

Veebeam 1.0: Technically Speaking

Veebeam is a complex product, the focus was to get great video quality off the laptop and onto the HDTV. It deploys a lot of technologies that a user is either unfamiliar with or unaware of due to how seamless Veebeam is to use. However to truly understand what Veebeam is and what it does we need to step into its technical world and why such a product would have been impossible, even just a few years ago.

Veebeam is built as a set of layers. Of which the highlights include:

- Wireless USB / Ultra Wideband / WiMedia
- GDI & MMX Colour Space Conversion
- DirectShow codec management
- H.264 Real-time encoding
- DLNA: Digital Living Network Alliance

Each of these technologies has been chosen for a very key set of reasons and by looking at them you begin to understand why Veebeam is what it is.

Interference

The biggest problem with Wi-Fi is that interference is a significant problem. It comes from a variety of sources, other 2.4GHz devices such as Bluetooth, other Wi-Fi networks or reduced capacity due to running an 802.11 network in mixed mode.

We (Veebeam) have developed a Wireless USB standard chipset that uses the UWB (Ultra wideband) WiMedia protocol to give us high speed sustained rates without interference. This technology has taken years of development but in return offers us significant benefits over Wi-Fi.

The range is shorter as the power-per-bit is less than 802.11; meaning it's more suitable for sustained use on battery based devices. The con is range, however when you consider you're typically controlling a laptop on the sofa then throwing the video / web onto the TV we didn't see it as a significant loss. In fact, reduced range means less interference/loss-of-capacity from similar devices; with 802.11n, your neighbour's neighbour's neighbour can interfere.

By using Wireless USB (through a standard USB port) we get the bandwidth we need and the ability to work with older laptops. Of course as 5GHz Wi-Fi becomes more prolific and 802.11n becomes dominant we may not need it.

Rules of Engagement

There are two key modes within Veebeam:

- Desktop
- Player

Desktop mirrors (unsurprisingly) the desktop on the TV; be it with a slight lag.

This is due to the technologies involved. We capture the desktop in real-time, convert from RGB->YUV and then encode using a high-speed H.264 profile @ 1080p. This is then sent over to the receiver using DLNA as a protocol. We will explain the lag later, but in short it is needed for a great video experience.

Long story short, whatever you use to view video / stream from the internet can come out on the TV (along with sync'd sound). Therefore you can throw your latest CBS/NBC shows for example direct to the TV for a more sane viewing experience.

The other mode in Veebeam is a **Player**. Unlike your regular video player, this one is very special. It's capable of cracking open any file you have a codec for.

If within that file there's an H.264 stream then we'll repack it into a standard container and send it without transcoding. If it's not in H.264 we'll decode to YUV, encode using our same H.264 technology and then send it.

The result is that whatever codecs you add to your PC, we'll then use them intelligently and with minimum loss to quality (in most cases, no loss as we only repack). We see this particular useful for future proofing and flexibility.

Why no Aero Effects?

This is something that confuses people greatly. When you connect and start using Veebeam we turn off the Aero Effects. This is done for a very important reason and common to all mirrored mode drivers.

Windows Vista / Windows 7 have a component called the Desktop Windows Manager (DWM). It's responsible for both drawing and compositing what you see on the screen. Its architecture is very sophisticated and it understands all sorts of things, including how to off-load tasks to the GPU (graphics hardware) that would cause the CPU (processor) strain.

Unfortunately that comes at a cost. If we want to grab the desktop and send it to the TV we have to go all the way to the GPU to grab it. And here's where it gets really complicated. To grab the desktop from the GPU we have to let all the pipelined processors finish what they're doing, lock-down, then grab the desktop, then pass that all the way back over the AGP/PCIe bus into main memory.

The upshot is this is slow. S-L-O-W. To compensate for this unfortunate problem we have to tell the DWM not to apply Aero Effects. The desktop is then rendered in main memory before being copied to the GPU and we gain a SIGNIFICANT performance benefit in our **Desktop** mode.

To be honest, when you consider what Veebeam is trying to do, this isn't really an issue. A user typically wants to watch streamed video and the Aero Effects don't apply at all when this is done. We restore Aero Effects when the dongle is unplugged, so session speaking, you get the best of both worlds.

Why no Direct X?

The reason is almost the same as why we disable Aero Effects. The type of work we need to do, doesn't lend itself well to using Direct X hardware acceleration. Additionally, we gain a massive level of backward compatibility using GDI that we can't get out of Direct X.

Until the services step-up and give us what we need with backward-compatibility to Windows XP and older hardware it's just not a viable option.

Why is there lag?

This is probably the area of hottest debate; people assume we're just a replacement to the HDMI cable. We're more advanced than this, we're a video product.

Our goal is to allow you access to streamed and offline video content on your TV. To do this, we naturally fit in with DLNA. This is a standard a range of devices can support. We view this as an important part of both allowing the user to control our various modes and a way with can create exciting and diverse products in the future.

Unfortunately it comes with some costs and one of them is a heavy amount of buffering. This is where the lag is introduced.

We do however view this as a problem we'd like to improve. There will probably never be zero lag, but the ability to drag windows around the screen in close to real-time is important to us.

This said, when actually watching video, this isn't an issue. The audio is always synchronised and if you close your laptop lid, you can't see the slightly earlier content appearing on the PC. In fact, we alter the power-model on the laptop when Veebeam is in use, such that closing the lid doesn't put the PC into standby to assist with this.

Got No Game

A short word on this: **Veebeam 1.0 is not suitable for a primary gaming display.** Not just because of the lag, even if Veebeam had zero delay, the overhead of the work needed to encode and pass data would drag any modern game down on many PCs.

We believe the performance hit while gaming simply wouldn't be acceptable. We never designed Veebeam to be suitable for a primary gaming display, someone could use it to view what is happening. It's really a video product.

Why Use Veebeam?

Veebeam is a very unique product at the moment. There's nothing else out there that does this level of real-time repack / transcoding / encoding without the user being swamped with technical setup.

Consider what you need to do to set-up Wi-Fi. Our Wireless USB requires none of this, yet is secure with an interference free, superior quality-of-service.

Consider what you need to do to connect to our receiver. You just have to plug in the USB antenna. There's no technical configuration required at all.

Consider what you need to do to view your Desktop on your TV. Nothing, it will just appear.

The product is aimed at giving a user complete control over video. We don't sell a subsidised product into a walled garden, such as Roku and Apple. A user is not dependent on what codecs or file formats the hardware supports; we're **augmenting what the PC already does.**

As such, we have a very strong position in giving a video experience without walls or restrictions.

Veebeam 1.0 is designed to let you choose what you want to do rather than us.