HoloLens Hands-On: How We Built An App For Microsoft's Augmented Reality Headset

Frederic Lardinois
@fredericl / 2:13 AM EDT • May 1, 2015

Microsoft's HoloLens is no joke. We've now tried the company's latest revision of its unreleased augmented reality headset and even built an app for it. The new hardware, which Microsoft also showcased during its Build developer conference keynote yesterday, feels very solid and the user experience (mostly) delivers on the company's promises.

Earlier today, Microsoft gave developers and some of us media pundits a chance to spend some quality time with HoloLens by building our own "holographic application" using the Unity engine and Visual Studio.

HoloLens is all about augmented reality. It's about placing objects into the real world, which you can still see while you're wearing the headset. It's not a virtual reality headset like the Oculus Rift, so it's not about total immersion. Instead, it lets you see objects on a table in front of you that aren't there in the real world, for example, and it lets you interact with them as if they were real objects.

When you first see somebody who is using HoloLens, you'll probably think something is wrong with them. They'll walk around things you can't see, make random click gestures in the air ("air-tapping" is what Microsoft calls that) and probably ooh and ahh a few times.

Until you've used HoloLens — and very few people have — this all sounds very abstract. But once you strap it on and use it, it's indeed a bit of a revelation.

The early (and highly positive reviews) from Microsoft's January event, where a few select members of the press got to try it, weren't an exaggeration.

Let me walk you through the experience Microsoft put on for us today. After locking all of our electronic devices in a locker (even smartwatches), Microsoft paired us up with a mentor who would walk us through the rest of the experience.

We all got our own HoloLens and a PC to code on and four members of the team walked us through building a simple HoloLens app. First, though, they taught us how to use the basic gestures that work across all apps: air clicks (for selecting stuff); gazes (which is really just about looking at things); and voice commands (which developers can easily customize).

With that out of the way, we loaded a couple of preset assets into Unity, made a few minor modifications to the scene, and then exported everything to Visual Studio. From there, deploying the app to the HoloLens (over USB) was just a matter of a few clicks.

The scene Microsoft had pre-built was a sketchbook with a few cubes on top of it, two paper airplanes that leaned against the cubes, and two paper balls that were suspended in the air over them (so they could bounce off the paper airplanes).

We started with a very basic, static scene and then added more interactivity from there. At every step, we would go to Unity, add new features, deploy the app and strap on the HoloLens again to take another look at our creation.

At its most basic, the Unity SDK allows you to simply place an object at a certain distance and height from where the device is pointing at when you initialize the application. That object is then suspended in that spot in space and you can walk around it.

We then added a bit of interactivity so the balls would drop when you clicked on them, as well as sound and — finally — the HoloLens' ability to detect objects in the world and have the whole scene and the dropping balls interact with it. Because HoloLens knows what you're looking at — and uses that to allow developers to place a small 3D cursor into the scene — you can easily use air clicks or voice commands to interact with objects.

That's really what HoloLens is about, too. We were able to drop the scene on a table for example. Then the balls would drop and roll off the sketchbook. So far so good, but the balls also dropped off the real table and maybe got stuck against the sofa next to it. That's very impressive, because that kind of real-world mapping is hard to do.

HoloLens constantly scans the world around you and then builds a (somewhat crude) model of it. That's how it knows where the ground is, for example. It doesn't know that a table is a table, but it does know that there's a flat surface there. The same goes for any other kind of objects.

While I originally assumed HoloLens — like most other augmented reality applications — needs some kind of QR code or markers to know where to place objects, it can actually recognize the world around you and then allows you to place objects where you want them. Using the Unity physics engine, it can interact with them.

For the grand finale, we added a new object to the scene that would explode when one of the balls hit it — and which then revealed a cavernous world underneath the floor with a stream and animated birds. That was obviously the most impressive demo and really showed the potential of HoloLens. You could even drop the balls into that cavern and see them bounce 20 feet beneath your feet.

The new hardware we used felt very solid and unlike the first prototypes, it's an untethered, standalone unit. You use a headband to strap the device to your face, and while the device is not exactly light, the smart weight distribution between the front and back makes up for that.

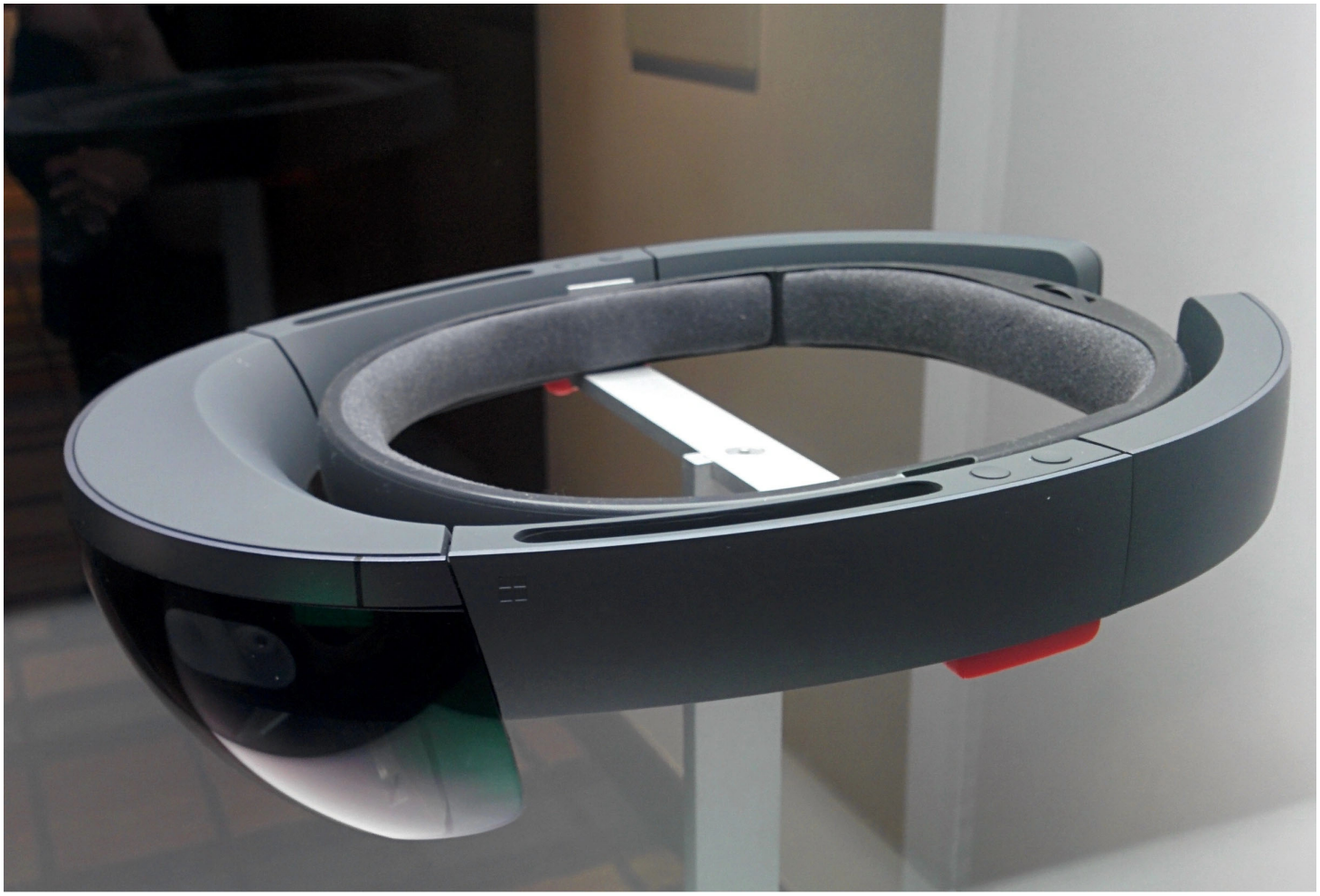
Microsoft admonished us to be careful with the units, but they didn't feel fragile at all. I wouldn't be surprised if this model turned out to be very close to the final production unit.

Going into the demo, I was especially interested in how well the holograms would lock into place. That, after all, is very hard to do, and if objects start randomly moving around as you walk around them, it's hard to feel like they were really there. I was surprised how well this worked. I found myself trying to kick the little paper balls with my foot after they had rolled off the table (but sadly, the algorithms are clearly not quite fast enough yet for this kind of real-time interaction).

There is, however, still a little bit of jitter to the holograms that is mostly noticeable when you get close to them. They really feel like they're locked in place, though, and that's the real achievement here, especially given that Microsoft is not using any markers for placing them. You can randomly move objects around in space. They will stay put. No QR codes necessary.

My best guess is that Microsoft is using one (or more) of Movidius' Myriad 2 vision processing chips in the device to be able to do the positional tracking necessary to enable all of this.

The one disappointment for me in trying HoloLens was that the field of view was very limited. This was not a totally immersive experience because objects would get cut off long before you would naturally expect them to drop out of sight. Some of the writers who saw Microsoft's first demo told me that they felt the viewing angle on these new devices was smaller than during the first demo. I can't verify that, though.



What did surprise me, though, was how little opacity there was to the holograms. I expected them to look somewhat transparent, with the background shining through. That really wasn't the case, though. They looked really solid and there was almost no transparency.

Today's demo obviously happened in a controlled environment, but it was nowhere as controlled as I expected. People moved around as four or five of us gathered around a table to look at our holograms and everything still worked really well.

At the end of the session, I came away very impressed. When I first heard about HoloLens, I thought this was a technology that was still very far away from being production-ready and I assumed that the demos Microsoft showed earlier this year were simply well-staged and had managed to pull the wool over the assembled tech press. Now, I wouldn't be surprised if Microsoft started selling HoloLens within a year (the company, of course, won't say when HoloLens will go on sale or at what price).