Another alternative could be an animal like a lizard that creates a strong contrast with your environment. It could be a visual aid you re-use in your game in multiple ways to communicate to the player various traversal options and the like.

Let's say you called it a "Scrambler Lizard" for the player and clue them in (either directly or by repeated use in the environment) that they can often be found where you can climb, hidden crawl spaces (that they have dug out), etc.

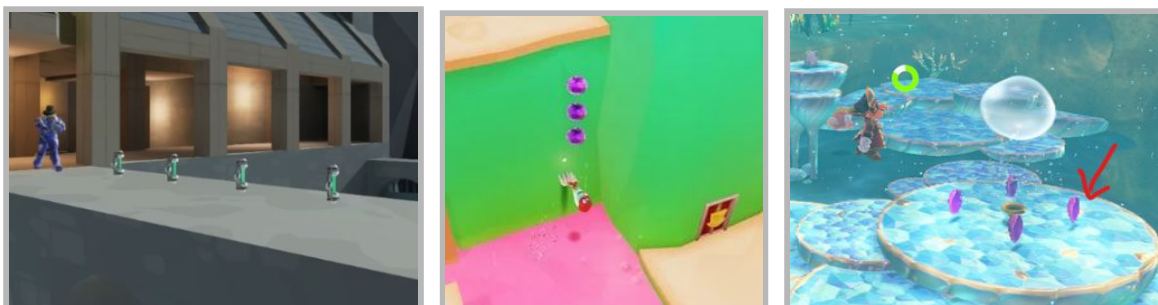


The benefit of this is that it would make sense if for example there's a bunch of scratch marks on climbable surfaces since the lizard keeps climbing up and down along it. It could also create motion (could crawl up & down a wall), sound, a silhouette, etc. as well as the association that the lizard equals potential traversal options.

### ✂ Breadcrumbs ✂

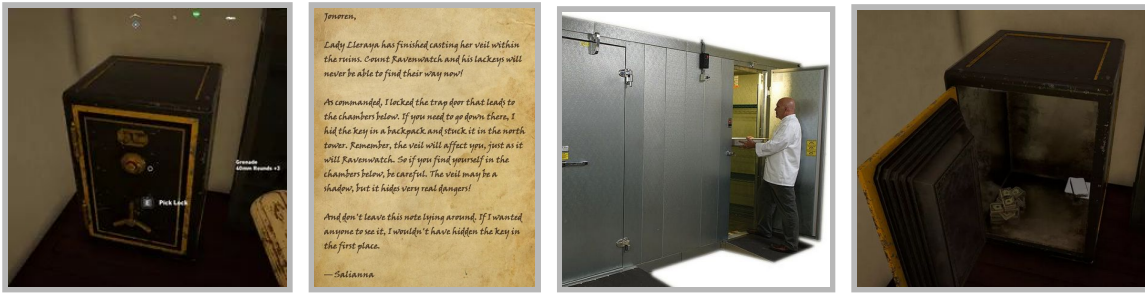
Breadcrumbsing is when you leave a trail of something (usually pickups or information) that leads the player in the right direction or towards secrets.

One way of doing it is to for example place pickups along a path to more directly guide players to use said path or use a pickup (or several) at important points (like the air bubbles in the image below on the right).



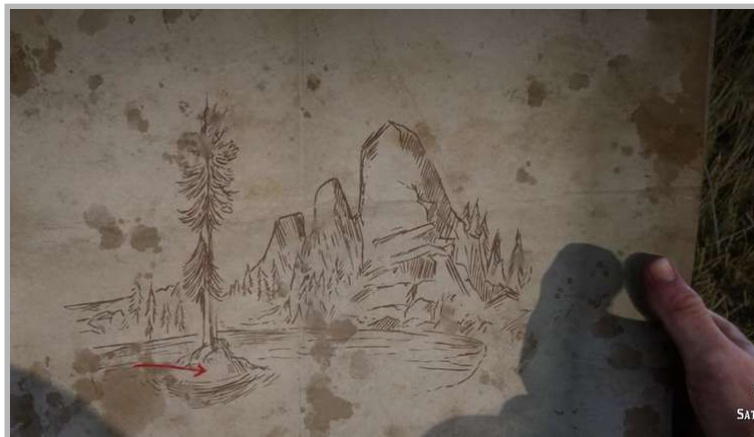
Another way is to leave clues in the environment and let players do some detective work. An example of this could be that you leave a note that describes how the chef stole the key to a locked safe and how the chef always hung around the freezer a lot. The player would then start investigating the freezer and find the key.





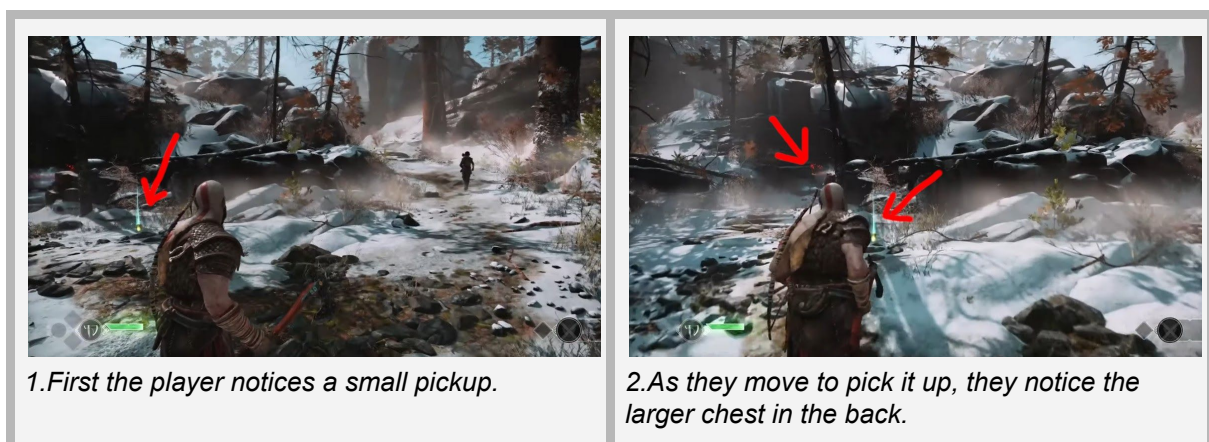
A breadcrumb trail like this can be effective if you want to have something in certain areas but they narratively don't make sense. Like why would a kitchen have a ton of money or loot in it?

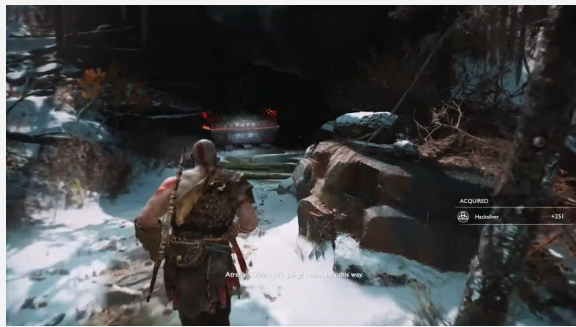
You could then place a breadcrumb in one location to loot stored elsewhere or use narrative to explain why loot is hidden in for example a kitchen.



Using treasure maps is also effective since it can add an extra layer of excitement. In an open-world game this can make your players explore and pay attention to their environments more as they're embarking on their own treasure hunt.

But breadcrumbing can also be more subtle and indirect. A good example of this can be seen in God of War (2018):





3. The player then moves to open up the chest.



4. Once they've opened the chest and turn around they can notice a side path from their new position that they didn't notice before.



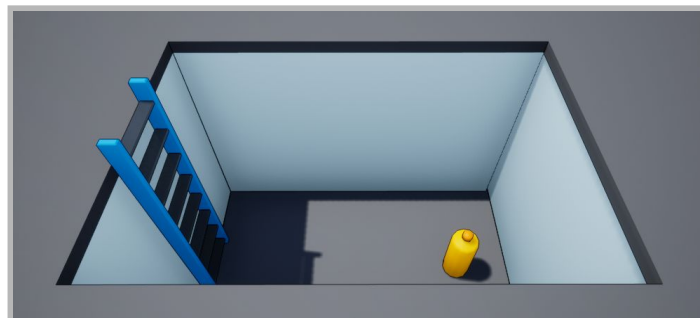
What makes this so effective and elegant is because when you chain elements together the guidance flows more naturally. The player feels like they discovered the side path on their own naturally.

But the situation would be very different if in image 1 the player saw the side path directly with a shiny chest in full view from the beginning.

An effective way to create breadcrumb chains is to create something harder to notice or reach as the end goal and then from the beginning think of how you can guide the player towards a natural reveal.

Creating situations where you can control or predict the players movement and camera view without making it feel like you're strong arming the player isn't always easy. But there's usually at least some situations in most games where the player is 'locked' into an action, like when they press a button or climb a ladder.

A fake dead-end where players only notice a path or item when they turn around is a positive form of bait-&-switch where the dead-end ends up rewarding the player.



*In this basic example situation, imagine that the player saw the collectible. They jump/climb down to collect it and when they climb up the ladder their view will in all likelihood be facing forward.*

## ✂ Teasing ✂



Teasing is when you show something of interest to the player that is out of reach, inaccessible or in the distance.

This could be the player's goal, a treasure, an object they can use an ability on (once they've unlocked it), etc.

You effectively inform the player of *what* they need to do or get, but now *how*.

This is generally good to use since it minimizes the risk of the player running around blindly not

knowing what their goals are, or they just blaze past an area instead of exploring it for secrets.

It's effectively *setup and payoff* since you spark player's curiosity and reward them at the end.



*In The Last of Us the devs frequently teased and guided players with landmarks. As the player progressed and saw the same landmark they could notice how they're getting closer and closer to their goal.*



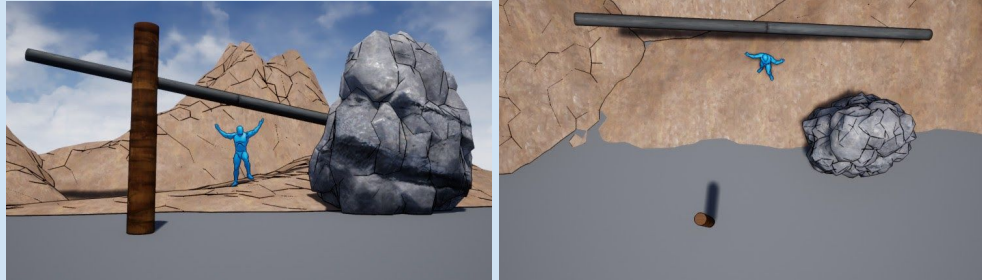
*Here we can see that the devs of Doom (2016) not only teased the blue keycard the player needs to get to proceed (blue arrow) but also a collectible (orange arrow). They also frame both of them*





### More on Framing

Remember that depending on the player's perspective objects can naturally form a "frame" around it. This means that your frame can consist of several pieces and not just a singular rigid one (like a doorframe or hole in a cave).



### ✦ Prospect & refuge ✦

We as humans are more naturally drawn towards places which gives us Prospect (a good view) and Refuge (safety). This kept our ancestors safe from enemies/predators and the elements, hence why we're drawn to it.



*Indoors and we can see a lot = We currently have Prospect & Refuge (primary P&R)*



*A castle on a mountain = Can offer us Prospect & Refuge (secondary P&R)*



This ties a bit into the idea of the “weenie” and as a landmark that can serve as a goal.

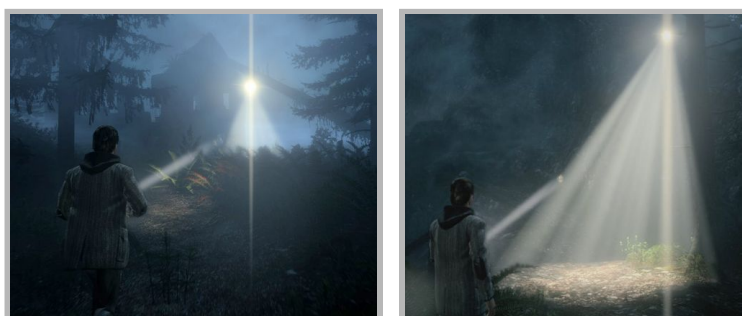
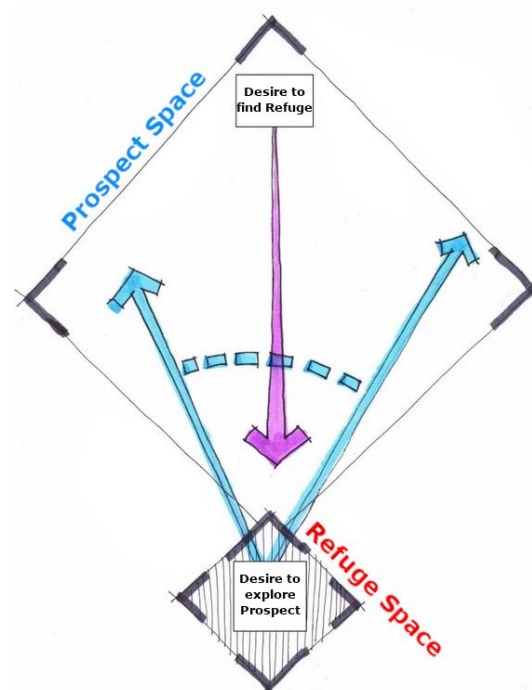
If something can offer us P&R we’re more likely to go towards it.

On the other hand, something that doesn’t seem like it will offer us P&R is less likely to draw us in.

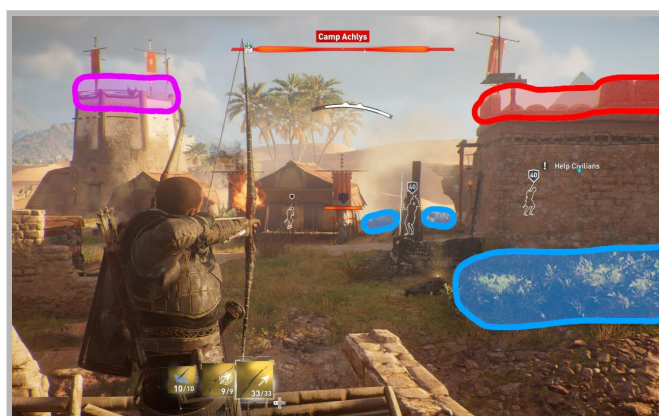
But this doesn’t have to be a castle on a mountaintop. It could be a guard tower we can get a good view at or a house/bunker we can find safety at.

This is especially true if we’re in a more hostile environment.

Players are also influenced by the context of your game. What in your game offers the player P&R? This can influence how players move around your level and can thus be used to guide them as well.



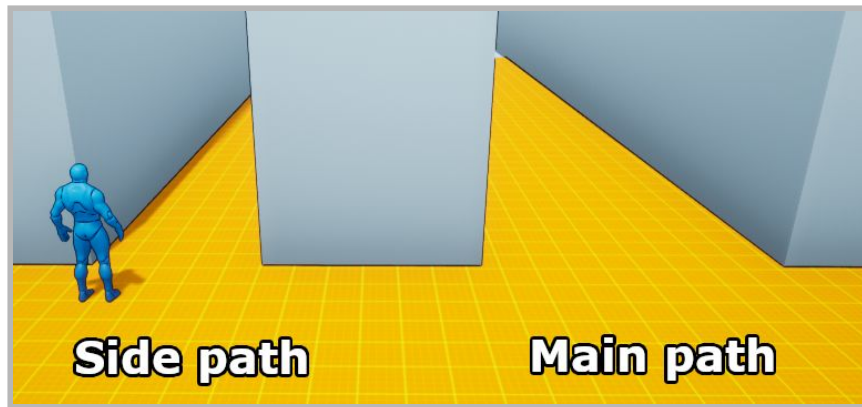
*The idea of Refuge (or Prospect) could be something the player learns about as they’re playing the game. In Alan Wake (2010) the player learns that lamp posts are Safe Havens where they’re protected from enemies and the hostile world around them.*



*In this example from Assassin’s Creed: Origins, the player can get Prospect, but not a lot of refuge on the wall (red) on the right. They can get Refuge in the bushes (blue) where they can hide, as well as some Prospect. They gain the most P&R from the tower (pink) on the left where they are both out of sight of the enemies sight lines and can get a good view of the enemy base.*

## ✂ Contrasting paths ✂

You can just visually communicate to the player that one path is more of a main path and another might lead to optional content, a shortcut, a new area, etc. by changing the size and 'theme' of said paths.



*The side path is more narrow than the main path, which can also help make these pretty much identical (visually, they have no actual markings to separate them) paths look more distinct.*

When you effectively theme paths you can make one path seem like it belongs to the area you're in (like a forest path) while the other has more civilization around it, it opens up more (isn't in the enclosed forest anymore), etc.



*While this image is made super distinct to showcase the effect, you still get the sense that the path on the right won't exit the forest since it feels like it still belongs in the forest. Meanwhile the other path on the left clearly has more civilization around it, giving it a separate feel and theme.*

## ◆ Pacing ◆

When you design your level it's important to balance pacing, predictability and that your level makes logical sense.

For example, if there's always a straight path towards your goal that quickly becomes uninteresting and boring, but if players have to travel through confusing maze-like levels they can easily become lost and confused.

If a level consists of non-stop intense combat for 1 hour it can easily become overwhelming, tiresome or even boring.

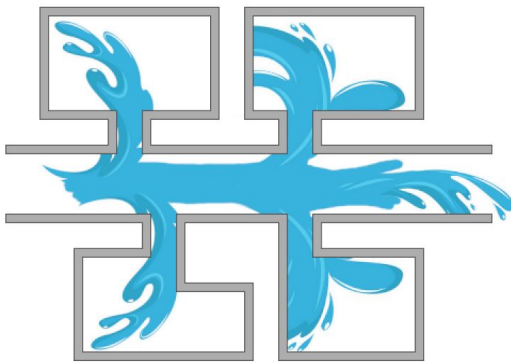
As such it makes sense to plan and keep an eye on the levels pacing from start to finish, using various tools and tricks to ensure the level is paced the way you want it.

### ❖ Pacing tools & tricks ❖

#### ✂ Flow in Singleplayer ✂

While I will touch on flow a bit here, most of the information on flow can be found in the **Multiplayer map design** section of this document.

This is because while a bunch of it is shared, a lot of it is either unique to Multiplayer or contradicts what you would want to do in a singleplayer level.



Closely tied to pacing, flow is about player movement and action. To make moving through a level more fluid and intuitive.

Players tend to flow through a level like a stream of water, but move more like a car.

What I mean by this is that if for example there's a hallway players will most likely enter the rooms closer to them first and not run back and forth, nor start at the very end.

So since players generally dislike missing content when they didn't intend to, it makes sense to for example not place an event that blocks off the level in the first room they can enter in a hallway.

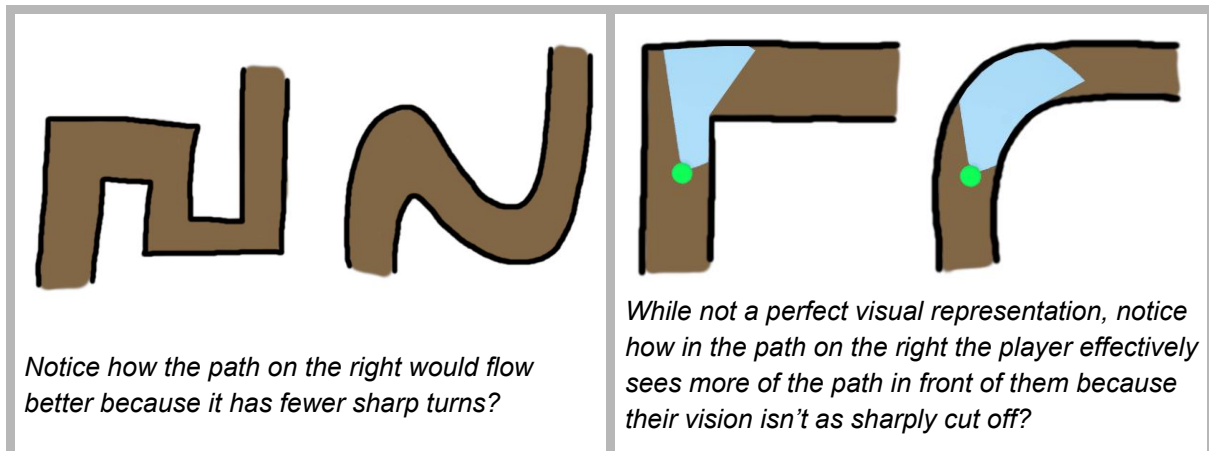
Instead putting this event towards end end of the hallway, signaling to the player where the goal is or when the game might continue/'cut-off' can be a good idea.

This could for instance be an NPC telling the player to *"meet them by the gate when they're ready"* or that they have to jump down a ledge to continue.

As for 'moving like a car', players will move through your level more smoothly if you don't construct it with a lot of 'sharp turns' a car would have trouble making.

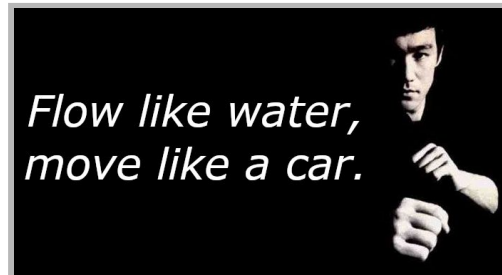
What this means is that when players is that if players have to make a sharp U-turn once they enter a room to go towards their goal or you have sharp corners, you will break or reduce their line of sight.

When the player moves through a level and you sharply reduce the line of sight in the direction they're moving or place the way forward outside of their line of sight this will cause the player to move with less confidence and slow down.



Additionally, making use of visual language can aid in player flow.

The better the player is informed of the path ahead of them and the more they can move with confidence, the better the level will flow for them. So consider when and where to push for more flow.



### ✦ Gates & Valves ✦

There are two common ways of controlling the pacing and access of a level, gates (hard & soft) and valves.

#### **Hard Gate:**

Halts progress until something is completed, like finding a key or pressing a switch. Usually takes longer to bypass than soft gates.

Often used when you want players to explore the area, have them enter areas they wouldn't normally to reach their goal (like entering a mine to find dynamite to blow up rocks blocking the road), etc.





**Soft Gate:**

Intended to briefly slow the player down. This could be pushing down a tree, turning off a valve or needing your AI partner to help you get the door open (boost them up so they can drop down a ladder), etc.

Handy when you briefly want to slow the pace down, have some character banter, etc.



**Valve:**

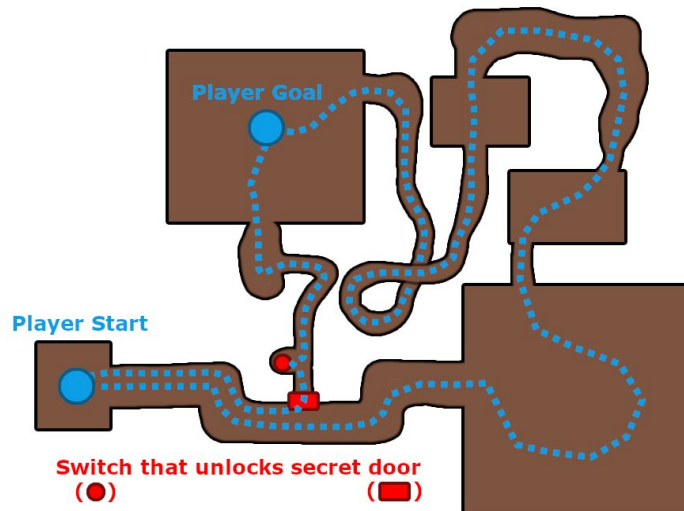
Areas that close off behind the player (can later be opened or not, depending on your intent). This can be having player jump down a cliff ledge, a part of the level gets destroyed behind the player, they slide down a rope, a crawlspace, etc.

These can be effective to keep pushing players forward, allow devs to load/unload certain parts of the level, in Multiplayer (combined with blocked LoS) to protect player spawn rooms, etc.



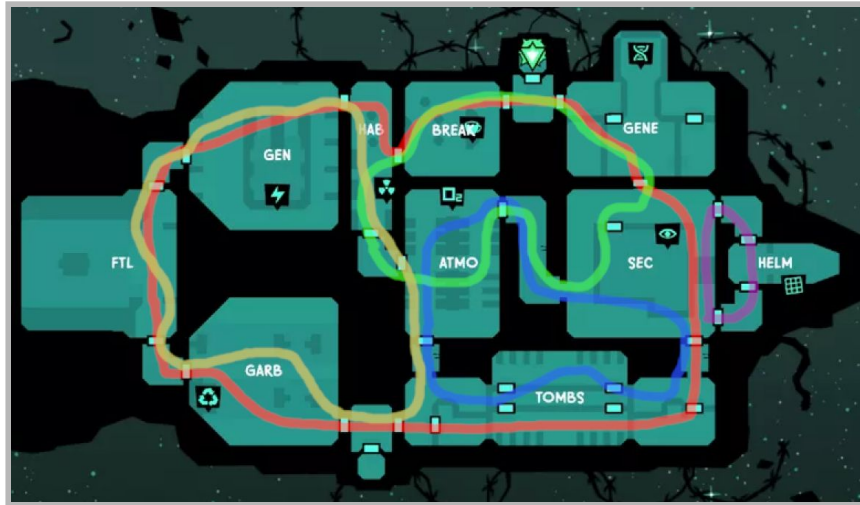
✂ Loops ✂

To minimize pointless backtracking through empty areas (like dungeons the player has cleared), having the level loop around on itself can be an effective way of reducing it. This could also just be a rope the player can slide down to the entrance with, a teleporter or something along those lines.



Example of how a dungeon could form a loop.

Games like *Void Bastards* and *Hitman 2 (2018)* also has several loops throughout the level (sometimes referred to as 'swiss cheese', since it's full of holes). This opens up for more freedom of exploration and reduces the risk of forced backtracking.



*A level from Void Bastards.*

### ✂ Recycling levels ✂

Re-using your levels and areas can be a smart move for several reasons. It saves you time from having to re-do as many levels from scratch, it can make levels more unpredictable, it can result in power shifts and more. What's important however is that you try and make it feel and play differently, so the players experience is different. It could even include new mechanics on the second-go. A few examples below:

#### **Power shift / Reversal**

Ex, the player attacks a fortified enemy position with a machine gun emplacement. Once the player defeats the enemies here, they get to use the machine gun against the enemy reinforcements.

#### **Playing the level backwards**

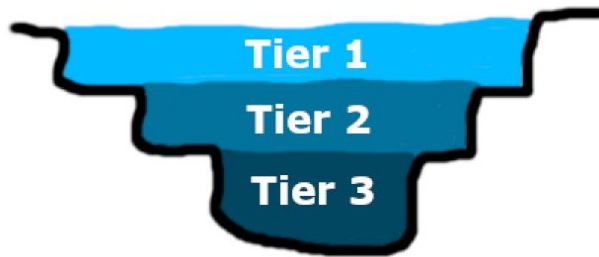
Ex, the player breaks into a hidden dungeon to steal treasure. On their way in they have to both platform and avoid traps. Once they get the treasure enemies swarm the dungeon and you now have to fight your way out backwards through the level (which on its own can offer a different experience).

#### **Revisit**

Ex, the player revisits a previous level/area, but it has changed. This could both be the challenges the player faces and visually. If it's an open-world game this could happen over a larger span of time.



In *Yooka-Laylee and the Impossible Lair* players can modify the levels which opens up new rewards, different visuals and a different way to play the levels.



In *God of War (2018)* there's a huge lake in which the water level will gradually lower as the game goes on, which reveals new areas while still keeping the prior areas accessible.

## ❖ Planning & structure ❖

### ✂ Intensity, variety & time ✂

There are 3 core elements you should be aware and think of when planning out the pacing.

#### Intensity

- The challenge and effort required, along with the impact. Easy puzzles and combat encounters are low intensity, while difficult puzzles and boss fight are high intensity. Keep in mind that the narrative can also have intensity.

#### Variety

- The type of activity or things that happen. Varies from game to game, but in an action-adventure it can be going from platforming to stealth to combat.

#### Time

- The minimum and estimated average time it takes to finish something (like your level).

These 3 things can also happen on a *Micro* (small, like a single activity), *Meso* (medium, like your entire level) and *Macro* (large, like the entire game) scale.

So there can be different intensity in a single combat encounter (Micro), how long it takes to finish a level or quest (Meso) and variety across the game (Macro).

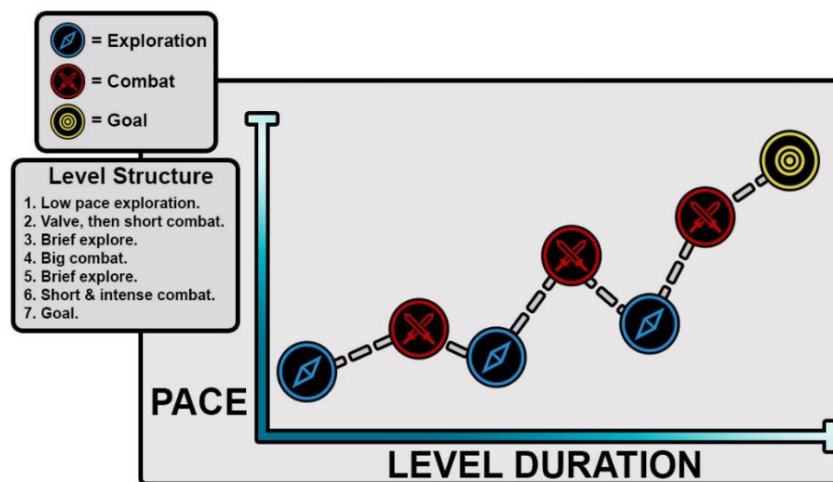
Balancing these things at the different scales is important. It helps keep your players engaged by not burning them out, boring them or stressing them out too much. If for example you and another designer each place your high intense combat focused levels back to back that could potentially have a negative effect on your players. Even awesome things can become routine and boring if we're exposed to it too often or too long.

Also, even if you're not directly involved in the narrative, it's good to keep it in mind. If the narrative team wants to hit certain narrative beats, make the player feel certain emotions, etc. you don't want to create a disconnect with your level.

If a character close to the main character died at the end of the last level, making your level start with a bang and be high intense joyride might not be ideal.

Balancing and keeping all of this in mind can seem overwhelming, so let's look at how we manage all of this.

### ✂ Beats, charts & timelines ✂



Charts are probably the most common tool to use when planning and structuring your pacing of a game, level, combat encounter, etc. They help give you an overview of what you're working on and allows you to set goals (for example, a combat section should roughly take 3 minutes).

The chart itself is made up of "beats" which are specific points on the chart that indicate what is happening (see image above for example).

You then arrange these beats together so they form peaks and valleys along the graph.



Probably the most common form of pacing structure is similar to the one on the left (see image).

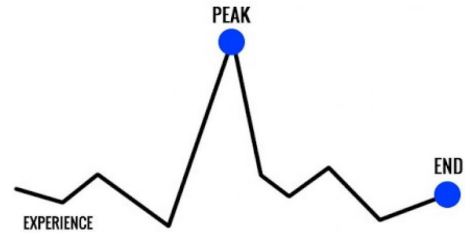
It contains peaks and valleys and a constant escalation until it reaches the climax. It's typically a pretty safe structure to follow if you're unsure of what to do or want to play it safe.



This doesn't mean you can't change things up and have extended peaks/valleys, start with a high intensity beat, etc. In fact, it's good to change things up (at least every now and then) so your game doesn't become too predictable.







Just make sure you keep the player's experience in mind. A 50 minute non-stop high intensity level might seem interesting to you, but would players actually enjoy playing it?

Something also worth considering is the Peak-End rule. As humans we tend to remember the peak and end of our experience with something the best. So for example, it could be okay to have a little more uneventful and 'dull' level if the peak and the end is memorable and satisfying to the player.

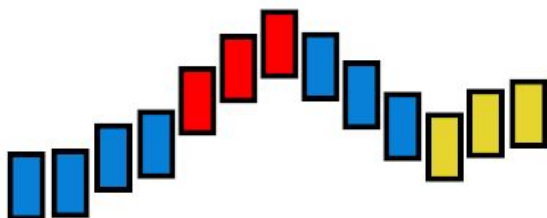
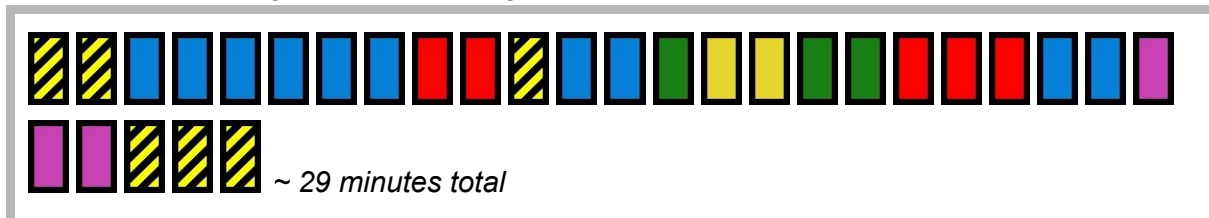


Creating a timeline can also be useful, especially when used in together with the pace chart. You could for example use colored/patterned boxes to represent an event and time. In the example below, each box represents roughly 1 minute of playtime.

**Explanation:**

Cinematic 	Exploration 	Combat 	Puzzle 	Boss 	Platforming 
---	---	--	--	--	---

**Example timeline** (goes from left to right and top to bottom):



You could even arrange the blocks so they also form the desired intensity, thus giving a pretty accurate idea of the pacing and events.

Adding some notes for the different beats that describe them would also be a good idea.

So even before touching the game engine you could make a rough plan of your desired pacing and events, which can help you when you create the different areas of the level so they match the event, time and intensity.

This can also help you keep track of roughly how long your level will be, which is important to keep track of.

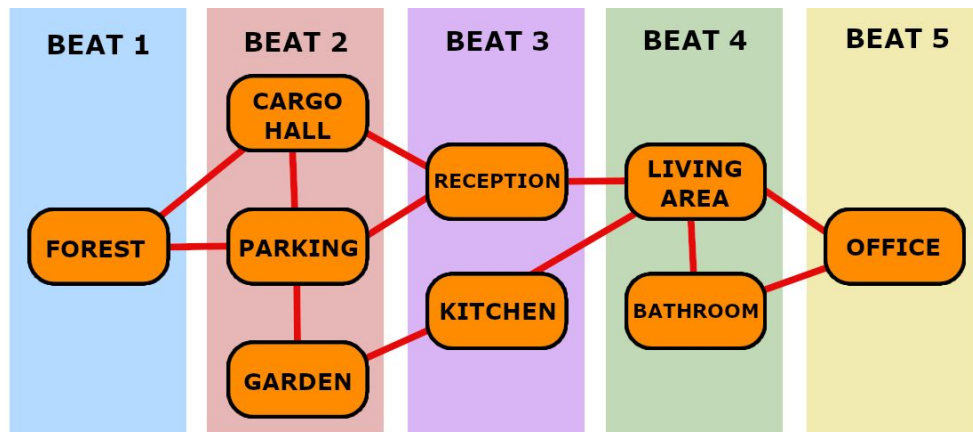
It varies from game to game, but in many Action/Adventure style games (like Uncharted 4, Batman: Arkham Asylum, Titanfall 2, Doom Eternal, etc) they tend to land around 30-50 minutes in length (with an average of 40 minutes). So if you were to plan a level for that kind of game and you planned out a 20 minute or a 90 minute level, you might need to add or cut some parts.

But don't get too fixated on creating perfectly balanced pacing graphs. Remember to spice things up a bit and throw the player a curveball from time to time.

### ✂ Non-linear pacing ✂

Potential pacing issues can however arise if you open up your levels with alternate routes. Even if you manage to ensure that the player doesn't get lost, you would still want to keep the overall pacing of the level intact.

A way to do this is to break the areas down into beats.



You would effectively design multiple linear levels and combine them. While not foolproof, this does help control the pacing a bit better.



Things get more complicated in an open world however.

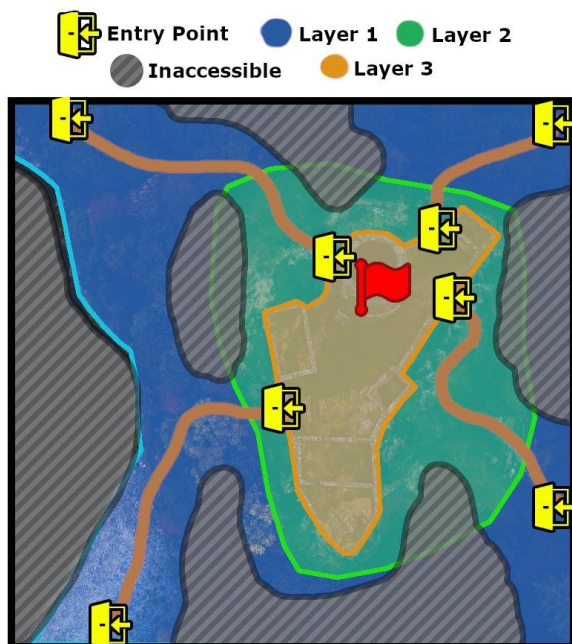
With tons of map markers on the map it can often lead to players just choosing 1 activity ("I guess I'll just go collect a bunch of collectibles") which can mean they spend extended extended periods of time doing the same kind of thing, with the same pacing.

So it can make sense to be careful with how the available choices the player has is handed out. That way you minimize choice paralyzation and can to some degree steer the pacing.

So disregarding map markers for locations, the player would generally be drawn towards interesting points they can see.

If too many points are introduced this could potentially lead to choice overload. Compare having 3-6 interesting locations you can see and go to and 20, all screaming for your attention.

Having fewer Macro points of interests that then reveal several Meso points of interests once you reach it or get closer would reduce overload and help you better predict what the player might do.



Now assuming the player is heading towards a point of interest/landmark/goal, how do we ensure the pacing for the player up to said point is decently paced?

You can limit where the player can be at specific points (for example entry points) and also layer your area.

So while you can't control their entire flow, you can more easily predict where the player can be and design around what they will see where and when to a degree.

The layering can be as dense (or not) as you want, but some form of layering is usually a good idea.

This limited entry can also be more organic, like you can only airglide over a certain part of a castle wall from one specific point instead of having doorways and linear pathways.

While restricting all (or none of) your areas (open-world or not) in ways like these might not suit your game, it can be handy to work with restrictions, layering and the like.

Games like *Batman: Arkham Asylum* used restrictions to great effect. It had a more open "world" the player could explore, but used buildings and the like to deliver a well-paced and more directed experience, which resulted in many memorable moments for players.



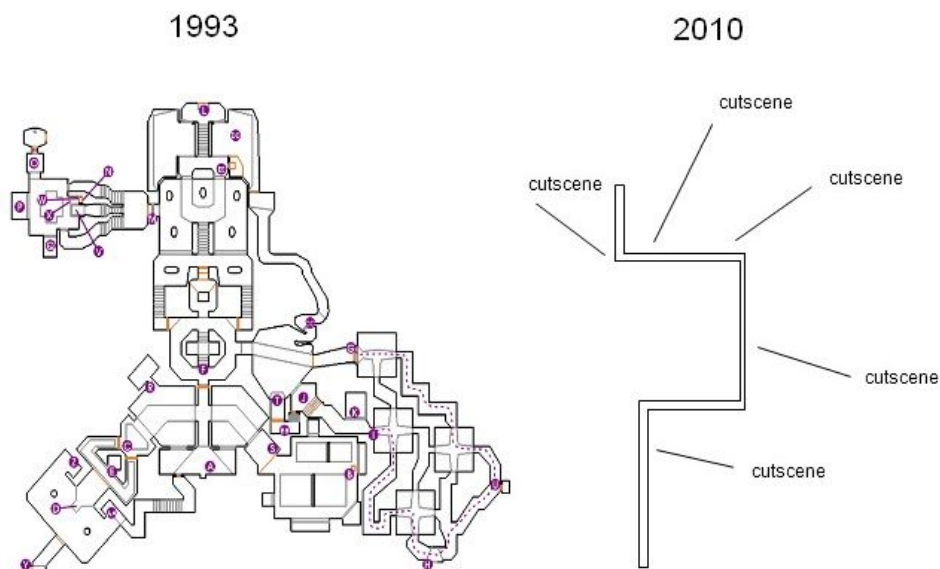
## ◆ Enhancing the experience ◆

### ❖ Choices ❖

Balancing the pacing of the experience, the flow of the player's movement and the choices the player can make in the level are both difficult and crucial.

They also go hand in hand. Too much choice can make it harder to control the pacing and flow, while too little can make it feel too linear and dull.

#### FPS map design



Above you can see a popular image that has been floating around for a while. On the left is the level E1M6 from Doom (1993) and on the right the “average modern FPS level”. It's an inaccurate and biased image that doesn't accurately represent reality, but it *does* however accurately demonstrate how player's *experienced* old and new level design.

The Doom level is actually linear in structure and classic FPS level design (including Doom) often relied on finding keys/switches for gates to proceed (as in not being completely open). Players were effectively given small chunks of more open spaces to explore. But in newer games, players *felt* like they were too railroaded, too controlled. The level on the right (see image) is their *experience* of modern levels, not what the levels actually look like.

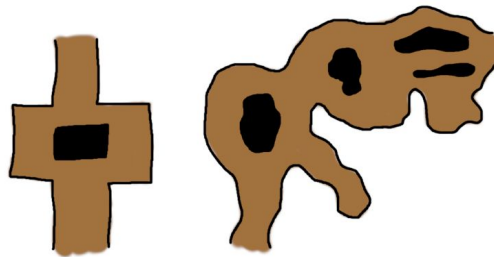
The key to breaking this perceived linearity is through choice. That players feel like they have a choice in how they move through the level. That the designer doesn't heavily 'control' them to the point where they aren't even allowed to get a little bit lost.



So how can we add choice to our levels?

More frequent alternate routes, branches or even dead-ends help break up the sense of linearity. Even 'pointless' paths help break the feeling of linearity a bit.

If you can, try and make these different routes feel, look or function differently. Going down one path could trigger an event (like a dialogue with an allied NPC) making it feel to the player that "if I didn't go down this route, I wouldn't have heard this comment from my ally".

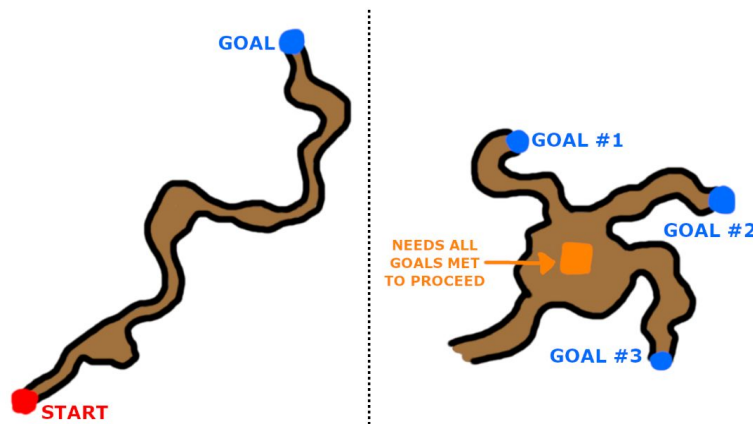


Examples of simple branching paths and dead ends.

Another element that can make a level feel very linear is if it involves moving towards a goal (for example, a building in the distance).

Although this can be a very effective structure for a number of reasons, if most levels (or even the entire game) follows this structure it can add to this sense of linearity.

So it can be worthwhile to try and break this up a bit by for example creating a level (or part of a level) that requires the player to explore more or to complete several smaller goals to proceed. If they can finish the goals in any order they like, all the better.



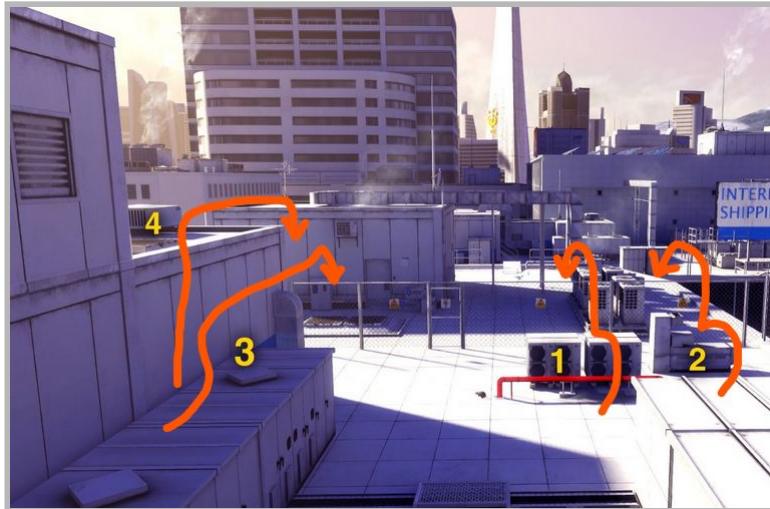
Left: A simplified version of a level where the player moves towards a goal in the distance.

Right: Example of a more open structure. You might need to find 3 parts to fix the elevator in the middle that takes you to the next level. You could find the parts in any order.



Part of the reason why levels like the one in Madagascar in Uncharted 4 was so well-received was because it broke up the more 'linear' structure of the game. It allowed players to explore a larger space and approach combat/stealth encounters from more directions (of their choice).

Adding choices to how the player can approach the same situation can also help make levels feel more open and give the sense that the player is in control. In for example *Mirror's Edge* (see below) the devs frequently offer players several alternatives in how they can move through the level.



- 1: Leap across the fence using the pipe and ventilation.
- 2: Leap across the fence using the ventilation on the right.
- 3: Wall-run and then jump off the wall to get over the fence.
- 4: Run up the wall and then jump over the fence.



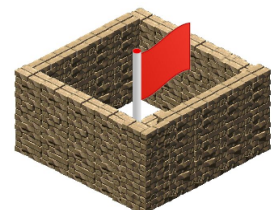
Give players small immediate choices, bigger choices and risks when they play. You want them to THINK and make choices that suit their playstyles.

Having players weigh risk versus reward helps make your levels more interesting. They don't need to be equal in terms of difficulty or balance either, so long as the reward roughly feels worth it and the challenge fair. You want players to consider the risks and think: "I could go for option A to be safe, but..."

These could be layout choices (observing the map), different approaches (stealth, platforming or combat), prioritizing which enemies to take out first, etc. These kind of choices also makes the level more replayable, especially if your level contains several smaller choices (not just a single bigger one that takes 20 minutes to reach).

But be careful of too many choices or 'meaningless' freedom. If there was a goal inside of a castle wall and you could approach it from any angle in the same way by just climbing over the wall, the players choice of where to enter from is greatly downplayed and lessened. Ultimately their 'choice' wouldn't matter.

If instead the gate had guards guarding it on one side, there was a hole in the wall on another and there was a cliff the player could airglide over the walls from then each choice becomes more interesting and meaningful.



## ❖ Enhancing with audio & visuals ❖

While it might initially seem like things like visual and audio quality wouldn't fall in line with what a level designer should know, it's important to understand how it can influence how players experience your level.

This is especially true if it involves the first impression of the game, a new area, etc. If their initial impression is "this is a high quality game/level" then they are more likely to forgive minor hiccups after their initial impression.

In Multiplayer this could be the starting point of the level, where it could be a beautiful panorama. Since it usually doesn't matter what the starting area looks like level design wise you can try and make it more impressive/detailed.

Players will also perceive similar subsequent, but less impressive events/visuals to be roughly as impressive, since those things become more routine and won't be as carefully scrutinized.

An example of this can be seen in *Dead Space 2* (see images & videos below).

Here the devs made the first depressurization scene more visually impressive with the glass breaking, the room getting ripped up, visual effects going off, etc.

In future depressurization rooms it's mostly just loose objects in the environment, enemies and the player that's affected.



*The first depressurization:*

<https://youtu.be/UyfLg5aN09o>



*Subsequent depressurization:*

<https://youtu.be/1MyxnIRXNTc?t=39>

The aesthetics also factor in to how visually interesting a world or level is.

If it takes place in more bland locations the player has already seen in lots of other games (factories, warehouses, city streets, etc.) it will make your game/level feel bland and uninteresting.

This doesn't have to ripple out across the entire game, just a single level (or a part of a level) can stand out in this way if your game wouldn't allow for it on a larger scale.

This unique element doesn't have to be visual, it can also come in the form of unique mechanics or what you do in said level.

An effective approach to make an environment feel both familiar and unique is to combine 2 familiar things into one. This also helps prevent your environment from becoming too weird.

A good example of this is the popular "Space Western" setting which throws the wild west or frontier into a sci-fi setting in space and/or together with robots.

Something so simple has served as a setting from works all over the world in most forms of media (see below for examples).



So don't be afraid to mix things up, especially if it can result in an interesting contrast. A frozen volcano, an indoor-outdoor location, pyramids in the ocean, etc.



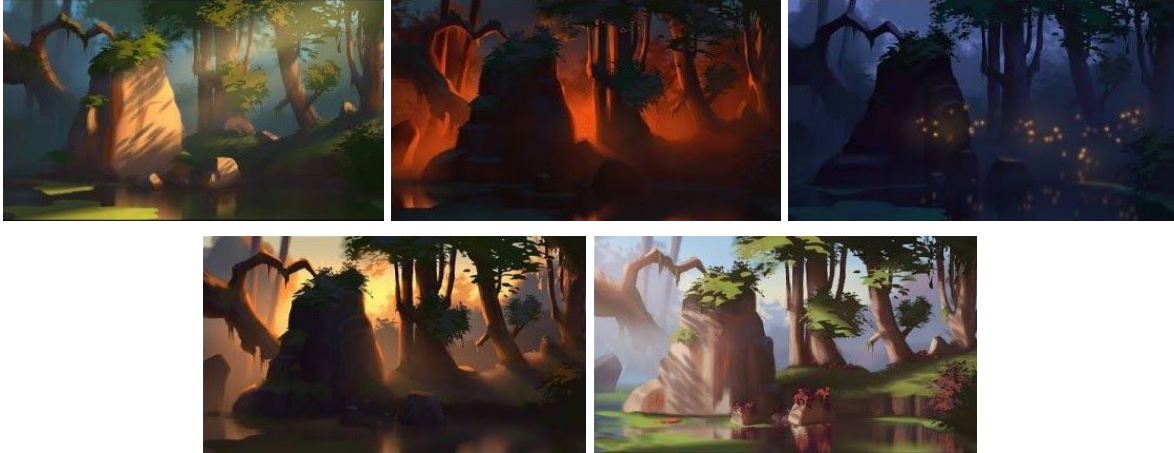
Other things to keep in mind that can enhance the experience:

- Imperfections indicating that something is "off" or more chaotic can make a level more visually interesting and atmospheric. Environments being less rigid and 'boxy' also helps it feel more real and interesting.
- Small details like weather effects, time of day changes, reflections, dust, rain, fog, etc. to enhance the mood and realism.
- Motion helps breathe life into a scene. Without it, a location can feel fake or boring, since reality is rarely completely motionless. Be it things blowing in the wind, small rolling debris or natural disasters like tornadoes.
- Don't forget the importance of sound in level design (voices, ambient, sfx, etc). Be it background noise, hearing events the player can't see or hearing enemies rummaging around in the room ahead.



## ✂ Color, lighting & mood ✂

Color & lighting can be a very effective tool to create different moods, emotions and make areas feel more distinct.



*Same scene, different colors and lighting.*

It's less about what the color, lighting, etc. might "mean" and more about how it makes you feel and in what context (for example, the color red can represent both health (like a first aid kit) and danger to a player).

Picture you were heading to the hospital for a checkup. Imagine if the hospital looked like in the image on the left (see below).

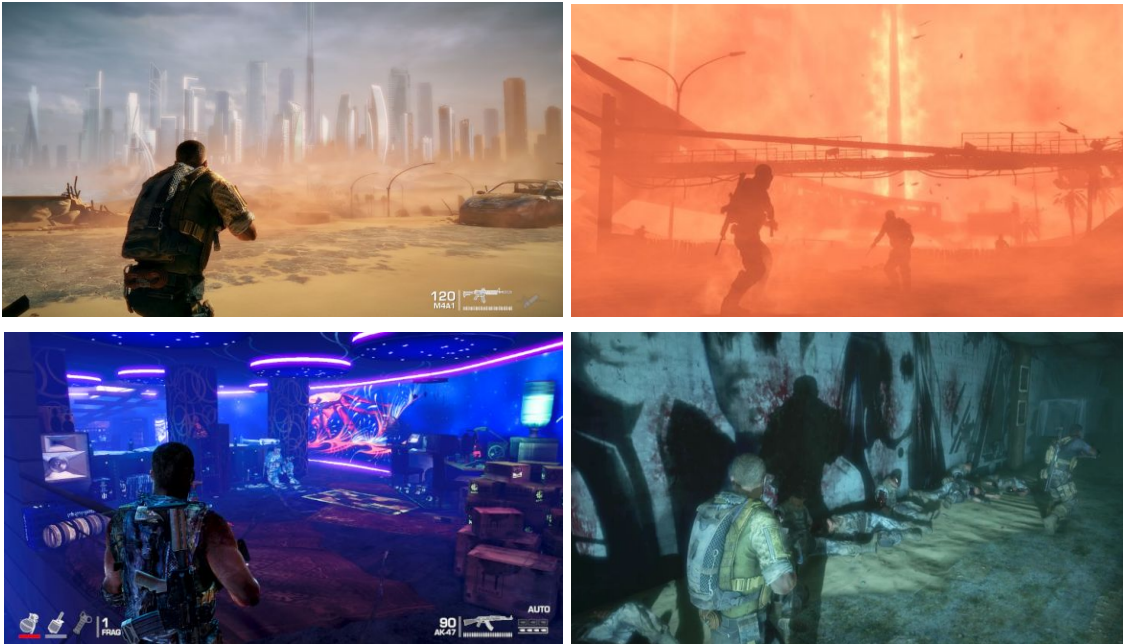


Normally if you were visiting a hotel or something you might've thought nothing of it, maybe even consider it cool or neat. But because you're now at a hospital for a checkup, you would automatically be a little nervous and the more dimly lit hallway with stronger contrasts and repeating lines everywhere could make you feel even more nervous and uneasy.

Now look at the more brightly lit hospital hallway on the right that has a more pastel color palette. Which 'hospital' would make you feel more at ease?

You can also make use of this to your advantage. For example, make normally unsafe areas feel safe, which can create a sense of unease.

*Spec Ops: The Line* (see below) frequently used things like color, saturation and contrast to evoke different moods and emotions out of the player.



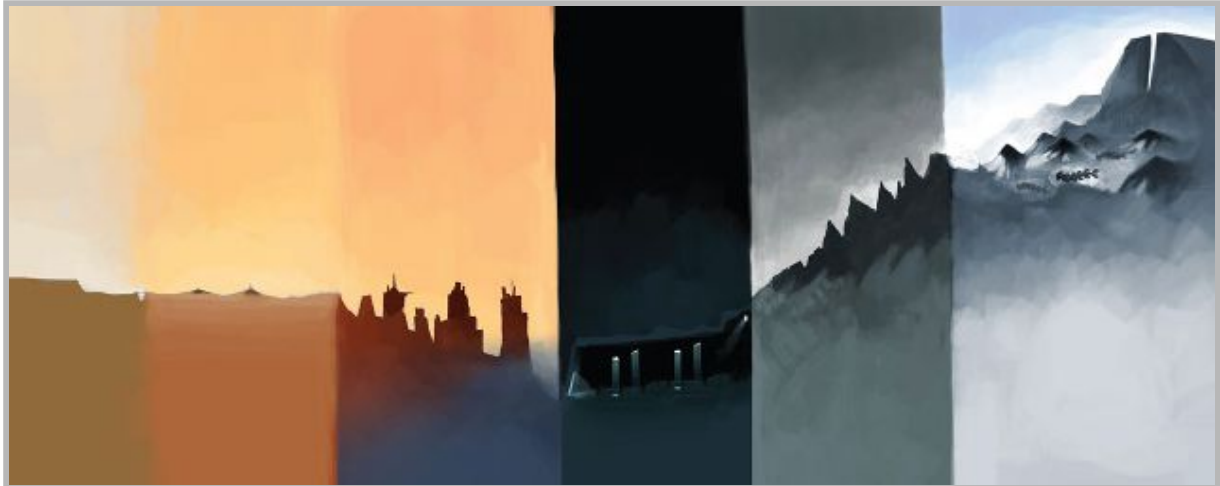
*Spec Ops: The Line*

*Mirror's Edge* used things like color and shapes to create a different sense of place. Notice how even simple color changes to the hallways and where the color is applied makes them feel and look different.



*Mirror's Edge*

Games like Journey even used color to give a sense of progression that also aligned with the mood of the game shifting.



## ❖ Narrative ❖

There are basically 3 main ways of telling a narrative in games:

- **Explicit**  
Told directly to the player. Cutscenes, mission objectives, dialogue, etc.
- **Implicit**  
Implied to the player. Things in the world like blood trails, a trophy on a hunters wall, worldbuilding, etc. Typically applies to environmental storytelling.
- **Emergent**  
Things that happen because of the player, i.e. “Player stories”.  
Choices made when they play.  
Different paths, weapons, etc. Eureka and discovery moments.  
Often created when systems interact with systems.

For level design, environmental storytelling is probably one of the more effective ways to tell a narrative or worldbuild.

This is because it doesn't interrupt the player, it makes the world feel more alive, can allow for some detective work, can quickly and effectively communicate information, etc.

When it comes to isolated local events, it can be thought of as a crime scene where the player is the detective, trying to piece together what happened in their mind. Preferably the clues should be easily noticeable for the player so they can accurately recreate what happened.



This can also be used to forewarn or teach the player of upcoming danger, as well as creating the sense of safety by placing barricades, sleeping bags, health kits and the like in a safe room.

Something as simple as a wanted poster of the player's avatar will clue players into the fact that if they're spotted, trouble is sure to follow.

If the player sees a bunch of dead bodies all of a sudden they know they should expect danger, while a text box popping up telling the player of this would be more jarring and interrupt the players pacing and flow.

Environmental storytelling can also be used for worldbuilding. This is less like a crime scene and more of a collection of larger and smaller details that help paint the picture of what the world is like.



You also need to be careful of not breaking the narrative or context of your level, that everything feels coherent. Basically, things in your level needs to 'make sense' to the player.

For instance, for gameplay purposes you might want to reward players that go off the beaten path and explore the dirty alleyways, so throughout your level you place a bunch of loot like gold coins, jewelry and the like in those areas.

But does it really make sense that in these poor areas a bunch of valuables are strewn about, not even 10 meters away from the beggar asking if you could spare them a few coins?

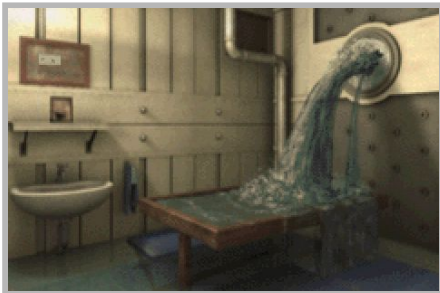




## ❖ Emotions ❖

Humans are very emotional creatures, even though we might consider ourselves to be very rational. Not only can our behaviour and decision making be influenced, but things that happen in the game can be felt more strongly and create lasting impressions.

Fear, panic and discomfort can be particularly effective tools.



If you're fighting enemies in tight corridors that will most likely trigger panic and claustrophobia.

An area becoming flooded to gradually remove breathable air can also trigger panic and claustrophobia. If these tighter areas then open up into more open spaces it can give a sense of relief and freedom. Just be careful not to make it too stressful when teaching the player something new.

The shape of a space can also matter. A big square room is more likely to have players linger while a narrower hallway with something in the distance is more likely to have them advance.

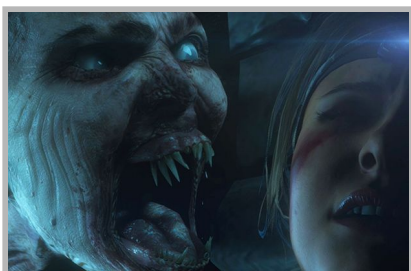
Acrophobia (fear of heights) can also be effective. This can be seen in Half-Life 2 in the section you have to travel across the underside of a bridge.

If we think back to the section in this document on Perception we know some tricks for how we can create a sense of depth (height) to help trigger people's fear and unease of heights.



In real life we also naturally tend to get a bit uneasy when someone (especially someone we don't know that well) gets too close and cross into our personal space.

This naturally means that when enemies get real close to us this can cause discomfort.



So think of what can cause fear, discomfort, happiness, relief, sadness, etc. and try and apply it when you feel that it can enhance the players experience and push the theme of your level even further.

## ❖ Spicing it up ❖

### 🔧 Wow moments 🔧

Cool or interesting events that can help pace the level and narrative. It could be 'setting the stage' of the level, spice things up in the middle or serve as a 'reward' at the end of the level. It can both be an interactive or non-interactive (like a cutscene) moment.

A 'disaster' moment can for example be a plane crash or a natural disaster like an earthquake. The Uncharted series is filled with these kinds of "wow moments", also commonly referred to as set piece moments.



The scale of a moment or event can also influence it (be it size or quantity).

For example, compare 5 soldiers marching in the distance to 10 000 soldiers (or a military parade). Playing a level in the midst of a war or raging firestorm can also result in the level itself being a "wow moment".

Impressive scenery like a beautiful view of the landscape, you getting out of a burning city and seeing it from a distance, etc. can also act as a "wow moment". These are commonly referred to as "vistas". They can show off where you've been, where you're headed, your progress (how far you've come), etc.

If you can turn these wow moments into gameplay then even better, since then you can more closely align what they *should* be feeling to what they're actually feeling.

Watching a cutscene of the player avatar being inside of a collapsing building wouldn't nearly give us the same experience as controlling our character as the building is collapsing around us (like the devs did in Uncharted 2, see below). In fact, many players would probably even feel 'safe' since they know "nothing can happen to them" in a cutscene.

But in Uncharted 2, when the player avatar Drake says "Oh shit!" we're right there with him saying the same.



Video of the building collapsing in Uncharted 2: [https://youtu.be/pWi7nG\\_45BU](https://youtu.be/pWi7nG_45BU)

## ✂ Reactivity ✂

One of the biggest strengths of videogames compared to other forms of media is reactivity. The ability to pick up and throw something in a game is one thing, but if the player then throws said thing at a character and they react the world feels more alive because it responded to something the player did. But if the characters didn't react this could make the game's world feel fake and 'gamey', because they expected a reaction.

Something so simple as physics can add a lot to a game.

When tiny stools blocks the player from moving, shooting a cup doesn't destroy it or cause it to bounce or when grenades doesn't damage the environment in some way instantly makes the game world feel less 'real'.

This includes being able to interact with things like soda machines and computers, things the player 'expects' to be able to interact with..

A game that showcased how much physics can add to the feel and enjoyment of a game is *Control*.

The player could bump into chairs and have them topple over or shoot up a table so it gets visibly damaged.

After a battle in *Control* the player could not only feel their impact on the world around them, after a battle they could visibly see the aftermath. It *looks* like a battle just took place there.

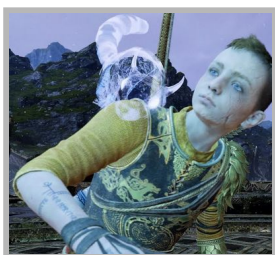


*Physics showcase from Control:*  
<https://youtu.be/ox5NOqOkg64>

Reactivity could also be 'pointless' interactions like being able to walk up and examine things or turn on/off water faucets.

It can also be the game's NPCs or even the player avatar that comments on the things the player does. Like if the player jumps into a pool of water when they need to hurry the NPC ally could say "*Um, what are you doing?*".

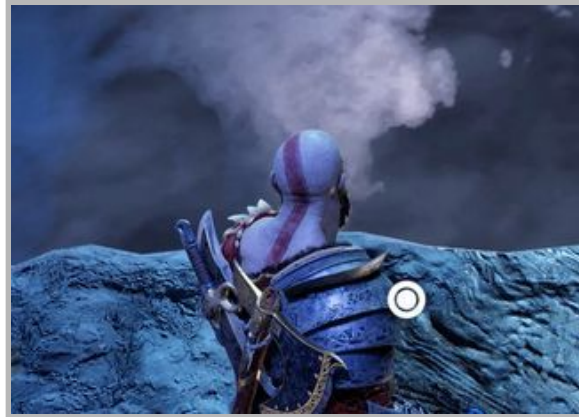
This is basically what happens in the "Marco Polo" reactive event from *Uncharted 2* (<https://youtu.be/AZ9LI3Xzrls> ).



In *God of War (2018)* there's a spirit animal you can summon that will give you health items.

But if you summon it when you're at full health it will comment on how you're not even hurt.

Something else *God of War* did was implement another form of reactivity, which is to have 'stupid' things the characters in the game wouldn't do, but could be fun for the player to try 'to see what happens'.



*The game warns that if players stray from the world tree path and fall off the edge they will die. The developers still allowed players to do precisely that and implemented a brief scene for it. As expected the player died, but they were allowed to do it and were 'rewarded' with a scene. Additionally, if the character Mimir is with them at that point, he will comment "Oh dear".*  
Video demonstrating it: <https://youtu.be/2xUcvaSxx3Y?t=30>

A problem with many forms of reactivity though is that it's the non-guaranteed reactivity that often works the best. Meaning a lot of the work you and your team put into reactivity can be missed by a lot of players. This can often mean that you need to spread out and add a lot of reactivity to ensure players encounter some of it.

Initially it might seem that it can be a lot of hard work "players won't even see", but it's these sort of details that help make games feel special and elevates the players experience. After all, the details are where the players fall in love your level/game.

### ✂ Exploration ✂

Exploration and finding things is very appealing to a lot of players.

To help motivate exploration you can place valuable items or resources in your level, some more obvious and some more hidden (so it feels more rewarding when the player finds them).

Don't be afraid to make some items really well hidden or use duds/dead ends. Like a bandit opened a treasure chest, but you can find hints to where they went and you can find their corpse with the stolen treasure.

Placing pickups more around the edges of rooms or only becoming visible if the player turns around to look at the way they came can also be effective ways to spice things up, so long as it doesn't get too predictable and weird.

The rewards don't always have to be 'physical' though. They could be alternate paths, shortcuts, vistas, flanking routes, etc.

Possibly even narrative rewards (like quests or audiologs) or jokes/easter eggs.



## ◆ Designing for combat ◆

While what I will cover in this section will mainly focus more on first and third person shooter level design, a decent chunk of it still applies to other types of games.

### ❖ Vantage point ❖

Usually it's a good idea to show off the combat space (or area the player needs to sneak through), including possibly showing off enemies.

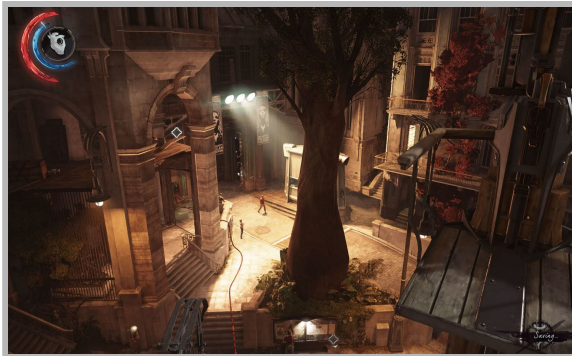
This is so they can more easily orient themselves and move with confidence through the space, along with potentially plan their approach. Basically see their goal, opportunities, risks/threats.

Opportunities can mean workarounds, cover, hidden entrances, sandbox 'tools' (traps, destructible objects, etc), eavesdrop points.

Risks/Threats can mean enemies (which enemy types are there, which enemies to take out first, is there an enemy emplacement to watch out for), alarms, snipers, cameras, etc.

But be careful of how little/much you show a player at the start of an encounter. Make their options clear, but don't overwhelm them with choices and information.

You can also opt to have the player move through an empty area first and on their way back it becomes a combat space (so they're already familiar with it, since they walked through it).



◀ *Dishonored 2* | *Uncharted 4* ▶

But said vantage point doesn't need to be high up. It just needs to provide a good overview of the area. So it could just be tall grass the player can hide inside or a safe room with bullet-proof glass windows. Vantage points should also be present indoors if needed.

But you need to be careful so you don't make the vantage points too powerful.

If the vantage point allows the player to not only see all of their enemies in an elevated position and in cover, that can easily result in the player just sitting in place and killing all enemies from the vantage point.

They would then just walk through the combat/stealth space you created, not only 'wasting' your effort, but also making the encounter far less interesting.

So to avoid this, generally try:

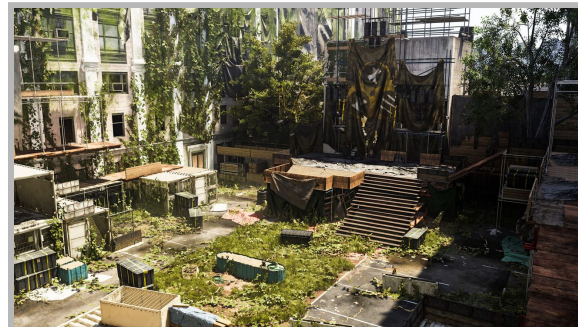
- Make it impossible for the player to kill enemies from that position (for example, the player is behind bullet-proof glass so they can only see their enemies).
- Make it a bad place to fight (for instance, they're very exposed with no cover or the cover only conceals them, like tall grass).
- Have reinforcements that spawn in a position that makes the vantage point a bad place to fight from.
- Don't make it possible to kill all enemies from this spot (enemies are inside of buildings, behind shields that would require the player to flank them, etc).



*The tall grass only offers concealment (if undetected) and the wooden boxes are brittle cover that the enemies can destroy. This makes it a risky spot to start a fight from.*

Of course vantage points might not suit your game, you might want enemies to ambush the player, etc. so it's up to you to decide if and when to use vantage points.

## ❖ Combat Spaces ❖



*Examples of combat spaces.*

When designing a combat space you generally want to try and provide choices to the player. This is to allow players to feel like they get to play 'their way' and that *their* decisions in combat lead to victory.

What this essentially boils down to is to think about the different playstyles players have and also opening up for player movement during combat.



Some examples of common playstyles:

### **Rusher**

Likes to aggressively charge down the most obvious play path to fight enemies from short to mid range.

### **Sniper**

Likes to find good spots with good sight lines, usually a fair bit away from the enemy, to shoot them from afar.

### **Ninja**

Likes to take the less direct paths (through small passages, over catwalks to the side, etc.), often trying to flank the enemy.

### **Opportunist**

Likes to take in their environment for opportunities. This often involves interactables. For instance, like running up to the mounted gun on the side, shooting gas pipes to flush enemies out of cover, shooting down stalactites so they fall down on enemies, etc.



### **Interactables**

Try and place several interactables of each type (or a variety of different ones) in your combat/stealth spaces.

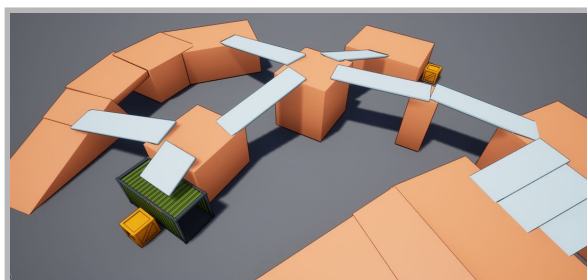
This enables more opportunities for the player (and enemy, if you so desire).

Interactables can be noise distractions, explosive barrels, gas leaks, etc.

You then try and shape your combat spaces so it enables these different playstyles.

The combat spaces don't always need to allow for every playstyle every single time, but keeping the idea of *"can this combat encounter be played in a different way?"* in the back of your mind is a good thing to do.

Also, try and design your combat spaces with verticality (different 'layers') since it helps make them more interesting. Flat spaces tend to be perceived as pretty dull.



To make combat feel more varied and less static, consider having a more dynamic combat space. Basically have the player feel that they need to look and move elsewhere during the fight.

For example:

- The combat space could change. It could get destroyed (pillars falling over creating new cover, areas become flooded, etc.) or things could move around (platforms could be lowered/raised).
- Enemy reinforcements enter from different positions or in new ways (like from a dropship that drops enemies behind where the player entered).
- Enemies can fall back deeper into the combat space or into the next room (so you can effectively break up the combat space into several smaller ones).

To further spice things up you can throw a curveball that affects one of the players 'senses'. So things that affects how the player (positively or negatively):

- See
- Move
- Shoot/Attack
- Take cover
- Hear
- Get Hurt

Negative perks could be an area that is filled with thick smoke that limits the player's ability to see. It could also be quakes that shake the ground causing the player (and/or enemies) to move slower.

A positive perk could be to allow the player to control certain pieces of cover so they can lower and raise it (i.e. lower it if enemies hide behind it, or create new cover for the player).



### Avoiding predictable patterns

Try and avoid scenarios and patterns that is too familiar to the player, especially in close proximity to one another.

Basically avoid *Enter room ▶ Door closes ▶ Enemies Spawn ▶ Defeat enemies ▶ Door open*, especially in a single level (or repeatedly over the course of the game).

To slightly change it you could have the player fall into a room with enemies in it, the player does something that sets an alarm off and activates a lockdown, etc.

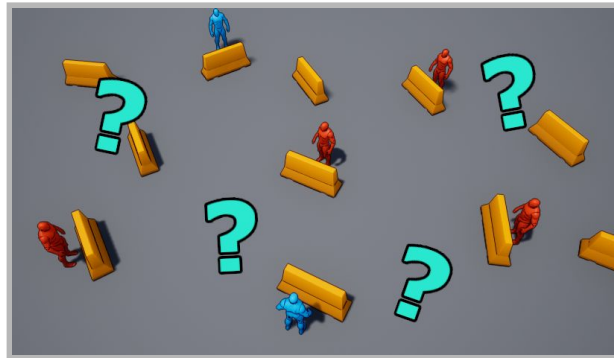
Patterns in structure like *Hallway ▶ Room with enemies ▶ Hallway ▶ Room with enemies* are also good to avoid.

Varying the scale and type of areas can also help spice things up. For example *Hallways ▶ Big room ▶ Hallway ▶ Outdoor area ▶ Medium room*, etc.

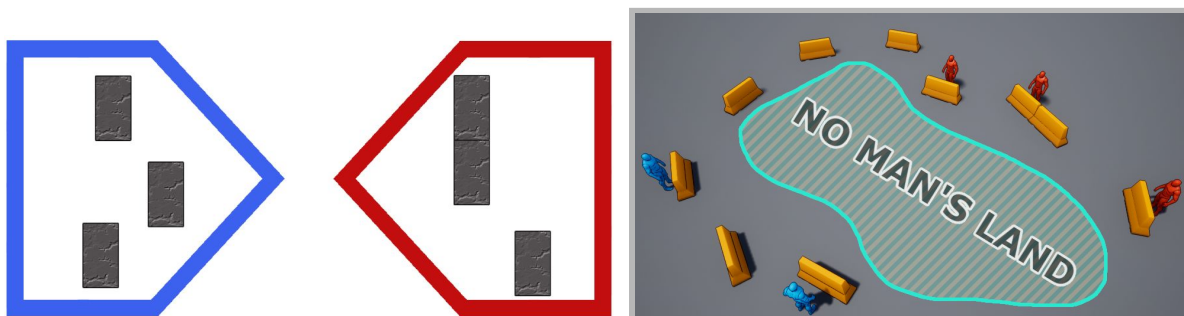


## ✂ Combat fronts ✂

When designing combat spaces it's important to consider the combat fronts. Basically, for each side (PvP/PvE) where are the fronts, flanks, rear? Not having these things clearly defined easily leads to uncontrolled chaos.



*Notice how it becomes hard to define where each 'side' is and it looks very messy.*



*Notice how it became easier to visually read and understand the space when it was organized and separated with a "No man's land"?*

Where these two sides meet forms the engagement line (the area where combat is expected to happen) with each side separated by a neutral zone (No man's land).

This neutral zone is generally a dangerous area with no clear benefits that the the opposing sides have to push through or bypass. This neutral zone could even be inaccessible (for example, a pitfall).

As you're creating the combat front also consider where the enemy is expected to attack from and how the player (and enemy) will move through the space (which areas will be more or less appealing).

As mentioned previously, you can make combat more exciting by moving the combat front mid-combat.

For example, having the player drive the enemy back into a retreat, so they move to a new position where they will also gain an advantage (like reinforcements).

If the player has allies, they could also push up along with the player.

An example of this is from *Tomb Raider (2013)*. Here the battle is fought on two platforms that are separated by a cliff, with enough cover for both sides to offer some safety.

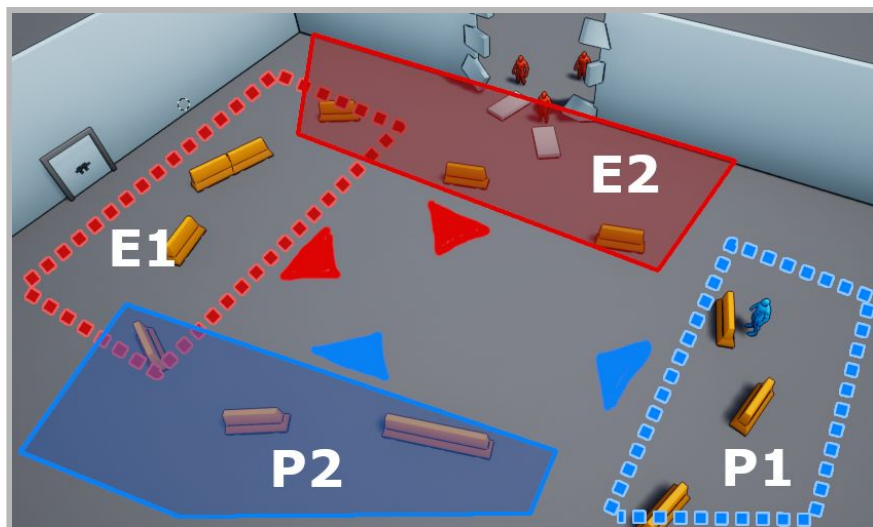
In the first part of the battle it's fought at range with the enemy at a higher elevation and constantly throwing grenades at the player, forcing them to move.

In the second part of the battle enemies slide down via ropes on the sides to flank the player, changing it up to make it more close-ranged.



Another example is to have enemies blow up a wall or door to attack the player from a different direction. The explosion would also let the player know that enemies will come from a new direction.

This is so it doesn't feel like the enemies 'appear from nowhere', which can happen if they don't 'announce' their entry if they spawn from out of sight (behind or to the side of the player).



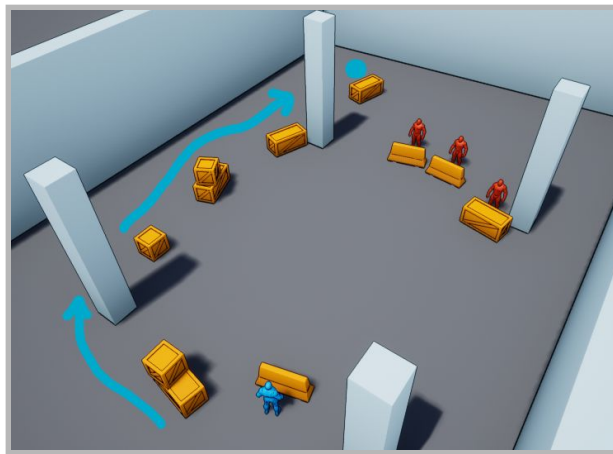
The first combat front takes place between **E1** (enemy) and **P1** (player).  
The second combat front is moved to **E2** and **P2** when the enemy blows up the wall.

## ✂ Flanking ✂

Flanking routes should serve as the bridge(s) in the neutral zone between the combating sides.

As such, most of the cover in the neutral zones should be focused around the flanking routes. But the cover along the flanking routes should preferably not be good permanent positions (which would defeat the purpose of the flanking route).

It's important to note that the flanking route should preferably have cover placed along the enemy flank (so you can actually attack them on their flank). See image below for example. The route should also be noticeable (with visual/audio helpers, if necessary) to the player from their own side of the combat front (so they know that flanking is possible).



However, generally you don't want the flanking routes to be instantly available. The player should have to work their way through their own frontline a bit first (or after a set period into the encounter it becomes available).

For example, Titanfall 2 could place a pit in the middle of the neutral zone and allow the player to wall-run across or take the longer route with more enemies.



The devs in general liked getting the player to use their mobility (double jump, wall-running, climbing, sliding, etc.) to reach/use the flanking routes.

To make flanking routes even more interesting they can basically be turned into goals themselves or as things that change up the pace of the fight a bit. For example in *Gears of War* there's a level where there's an enemy in a machine gun nest. In the flanking route there's an enemy sniper the player first has to bypass and then they can use said sniper position to take out the machine gunner. Not only does the player get to take down that pesky sniper, they also get to turn the tables on the machine gunner.

## ❖ Cover ❖

The main function of cover is to block line of sight and/or attacks, which is important in both stealth and combat scenarios.

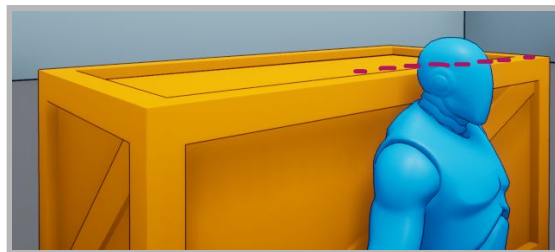
What this generally means is that the more damage blocking and less visible a piece of cover makes a player, the safer they will feel.

This makes how cover is implemented very important when it comes to creating good combat spaces.

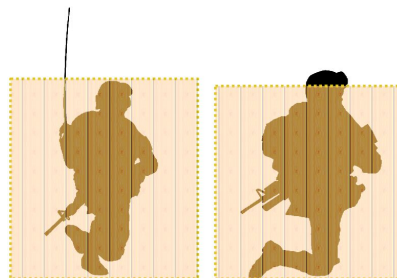
There are basically 3 types of cover:

- Blocks vision & damage. Hard cover, like rocks and stone pillars.
- Blocks vision, but not damage. Soft cover, like bushes.
- Blocks damage, but not vision. Hybrid cover, like energy shields.

In addition, the metrics of the cover should be checked and standardized so you don't end up in situations where the players head is exposed while in cover (and if unintentional, enemies).



*If the player can get shot in the head while in cover by enemies (in PvP or PvE) it can feel unfair and weird (“Why isn’t my character taking cover properly?”).*

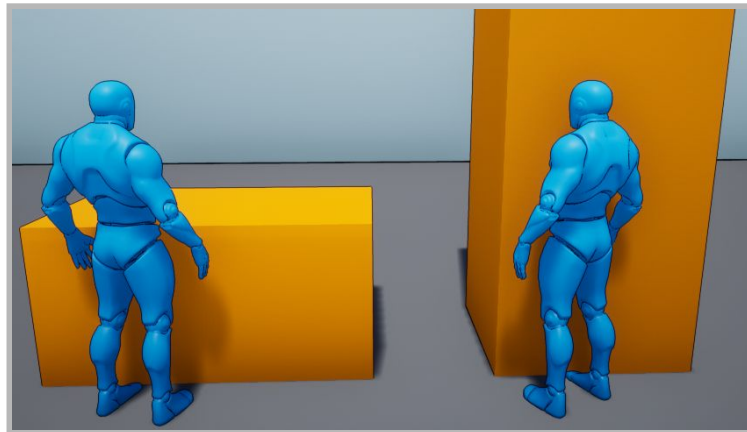


*It's usually a good idea to indicate where enemies are, even when they're behind cover. You could make it so that enemies stick out a bit when in cover, but that would mean that the player can attack them in cover (which you might not want). Instead you can have parts stick out which the player can't damage (like an antenna) instead.*

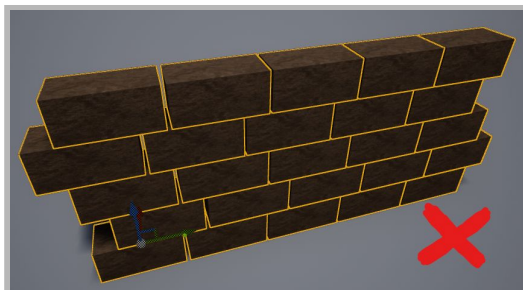


The height of the cover is also important to keep in mind, since it can greatly affect the players (and enemies) options.

For example, low cover allows for a better view, firing over the cover and to vault over it. Meanwhile high cover only allows the player to shoot at the corners of the cover.



Cover should also be implemented so it works well with the games core mechanics and preferably documented (in editor, with for example comments or color coding) how the cover is planned to be used (vaulted over, ran around, etc). Both to remind yourself and inform your team so the design intent remains intact.



Making the cover assets modular can help make it more nuanced and varied (so you don't just have a single short and long box).

However, you also need to consider performance (keeping asset & poly count down) so making cover out of individual sandbags isn't ideal.



*Making cover pieces that can fit together into larger chunks and allow for modular cover placement can help make a space look better and allow the designer more flexibility when creating the combat space.*

*But while things like sandbag or brick walls should have larger sections, singular bags/bricks can help 'glue' things together or to add some visual variety, so they're still good to have.*

Also, when constructing cover try and avoid crates/boxes. They're uninteresting, often don't make much sense in the location (especially in large quantities) and since they're so common it can take players out of the experience if they see a bunch of crates.



*The dreaded wooden crate.*

Instead try and make the cover look more like it organically fits into their environment.



That said, even when making more natural looking cover and while sticking to the proper metrics is good (for readability and so it works as intended design wise) it can still be problematic.

This is because it can very easily look fake/'gamey' and it can scream "this is a combat space". In some cases you might want this, but usually this can take players out of the experience, telegraph what's going to happen (taking away any planned surprise), etc.



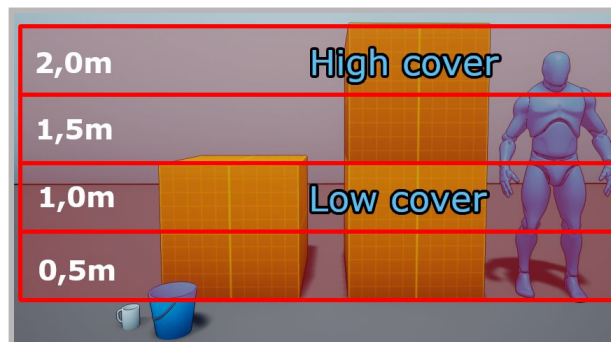
*Here the player instantly knows how the negotiations will go with the NPCs they're meeting since it contrasts so heavily from the areas they were just in and that it looks like a combat space.*

You basically want to avoid this feeling of a space being 'designed for combat' and that it looks more natural.

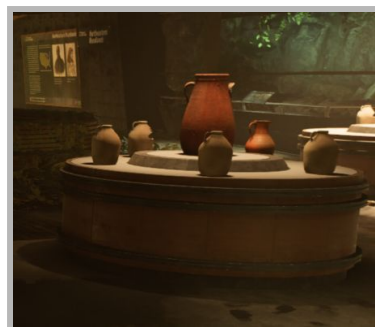
A way to break this up is to use these pieces of cover even in non-combat spaces and to add more elevation variation (while still keeping the metrics in mind).

Basically try and make your world look more coherent (like the rest of your game) and natural, so you don't get a sharp contrast between a non-combat space and a combat space.

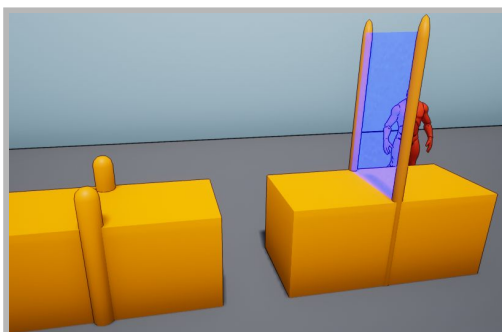
If your combat spaces only have most of the main objects in the environment be of Low cover, High cover or pillars (see image below) and the rest of your game doesn't that may cause a disconnect.



In the image above, the risky area is the 1,5m one. This is the area you want to be careful with when trying to break up your cover. This is because you don't want to make Low cover look like High cover and potentially mess up things like not being able to peek up behind cover to shoot or not being able to vault over the cover.



*Using small props can also help break things up a bit and give the sense that this waist-high blocky/circular shape actually serves a purpose and is used. Having these objects be breakable or 'loose' so they can be knocked away by gunfire and explosions would be a good idea though.*



You could also use solid elements that are a part of the cover, but don't disallow vaulting or vision from the entire piece of cover. Parts that stick out into the 1,5m area.

These could potentially include solid blocking 'sheets', glass or holographic screens.

In other words, only changing small sections, not the entire cover piece.

## ✖ Cover placement ✔

We already talked about 3 different types of cover (hard, soft & hybrid), but cover can also have 3 different motion states and a dynamic state:

### **Static**

Fixed in place and won't move. This is generally what most cover falls under.

### **Moveable**

Can move depending on various factors. For example a pushable crate.

### **Moving**

Moves on its own. Like a moving vehicle.

### **Dynamic**

Has special interaction or properties. For example, destructible cover.

A static indestructible piece of cover has no special dynamic. Meanwhile cover that can get destroyed or blow up can force the player to move around more and to also flush enemies out of cover more easily.

This destructibility can also be applied to things like walls and doors, which can result in more chaotic and unpredictable gameplay.

Moveable and moving cover can really help make a level or space come to life.

An example of this is in *Gears of War 2* (Indigenous Creatures level) when the player has to use a huge worm like monster as cover that will walk of its own.

The player can direct the creature by shooting down glowing fruit that the monster will move towards to eat.





When it comes to placing the actual cover, you should consider:

- The viable/intended ranges of the combat space (short, medium long/sniping).
- Having more than one piece of cover for both the player and enemies gives them more options and make the combat space more interesting and unpredictable.
- One (or preferably more, so the player has options) pieces of cover should be available to the player at the start of combat. This allows them to assess the situation and decide how they want to approach it.



- The placement of cover (like in the image) to for example allow the player a chance to assess the situation at the start of combat, can result in the game becoming predictable if you're not careful (as we mentioned previously about how players can instantly tell combat is about to start).  
To avoid this you can make use of dynamically spawning cover (a pillar falling over), the player character automatically creates cover (flips a table over) and place cover in non-combat spaces.

To control the difficulty of a combat space, you can manipulate the cover in certain ways:

### **Amount**

More cover means it becomes easier for the player (and the enemy, if they have access to more cover on their side) to move around and attack. Less can result in more risky open spaces.

### **Size & coverage**

Bigger and taller cover offers more protection, as does cover that protects the player from more angles. Smaller cover with less coverage from several angles does the opposite.

### **Concealment**

Some (or all) of the player/enemy cover only blocks line of sight (like bushes), which offers no protection.

## Spacing

The distance between different pieces of cover can make moving from cover to cover more (or less) risky. Generally the distance between cover shouldn't be too far apart (see the Metrics section of this document for idea for tools to help you with this).

You need to consider the player's speed so they can move between cover to cover relatively safely if you want players to have better flow in combat.

If the player feels that other pieces of cover is "too far away" they might end up staying put.

## Temporary cover

Destructive cover can force the player (or enemy) to keep moving, as can cover that moves on it's own.

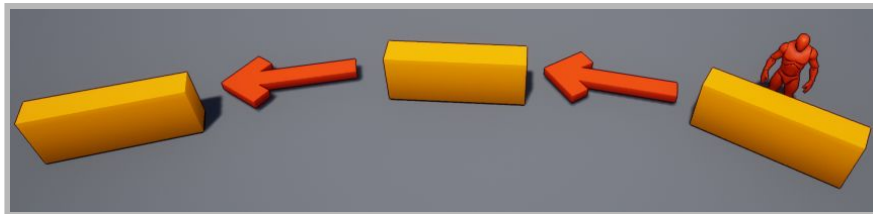
The cover could also be timed to disappear after a set time or alternate (for example, blue cover boxes appear when the red cover boxes disappear), further promoting movement.

# ❖ Enemies & encounters ❖

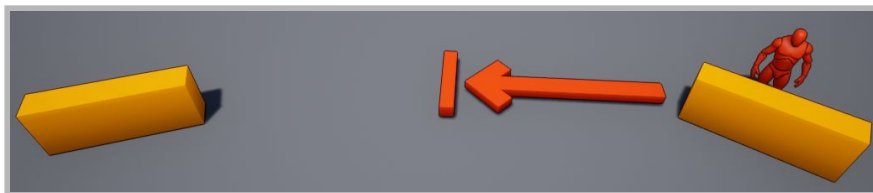
## ❖ Enemies & cover ❖

When placing cover you also have to consider how the enemy AI will use it and how the placement of it can affect the difficulty of the encounter.

For example, more cover options for the enemy will mean they will move around more, which will make the encounter harder than if they just sat in one piece of cover.



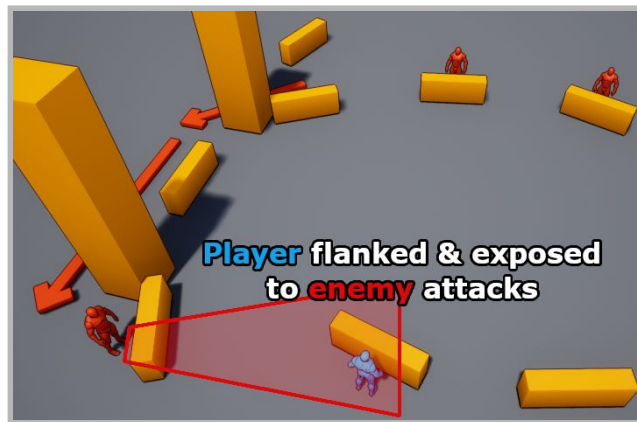
To prevent this (assuming the AI isn't set up to run between cover regardless of distance) you can break up cover to reduce enemy movement options.



This prevents you from having to fiddle with the AI to create specific solutions for specific encounters. This also benefits the player since their idea of how the enemy works isn't broken (for example there's tons of cover around the enemy and in some encounters they move towards it and sometimes not).

This however still allows for situations where if an enemy is flushed out of cover (for instance, the player uses a grenade) and another piece of cover (which was previously too far from them) could now be a viable piece of cover to move to.

In addition, keep in mind of how the cover will work for the enemy, not just the player. A good flanking route for the player can be a good flanking route for the enemy as well and if you're not careful you can make encounters unintentionally difficult.

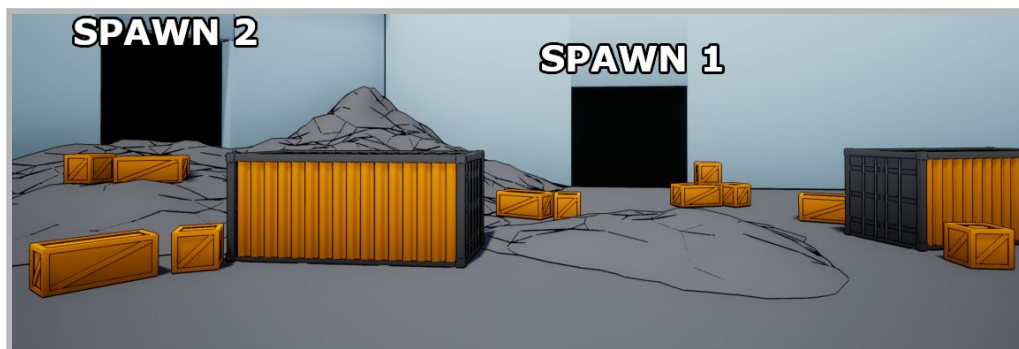


#### ✂ Spawning & tracking enemies ✂

How enemies spawn is important since if not done properly can lead to frustration, too easy or too hard encounters.

If enemies only spawned in the center of the player's field of view, close to where the player is aiming then it would be fairly easy to just gun them all down or take them all out with a grenade.

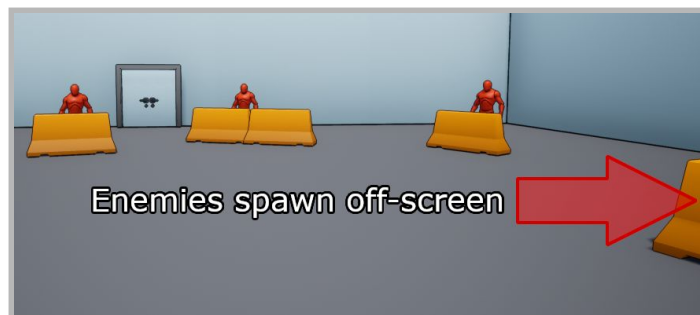
But if we instead split the enemies spawn point up but still keep both on the same screen (so it doesn't feel like they're secretly spawning) it becomes more interesting.



*Imagine if 5 enemies only spawned at **Spawn 1**. It would be easier (and less interesting) than if 3 enemies spawned at **Spawn 1** and 2 over at **Spawn 2**. In this example both spawns would still be in the same field of view, so the player could keep track of both enemy spawns.*

But if we wanted to make it harder, the enemies could spawn from off screen. This can however run the risk of the player feeling like enemies “spawn from out of nowhere”. Depending on what you’re going for, this might be intentional.

But usually some kind of indication of enemies spawning (using for example audio) is needed, especially if it’s not clear *where* enemies can spawn from. If you for example had a bunch of locked doors all over your combat space and some of them spawned enemies in some encounter and some not, then it becomes even harder for the player to predict if and where enemies spawn. Especially without some kind of notification and *especially* if the player is already fighting other enemies.



Having enemies spawn at different elevations can also make things more interesting *and* more difficult. Not only because player have less of a tendency to look up, the elevated position also gives them an advantage over the player (we’ll explore this more later in the *Line of Sight* section).



You can take these things into consideration and also alter it so the spawning area for the enemies can be less advantageous to balance it out.

For example, enemies spawning off-screen might not have access to a lot of cover that allows them to easily get close to the player or the enemies spawning at the elevated position don’t spawn with any cover.

As we’ve mentioned previously using things like dropships can help signal when new enemies spawn, as can forcing enemies to climb/jump down from elevated positions.

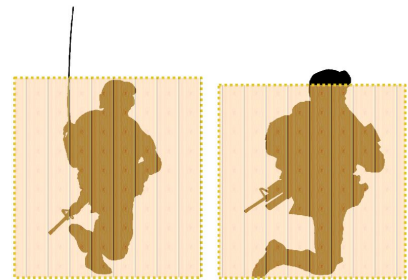
The goal should be to make it feel fair to the player.



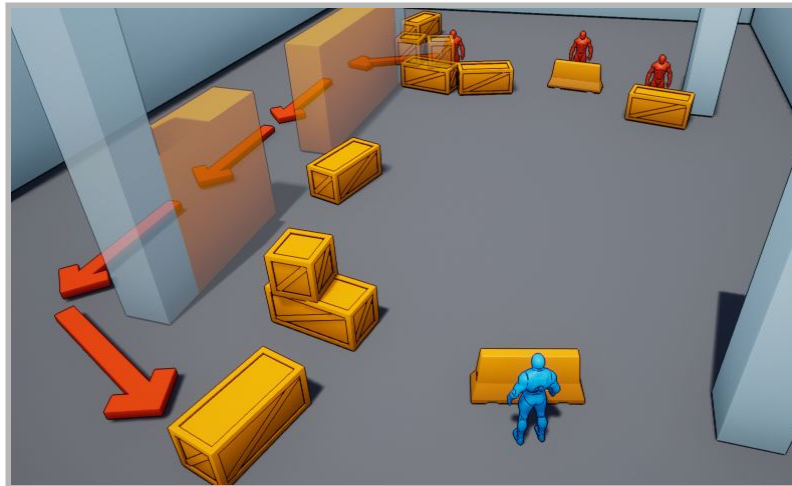


How easy it is for the player to track enemies on the battlefield also influences difficulty.

As we've mentioned previously, allowing the player to see some part of the enemy while they're in cover helps the player track the enemy (see image on the right).



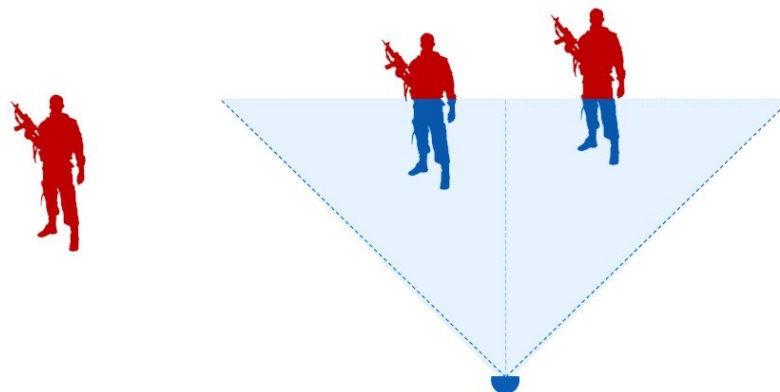
Low cover allows the player to more easily see the enemies movement (like when they're flanking) while high cover hides information from the player.



*Vision blocking high cover makes it harder for the player to keep track of enemy movement, which can more easily lead to the player getting overwhelmed.*

How far apart enemies are (and can be) also influences the players ability to track enemies. If enemies are limited to only move around in a single concentrated area the player can see the entirety of in their field of view, tracking enemies will become easy.

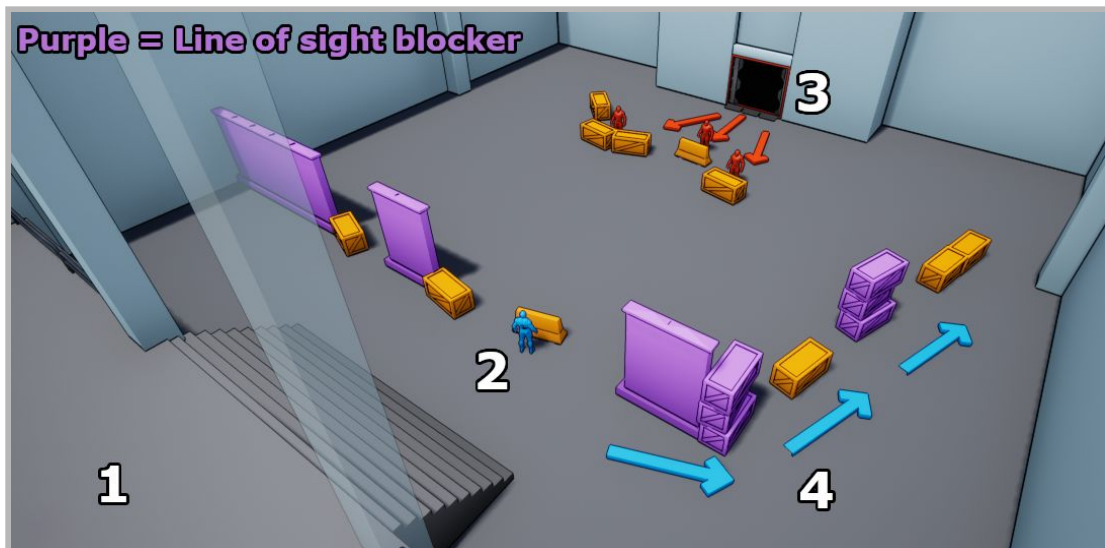
If on the other hand enemies either spawn or can move to different locations in the combat space (that brings them outside of the players field of view) that will ramp up the difficulty of tracking said enemies.



*The enemies on the right are both inside same field of view of the player, but to look at the enemy on the left they would have to stop looking at the enemies on the right side. To keep track of both the player would have to constantly shift their focus back and forth.*

*If the enemies also moved around then the player can even lose track of said enemies.*

To look at an example of an easy and basic combat scenario, see the image below. Here we have centralized enemies with more limited low cover options, LoS blockers (in purple) to allow the player to disappear from the enemy LoS (line of sight), a No Man's Land/Neutral zone in the middle and a flanking route (mainly for the player).



1. The player enters the area here. From the elevated position they get a good view of the room.
2. When the player reaches this point, enemies will spawn from 3 from the single large door in the back.
3. Enemies cover options are limited and close together, making it ease for the player to track them.
4. The player has access to a flanking route that will expose the enemies sides.

But if you wanted to increase the challenge for the player, here are some examples:

- Having many movement options for the enemy, making them harder to hit.
- Using a no man's land around the player, discouraging them from moving.
- Allowing the enemy to flank and attack the players exposed sides, forcing them out into this no man's land.
- Having multiple spawn points for the enemy on opposite sides of the map, off screen from each other.
- Having multiple avenues of fire which means the player has to balance many focus points.
- Giving the enemy the height advantage on the player.
- Positioning line-of-sight blockers to break lines of sight to the enemy, allowing them to be lost.

Just keep the balance of the encounter in mind, so that it feels fair to the player.

## ✦ Encounter design ✦

When a player is tasked with defending themselves breaking encounters up into waves that increase in complexity and difficulty is generally a good approach to take.

Usually the first wave is weak and is there to allow the player to mentally absorb the space and assess the situation.

### Enemy types

What enemy types to use during your combat encounters is important to consider, since different enemy types can drastically change the difficulty and pacing.

Enemies have individual difficulty (i.e. how difficult a normal rifleman is) and can become more difficult to deal with when working together with other enemies.

For example, an enemy that has a weak pistol and just keeps throwing smoke grenades isn't that dangerous on their own, nor might the enemy with a shotgun. But when the shotgun enemy can advance under the cover of smoke they could become a serious threat.

So consider the different enemies effective ranges (short, medium, long), their strengths, weaknesses and how they can work together when designing your combat encounters.



The difficulty also increases when you introduce a new enemy type for the first time, since the player isn't familiar with what they can do, how much damage they deal, their behaviour, etc.

As enemies of different types spawn the player should be able to discern what types are in the mix so they can formulate a counter-tactic around it. Meaning if you decide to spawn new enemy types in the second wave of enemy reinforcements it might not be ideal to discreetly spawn a flamethrower wielding enemy in the players flank.

But how close/far you space enemies between themselves and the player also matters, since the player might need to keep track of several high threat enemies.

Also, be mindful of not spacing/spawning enemies too far away since it can drastically reduce them as a threat (20 seconds for a shotgun enemy to reach the player, for example).

## Waves & pacing

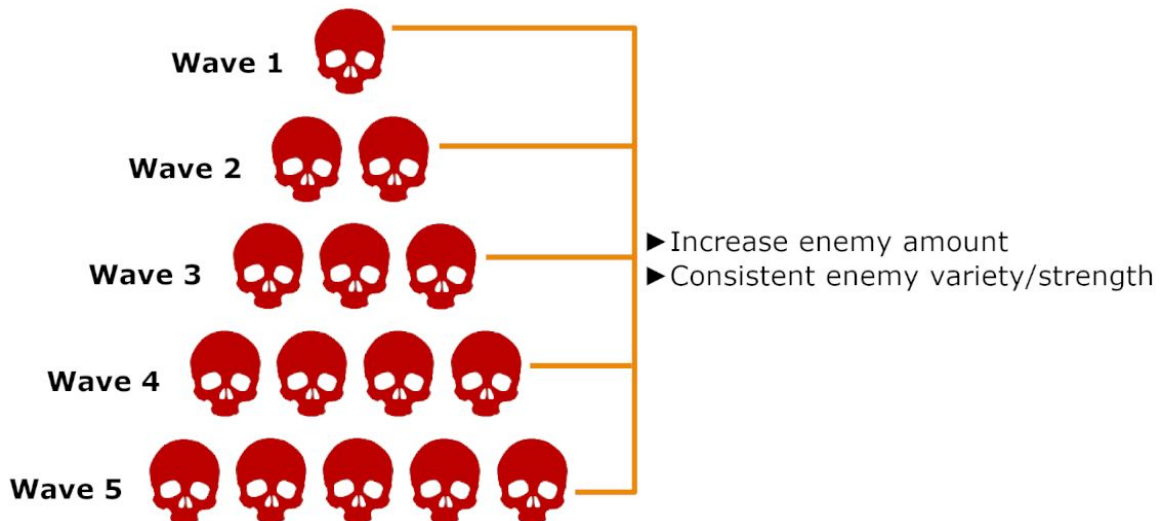
Breaking encounter up into waves (each one consisting of one or more enemy) will help you pace the encounter.

Giving waves a theme (short range round, sniper round, etc.) can help prevent the rounds from becoming too chaotic (like mixing short range, snipers and another enemy type) and also more tailor made for a certain kind of challenge/experience.

For example, one encounter could have a machine gun turret enemy and snipers. The challenge would be to get up close to the enemy since that's their weakness. You could then design the encounter/level to provide flanking options, create smoke screens, receive fire support from allies, etc. to help them deal with these enemies.

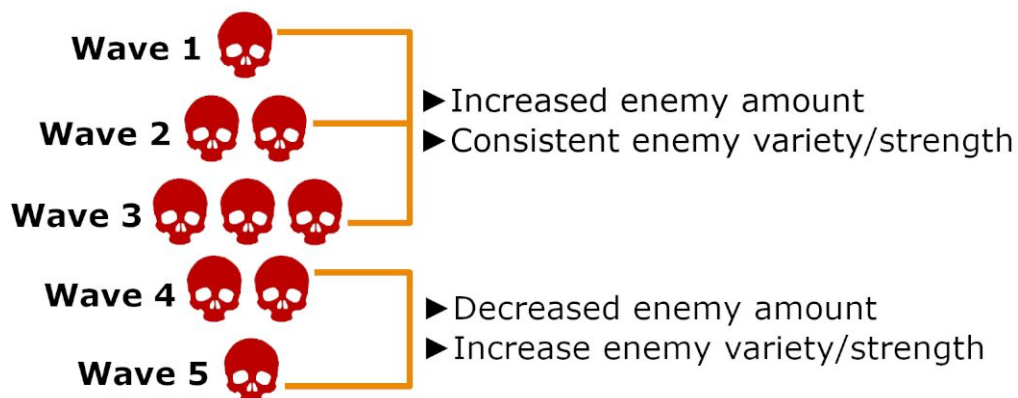
You can even play around with ideas like making the final wave unbeatable, forcing the player to flee or move towards a revealed goal.

Two ways to change the intensity of a fight is either via the triangle or diamond pattern. The triangle pattern keeps the types consistent, it just increases the amount of enemies.



The diamond pattern on the other hand changes the types/strength but to keep the difficulty in check also reduces the enemy amount.

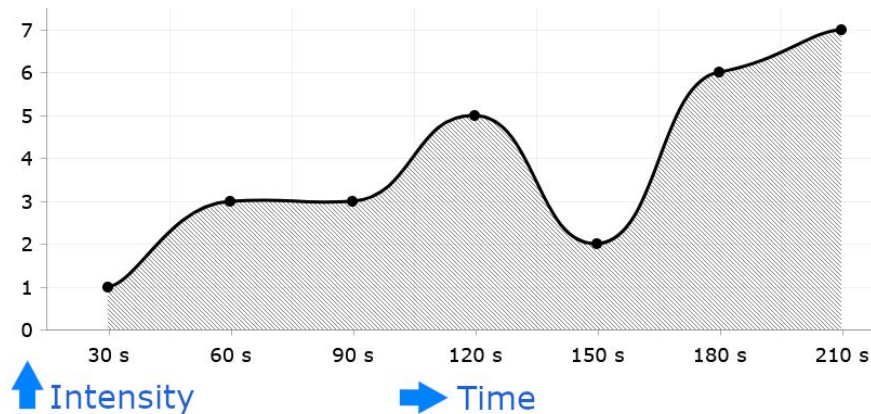
Generally the diamond pattern tends to be harder than the triangle one.





But a consistent gradual increase in difficulty isn't always ideal. Not only can it limit what you can do, it can also make encounters too predictable.

So how you pace your encounters also matters. Not only to give the player some time to breathe but to also allow for narrative, introduction of a new enemy type, to replenish ammo, etc.

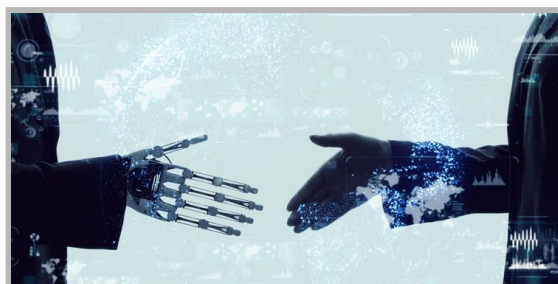


While it's a good idea to have peaks and valleys along with increased intensity, it's up to you to decide how to arrange them, as well as how long to hold on certain peaks and valleys.

### 🔧 Allied NPCs 🔧

Allied NPCs should ideally never take away from the player's enjoyment and satisfaction. If a player works hard during a fight but has a cool moment 'stolen' from them by an allied NPC it can feel a bit sucky.

Allies should also ideally never feel like a burden for the player, especially if it comes to the point where the player feels like they have to babysit the NPCs. Occasionally it might be acceptable during a specific narrative moment, but it should be handled with care so it doesn't frustrate the player.



Having allies block the player's movement is also generally very bad. It can not only make moving around the level for the player more annoying (getting blocked in) but if it happens in combat and the player gets hurt or even dies because of it then that can very easily lead to frustration.

Depending on the game, having allies be more or less locked to certain pieces of cover and not move around that much might be a good idea, since the risk of messing things up for the player is reduced.

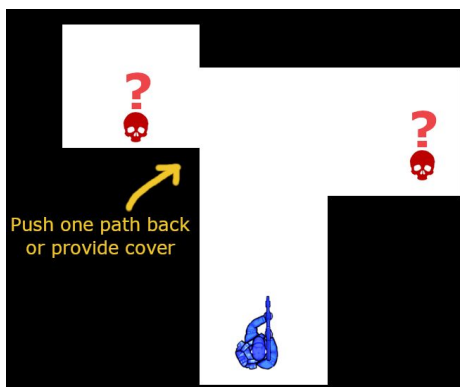
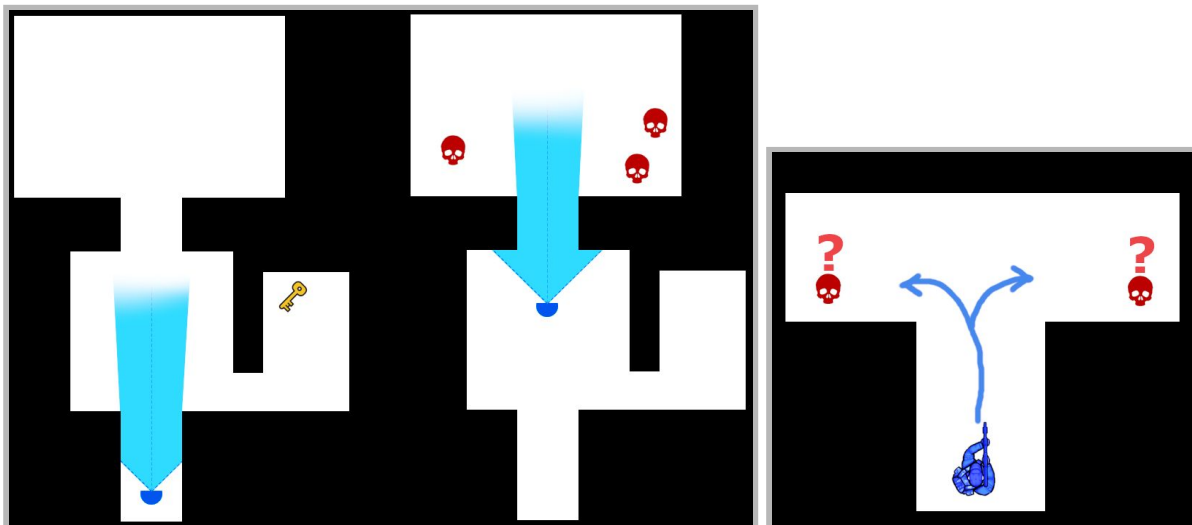
Allies can also be effective to steer players in the right direction by for example having them rush to where cover is at the start of combat. This also gives the player a clear understanding of where the player/allied front is and where the enemy front is. However, try to let players control the flow of the combat. Let players decide when to advance a push, etc.

## ❖ Line of Sight for SP & MP ❖

### Horizontal LoS

Consider the players Line of Sight, both in terms of depth and what can block their view. The further away the player can be notified of various elements (like enemies and pickups) the easier it will be for them to react and plan for it.

This can directly influence the difficulty of encounters and can cause things like 'door problems', when the moment the player walks through a doorway they're attacked by enemies, etc.



T-junctions can also be problematic, especially in multiplayer. If the player has two or more choices and choosing one of them doesn't give them some form of protection from the other it can lead to lucky/unlucky situations that don't feel fair if they decide to look left but an enemy was on the right (and the player gets shot and dies).

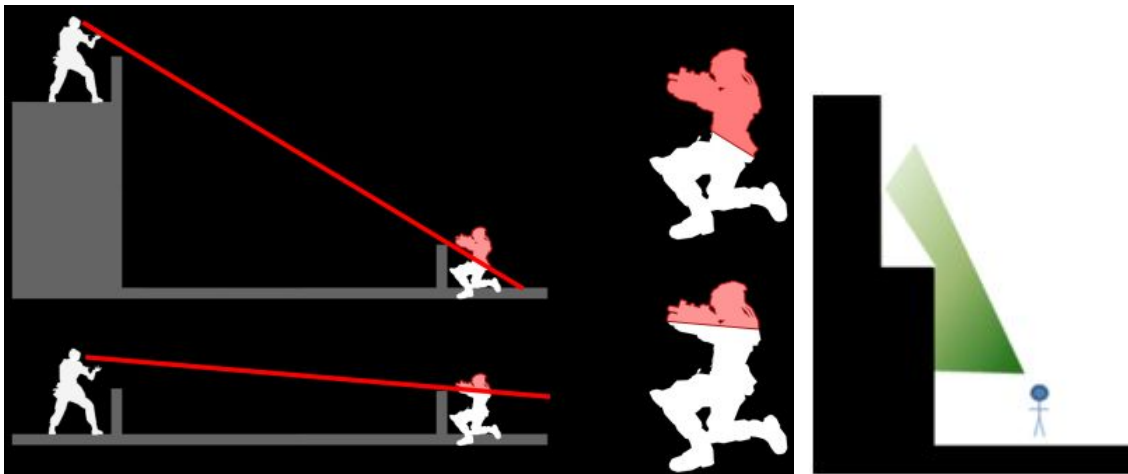
The player can only focus on one point at a time and making them vulnerable from several points at once can very easily feel unfair.

The solution is to push one path backwards/forwards a bit or to provide cover, so the player can more easily focus on one corner at a time.

## Vertical LoS

Elevation and angles also influence how effective cover is. Higher elevation generally gives a defensive and offensive advantage.

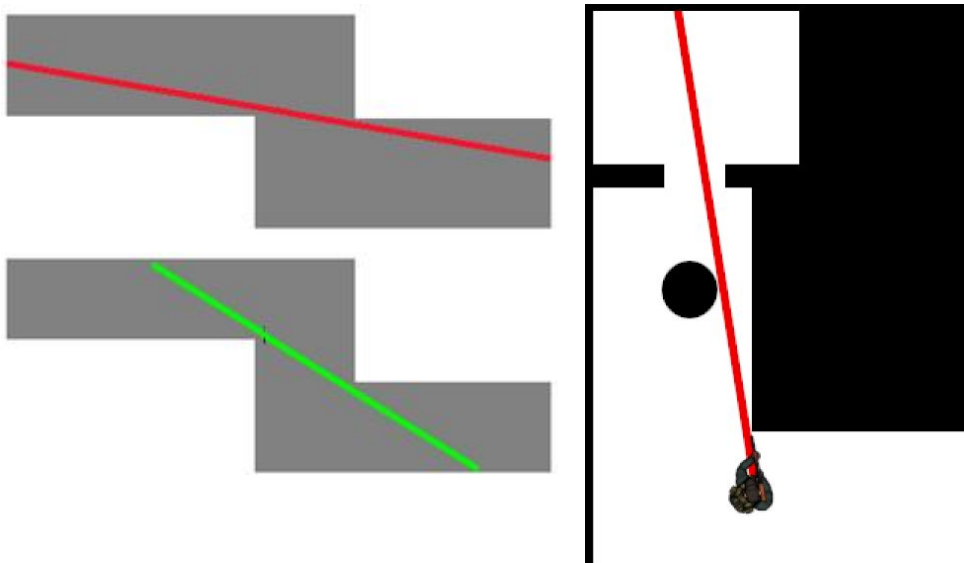
Elevation usually affects how much the player can see of their environment (i.e. from a tower you can get a better sense for the surrounding environment).



## Tight angles

Tight angles can often cause overpowered sightlines, especially for low time-to-kill games (CS:GO, Rainbow Six: Siege, etc).

You can avoid these by slightly shifting your layout a bit and double-checking the areas where potential problematic and unintended sightlines might appear.



*Above are examples of how you can accidentally create unintended sightlines that can imbalance your level.*

*The bottom left one is what you should try and aim for with corners, if you want to control it better.*

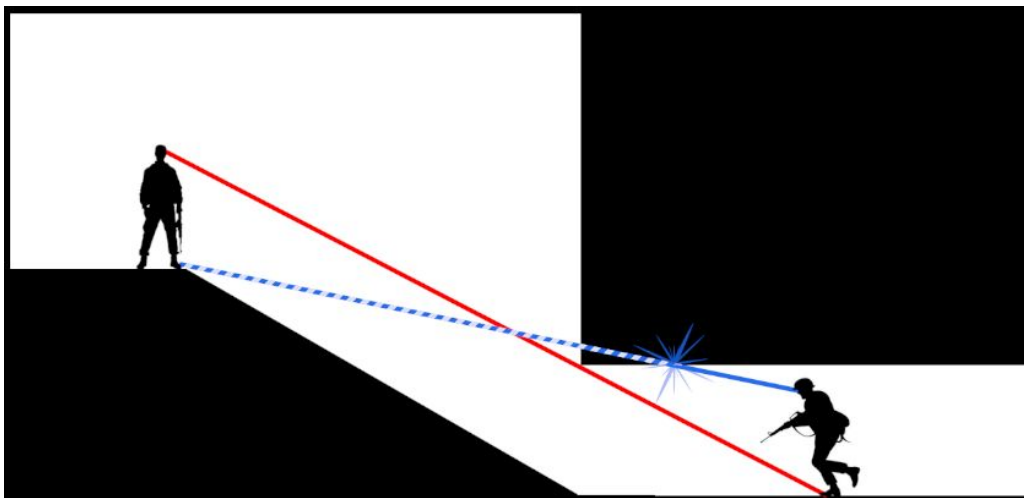
Pixel/Touching sightlines that are not intended should also be avoided. This is less important in singleplayer, but in multiplayer can cause unintended imbalances in the level.



## Ramps

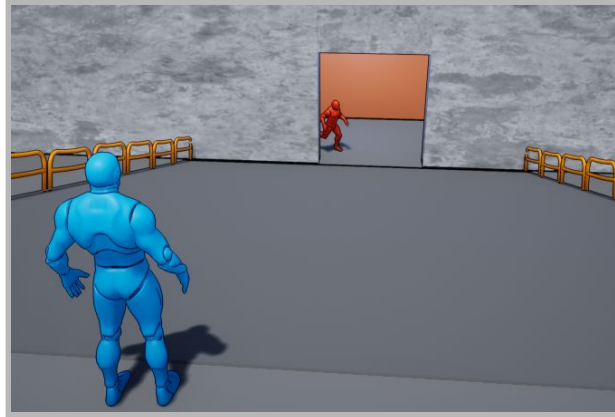
Ramp (and elevated positions) sightlines is also something you need to be aware of. If a player at the top of a ramp can see an opposing player (their legs) before they can see them that can create an imbalance in the design.

This is generally caused by a higher elevation of one player while something like a roof is blocking the other players view.



*Notice how the player in the elevated position will notice the other player (and be able to attack them) before the player approaching the ramp can even see them.*

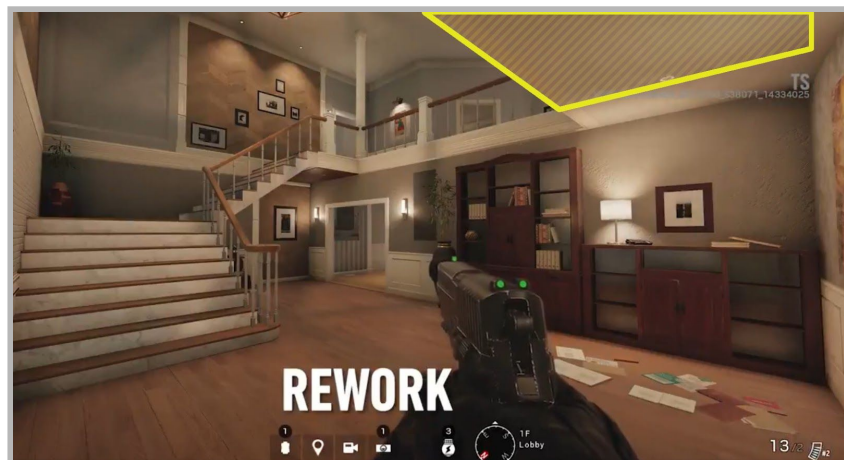
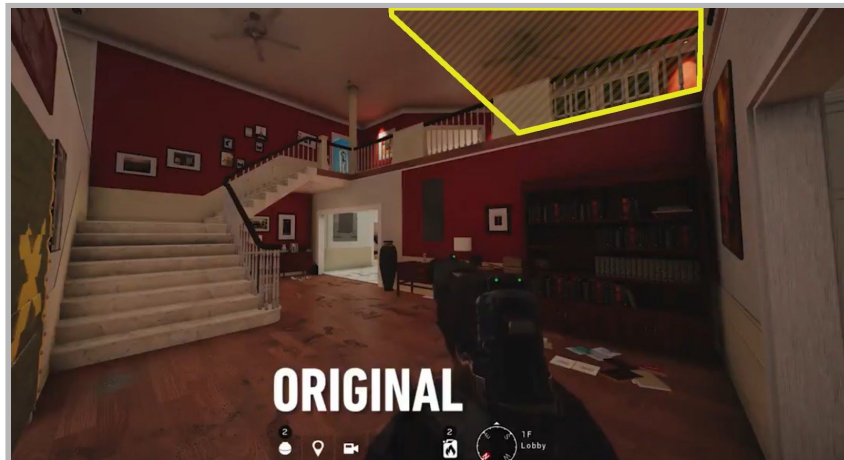




*Instead, it would be better to aim for a more simultaneous reveal. In the example above, instead of a forward facing tunnel or opposing ramp, it's a sideways hallway.*

### Corner angles & verticality

Particularly nasty angles is when they also include different elevations. One such angle was fixed by blocking it in *Rainbow Six: Siege* (see below). It combines horizontal and vertical advantages for the player on the elevated position. If it's high enough, it also includes the tendency players have of not looking up and them having to dedicate their sightline to look up and cover less horizontal space that's on the same floor level as them.



## ◆ Multiplayer map design ◆

### ❖ Overview ❖

While a lot of what we've talked about in this document still applies when creating multiplayer levels, some things either are more important in MP, is unique to MP or contradicts what you might want in a SP level.

We'll start by looking at some of that (including some good approaches to take) and get deeper into it as we go.

#### **Focal points**

They serve as navigational aid and landmarks for the area (meso) and map (macro). Additionally they act as gameplay and artistic focus for the area/map (i.e. "the lighthouse", "the crane", "the docking bay", etc.).



Most areas should have focal points, space for a decent number of players to fight in (4-8), a decent number of terrain variation/options and 2 (preferably 3) entrances/exits.

#### **Points of Interest**

Generally you want to have one or more areas that are more desirable than others in some way. This could be sniper perches, pickups, bunkers, etc.

In modes like Capture the Flag and Domination these points are automatically created. Since these areas are desirable and advantageous in some manner, you also need to balance them with some kind of weakness.

#### **Verticality**

Increases amount of choice and 'gameplay per square meter' a map has.

Usually elevation differences of 3m and 6m (up or down) helps make levels more interesting and fun, just keep sight lines and balance in mind.

## Cover

Players should generally not have too long unbroken line of sight, so large open spaces should be broken up by full cover. This allows players to advance without being vulnerable for long periods of time.

But if you want to promote risk/reward in an area (a pickup in the center of an area) then you can loosen up the cover a bit.

## Modes

Does your level need to support multiple modes and player counts?

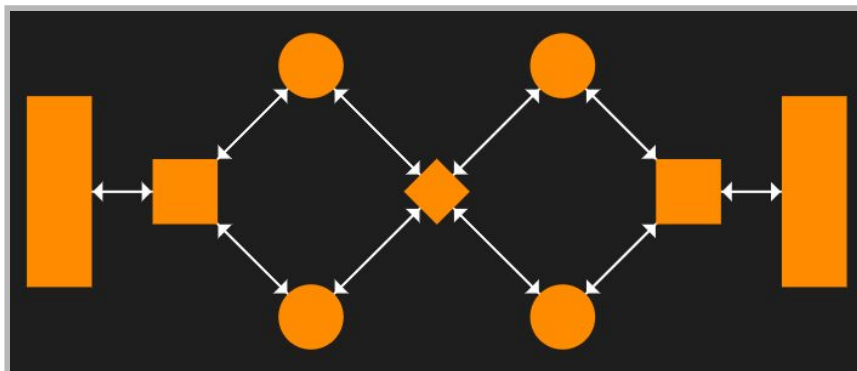
Can it be dynamically modified to allow for this? For example, a symmetrical Capture the Flag level would be relatively easy to reduce in size to create a tighter arena to fight in.

## Connectivity/Flow diagram

How should the different areas connect? It's generally a good idea to first start out with a simple flow diagram just to get a sense for how the map and areas will flow together.

As you're doing this it's a good idea to consider and plan out:

- Global flow (*pickup placement in for example arena shooters*)
- Local flow (*focal points, terrain advantage*)
- How to encourage players to make good decisions
- The entrances/exits (*2 minimum, preferably 3*)
- Good deathmatch maps need to connect to everywhere from everywhere.
- Good node/CTF (Capture the Flag) maps strategically block off some connectivity (to make pursuit and defence more possible). CTF requires that the flag carrier have places to run, but not hide.



## ✂ Readability ✂



One of the most important things you need to remember when it comes to multiplayer levels is that they're easily readable and aren't visually complex.

In multiplayer games where even a single second can matter, not easily being able to make out where other players are often leads to frustration and unease.

This includes nighttime levels, which might initially seem moody and pretty, but can easily cause serious readability issues and a strong sense of vulnerability.

This tends to result in nighttime levels become amongst the least popular levels in the games they're in (while the most popular maps are always well-lit).



*Dark night levels often result in poor visibility and lack of contrast.*



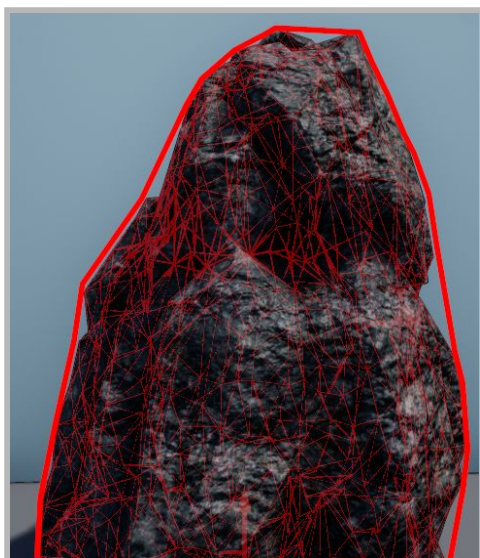
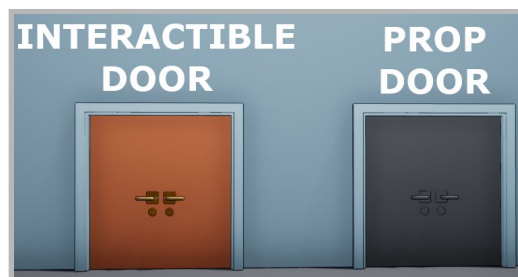


But that doesn't mean you can't have nighttime levels (or similar areas that have the potential to be dark or lacking in contrast). You just need to make sure they're well-lit and are easily readable.



"King's Row" from Overwatch takes place at night, but still offers good lighting and readability.

Also be clear with your visual communication. For example, if doors are interactable make sure the non-interactable doors are separated from the ones that can be used. Prop items should also preferably blend more into the environment than things the player can interact with.



While not directly tied to readability, make sure your *collisions* on objects that can block attacks (like bullets) is accurate.

If a player is behind one of these objects and shoots through parts that don't block their line of sight but their bullets end up hitting the 'thin air' right in front of them (i.e. the object) that can be frustrating. Especially if they end up dying because of it. This can also make certain pieces of cover more powerful than intended.

Even just small inaccuracies (like in the image on the left) can cause problems.

## ❖ Level structure ❖



### Keep layouts simple

Popular MP levels tend to be more simple in terms of layout. A complicated maze with tons of forks, hiding spaces, alternate routes, etc. makes it harder for people to remember and can stress them out more. Generally two or three main routes is enough.

While knowing the map well should be rewarded, it shouldn't be overly important to the point where it can greatly influence who wins and loses, since the players skill and not extensive map knowledge should be a stronger factor. Meaning tons of secret routes might not be ideal, or just secret areas/things in general that can offer a strong advantage. Even teleporters can be iffy sometimes since players would need to know where they go to offer up a fair fight. There are usually better solutions instead of using teleporters.

Basically *less is more, enough is enough*. Design what is necessary and don't go much further beyond that as a rule of thumb. This doesn't mean you should never make more complicated designs, but you will most likely need to spend more and more time on iteration and playtesting (potentially even bigger reworks, if you're not careful).

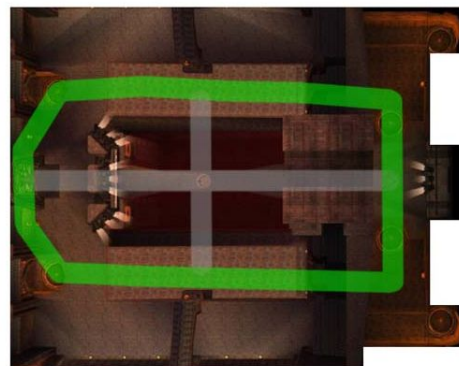
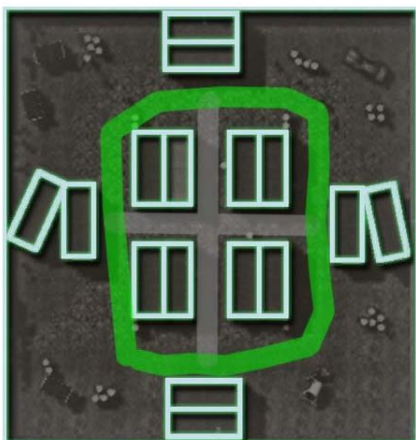
Next we'll look at some of the more popular (that players like to play) level structures.

### ⚡ The Circle ⚡

These types of levels tend to be circular in structure when it comes to the main path and are effective when you want to promote a lot of action and movement (deathmatch style maps).

Generally you also want to place pickups and the like along the circle to keep players moving.

Connecting the paths along the circle allows players to move between different parts of the circle.

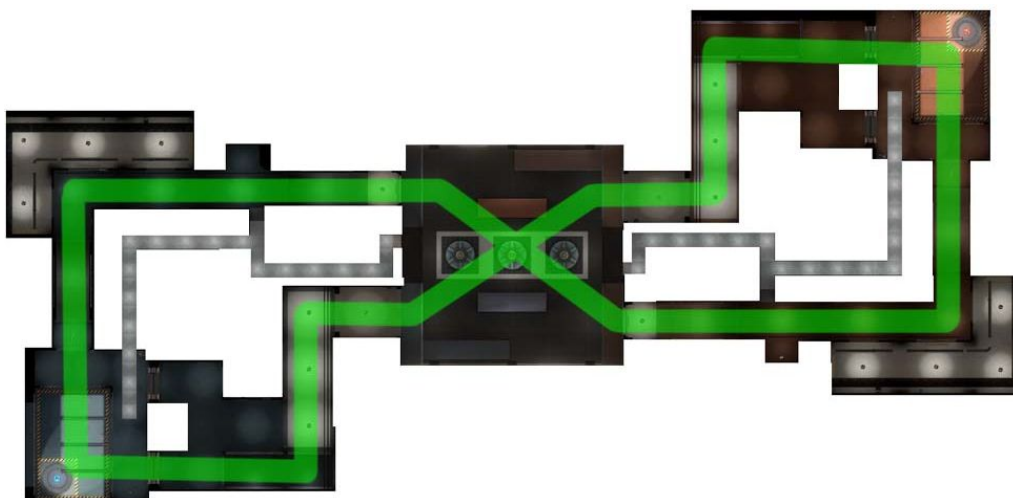




❖ Figure 8 ❖

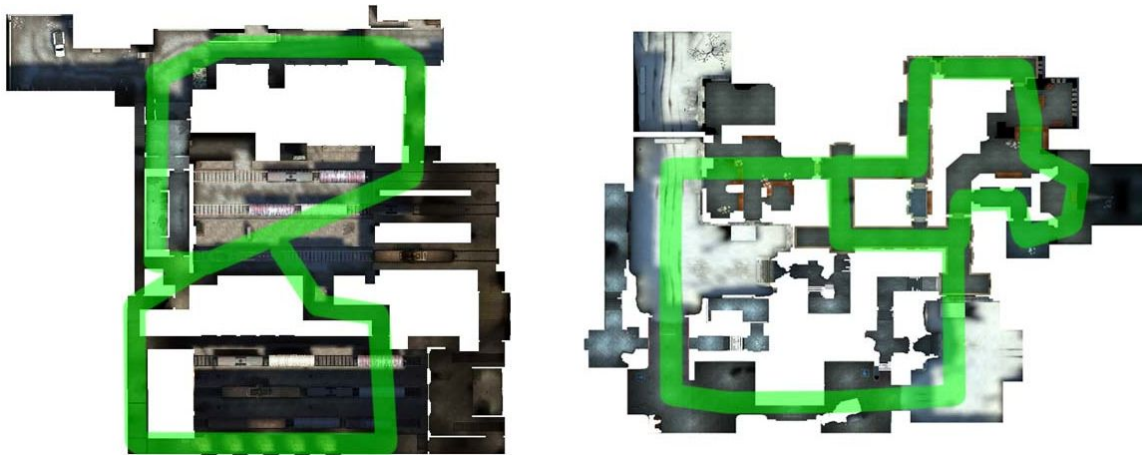
A variation on the Circle is Figure 8, which is when you basically connect two Circles with one another. This structure is good for more tactical team play where both teams meet in the middle of the map and have to coordinate more. So it's ideal for say "capture the flag" or "bomb planting" modes.

The "Turbine" level from *Team Fortress 2* is an example of this. A player wanting to steal the "flag" (in TF2 it's called *Intelligence*) will need to cross the intersection in the middle, which is where the enemy can more easily spot them. Meaning they need the support of their team to increase their chances of bringing the "flag" back to their base.

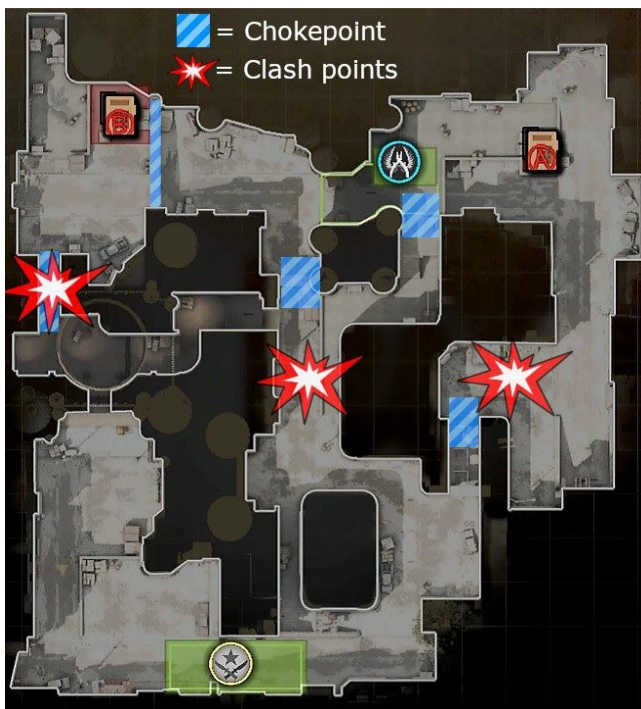




The "Train" and "Office" levels from *CS: Global Offensive* have both a more loose Figure 8 structure and more paths which makes map control harder, meaning greater teamwork is required.



### ❖ Clash points & Double Clash ❖



Clash (a.k.a. Meeting) points are where teams will meet if they both move from their own spawns at the same speed, meaning this is where combat is most likely to happen.

Choke points are narrower areas where teams have to focus their fighting and push through to the objective on the other side.

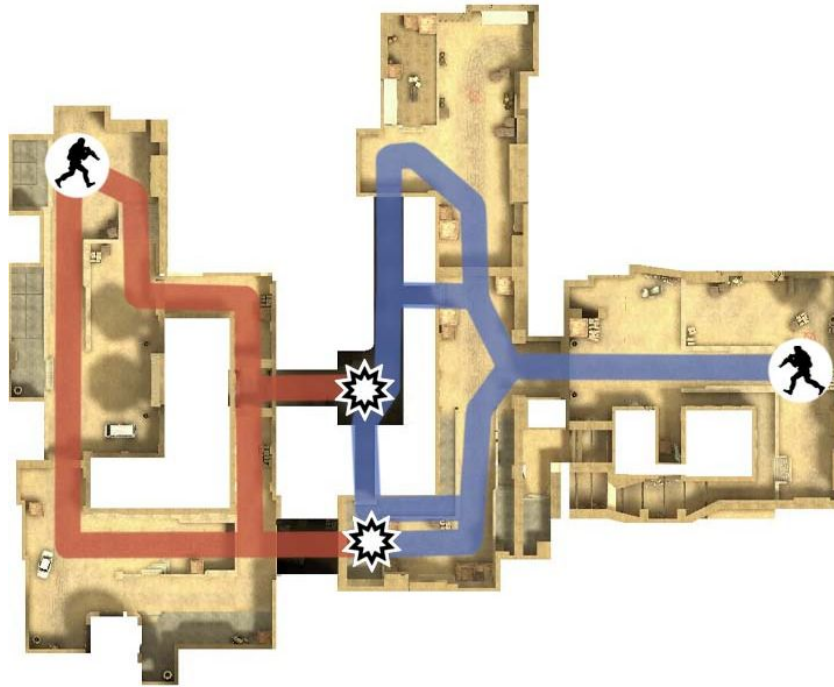
Generally you shouldn't have too many Clash points, while you can have a few more choke points since they're typically found behind Clash points.

### Double Clash

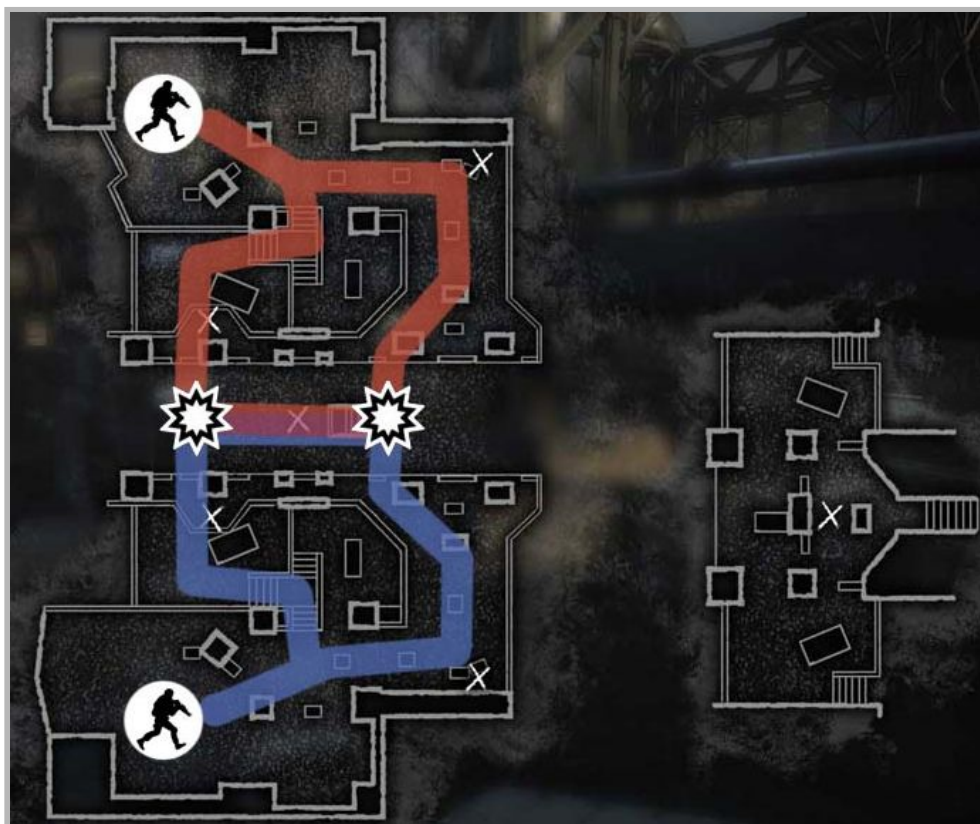
These are maps with 2 points where teams will Clash, usually with the overall shape of Figure 8. These types of levels are generally suited for round based modes or when players need to break through the opposing team.

The level "Dust" from *CS: Global Offensive* is one such map.





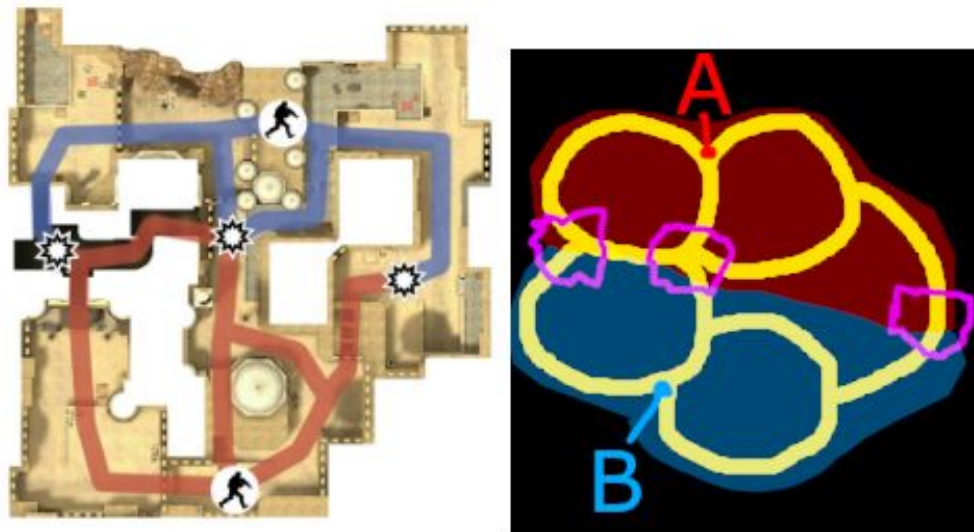
Another example is "Tyro Station" from *Gears of War*. Here teams will either meet in the tunnel below or at the station above. If a team only focuses on one that opens up for flank attacks or to avoid the enemy team entirely.



By adding another Circle to Double Clash you can create a quite versatile map that is suited for modes with a constant circulation of players, like breakthrough or domination style ones. Two examples of this are "Dust 2" (left) and "Aztec" (right) from CS: *Global Offensive*.



Do note however that you want to avoid creating 4 or more main Clash points between teams, as that can easily overcomplicate things and make it less interesting. Below ("Dust 2" from CS: GO) is effectively the limit you should stretch your design to.



*Dust 2 is basically two Figure 8 fused together, with a half-Circle connecting them at the end.*

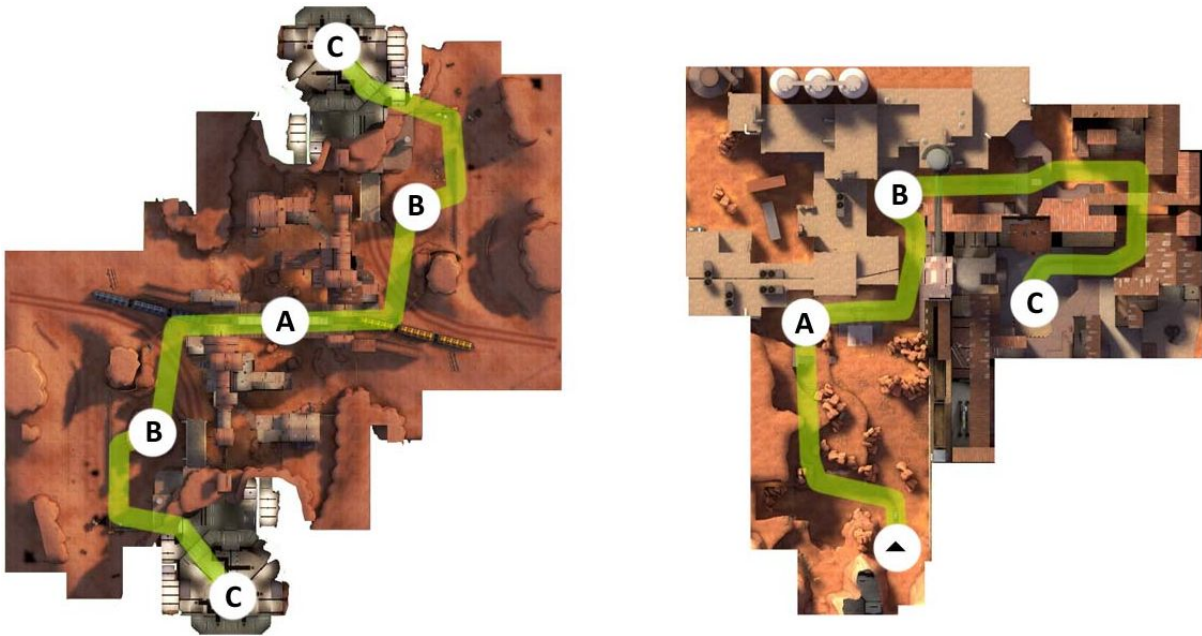


## ⚔ Tug of War ⚔

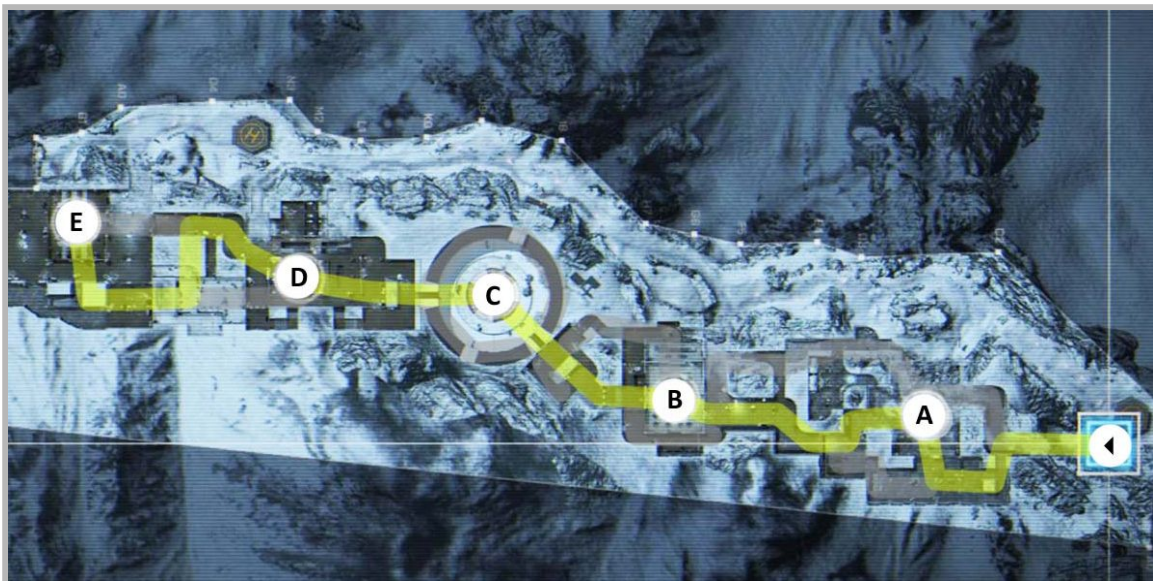
This is basically a serpentine shaped level that boils down to each team trying to push the opposing team back into their base. Each point is basically a small 'arena' players will fight in.

Control Point and Payload style maps from Team Fortress 2 are good examples of this.

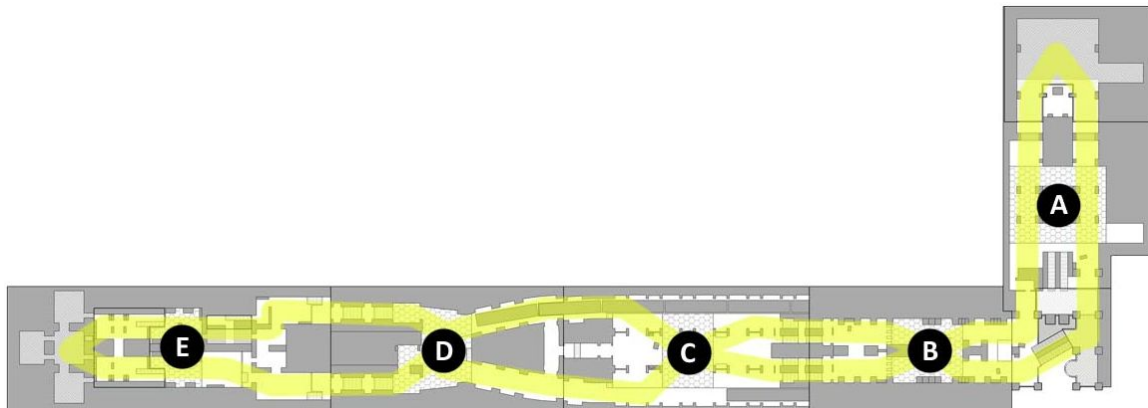
"Badlands" (left) is a capture point level while "Badwater Basin" (right) is a payload level.



Another example is from Battlefield 4.



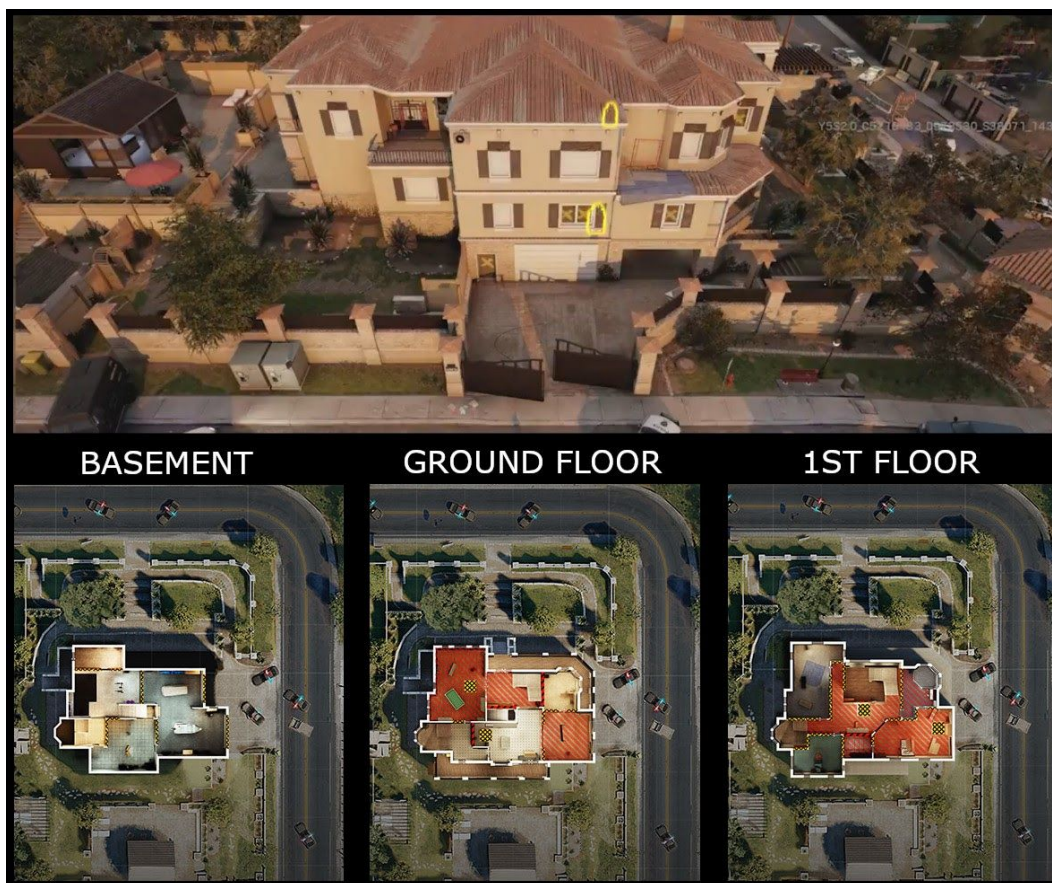
A variation of this can be seen in *Ghost Recon: Phantoms* where each point has two paths leading up to it, making teamwork more important. Both in terms of defence and to break through.



✂ Defence ✂

Making levels focused on defence where a team has some kind of unique advantage (powerful weapons, ammo, buttons to cause air strikes, fortifications, etc.) is another way to make interesting levels.

*Rainbow Six: Siege* is focused on this kind of level, where one team defends a building from the opposing team.





CS: *Global Offensive* also makes use of this for the hostage rescue levels, like "Militia" (see below, left) and "Assault" (see below, right).

There the terrorists have their base at the other end of the Figure 8 where they defend the hostages from the other team.

The reason why it doesn't just take place in this building is to allow the terrorists to intercept the other team on their way to the extraction point with the rescued hostages.



To make it even more interesting, consider spreading out advantageous positions inside the building that's being defended.

In "Checkout" from *Gears of War 3* the location is effectively a circle/ring to allow good circulation but also have several positions of stronger cover.



## ❖ Balance ❖

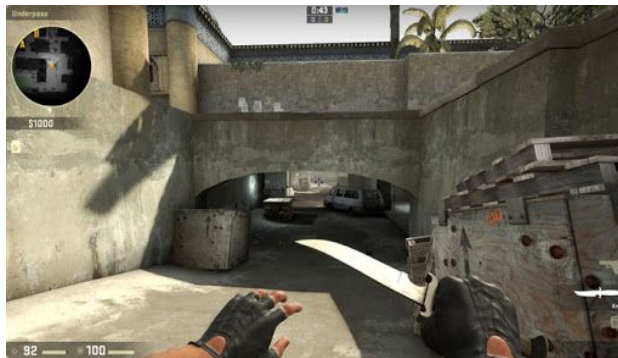
A lot goes into balancing a level. While I'll provide a grab bag of some good things to consider, the most effective way to balance a level is through constant iteration and testing.

### Chokepoints

They basically funnel players into a more narrow/small area where teams either try to defend or push through the area towards a goal. They also act as separators between different areas. These can be useful since they will 'force' players to fight together (instead of individual 1v1/1v2 skirmishes across the map).

This also means that they should be present before objectives.

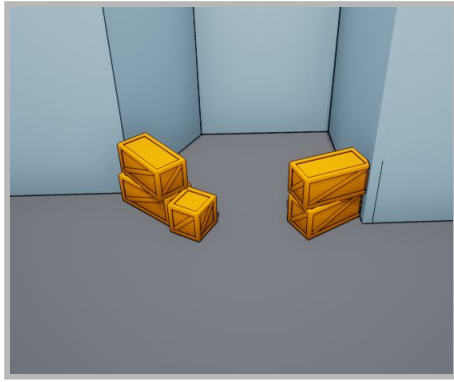
Keep in mind that it's usually a good idea to have several chokepoints leading up to an objective (not just a single tunnel as the *only* way to the objective, for example).



### Head peaks

Avoid head peaks (cover that blocks everything but a player's head, offering great cover and can make it harder for players to make out enemies. This is mainly relevant for first-person shooters.





### Strong cover

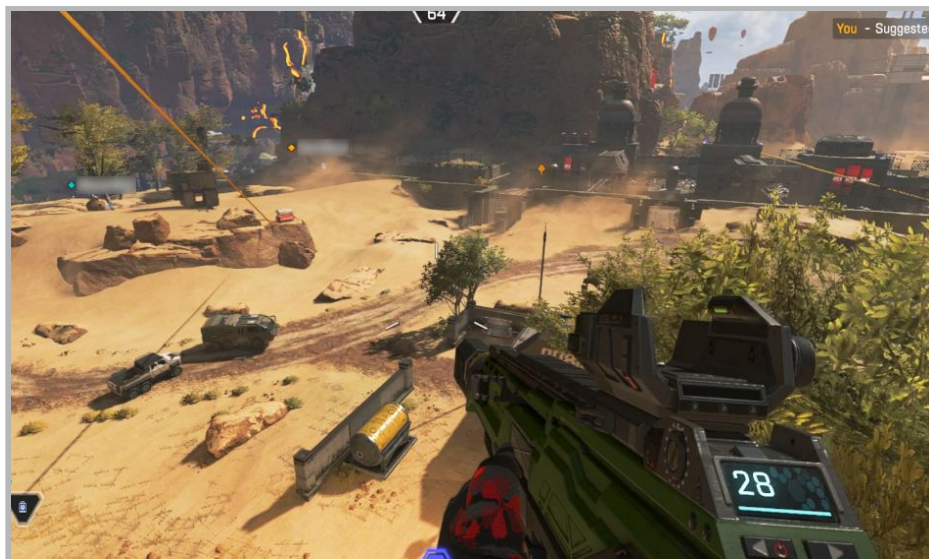
To balance out certain strong cover spots you can make them more enclosed so they're more susceptible to things like grenades and if they have to leave the cover they're very exposed (for instance, no nearby cover).

### Large open spaces

When it comes to large open spaces, things like movement is very attention grabbing. So a moving player would be instantly noticeable for a player hiding in a good sniper position overlooking a large open space.

To help balance this out, introducing random moving elements like dust, tumbleweed, etc. (so not all movement is player movement) will make players not as easily spotted when moving.

As a bonus, this also helps make your environment look less static.



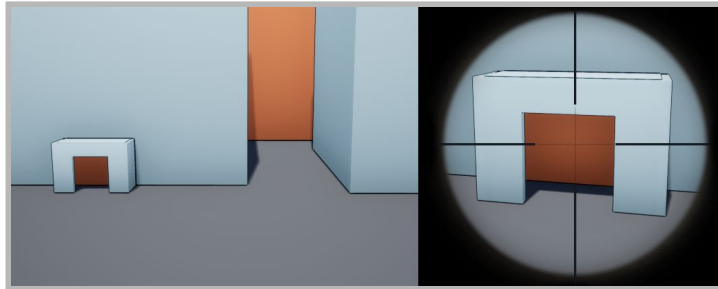
### Notes on elevation

While elevation is generally a good thing and makes levels more interesting, you have to be aware of how it influences how players can use sound, radar/minimap, etc. to detect enemies and how verticality can make this harder (since they might be able to identify direction, but not elevation).



## Sniping

Try and divide potential sniper targets (like hallways, doorways, etc) so that several spots aren't visible when scoped in with a sniper. This not only make it more interesting for the sniper, but it also forces them to not constantly be scoped in (or shift their attention frequently) and helps balance out the sniping spots.



## Supporting different playstyles

Ideally your levels should allow for different playstyles to become balanced.

If for example one section of the map only supports a sniper playstyle, snipers will dominate that portion of the map.

Refer to the example playstyles in the *Designing for combat* section and also consider what kind of playstyles your game allows, then try to enable that across your level.

While not perfect "2 Fort" from *Team Fortress 2* allows for multiple playstyles.

Snipers can duke it out on the balconies, the Scout can double jump and jump across the roof on the bridge, you can move through the bridge with the cover of the roof or you can move through the underground tunnel beneath the bridge.



### Risk & reward

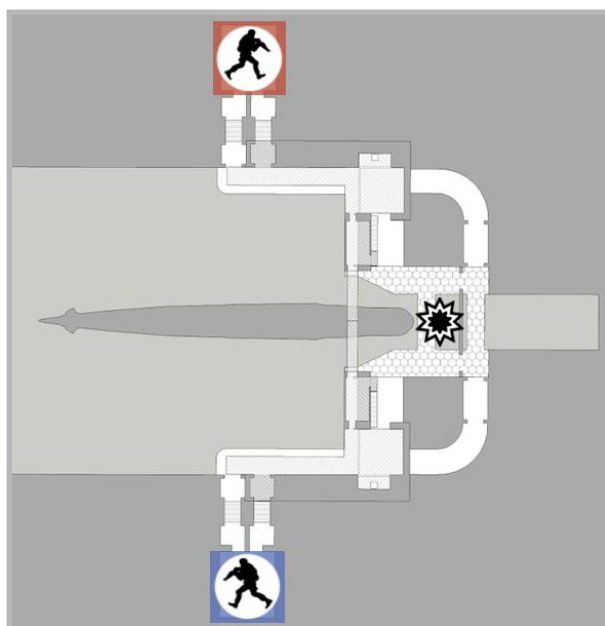
Risk & reward in level design can be present in the routes as well. So the risky part could be the walkway you move across to the 'reward' area could be more exposed, you create a lot of noise or it could be that you take damage in some way (falling down to the 'reward').



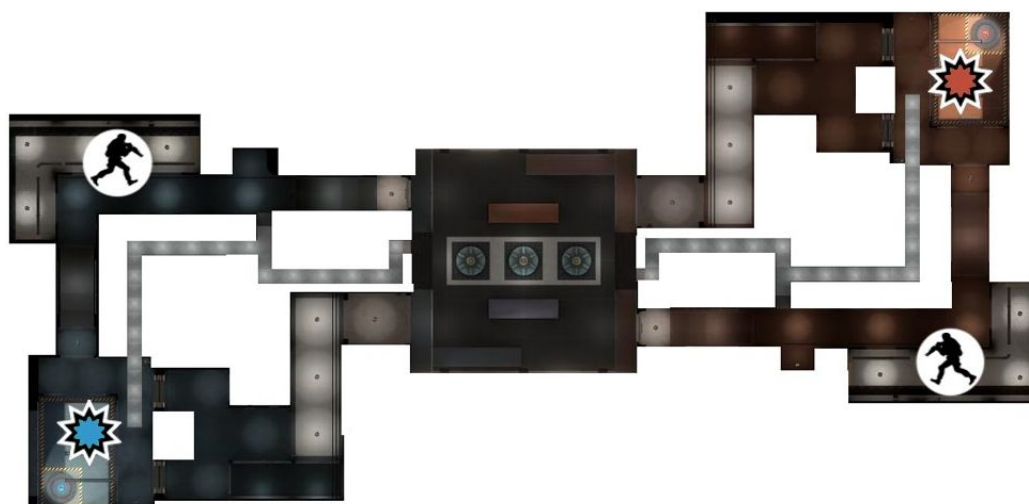
## ❖ Symmetry & Asymmetry ❖

The easiest way to balance a map is to mirror both team's sides, creating symmetry. This creates equal starting conditions, makes it more predictable (easier to learn, since both sides play the same) and it's easier to predict where the Clash point for both teams are (since they will run an equal distance).

An example of this is the "Sub-Pen" level from *Ghost Recon Phantoms*, which has 3 different paths in the center, with the control point in the middle of the map (by the Clash).



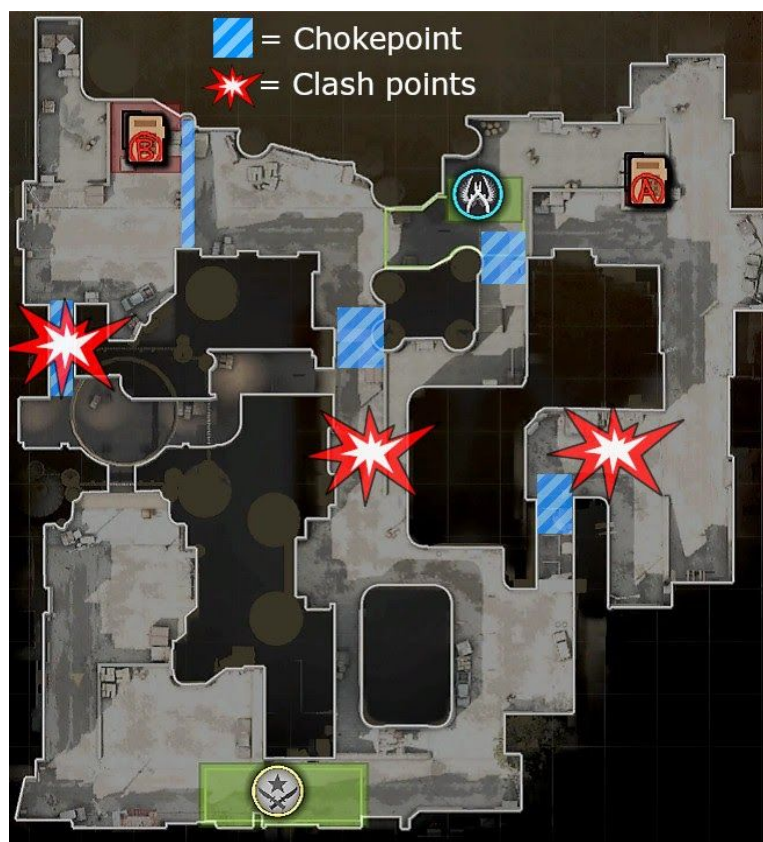
Inverted symmetry also works (effectively rotating one part of the level 180 degrees). "Turbine" from *Team Fortress 2* is an example of this, where they place the spawn and capture points a bit away from each other as well.



A way to make it feel slightly less artificial is to tilt and shift the symmetry a bit. For instance "Nuke Town" from *CoD: Black Ops*. Here they tilted the level's symmetry as well as rotating and moving the building on each side to make it balanced.



A well-rounded asymmetrical level is Dust2 from *CS:GO*. The Clash points and Chokepoints are balanced so they don't give one team an upper hand over the other. Player's meet each other at balanced spots at the Clash points and the Choke points don't offer superior cover or sight lines to one team over the other.



## ✂ Time ✂

A way to help balance asymmetrical MP levels is by timing how long it takes to reach certain points, making sure they align for each side.

If not properly balanced this can mean one team can reach objectives or Clash points faster than the other team. This can for example lead to one team being able to capture the point faster, secure the Clash area (setting up in cover), setting up ambushes, etc.

Some suggestions on how to control this:

- Moving the spawn points forwards/backwards.
- Alter the routes so one is shorter or longer.
- Slow down/speed up one team in some way. Ladders, water, closed doors, etc.

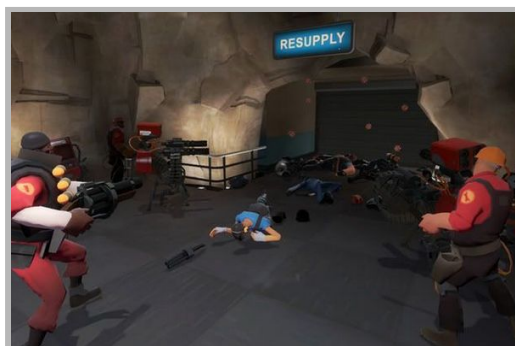
A generally good sweet spot for Clash points is roughly 5-12 seconds. If it takes too long to get into the fight it can create dead-time.

## ✂ Spawn points ✂

### Spawn Camping

Preventing teams from being able to spawn camp the other team is important.

While it might be fun for the ones spawn camping, the team that is stuck in their spawn room runs a very high risk of becoming extremely frustrated.



To prevent this, you should generally stick to two rules of thumb:

- Always have at least 3 exits from a spawn room at different locations and/or elevations. The doors should prevent enemies from shooting through the doors when opened.
- Enemies should not be able to enter the opposing team's spawn room. To visually represent this you can make use of magic or energy to prevent access/damage. But if your game strives for realism then making use of elevation can help to have players drop down over a ledge, etc.

## Changing Spawns

Swapping or changing spawn points is another way you can balance a map, often used in Control Point and Payload modes.

In for example "King's Row" from *Overwatch* as the attacker pushes deeper into the level and takes over control points they receive new spawn room further ahead. Meanwhile the defender's spawn room is pushed back.

This helps keep the pace of the match going by not forcing players to run long stretches after each death while keeping the action focused around the objective.



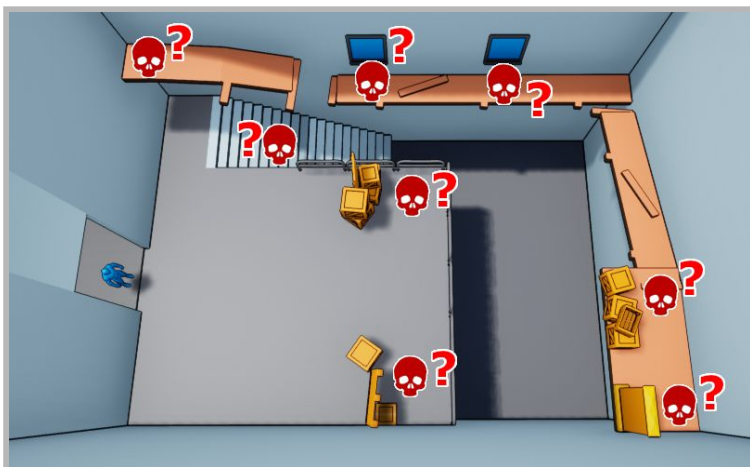
The devs gave the defenders a slight edge to allow them to set up before the attackers can attack. But on the other hand, the attackers have several routes to attack from and cover to hide behind, which makes it harder for the defenders.

To make the final push more intense, the defenders spawn room is often placed very close by to the final objective.

### ✂ Potential Player Positions (PPP) ✂

If spaces have too many PPP it can easily become overwhelming.

This happens when the player becomes vulnerable from too many directions and have to scan for players in too many locations when entering a new space (usually more open, be it indoor or outdoors). This can easily lead to a strong defender/camper advantage and a sense of unfairness for the overwhelmed player.

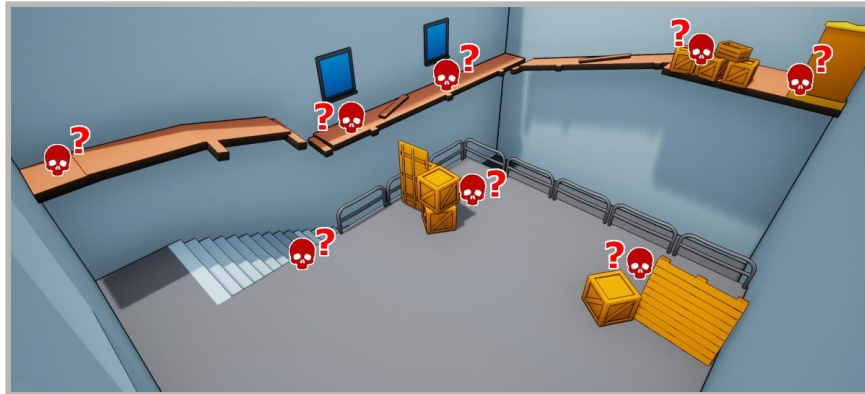


PPP are basically points where the defending/camping players would have an advantage over a player entering the space and would thus most likely be in. This is usually behind cover, around corners, elevated positions, etc. In other words, they're basically attentional goals.

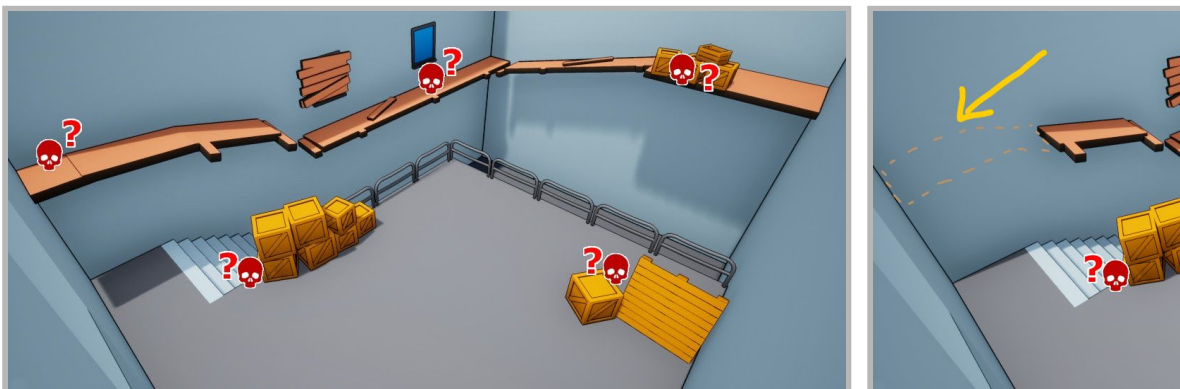


The way you fix this is by reducing the number of PPP and/or gathering them up (so the player can see more of them on the same screen, without having to turn around to look elsewhere).

You can also divide up the space physically (like with a tall wall or barrier) or with concealment (something players can shoot through, but can't see through or hide inside).



*A bit too many PPP.*



*A more manageable amount of PPP (see left). But depending on your intent you might also want to remove the elevated one in the corner (see right). This is because they tend to be pretty strong and would be the only non forward facing (from the entrance) PPP.*

But be mindful of visual complexity (i.e. visual noise) and other things that makes it harder to spot players (hiding in shadows, camouflage amongst objects of similar texture/color to the player, etc).

This is because it takes longer to spot (or basically 'check') each PPP and can even result in the player not seeing the enemy.

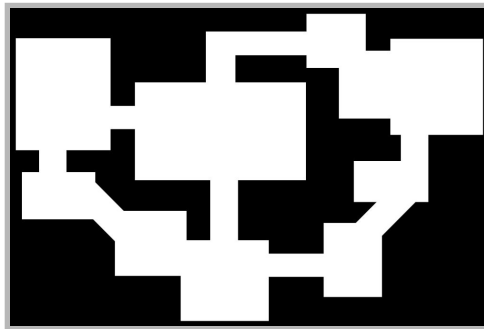
## ❖ Flow ❖



### Level flow planning

When planning out the Flow of the level, think about questions like:  
What's the ideal amount of players?  
How big is the map?  
What playstyles should be encouraged?  
What weapons do you want players to use & how often? (weapon balance)

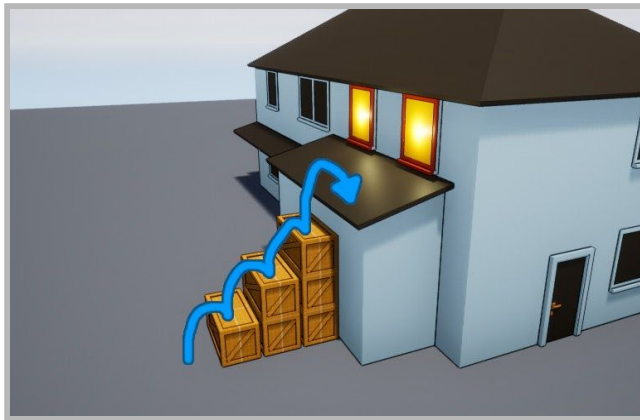
It's common for junior designers to make more blocky levels. Not only does this end up looking artificial, this can result in players getting stuck on parts of their level. Sharp 90 degree corners tend to force players to slow down to peek around corners as well, which might not be desirable.



But it's actually not *that* difficult to get away from more blocky designs. As we mentioned before, design your level with *forward momentum* in mind (i.e. as if the player moved like a car).

An example of this is from *CoD: Black Ops 2* where the levels have more smooth corners and turns.



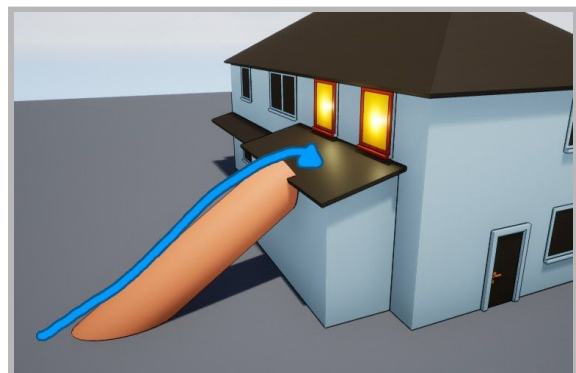


This of course also means that you shouldn't clutter the path in front of the player too much or create scenarios that bogs the player down.

For example, if the player needs to press the jump button (or climb) 3-4 times in a row to cross an obstacle that slows them down immensely and makes them very vulnerable, partially because they're so focused on the act of climbing/jumping.

Instead, try taking the end goal (reaching the elevated position in this case) and smoothen it out so players don't lose their momentum.

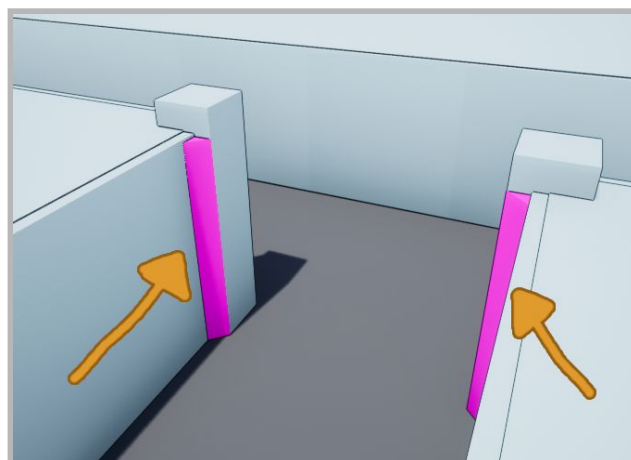
For instance, use something in the form of a ramp like a fallen tree, a boulder, planks, etc.



Another thing you need to be wary of is that players dislike getting stuck on geometry or slowed down. Imagine if you try to escape a room that another player threw a grenade into and you got stuck on a chair or the side of a door and died. This would most likely annoy you.

To fix this you can remove (or move) clutter that the player could get stuck on and also use collision helpers (see image below).

Basically you would determine the players intent and movement flow and add invisible collision that would 'nudge' the player in the direction they would want to go.

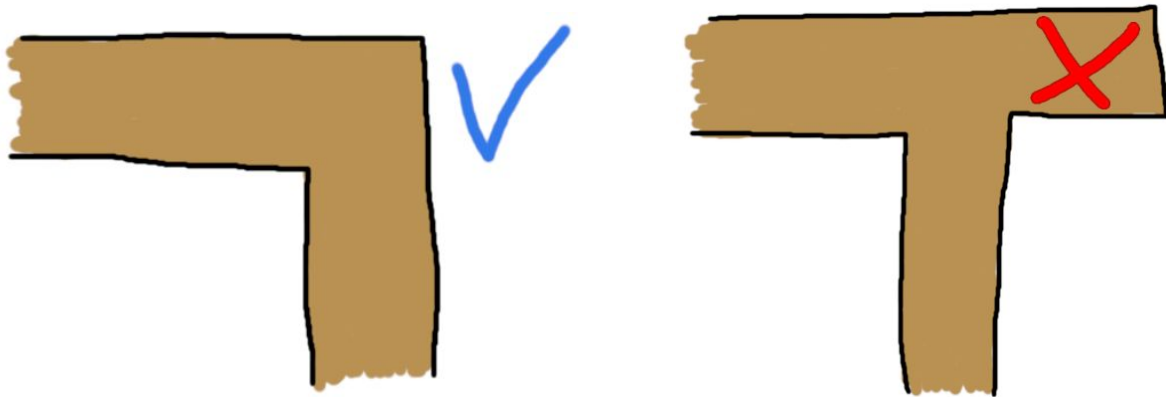


Avoiding 'useless' space is also a good idea. These are spaces the player doesn't have any real reason to go to. This includes dead-ends.

This is because they don't offer anything interesting to the player, no tactical choice.

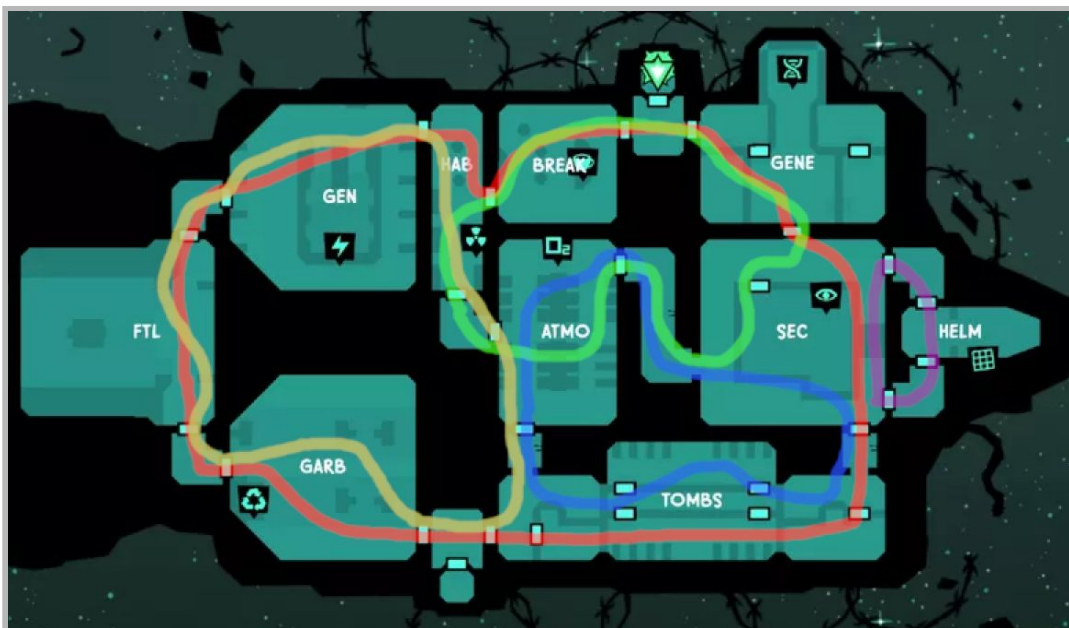
If you have this kind of 'useless' space, then consider removing it or incentivise players to go there with for example a risk-reward scenario.

Doing so will make your map more intuitive and flow better.



Loops are also highly useful in MP level design (and combat spaces in general).

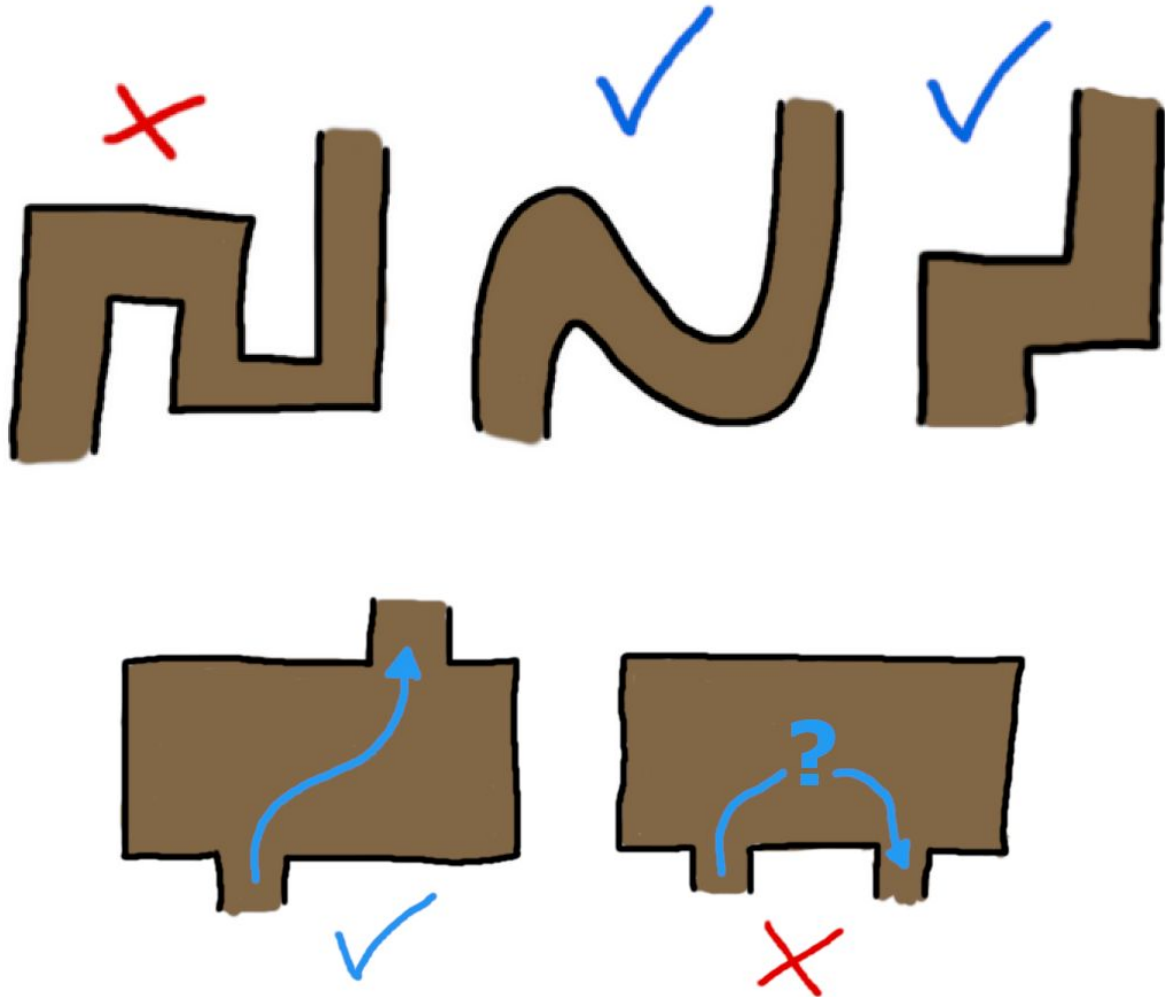
They help provide players with options, help remove dead-ends and keeps players flow going.





Flow in general is about keeping players forward momentum going.

So just playtest your level and check for when you or other players feel like they have to step on the 'car brake' or slow down. When entering a new space a path forward should generally be somewhere in front of the player.



### Space to move

The playspace should have enough room for players to move both solo and in groups, especially in areas where combat and a lot of player movement is supposed to take place.

Consider the space required to allow for the kind of gameplay you want when making enclosed spaces (like hallways and rooms).

Then if needed, consider making rooms larger, doors broader and stairs wider, etc.

## ◆ Briefly on accessibility ◆

While it's beyond the scope of this write-up (I might do a separate one on accessibility in the future), it's good to read up a bit on accessibility so you don't needlessly and unintentionally make your game/levels harder to play for your potential players.

A few places you can start at is:

Douglas Pennant's GDC talk on color blindness and how to design to support it:

<https://www.youtube.com/watch?v=KbFs9ghIIEI>

Color blindness simulation tool:

<https://www.color-blindness.com/coblis-color-blindness-simulator/>

Game Maker's Toolkit video on designing for accessibility:

<https://www.youtube.com/watch?v=xrqdU4cZaLw>

UX designer and Accessibility Specialist Iam Hamilton gives a talk on accessibility:

<https://www.youtube.com/watch?v=umU9xavDJqQ>

A Game UX summit talk on the importance of good UX design for mobility accessibility:

<https://www.youtube.com/watch?v=006jfzCrMtQ>

Brian Allgeier from Insomniac Games talks about how they embraced accessibility on Spider-Man:

[https://www.youtube.com/watch?v=gJ\\_RdQ2BSx0](https://www.youtube.com/watch?v=gJ_RdQ2BSx0)

Great site that breaks down how to design for various levels of accessibility:

<http://gameaccessibilityguidelines.com/>

Helps you understand what accessibility issues various people can experience in games (Scroll down a bit on the site):

<https://accessible.games/accessible-player-experiences/>

A collection of articles, resources, guidelines, etc. on accessibility:

<https://igda-gasig.org/how/for-developers-researchers/>

---

*I hope the contents of this document was useful to you.*

*Thank you for reading!*