www.matthewball.vc /all/cloudmiles

Cloud Gaming: Why It Matters And The Games It Will Create — MatthewBall.vc

May 5 Written By Matthew Ball & Jacob Navok

61-78 minutes

There is a tremendous amount of confusion around cloud gaming: what it is, why it matters, what it will allow, who can deliver it, how much it will grow the market, etc. This is further complicated by misunderstandings around how online multiplayer works in the first place and what will and won't change when a "console moves to the cloud." This leads to bad headlines and excessive hype.

The essay below is focused on explaining the overall future of cloud-based gaming, the experiences it will enable, and the first new "genre" of game it's likely to create. To do so, it's helpful to run through a few other ideas: **(1)** How online multiplayer actually works; **(2)** Why it works this way; **(3)** How cloud game streaming works; **(4)** What cloud game streaming is often confused with; **(5)** Why cloud game streaming is still so hard, costly, and imperfect - and when that might change; **(6)** Who will want cloud game streaming and what sorts of experiences will be needed in order to drive mass adoption; **(7)** What a cloud game is and allows. The **eighth** chapter then explains the likely first cloud-based genre, "MILEs," which has the potential to engage millions of new "gamers" worldwide and in the same live game.

Depending on your familiarity with the above topics, it might make sense to skip certain early chapters. However, the throughlines are nevertheless important.

#1: HOW ONLINE MULTIPLAYER GAMING WORKS TODAY

For the average person, the most complex computational experience they will participate in or see in a given week is an online multiplayer video game. What makes these experiences so remarkable - and possible - is the fact that online multiplayer video games actually involve very little "online" work.

When a player participates in an online video game, their local (i.e. nearby and touchable) machine (e.g. iPad, PC, Xbox, Switch) is actually running everything we consider to be a "game." This includes everything - the game logic, the AI, the physics, the image rendering (i.e. visuals), the audio, and so forth - just like an offline game. This is why players must install 40GB game files to play online-only titles like *Fortnite* and then keep updating them with many more gigabytes of locally-installed data as the online-only experience changes.

Ultimately, the role of remote processing - the online multiplayer server - is to relay each player's positional data and inputs to each of their individual machines and then essentially referee process conflicts when they occur.

An example helps put this in context. When a player shoots a rocket launcher at a tree in *Fortnite*, this information (e.g. the item used, its attributes, and the trajectory of the projectile) is shared with all other players in the game. Each of their local machines then processes and acts on that information: they show the explosion, determine whether their player is harmed, remove the tree from the map, and allow the player to move through where it once was, etc.

In practice, players might not even see the same visual explosion, even though the "same" explosive hit the exact "same" tree at the exact "same" angle at the exact "same" time, and the exact same logic was applied to process the cause and effect. This variation doesn't matter, and it's odd to consider how a shared experience can vary with such trivial items. However, this inconsistency exists precisely because most information in an online multiplayer game (or shared simulation, if you will) is trivial. What matters are consistency, fairness, and technical resiliency (i.e. playability). Anything that isn't core to that experience is deprioritized in order to preserve processing power for what does matter.

This same approach also means that different machines can disagree about consequential events too. To continue with the example above, it's possible that Player 1's console might determine that Player 2 was killed by the explosion that destroyed the tree, while Player 2's console would say Player 2 took significant, but not fatal damage. Neither console is "wrong," but the game obviously can't proceed with both versions of the "truth."

In most cases, this happens because it takes time for an input to travel from a controller to its console and from one console to another console (these gaps are called "latency"). As a result of these delays, Player 2's console processes Player 2's evasive maneuver before this information travels through the web to Player 1's console, where that decision to move is then re-processed for Player 1. Although the time it takes for this information to travel, be processed, and visualized happens so quickly that a human wouldn't be able to see the difference between Player 1's screen and Player 2's in real-time, it can nevertheless mean Player 2's character is in a fatally different position. The difference between a bullet hitting and missing a target, after all, is only a few pixels.

In truth, all players are looking at slightly out-of-date positions of one another - which means their local machines are processing the cause-and-effects of these out-of-date positions. Most of the time this doesn't matter, but it often does. In this case, the online server determines the "truth" and sends out information to re-align all players. Most of these changes happen so fast that players will never even know. Indeed, there are many neat tricks to cover this up (e.g. consoles will often predict what is likely to happen and visualize it before it occurs, then skip frames or even "rollback" an in-process visual to "correct" the simulation).

Notably, this process is unique to online gaming. When four players play *Mario Kart* together on the same Super Nintendo with a split screen TV, physics are always correct and consistent because the same console and same cartridge process all decisions. No synchronization or distinct processes occur. There is only one source of information and thus no opportunity for disagreement.

#2: WHY ONLINE MULTIPLAYER GAMING WAS DESIGNED THIS WAY

Today's approach to online multiplayer gaming can seem wasteful. Why should 100 consumer-grade machines (again, to continue with *Fortnite*) all be performing the same "work"? Especially given that conflicts arise when re-re-re-re-reduplicating this effort? Why doesn't the tech-media company responsible for this experience instead use their own billion-dollar, industrial equipment instead of pushing it to small consumer hardware wrapped in dyed plastic covers? Or at least offload the processing to Amazon's infinite server stacks?

Consider the huge limitations that come from relying on the end-user's local machine beyond just reconciliation issues. When processing occurs on a consumer-grade device, the richness of the experience is tied to it too. A PS4 (released in 2013) can play *Fortnite* with a player using a sixth generation iPad (2018), but the iPad user will have 1/3rd of the frame rate and see 1990s-era graphics. Further still, the iPad will choose not to render much of the experience (e.g. another player's skin) as it needs to prioritize processing power for core gameplay. And because consumer hardware has consumer-grade processing power, online multiplayer worlds have to be severely constrained in their scale and complexity. A 2020 iPad Pro or even PS4 Pro can only track and manage so many details all changing in real-time as part of a hard-to-predict simulation. And the complexity of these games ends up at least partly limited by the lowest supported devices.

There are numerous reasons for this ostensibly byzantine approach. However, most stem from two points.

First, the Internet is not reliable. Your connection is not reliable. Packet transfers (how the internet sends data) are not reliable. Two players (Player Y and Player Z) could be in the same *Fortnite* match and both located in the same Manhattan building, but their packets might be routed through Chicago and NYC respectively. As a result, Player Y's data will take longer to be transmitted to the Epic Games server (and back) than Player Z's. In addition, different delivery routes mean that if a network hiccup occurs on a cable or wire center in Chicago, only Player Y will be affected. Or perhaps Player Z's son starts a 4K stream of "Frozen II," leading their home router to prioritize the son's packets and deprioritize Player Z's. Or perhaps Player Y's neighbor begins downloading an 80 GB game off Steam and takes up all of the bandwidth in the copper wire that supplies both of their Internet.

Second, the Internet itself was not designed for continuous, persistent connectivity, especially across multiple participants where milliseconds matter. Instead, it was built to share static, non-live files from one computer to another.

Because of this, multiplayer was designed in the early 1990s with the packets as small as possible and as much information as possible redundant. Multiple "truths," in other words, are a feature, not a bug. For example, in a racing game, Player 1's console will assume Player 2's car will keep moving forward at a predictable speed. If Player 1's console then learns Player 2 actually pressed the brakes suddenly, it will stop rendering the car as originally predicted, and suddenly Player 2's car will appear in its "correct" position, with no interim video frames. The benefit of this is that if any player's data/internet is interrupted, the game is able to proceed and no cars suddenly stop or disappear. If such tricks didn't occur, real-time games (versus, say, card games) would be unplayable.

#3: WHAT IS "CLOUD GAME STREAMING"?

Cloud game streaming brings the majority of the gameplay processing to a data center outside of a player's home. This data center could be little more than racks and racks and racks of consumer-grade consoles stripped of their enclosures or built around a fundamentally different (and better optimized) set of servers and related infrastructure. But what matters is that it's in this data center that game logic, the AI, the physics, the image rendering (i.e. visuals), and the audio are all processed for the end-player.

Remote data center processing doesn't mean that packets are no longer routed through different hubs, or that an aged iPhone is suddenly as powerful as a brand new PC. But through cloud game streaming,

users will be able to play a high-powered AAA game that typically requires a two-square foot console or high-end PC on anything from an old Android to an underpowered Chromebook. It also means players no longer need to worry about whether their specific device is compatible with a specific game, whether they'll play a "good" or only competitively-tolerable version of the game, or whether they'll need to upgrade or replace their device to play a sequel. Games - all games - will just *work*.

This is possible because what a player sees while cloud game streaming is essentially just streaming video. In fact, this video is not altogether unlike watching live, digitally delivered sports on ESPN+. But in this case, the viewer is also moving an analog stick or pressing a button based on what they see in this video, with these inputs then sent to a remote computer that interprets these inputs, renders new visuals as a result, encodes these renders to video, then streams it back live to the player's screen. This typically happens so seamlessly that it looks like the game is being processed right in front of the player rather than being interpreted, rendered, and sent back to the player from miles away.

Furthermore, the shift to server-side processing hardware means more than just the ability for an iPad to access an experience equivalent to that of a high-end consumer console. Instead, both devices can access even greater and more efficient industrial-grade processing power. "Remote consoles" *could* always be updating, upgrading, and patching, too - and without the user needing to do a thing.

However, cloud-based streaming does not mean there is now "one version" of a game being processed. Cloud gaming is often thought of as swapping a household power generator for grid access to a remote nuclear power plant. In truth, it's more like accessing a remote building full of individual high-quality generators when needed. Today, two online *Call of Duty* players are both processing their own simulations via in-home hardware. If they shift to cloud game streaming, there are still two versions being processed, only now it's happening outside their home. As a result, processing and other latency-related conflicts can still occur (though they are fewer and less significant as data centers have better equipment and shorter distances between one another than consumer homes do).

#4: WHAT IS "CLOUD GAME STREAMING CONFUSED WITH?

Due to the popularity of cloud content services like Spotify and Netflix, many believe that cloud game streaming is an all-you-can eat subscription bundle of their favorite games and countless others. However, Spotify and Netflix are so compelling because they package together two different things, content *and* streaming content delivery, and then offer access via low cost, AYCE monthly prices. These three elements are separate.

To use an example, downloading a video from Amazon Video for offline playback is digital delivery. Playing this video on-demand without downloading it in advance is cloud streaming. Accessing this video for free as a Prime subscriber is a cloud video streaming subscription.

To this end, we see a mix of so-called "cloud gaming" services and models in-market today, only some of which are actually cloud game streaming.

1. Google Stadia Pro ($10/month) or GeForce Now ($5/month), for example, are cloud game streaming services that still require users to buy individual games (though some are free). In other words, they change how gamers access gaming hardware, but not how they access (i.e. buy) gaming content. The lack of the latter is widely seen as a key barrier to adoption, especially given the services lack compelling exclusive content too. Microsoft's xCloud is likely to resemble this model (though per later in this section, bundles are likely too)

2. Xbox Game Pass ($10/month) and Apple Arcade ($5/month) are All-You-Can-Eat digital game subscription services. These services are often called "Netflix for Games", but subscribers still need to download and install these games locally. Accordingly, they're a version of Netflix where the only playback option was via "download to watch offline."

3. Sony's PS Now ($13/month), meanwhile, is a cloud game streaming service that also includes an AYCE gaming catalogue. Even still, subscribers don't need to play many of these titles via cloud delivery - they can also be downloaded for local processing. In fact, the vast majority of PS Now playtime is believed to be via local installs.

4. Sony and Microsoft have also suggested that physical console owners who buy a la carte games will be able to stream those games to other devices at no additional fee.

This construct might seem weird. For the most part, consumers aren't familiar with separate bills for distribution/delivery and content. Pay-TV is a good example. Households don't "buy" access to Comcast's coaxial cables ("distribution/delivery") for $20 and then add various television networks ("content") atop it. Instead, Comcast sells a package of both and then relays part of the proceeds to the

TV network. Even when premium networks like HBO, Showtime, and Starz are added a la carte to a TV package, the distributor is the one who sells it, and they retain a third of the fee.

This model extends into “direct-to-consumer” video services too. When a customer buys HBO Now from Amazon Channels for $15, Amazon keeps ~$5 to manage delivery. When a customer buys HBO Now direct from HBO, the company charges $15 and spends several dollars on its own delivery costs. It is, however, technically possible to unbundle these products. Imagine, for example, sending Amazon $15 per month for the *ability* to watch content through Amazon Video, and then adding the likes of HBO for only $10. Stranger still, you might go to HBONow.com, buy HBO for $10, then need to connect it to a paid technology provider (e.g. MLBAM or Apple TV+) in order to watch it.

The unbundling of content, access, and services is common in gaming due to the diversity of parties and technologies needed to enable online multiplayer experiences. Accordingly, consumers are not just used to paying multiple parties, they’ve even come to accept paying the same party multiple times. You have to buy a PlayStation console from Sony and then a PlayStation game from Activision, Sony, or another party to have something to do with your console. And if you want to play multiplayer, you then need to buy the monthly PlayStation Plus subscription from Sony. In some cases, you then need to pay the publisher a monthly content subscription for their game too.

Given this complexity, everyone agrees that simplified packages are inevitable over time. Microsoft already offers bundles of Xbox Live (its multiplayer service) with Xbox Game Pass and even a monthly rental for its Xbox One console. Bundling options with xCloud, Microsoft’s forthcoming cloud game streaming service, seems likely. To return to the Amazon Video example above, it’s clear most consumers prefer streaming all-you-can-eat content bundles.

At the same time, we shouldn’t overestimate this model’s fit to gaming. Many of the biggest games today are free-to-play. As a result, moving them to an all-you-can-eat bundle isn’t transformative. And even when a game isn’t F2P, an enormous share of total revenue can/does come from micro-transactions. This monetization, let alone game design, doesn’t port well to AYCE.

In addition, gaming playtime is enormously concentrated among the leading titles. As much as audiences rewatch *The Office* or replay Taylor Swift’s greatest hits, it’s unlikely that it would be as much as 60% of a user’s playtime. Video and music consumption is heavily diversified and usually unplanned. Gaming is the reverse - most users just play *Fortnite* (or their favorite) every day. As a result, the benefits of avoiding a local install or upfront price are less compelling, as are, in theory, recommendations.

The biggest challenge, which is specific to cloud delivery, is the cost. It doesn’t cost Netflix or Spotify much more if a user watches or listens much more, so the AYCE model works. Cloud game streaming, however, is incredibly costly on a usage basis. This makes flat, AYCE pricing quite challenging. More on this in the next section.

#5: WHY IS "CLOUD GAME STREAMING" SO HARD, COSTLY, AND IMPERFECT? AND WILL IT GET BETTER?

For all the apparent absurdities of locally/redundantly processing online multiplayer games, it’s remarkably hard to ensure a comparable quality of experience (e.g. latency, reliability, fidelity, etc.) via cloud processing.

However, shifting from the synchronization of small data-packets of positional data to inputs of enormous video files means exponentially greater sensitivity to internet slowdowns, and in turn, more hiccups. The impact of these hiccups, too, are much greater. Buffering is annoying in a movie; it will cause you to die in a game. When aiming a highly precise gun-sight against a fast-moving target, frame drop and lags quickly become intolerable.

These challenges might sound surprising. If cloud streamed games are “video,” tens of millions of consumers watch 4K YouTube videos on demand at 60 FPS every day, and all the hard game processing work is done at an industrial supercomputer, why is gaming so hard? To answer that, most video is (1) Fully encoded, analyzed, compressed, etc., long before it’s delivered to the customer; (2) Not livestreamed, and when livestreamed, need not be delivered in absolutely or even approximately in real-time; (3) The same for all viewers; (4) Predictable in demand; and (5) Watchable at relatively low bitrates/quality levels (e.g. 360p).

These attributes produce considerable added complexity, cost, and service assurance risk. But more important is how it produces unique challenges that precludes using many of the “tricks” used by music and video streaming services to manage costs and deliver reliable user experiences.

Challenge #1: Data Center Costs and Inefficiencies

Server-side rendering is incredibly expensive. The GPUs themselves are quite costly and data centers have notorious heat and electricity requirements. It's one thing when a game console is at home, by itself; it's another when it's sitting in a rack with many others, generating tremendous amounts of heat from its graphics cards. Most estimates suggest at least $0.35-40 per user hour. This means that just the direct marginal *server* costs of 20 hours of gaming is $7+.

Cloud game streaming also has fewer opportunities to achieve "scale efficiencies" than most cloud-based content solutions. Players cannot be too far away from a remote data center, or latency increases to the point of intolerability (see above - this is also why multiplayer typically sorts players on a regional basis; there's too much latency from greater distances). This means a cloud game streaming service can't just have one "hyper-scale" data center. Many locations are required around the country and globe. This requires enormous capital investment and makes upgrades and technical fixes harder to implement.

Furthermore, cloud game streaming servers spend most of their life with excess capacity due to the need to "plan for peak" demand. If there are 75,000 players using a service at 8PM on Sunday night, the local data center needs to support 75,000 players. At 4AM on Monday, however, there might be only 15,000 players, and thus 80% of capacity goes unused. At most times, less than 40% of these racks would be in use. As a consumer, you can purchase a $400 GPU and let it sit offline as much as you want, but data center economics are oriented toward optimizing for demand.

All cloud-based servers face the challenge of optimizing for utilization. This is why AWS, for example, gives customers a reduced rate if they rent servers from Amazon in advance ("reserved instances"). Customers are guaranteed access for the next year, because they've paid for the server, and Amazon is pocketing the difference between Amazon's cost and the customer's price. AWS's cheapest Linux GPU reserved instance (equivalent to a PS4) costs $2,709 for one year, before video bandwidth.

But if a customer only wants to access servers when they need them ("spot instances") the reverse is true: the customer isn't guaranteed to get the server they want. The customer can rent what Amazon has available, but if they're out of a particular server in one region (GPU instances are particularly limited), the customer will have to try another place or another time. This is acceptable because the customer hasn't paid anything yet.

The problem for cloud gaming is that players expect to play when they want. The player has already paid for the services, either via subscription or by buying their game in the case of a service like Stadia. If the service doesn't have GPU servers available when the player wants, the player will buy their next game back on a locally-processed PlayStation, where they don't have to worry about their home game console being "unavailable." So a cloud gaming platform can't optimize for demand like a standard cloud service, unless they intend to charge the user 7x the price of a PS4 for one year of access.

Note that off-peak capacity can sometimes be sold to universities or other organizations that do batch processing, but at lower rates. This is because the hours with the greatest excess capacity (e.g. 2-10AM) have little overlap with the working hours/needs of nearby businesses.

Challenge #2: The Unique Limits of Live Game Streaming

Netflix and other video services can use several tricks to manage the end-user experience.

One option is to push several extra minutes of a video to their device using spare bandwidth. As a result, a temporary loss of Internet connectivity can occur without the user even knowing; their video just keeps playing. This obviously cannot happen to events that haven't yet happened - there's nothing to pre-load. As a result, this solution isn't possible.

But even when content isn't pre-loaded far in advance, all video playback is "playing" a second or so after it's "streamed" to a device. Unlike the example above, this caching strategy isn't designed to bridge a temporary loss in connectivity. Instead, it's to manage packet delivery issues. In reality, data doesn't arrive complete and in order. A viewer might receive video frame 12,070 before 12,105, for example. But as long as the video has buffered or has a delay, local software (e.g. the Netflix app) can re-order these frames before they're seen. This even occurs with "live" video; sports fans want to watch "live," but that doesn't mean they can't be 1-3 seconds behind the ballpark. However, games must be real-time, or the player will die. And if a packet is missing in real-time, correction requires skipping the missing frame (creating a jittery experience) or waiting for it to arrive (adding latency).

The caching issue also prevents game streaming services from using Content Delivery Networks (or “CDNs”) to ensure reliable, non-buffered video experiences. CDNs essentially “manage” traffic by deciding which broadband networks will be used to route traffic to and from the user. This ensures greater delivery reliability, but also increases latency as networks with spare capacity are prioritized over those that are shortest, and as a result, all content is cached.

Another, similar solution used by video providers is a strategy such as Netflix’s “Open Connect.” This allows various telecoms, like Comcast or Verizon, to host a video service’s content inside their network. This means that when a Flordian Netflix subscriber loads Stranger Things, this content doesn’t need to travel from regional Netflix’s data centers, but instead only a few blocks or miles away at a local center owned by a telecom company or other third party. This reduces the risk of technical hiccups, as well as the cost of bandwidth for both Netflix and the internet provider, which means the latter can also avoid unnecessarily clogging their network. However, a cloud game streaming service can’t locally pre-load content that hasn’t yet happened.

The closest one can get to “Open Connect”-like experiences is to set-up myriad mini-data centers across the country/globe across various Internet backbone sites. However, this is not storing files like Netflix is doing - it’s what’s called edge computing. This equipment requires enormous capital investment and makes upgrades and technical fixes harder to implement. In addition, this further exacerbates the aforementioned peak problem. Now capacity has to be optimized specifically for Boston and Philadelphia, rather than the northeast overall.

The final tool is offline compression. Non-live video services like Netflix receive all video files hours to months before they’re made available to audiences. This allows them to perform extensive analysis that allows them to shrink (or “compress”) file sizes by analyzing frame data to determine what information can be discarded. Netflix’s algorithms, for example, will “watch" a scene with blue skies and will decide that if a viewer’s Internet speeds drop, 500 different shades of blue can be simplified to 200, or 50, or 25. The streamer’s analytics will even do this on a contextual basis - recognizing that scenes of dialogue can tolerate more compression than those of faster-paced action, for example. This is multipass encoding. However, games don’t have this opportunity. The content is live and needs to be encoded for transmission as fast as possible. This means they’re not optimized and require greater bandwidth availability.

Challenge #3: Enormous Bandwidth Costs and Operational Burdens

The cost of delivering cloud game streaming also goes well beyond server investments and infrastructure operations. Even at tremendous player scale, the total cost of delivering online multiplayer positional and input data is relatively inexpensive (the real cost is pushing updates and initial game files). Sending 60-120 FPS high definition video frames per second is many times more costly. And as discussed earlier, live game streaming services lack the luxury of multipass encoding. This means that a 60 FPS video game stream isn’t just 2.5x larger than Netflix’s 24 FPS video files.

In aggregate, the cost of bandwidth is believed to be up to twice as great as that of compute and rendering - an already high $0.35 per hour of playtime. This is connected to why the big technology companies are seen to be the only viable suppliers of core game streaming infrastructure/services. Amazon, Microsoft, and Google, for example, already pay lowest bandwidth pricing from private networks. After all, they’re the largest customers. They’ve also spent 10-20 years purchasing largely dormant fiber optic cable (“dark fiber”), for which there’s only an internal cost to use rather than a literal one. In some instances, the Big Three will also form “peering” agreements with third party networks, allowing each party to use one another’s fiber for free.

Additionally, the networks connected to the data centers need to be re-prioritized for cloud game streaming. To explain this, it helps to return to our earlier example of two Manhattan-based players whose data was routed through Chicago and back in order to avoid local congestion. This is the default and dominant traffic model, as there are almost no web experiences that are materially affected by milliseconds. When a web user loads a Facebook page, for example, they don’t notice or care about the difference of 30 milliseconds. But as discussed above, milliseconds are material in video gaming.

Because so few applications require real-time delivery, the financial cost for specific routing is quite high, as is the operational burden of negotiating specific routing with dozens of individual carriers. This can make sense for, say, algorithm-powered high frequency stock trading funds (some of which have literally funded the laying of dedicated fiber optic cable across the Atlantic Ocean). However, it’s challenging for consumer-grade leisure services. This is another reason why only the most cash-and-dark-fiber-rich technology companies are expected to lead in cloud game streaming.

Challenge #4: Consumer-Side Obstacles and Costs

Finally, there are the consumer-side issues. The “last mile” problem is well known. Even if everything runs perfectly up until a household’s curb, the last dozen feet of copper wire could be flimsy and introduce intolerable latency. The “last ten feet” problem is even more acute here. In-home WiFi is almost guaranteed to be terrible, adds a lot of choppiness in the way it processes packets, and doesn’t properly prioritize traffic based on ultra-low latency requirements. Furthermore, cloud game streaming requires significantly higher broadband caps and speeds. Today, playing *Fortnite* (an online-only game) requires a fraction of the broadband of Netflix when playing (though admittedly there are regular patches downloaded). The reverse is strongly true here; cloud gaming takes up significantly more bandwidth than playing locally. Netflix’s peak quality (4k, 60 FPS) requires 3.5-7GB of data per hour and is only available for a small portion of its content and playable only on a few television sets and iPad Pros. Conversely, Google Stadia’s *minimum* quality (720, 30 FPS) is 4.5GB. The quality most local gamers expect (1080, 60 FPS) starts at 12.5GB. Most homes can’t support this, and others will need to spend far more to have it.

And there is the problem of network asymmetry. Consumer networks are designed at the infrastructure (i.e. wires to your home) and hardware (e.g. model/router) level for substantially higher download speeds and volumes than upload speeds and volumes, and to prioritize the former over the latter. This is consistent with almost all consumer behaviors to date, but when cloud game streaming, a system needs to be able to download large amounts of data while simultaneously sending inputs as fast as possible.

The biggest challenge may be the actual pricing for cloud game streaming services. This delivery model has a problem unique to both gaming and streaming media overall: high marginal costs. Every hour played is estimated to cost, on a purely direct basis, $0.35-40 of infrastructure usage costs and potentially much more in bandwidth plus infrastructure amortization. This is orders of magnitude greater than traditional online play. As a result, it can neither be given away for free nor does one-size-fits-all work particularly well. To this end, it’s likely that the consumer cost for tens of hours of gameplay per month will be unaffordable to most present-day gamers, which means cloud game streaming services may have to subsidize, eating into profitability.

This is particularly problematic because gamers today are incredibly resistant to and largely unfamiliar with tiered and per-use payment. Consoles and games, for example, are an upfront fee for unlimited play. Season passes and online multiplayer subscriptions have a defined duration, but during this time, there’s no cap on play.

#6: WHO WILL WANT "CLOUD GAME STREAMING, WHEN AND WHY?

Ultimately, there will be tradeoffs to exchanging local processing for cloud-based processing for the foreseeable future. Indeed, Xbox Head Phil Spencer continues to try and manage expectations for the timeline around cloud game streaming. In August 2019, for example, he told GameSpot, “I think [cloud game streaming] is years away from being a mainstream way people play. And I mean years, like years and years."

After all, cloud game streaming technology isn’t actually that new. G-Cluster began in Europe in 2000, was purchased by SoftBank in 2004, and was streaming games in Japan for close to a decade before expanding through Europe via Orange in 2014. OnLive arrived to much fanfare in 2009 and died by 2012, with Sony scooping up the patents. PlayStation Now started life as Gaikai in 2008, was acquired by Sony in 2012, and began streaming PlayStation back catalog in 2014.

Game publishers and service providers have also tried their hand at cloud game streaming. Nintendo and Square Enix streamed Dragon Quest X to the 3DS in 2014, while Nintendo and Capcom brought Resident Evil 7 to the Switch in 2018. EA and Comcast launched a game streaming service to set top boxes in 2015. Square Enix launched their cloud game streaming platform Shinra Technologies in 2014.

Internet speeds, reliability, and data caps have all improved substantially over the past few years. However, it’s not yet clear that the perks of cloud game streaming are sufficiently appealing given the tradeoffs. Even after it offered 90 days free and added in more AAA games at no additional fee, Google Stadia doesn’t seem to have picked up many customers. Six years after PS Now launched, despite a 50% price cut from $20/month to $10/month late last year, the service only has 1MM users - a small fraction of the 110MM PlayStation 4s sold to date and 36MM PlayStation Plus user base. Notably, Microsoft’s Xbox Game Pass, which is cloud-based game subscription rather than a cloud game streaming subscription, has 10MM subscribers out of 50MM total Xbox Ones sold (some subscribers would be PC-only).

This is important because many believe that cloud game streaming will massively expand the number of AAA gamers by making it easier and cheaper to play games and access gaming hardware. And historically, this is what happened as gaming moved to new hardware platforms - from arcades to consoles to PCs then online PCs and then mobile devices.

And there is without a doubt revenue potential in cloud game streaming. Many gamers will end up spending another $10-20 per month in addition to their physical consoles, for example. Others will spend more time playing now that they can play the full version of their favorite games wherever they are, rather than a downgraded version. And those who already play these titles outside the home will likely spend more money on them; there's no point buying a skin you can't see on your old iPad. The ability to skip lengthy downloads will also increase playtime and lead to the average gamer playing more games.

However, the total lift to the traditionally-defined gaming market is unlikely to be substantial for the foreseeable future. Over the past several decades and platform changes, the expansion in the size of the gaming population has come from continued (1) decreases in the cost of hardware runtime; (2) decreases in the cost of game playtime; and (3) increases in the number and diversity of casual games.

For hundreds of millions of players, the hardware cost is already zero (they already own iPhones are iPads, the focal devices for cloud game streaming, or their family members do). What's more, numerous AAA titles are already entirely free-to-play (e.g. *Fortnite*, *Call of Duty: Warzone*). Cloud delivery might make the free-to-play experience better for those using iPads and iPhones to play, but it also means they become expensive. And at least to date, playing with 20-year old graphics doesn't seem to be an impediment.

In addition, most televisions will still require additional hardware in order to play a cloud game. Playing Stadia on a television, for example, requires a $60 Chromecast Ultra. And most players will still want console-grade controllers even if they stop using a local console. This requires another $60+ per player.

Furthermore, the games most likely to entice non-gamers to gaming are casual/social in nature, not computationally rich ones. Examples here include Guitar Hero, Wii Sports, Brain Age, Roblox, Words with Friends, and Angry Birds. Few non-gamers will be convinced to start gaming because *Call of Duty* is bigger, more realistic, and easier to access. To this end, it's important to emphasize that more than 50% of households already have a game console. There's a reason your mother, younger sister, or father doesn't play your Xbox One when you're not home.

Similarly, those that care most about computational power - today's AAA players of *Call of Duty* or *Uncharted* - are the least likely to substitute unreliable cloud-based hardware for reliable, low latency consoles. They might *add* cloud access to enjoy their games outside the home, but that's the aforementioned revenue upside opportunity, not upside in the number of players.

To that end, we can refer to Epic Games' Founder and CEO Tim Sweeney. He believes that although Internet reliability, speeds, and capacity are improving, local processing power is improving at a faster rate. As a result, "initiatives to place real-time processing on the wrong side of the latency wall [will remain] doomed to failure."

This opinion might be too fatalistic, but at least for the foreseeable future, it produces a hard-to-overcome “chicken and egg” problem. Today, all of the cloud game streaming providers have offered only games that could be (and are) played elsewhere. Every one of the hundreds of games that have been available on PlayStation Now - and there are hundreds of titles - can be played on a regular PlayStation.

Gamers buy consoles/PCs to play games. For years to come, hardware-based processing will have huge advantages, and most AAA gamers will continue to use them to game. If the vast majority of gamers are using local processing, the vast majority (and most important games) will be produced for locally processed hardware. Many of these titles will still be available for cloud game streaming services - indeed, Stadia and xCloud might even run the most beautiful versions of these games - but they won’t take advantage of the unique capabilities of cloud-based hardware. They will still be limited in complexity and scope, for example, and suffer from multiple truths.

This, in turn, gives console gamers no compelling reason to switch to cloud-only gaming, versus use cloud access as a way to play outside their home or on an iPad when the living room screen is busy. And as long as this is the case, game publishers will have no business case to make cloud-based games.

Eventually, however, we will start to see cloud-specific titles. Google, for example, continues to buy independent studios, and like every console platform before Stadia, it will need to fund money-losing exclusives to drive adoption and differentiation. This makes it critical to understand exactly what cloud-based games might offer those who currently enjoy locally-processed gaming and/or those who thus far, haven’t been interested in gaming. Success will require at least one of those two groups.

#7: WHAT IS A "CLOUD GAME" AND WHAT CAN THEY UNIQUELY DO?

Discourse around cloud games is confused by the tendency to name a “cloud game” based on where processing occurs. This is unhelpful.

Microsoft’s suite of upcoming Xbox Series X titles are designed to also play on Xbox One and xCloud. Obviously, these aren’t “cloud streaming games” even if you can play them via “cloud streaming”. If Microsoft cancelled the former two access models, these titles would *only* be available via cloud processing. But to call them “cloud games” would be to describe nothing about the games themselves, only their access model. After all, they’d be entirely indistinguishable from non-cloud games. To extend this logic to the extreme, Nintendo could update Legend of Zelda Breath of the Wild so that it couldn’t be played any way but via cloud-delivery. Nothing about the game would have changed except its access.

And even if Google’s Stadia, a cloud-only gaming service, produced an exclusive game, that decision alone wouldn’t make a game a “cloud game.” The decision to place game logic, processing, access, etc., in the cloud is exactly that: a decision. And this decision does not itself affect game design or game play. Similarly, much has been said about “instant play,” in which a gamer could move instantly from watching gaming content on YouTube into the game (and in the precise moment watched). This may be a fun feature of cloud game streaming, but this too doesn’t make the game itself a cloud game *unless* it affects game design in a way that requires cloud game streaming.

A helpful way to think about cloud games is not to consider where processing occurs, but where it *has* to occur. Here, it helps to offer terms:

- A **CLOUD STREAMED GAME** is a game that can, but need not be played via cloud processing. An example here is Resident Evil 7. The game is available via cloud game streaming on the Nintendo Switch (which lacks the processing power to play it locally), but it can also be played as a locally processed game on PlayStation 4, Xbox One, etc.

- A **CLOUD ONLY GAME** is a game in which cloud processing is the only option available to a consumer. An example here might be if Google Stadia bought exclusive rights to Resident Evil 8, a title currently planned for most major 8th and 9th generation consoles. In this sense, consumers *could* have played this title through local processing, but are not given that option.

- What’s interesting are **CLOUD REQUIRED GAMES** - games that can only be played using infrastructure, computational power, or technology that no consumer could afford or a consumer device offer. In a simplified sense, they are the games that only a “supercomputer” composed of interconnected racks of industrial servers might run (Amazon CEO Jeff Bezos describes these sorts of games as “computationally ridiculous”).

There are a few ways to describe what a cloud required game might produce. Most obviously, a game’s graphics can become far more realistic and detailed. It will be possible to see individual strands of

a multiplayer game is a shared, synchronous, and live simulation. Today's shift to remote, supercomputer processing means more players can participate in an even more realistic and complex simulation. However, synchronization remains an issue. In fact, it's exacerbated by complexity - not unlike the way more complex Rube Goldberg machines are more likely to misfire or break.

In a normal online game, every player experiences a unique amount of latency: some are 30 milliseconds, others 12, more still at 75 (it's particularly challenging if/as players span the globe). These margins might seem small, but the viability of a simulation depends on the coherent and reliable synchronization of all inputs and outputs.

To return to earlier in the essay, remember that while cloud gaming moves processing to remote computers, every player is still running a distinct version of a shared simulation, and that simulation needs to be synchronized in real-time. It helps that these remote centers are closer together, fewer, and use far better infrastructure equipment than the individual home, but synchronization still becomes exponentially more difficult as more computers, data centers, servers, and players are connected. Similarly, the more richly animated the world, the more AI it contains, the faster-paced the interactions, the greater the number of decisions to be processed, and the lower the tolerance for failure.

The truth is, the reliability, construction, and latency of modern networks still cannot process the sorts of games that most cloud-hopefuls imagine. The fact that Amazon's servers can produce a "computationally ridiculous" game doesn't mean players worldwide can receive it or share in that experience together.

But more importantly, cloud games can be much more than just "bigger" and "better" versions of locally processed ones. It's not odd that "bigger" and "better" is our starting vision for such games, of course. Every new medium thinks in terms of a slightly less-limited version of the prior one. The first TV shows, for example, were just recorded plays with bespoke camera angles and zoom-ins. But what really drives a new technology platform is how content is created specifically for it, rather than adapted to it.

Gaming, in fact, is the best example here. This is why every major technology platform shift has seen new game formats and developers/publishers emerge and dominate. The most popular games and hardware manufacturers of the arcade era (e.g. Atari), for example, didn't lead the console era. Similarly, the console leaders (Nintendo, Sony) don't lead in online multiplayer games. The online multiplayer leaders (Activision Blizzard, Valve), meanwhile, don't dominate mobile, and neither group currently leads in the live-ops games era. Every gaming "platform" is unique not just in how it removes the limitations of the prior platform, but in the entirely new capabilities it affords. This tends to require and reward new thinking, not adaptation.

To this end, the leaders of the cloud-gaming era are likely to share little in common with today's giants. But that doesn't mean we've no clue what the future has in store. In a funny way, we've already seen several potential prototypes of cloud required games. And the first one was "played" six years ago and used a 24-year-old game.

#8: MILES: THE FIRST "CLOUD REQUIRED" GAME GENRE

Proto-MILES

In 2014, Twitch found itself broadcasting the first mainstream-experienced vision of a "cloud required game" via a social experiment called "Twitch Plays Pokémon." TPP shared much in common with the average Twitch broadcast. A private (and anonymous) gamer set-up a locally processed copy of a game (1996's *Pokémon Red*), with his/her video feed shared to Twitch, which then broadcast the video stream publicly. But in this case, the aforementioned gamer wasn't - and couldn't - control the game. Instead, all decisions were chosen by the video stream's viewers. This included which direction the hero moved, the Pokemon that were sent into a match, which moves they used, or which items were deployed. At first, this was done on a purely democratic basis. Moving the hero up, down, left or right, for example, was based on which option received the most entries in the Twitch chat.

At first, gameplay was chaotic. New weighting methodologies were tested beyond just purely democratic voting (which led some users to protest these changes by constantly voting for counter-productive decisions, such as pausing and unpausing and pausing and unpausing the game). But eventually, the community of players found a functional rhythm. After 16 days, the game was completed, with a peak of 120,000 concurrents and more than 1.2MM players (the record for any video game, per Guiness Record). Since then, Twitch has "played" dozens of other *Pokémon* games, as well as many others.

TPP was run by consumer-grade hardware purchased by a regular consumer and located in their home. However, it was still a “cloud required game.” All processing happened outside the homes of every actual player, with each player participating by watching livestreamed video and sending in coordinated inputs. In addition, this structure supported what would otherwise be an impossible number of concurrent players. TPP only feels like something different because there was no *one* focal player, the graphics were lower-fi (not greater), and there was only one copy of the game playing. But none of these have anything to do with whether a game is being streamed, or whether it’s a cloud required game at that.

In the years that followed TPP, we’ve seen similar events that hint towards the future of cloud required gaming. In 2017, Reddit released a 1000x1000 (i.e. 1MM) pixel digital canvas which allowed Redditors to color in a single pixel every 5-20 minutes. The event had a time limit of 72 hours, after which “Place” would be forever locked in its current state.

The product of the experiment was stunning. A total of 1MM users participated, with 16MM pixels colored in and an average of 90,000 active contributors. Despite the volume of contributors, the diversity of options (while TPP was based on a pick one-of-four options model, Place participants could choose one of one million pixels and fill it in with hundreds of potential colors), and lack of any real instructions or goals, the Reddit community rapidly produced real art and coherent strategies. After the Reddit community “figured” out Place, it began to organize into factions, agree on shared objectives, and set to work to achieve them. These goals inevitably conflicted, leading to tribalistic fights over territory (and some groups simply wanted to prevent other groups from successful creation), but nevertheless, everything from the American Flag, to the Mona Lisa, and written text stories were produced and protected. Several creations were first attempted, then redone with greater detail and coloring. And again, the most any participant could do was color in a few pixels each per hour.

A timelapse of Place shows this entire history: the shift from chaos to small tests, real “art,” efforts to redo this art at better quality, successful and unsuccessful conquests, and the achievement of stability. (A great summary can be found here: http://sudoscript.com/reddit-place/). Crucially, the creators of Place, as with the developer behind Twitch Plays Pokemon, admit they had little idea what would happen. And how could they: it was up to the players.

moving hair on a character, blades of grass, or grooves in tree bark. Maps might memorialize every bullet hole, scuff, or smattering of blood, rather than have them fade away after a few seconds. What's more, each of these visuals might be unique to the exact moment and physics that produced it, rather than copy and pastes of the same template effect.

It will also be possible to have enormously larger maps and avoid the need for loading screens (often disguised via doors and other animations) as the player moves through them. Further still, games will be able to fully procedurally generate environments and content, rather than be constrained only to what a developer specifically designed. What's more, these experiences might be tailored to the specific needs, playstyles, and interests of the player.

One of the most hotly anticipated benefits is the ability for a game to include far, far more concurrent users in the same environment or match. Today, *Fortnite* maxes at 100 concurrent players, with *Call of Duty: Warzone* peaking at 150. There are several constraints to expanding this gap (more on this later), but local processing is a key one. There's simply a low limit to how many players (each of whom is constantly generating input and positional data) an at-home processor can track and render.

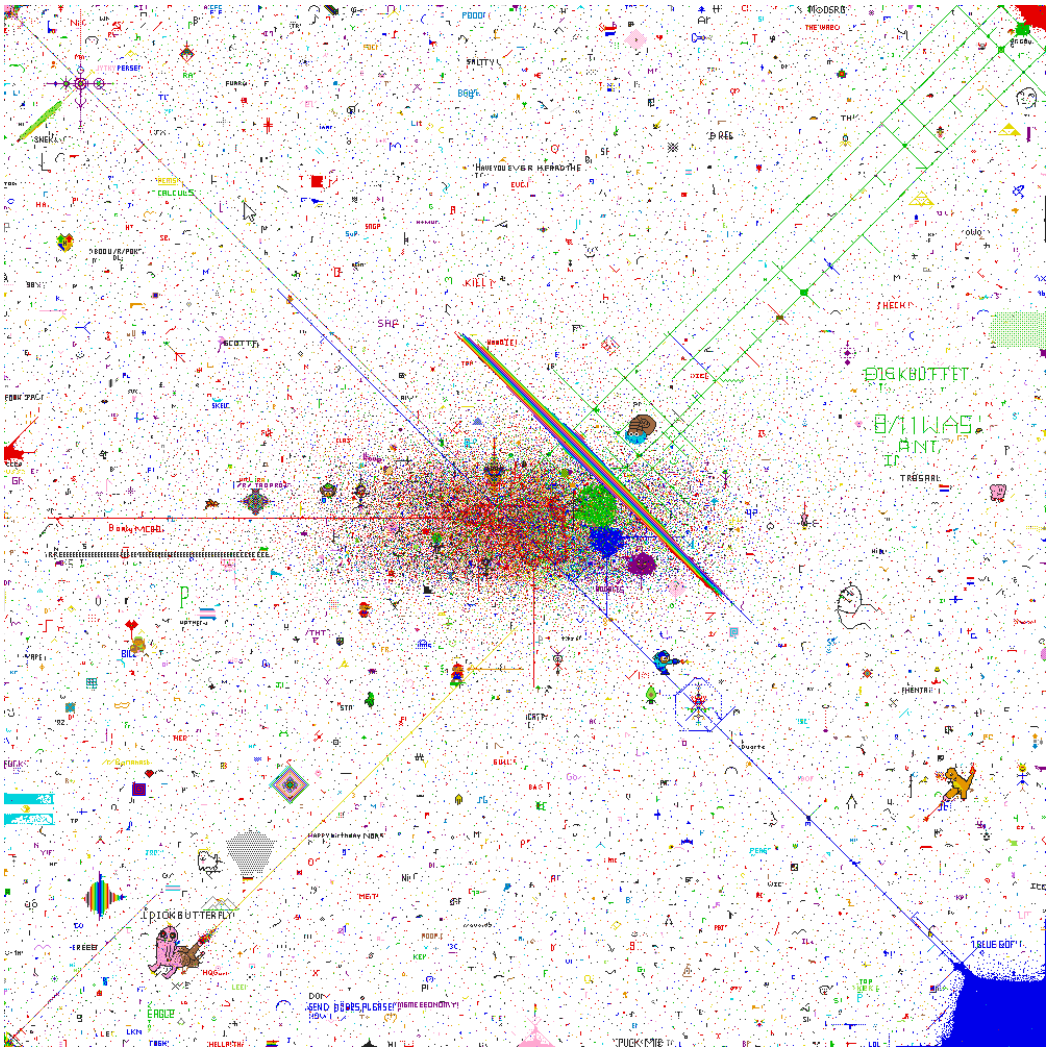
(Notably, while 11MM users participated in *Fortnite*'s live Marshmello concert, they did not do so together. In truth, these 11MM users were split across 120,000 or so separate instances of the event, each occurring at slightly different times and capped at 100 users each)

But more difficult and important than **what** is now possible is the **why**. It's not clear a much larger *Fortnite* map or 10x the players is "better." In 2014, Square Enix worked on a game with cloud AI and procedural map generation, and no load times under its Shinra Technologies banner. However, the company admitted that these advances didn't make for a better game. In fact, Square Enix found that the world they created lacked the warmth and sensibilities of a smaller, but more hand-crafted environment. It was a world "bigger than *Skyrim*, but less interesting." Meanwhile, games like *No Man's Sky* have delivered on the potential of procedural AI generation without needing to be cloud required. The removal of load times would delight any player, but they don't "make" a game.

Improved detail is equally misleading. More realistic games, especially to the degree of strands of hair or unique bullet holes, don't make a game better. The best selling game of the current, eighth generation of consoles, *GTA: V*, was made for and released on the seventh. It succeeds because it's fun and diverse, not because it's a realistic portrayal of organized and anarchist crime. The most popular AAA games globally, *Minecraft* and *Roblox*, look like games from the early 2000s and could mostly have been made then too. They're about creation, not simulating reality.

Put another way, imagine a version of *Call of Duty* where the implosion of a building is unique to the exact manner in which it was hit by an explosive, looks indistinguishable from the real world, and consistent for all players. This sounds cool, but doesn't make a difference to gameplay and is incredibly expensive to run via cloud. To this end, it's important to note the tremendous quality and capability of Epic Games' forthcoming Unreal Engine 5, which is based on fully local processing on consoles, PCs and mobiles devices. As a result, players get extraordinary visuals and complex capabilities without publishers or platforms needing to incur billions in delivery and remote infrastructure costs.

What's more exciting about cloud delivery is how CLOUD REQUIRED games allow developers to work *around* the limitations of network infrastructure, not despite them. As mentioned at the top of this essay,



Another famous event was produced by a mod of Grand *Theft Auto: San Andreas* run by a Twitch "speedrunner" (someone who tries to complete a level or game as fast as possible). Specifically, this mod allowed viewers to spend money to apply cheat codes that would either aid or harm the streamer's efforts - such as sudden increases in gravity or traffic. The most significant codes cost more money. Again this experience wasn't cloud-based - everything was locally processed on the single *GTA* player/streamer's PC - but the "game" was actually about large numbers of users interacting with a video stream in near real-time.

Explaining the MILE Game Genre

There will be many new genres created by cloud game streaming. However, we think the first is likely to resemble the experiences listed above - something we call “MILEs,” or “Massive Interactive Live Events.”

MILEs are distinct from most imagined versions of cloud games (e.g. a more realistic or complex *Call of Duty*) in that there is not a distinct simulation or processing of a game for every user. In most cases, there will just be one simulation and every player (and there could be millions of them) is a co-equal participant, as was the case in TPP. In other cases, there might be one or up to a dozen focal players with their own simulations, per a ‘normal’ game like *Call of Duty*. In these cases, the role of the audience is to tune in to drive/shape the live gameplay. The typical analogy used here is the Hunger Games, in which the audience sends items into the live game to aid their favorite hero or creates challenges for their opponents. These will be MILE hybrids. A MILExBattle Royale in the Hunger Games example.

(One of many ways to think about MILEs is Netflix’s Bandersnatch - only in this case, every decision made by a viewer would affect every other user’s experience, the “game” would never reset, every decision is permanent, and it would always be live.)

As shown by TPP and *GTA*, MILEs don’t need their gameplay to be processed in the cloud. Individual, consumer-grade processing can still be used. What matters is that the delivery occurs via streaming video.

But when MILEs are cloud-based, you can then take advantage of the “computationally ridiculous” power of a remote server. This might seem to be returning us back to the earlier problems of cloud game streaming. But this is where the fact that MILEs use only one simulation (or up to a few dozen in the case of a hybrid MILE) matters - this allows a cloud required game to deploy incredibly sophisticated AI and rich simulations, while ensuring real-time synchronization and avoiding network and home infrastructure-related issues.

The right way to think about this is to consider the earlier example of *Mario Kart* on a Nintendo 64. In this case, there was only one simulation (or version of the game) running, with each of four players accessing it. MILEs operate similarly: every player is tuning into a shared simulation, and then providing input into it. By avoiding the standard game model of myriad individual simulations (local or remote), a MILE need not synchronize simulations, only inputs. And per the start of this essay, multiplayer has been working and scaling under this model for decades. Note, too, this means that any platform can distribute a MILE. Twitch and browsers can, but so too could Disney+, Netflix, or Facebook Messenger.

The ways to interact with a MILE are up to the creators to decide. TPP had one view because the game was built like that; *Mario Kart* had four. You could have dozens of cameras or unique overlays that participants could tap into. Or, combined with cloud gaming level experiences, participants could have their own camera. This is a function of optimal design and cost, not technology.

This does mean the range of interactivity afforded by MILEs will typically be less than that of *Call of Duty* or *Legend of Zelda*. However, they will be designed for different goals. Instead of simulating the “real world” or creating a fully immersive experience, MILEs will be designed to support extraordinarily large numbers of participants and enable them to collaboratively tell a story and/or influence a shared world. In other words, they will be designed specifically for what’s uniquely possible via cloud-based gaming. However, it’s important to highlight that viewers will still have fully personalized experiences and control over how they watch or access a MILE. TPP had only one video stream because *Pokémon Red* had only one camera angle. *Mario Kart* showed that infinite views are possible from the same simulation.

Returning to what’s possible via MILEs, imagine, for example, an interactive Discovery Channel world populated by millions of real-time, evolving animals based on a centralized AI, each one “owned” by a viewer. For the foreseeable future, it will be impossible for these animals to be “played” by thousands of users, or for them to “exist” on local devices. But if all the users are connecting to the same cloud-based simulation, and then “tuning” into a cloud video stream designed for their non-player inputs, it begins to work. Under this model, the role of all viewers would be to observe or collectively act as “God,” or a pantheon of Gods each with different roles. There might be a team serving as the God of Weather; another, the God of Food. Some root for the mammals; others, the reptiles. Every God would own and control their own camera into the world, and every player could still have their own animal to protect and guide.

We can also imagine an interactive TV show that uses a cast of highly realistic characters that are actually played by a rotating cast of actors in motion capture rigs. This allows the TV show to “never” end, while also involving events and storylines that are impossible for a “real” human. It could be, in other words, a 24/7 version of LOST that’s capable of anything *Fortnite*, *God of War*, or *Pokemon* has done and more - all while looking even more real and allowing the audience to control the smoke

monster, the Island, or a Dharma Station. Some of this is possible today via local processing, but the characters could never look “as real”; their visuals would vary enormously by household and device (and often not work at all), and the cost would be prohibitive (and unnecessary).

Imagining the Future of MILEs miles away

It might seem odd to predict the future of gaming based on experiences that either were based on decades old games or even older technology. However, Twitch Plays Pokemon, Reddit’s Place, and *GTA: San Andreas*’s Cheat Code Speedrun help to reiterate an ongoing truth in gaming: social experiences, not technical capability, drive engagement. Each of these experiences, and dozens more like them, were fun because they were live and shared experiences that involved the audience. Over the past decade, video behaviors have increasingly shifted towards private viewing and on-demand payback. But the human love for communal, real-time events endures. It’s why we laugh more while watching a comedy in a theater, why few sports fans will watch a non-live match, and why unscripted TV experiences like The Bachelor and The Voice have struggled to shift to non-live, privately viewed SVOD services like Netflix.

Note, too, that the gaming industry has an extended history of creating/discovering new genres through forks and mods of old games. The most famous example might be the MOBA genre, which emerged out of a user mod of the RTS title Warcraft III. What’s more, it’s rarely the first mod or adaptation that scales. The most successful MOBA, for example, isn’t the first user mod, Defense of the Ancients, nor the sequel produced by Valve in partnership with DOTA’s creator. Instead, it’s Riot Games’ League of Legends, which was released six years later.

To this end, it’s important to see beyond today’s proto-MILEs. As games are made specifically for the format (rather than grafted to it), supported by technology purpose built for it (versus adapted to it), using information learned from prior player behaviors (instead of being hypothesis based), and for players already familiar with its interaction models (as opposed to discovering anew), we will see continuous advances in the diversity of MILEs, their appeal, and their engagement levels.

What’s more, MILEs-like tests have been happening for years at EA, Ubisoft, Square Enix (e.g. density demos that show off how many games a GPU can handle concurrently, how far they could push realistic rendering, what shared game environments where you could see other player’s views might look like, and what a shared world might entail). As always, it will take developers and platforms time and iteration to learn how to best design a MILE - what players find the most fun, the role of narrative, gamification, monetization, and so forth.

Most exciting is the way MILEs have the potential to dramatically expand the “gamer” player base. Unlike today’s AAA games, they involve the “player” without demanding their full attention. This allows gaming to move into other use cases. Consider, for example, how many people “watch” The Office while cooking, cleaning, or working. MILEs will allow the audience to lean in, lean back, or lean out as they choose.

Similarly, the skill requirements are much lower than traditional, AAA-looking/feeling games. It takes considerable time to become “good” at *Fortnite*, but helping change the weather in the desert of the Interactive Discovery sim doesn’t require skill. You can engage at a level that’s comfortable to you, at the location and time of your choosing. Furthermore, an audience's sense of “ownership” and “involvement” is incredibly powerful. Today, it doesn’t matter if a player beats God of War or makes a given decision in Bandersnatch. But voting in American Idol does.

In time, some of the technologies and concepts involved may even see MILEs take over unscripted reality programming too. Imagine versions of *American Ninja Warrior* where the audience controls the stage or chooses the contestants. Or *Who Wants to Be a Millionaire* where the audience chooses the questions, serves as a lifeline, or plays live, competitive mini-games to give their favorite contestants an extra lifeline in the first place. As always, gaming is taking over all other media categories.

Matthew Ball (@ballmatthew) & Jacob Navok (@jnavok)