Kinect Gigapixel Viewer

Quickstart Guide:

The Kinect Gigapixel exhibit allows for the display of Gigapixel images and for navigation with the Microsoft Kinect. It also allows you to change the Gigapixel Image through the included CML file.

Installation:

* Install the TUIO Kinect Complete utility: setup_tuio_kinect_complete-1.0.0.exe

* Install the KinectGigaPixel.air file. Adobe AIR is required: http://get.adobe.com/air/

Running:

* Run the TUIO Kinect Tray program. A window called “Contour” should appear on your screen. Move in front of your Microsoft Kinect once this window is open to make sure it is tracking blobs, you should see white outlines as well as circles with crosshairs appear over significant blobs. This will also give you the chance to check the ideal operating distances for your Kinect set up. Generally this is 6 ft. away from the Kinect.

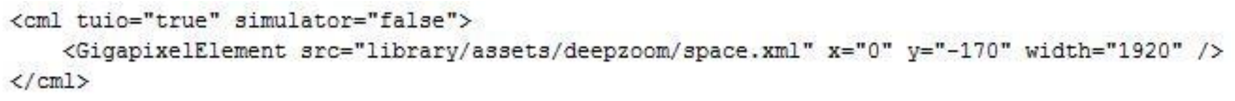
* Now you are ready to run the Kinect Gigapixel Element. Navigate to your installation directory for the exhibit, then open the “/bin/” folder to run the KinectGigaPixel.swf. This should open up the Gigapixel image, and you should be able to control it using motion gestures similar to touch gestures.

Editing:

* IF YOU ARE RUNNING WINDOWS 7: Adobe Air installations are automatically set to install into the Adobe AIR folder in your Program Files directory, which is set to Read-Only by default. To edit the CML, you will either need to change the default installation path, or do the following to give yourself write privileges:

    *Navigate to your “Program Files” in Windows Explorer and right-click on the folder, then select “Properties”
    *Click on the “Security” tab in the Properties menu.
    *Select the group or username that you would like to be able to edit files on, then click the “Edit” button.
    *In the lower box below users, click the “Allow” checkboxes for “Read”, “Write”, “List Folder Contents”, “Read & Execute”, and “Modify”, then click the “Apply” button in the lower right. Click “OK”.

* You can edit the exhibit through the “GigapixelElement.cml” file in the CML directory, "<Installation Directory>/bin/library/cml"



```
<cml tuio="true" simulator="false">
    <GigapixelElement src="library/assets/deepzoom/space.xml" x="0" y="-170" width="1920" />
</cml>
```

Joshua Hicks, September 6, 2012

* You must set tuio="true" and simulator="false" within the opening <cml> tag.

* To change the gigapixel image, edit the “src” attribute in the GigapixelElement tag to an alternate DeepZoom xml file, and be sure to include the matching folder branch for the xml in the "bin/library/assets/deepzoom" folder. For example, you may switch the src="library/assets/deepzoom/space.xml" to src="library/assets/deepzoom/wise2012-003-b.xml" to see a different gigapixel image. To create your own Gigapixel image, go to: http://openexhibits.org/?p=5893 or follow the more detailed tutorials at the end of this document.

* You may set the "minScaleConstraint" attribute in the GigapixelElement class opening tag in the CML to set the minimum size the GigaPixel scene may be scaled to. If minScaleConstraint is not set, or not set to a number above zero, the default value is .001.

* The GigapixelElement automatically maintains the aspect ratio of the image put into it. For this reason, when you want to set the width and height of the screen, it is recommended that you only set the width.

* If you adjust the width, or the image, you may need to set the "y" attribute to adjust the horizontal centering of the image on your screen.

* The GigapixelElement is automatically touch enabled and needs no Gesturelist attached to it to respond to the Kinect.

* The debug view for touch points is set in the CML to make it easier to see how interaction is coming through the Kinect. Occasionally a point may be noticeable in the upper-left corner, this is a known bug, and is not known to cause any usability issues.

Joshua Hicks - September 6, 2012

KinectGigaPixel Tutorial

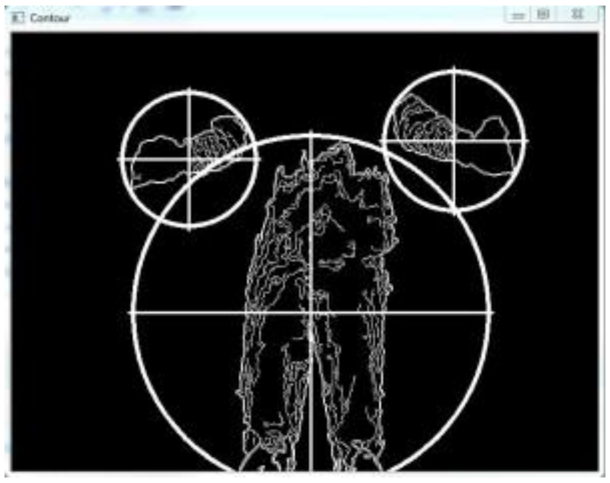
Part 1: Preparing the Kinect

The Kinect is an amazing new piece of technology for Natural User Interfaces (NUI.) What this means is that the Kinect is great for designing interfaces which basically disappear in front of the user. What most people are familiar with are Graphical User Interfaces (GUI), which have things like buttons, menus, cursors, and icons. The concept of the Kinect is to give a chance for people to break away from those things and command a computer using physical motions or vocal commands without even necessarily touching it, like you're some amazing computer wizard. Like Harry Potter meets Bill Gates.

At OpenExhibits, we have our TUIO Kinect Complete utility which opens up the door to start using the Kinect to interface with the framework instead of a touchscreen. Most recently, I updated the Gigapixel viewer to make sure it worked appropriately with the Kinect, and we have a fresh new package so you can use the Kinect Enabled Gigapixel Viewer after you've installed our TUIO Kinect Complete utility and do what everyone's wanted to do since the dawn of computing: wave your arms wildly at your computer and have it do something.

First thing's first, however. Before you can work with the Kinect, you will need our utility installed, which is included in the Kinect package. To install it, just run the "setup_tuio_kinect_complete-1.0.0.exe" file in the "air" folder with the AIR installer. The drivers used by this utility are the open source, LibFreeNect drivers, meaning you don't have to pay for them ever, but more importantly meaning you cannot have these and the official Microsoft Kinect SDK installed at the same time. So if you've installed the Microsoft Kinect SDK, install these drivers on an alternate computer, or uninstall Microsoft's drivers, or you will be sad.

Once you have this installed, you will need to run your "TUIO Kinect Tray" program, which will open up a window named "Contour." If you dance around in front of your Kinect now and look at your screen, you'll notice blobs with crosshairs or other possible markings appearing as you go in and out of the appropriate depths. This program is essentially a "blob tracker." The Kinect works through its two infared sensors, spaced apart much like your eyes, which allow it to measure depth down to the millimeter up to 12 feet (4 meters) away. This program works by taking that depth data and making educated guesses about what depth information in front of it is a person. This is not what's called a "skeletal viewer", which is a Kinect program that interprets educated guesses on a person's entire skeleton. This just tracks blobs, it's very simple.



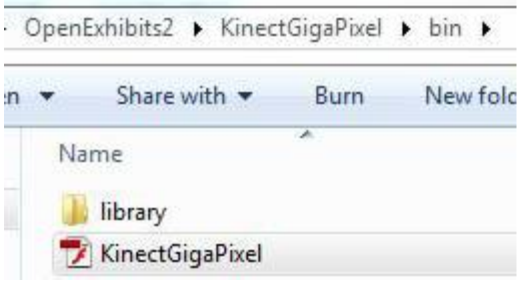
The blob tracker

I'll just say for now that you will want your Kinect set up on the edge of whatever you're placing it on, whether it's a desk, table, or shelf. You will want to stand approximately 6 feet away to achieve the ideal depth, though this may vary depending on conditions. I will talk more about the physical use of the Kinect later.

Now that your Kinect is set up and working, you will want to install the new exhibit, the Kinect Enabled Gigapixel exhibit. To install this, just run the KinectGigaPixel.AIR file located in the "air" folder of the project. This will automatically install the entire project for you.

For one, I feel I should point out that this package is actually using the GigapixelElement, which you'll see in the CML. The difference between an Element and a Viewer is that Elements are primarily one item, that do one thing. Viewers use elements, and commonly wrap them up in a frame, with info and close buttons, and allow them to be manipulated about the entire stage. You can have multiple elements in a viewer, you can have multiple viewers on stage, and you can have multiple elements on stage as well, but the elements will likely be harder to interact with without being wrapped in some sort of container or viewer.

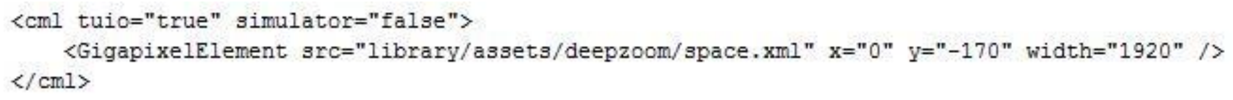
To run the project, make sure your Contour program that you installed earlier is running, then go to "<Installation Directory>/bin/KinectGigaPixel.swf" and run that file. The application should run full screen and you'll end up with a massive image that you can then float through using motion gestures in front of the Kinect. If it seems awkward at first, our third tutorial will go over understanding the experience and setting up your Kinect space.



If you can see your gigapixel image, that's great, but in this amazing package we included not just one, but two, two gigapixel images! But wait, how do you see the second one?

Well, if you're new to this, I'm going to introduce you to CML, which is very much like XML. If you're not new to this, this will be a helpful refresher. If you bothered to look at the Main.as file, you'll notice there's just not a lot there, that's because the OpenExhibits libraries handle mostly everything. What you want to do is take a look at the .cml file, which you can find in your "KinectGigaPixel/bin/library/cml/" folder, titled "GigapixelElement.cml", appropriately enough. You'll want to open that up in your favorite .xml editor, such as Notepad++.

CML (Creative Markup Language) is being developed for OpenExhibits as a way to easily lay out an exhibit, or application using our libraries to create complete, custom exhibits. When you open up the CML file, you'll find various things in tags using angle brackets that look somewhat like <this />, and these things can actually be combined in used in various ways to create custom displays, but like I said earlier, all you need right now is the <GigapixelElement/> tag. Within the brackets of your <GigapixelElement/> tag, there's an attribute called "src", this is the file that the GigapixelElement is looking up to find all its images to create the gigapixel image that you see. Right now the file it's set to should end in "space.xml".



```
<cml tuio="true" simulator="false">
    <GigapixelElement src="library/assets/deepzoom/space.xml" x="0" y="-170" width="1920" />
</cml>
```

So with your CML still open so you can edit, navigate to your "KinectGigaPixel/bin/library/assets/deepzoom/" folder. You should notice two folders, and two XML files. The folders are where the images that are linked to the appropriate XML files are. You don't need to edit these, you just need to learn where these files are and where they go in case you decide to make your own. Here, you should be able to see your "space.xml" file that's filling out the "src" tag in your CML. Now you can change that attribute in your CML tag to the other file in the folder, "wise2012-003-b.xml". Leave the rest of the folder branch in the filename, however, so it should look like this: "library/assets/deepzoom/wise2012-003-b.xml". Now if you go back and run your program again, you should have a completely different panorama.

Joshua Hicks, September 6, 2012

Now, this is great, and this information is also very useful if you're using the normal GigapixelElement or Viewer without the Kinect. You can also customize the program to display your own gigapixel images that you create, but what if you don't know how to make them? That's all right, I give a rundown in the second part of this tutorial on how to use Deep Zoom Composer to create your own gigapixel images out of large, high res images, or combinations of smaller images. In the third part, I'll give you a complete rundown of how to set up the physical space around your Kinect and how to interact with it to view the Gigapixel images.

Part 2: Prepare a Gigapixel Image

The Gigapixel Viewer allows a user to pan, drag, and zoom on gigapixel images of any variety. The Kinect itself is this amazing space age gadget designed to use depth information from whatever's standing in front of it to make user input available. Instead of touching your screen, touching your mouse, or touching anything, you stand approximately 6 feet away (though this may vary depending on your setup) and move your hands.

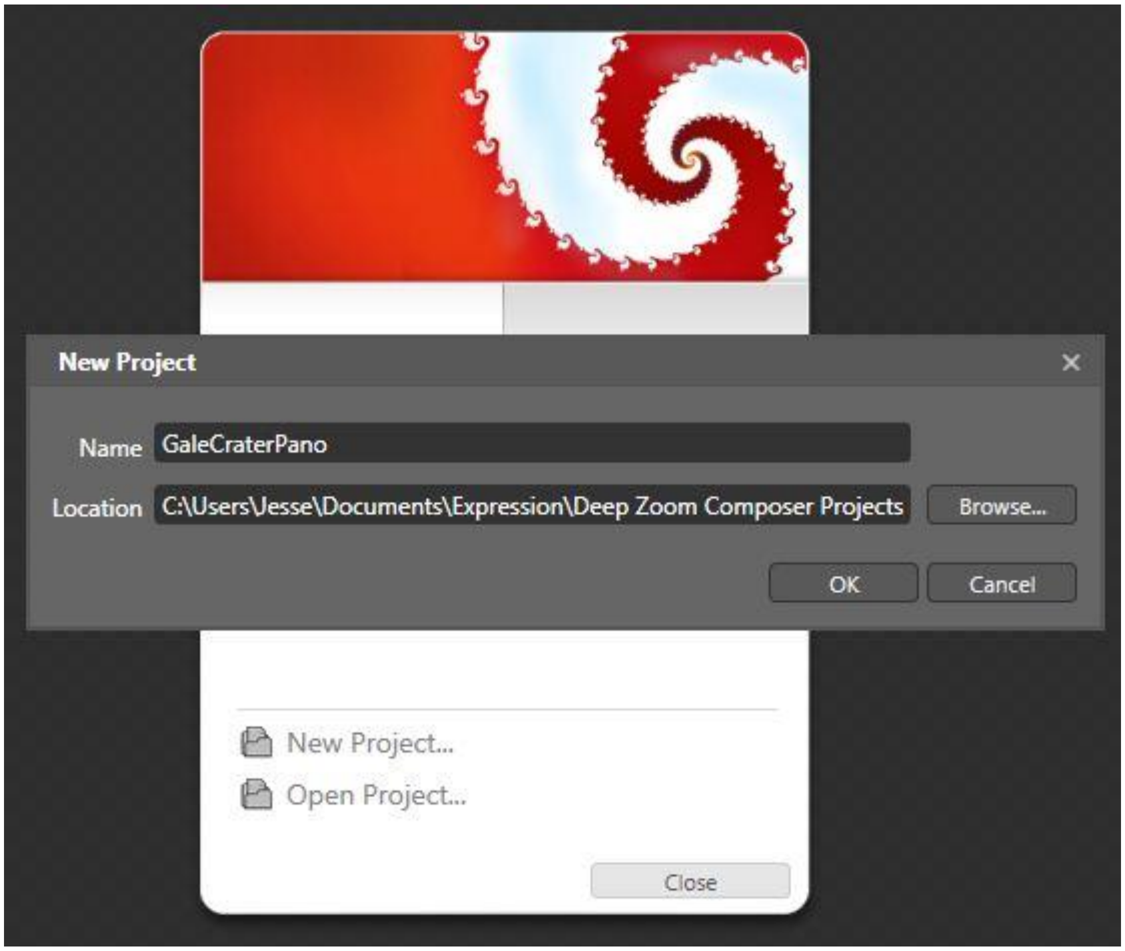
And most recently the Mars Curiosity Rover has started transmitting back images after this device of super-tech landed on the red planet in a combination of falling, parachutes, jet engines and a skycrane gently setting it down. It seems only appropriate that we take the images from this interplanetary device and set them up to view them on our natural user interface (NUI) device. This way, when people ask what you're doing while waving your hands around at a big screen, you can say, "It's all right, I'm just moving a camera around to look at Mars," because, you know, we live in the future and that's something we do now.

The first step is to get an image. There's multiple ways to get gigapixel images, the easiest one is to just go to one of NASA's many sites and just download one. They're huge, and easy to find. You don't go to Mars and act all shy about it.

What I'm going to do is take the panorama, which is fairly large to be viewed on its own. If it's not big enough for you, don't worry, the available pictures from NASA are only likely to get larger as currently the panoramas being sent back are made of thumbnails instead of the high-res images.

For those who are new to gigapixel images, a gigapixel image is usually either an extremely large image, or a series of many images that image processing software looks through, cuts up, and then branches out into different folders like a folder tree. As a viewer zooms in and out, it allows the computer to only need to load a small number of moderately sized images instead of working through one massive file of image data. We're going to go with just one of the panoramas for now, but you may try downloading all of the Curiosity's mast images to make a gigapixel image from full-res photos instead of one high-res photo. It sounds crazy, I know, but how much crazier is it than waving your hands around like a wizard to look at the surface of Mars?

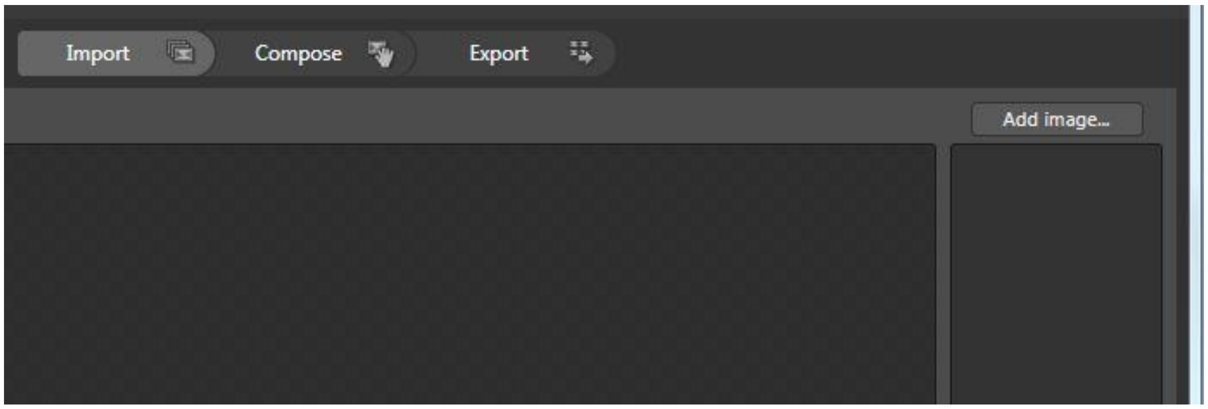
To create the gigapixel image now that we've got a large image, I'm going to use a program that's free for Windows called Deep Zoom Composer, developed by the lovely SeaDragon people at Microsoft, and free with Windows. Deep Zoom can output gigapixel images for various formats. What we want is an XML file and a series of branching folders.
Once you have your photo (and this can work with any large photo, or massive series of photos), you can open up Deep Zoom Composer, and start a new project.


New Project

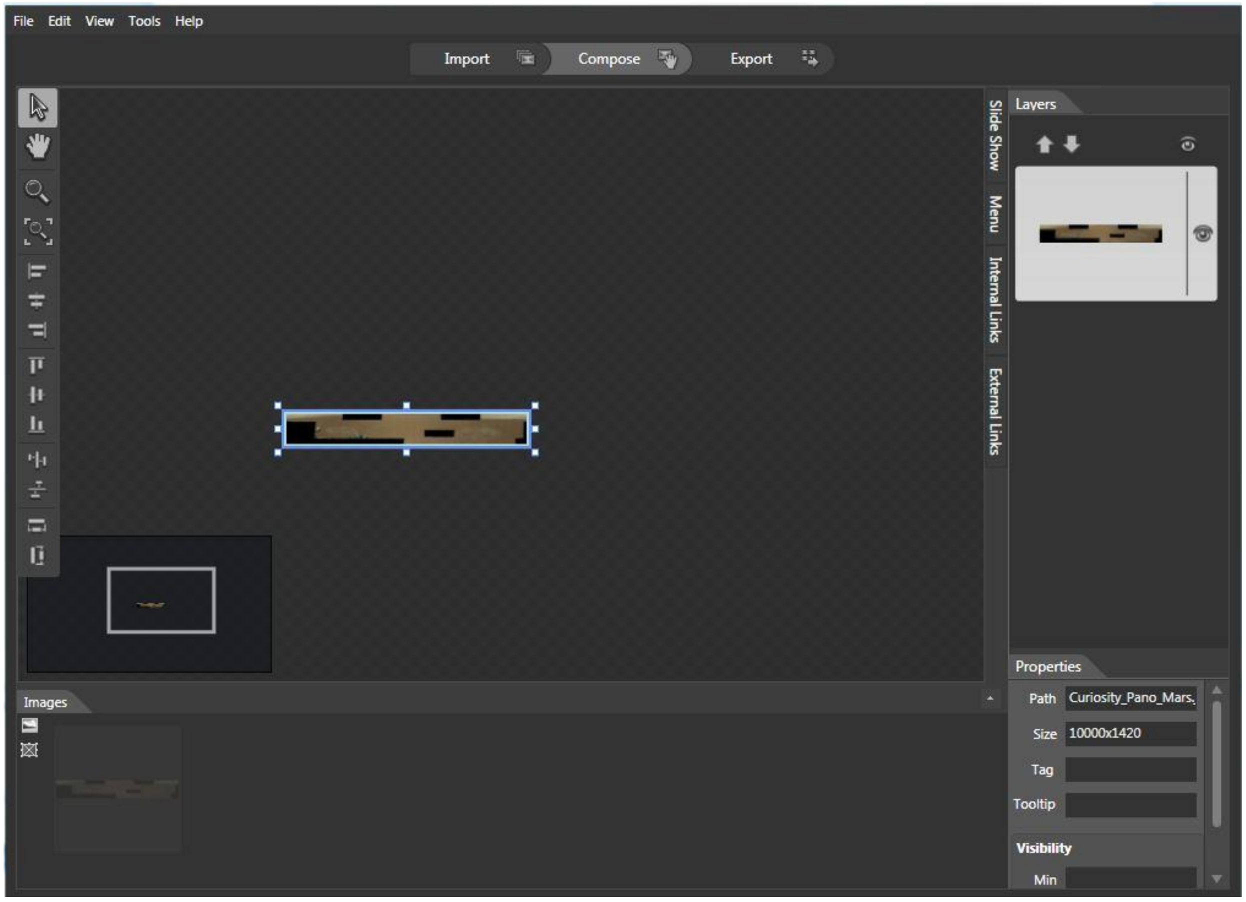
Name: GaleCraterPano

Location: C:\Users\Jesse\Documents\Expression\Deep Zoom Composer Projects

From here, it's fairly easy. The first step is to import the photo by going to the button above the list box at the right and clicking "Add image..." Then you just navigate to where you saved your large panorama, or go about clicking on your whole folder of images. Just to note: Deep Zoom Composer can be finicky with different file types. Some extremely large files it can handle easily, and some other files it just won't accepet. If you're having trouble loading a chosen image or series of images, try saving them in a different file format and then try again.



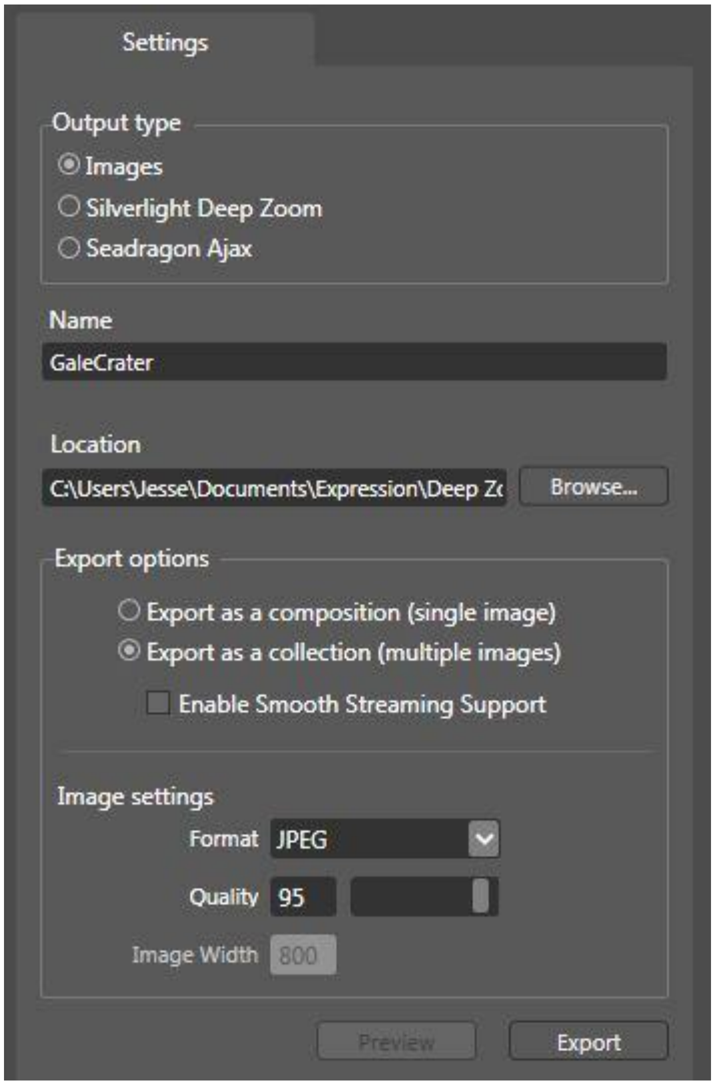
Now, the next step is either very short or very long. Since I only have one image right now, I'm going to go to "compose", and drag it out onto the screen. A little window in the bottom left shows me where my image is relative to the rest of my composition. But I don't care, because I only have one picture. As you can see though, I have a lot of room to expand if I ever decide to take on all of the Curiosity's high-res images. Since I only have one picture, I'm going to go on to "Export".



Path Curiosity_Pano_Mars.
Size 10000x1420
Tag
Tooltip

The Export page is fairly simple. Here's the important thing you want to know for using the gigapixel viewer: you want "Images" selected, as this is what's going to give you your XML and branching file structure that the gigapixel viewer likes to work with.

Once you click "Export", you will be asked if you'd like to "Learn More" or view your images folder. Lets look at the images. The first thing you get is this weird folder tucked away on your hard drive with all these files which you may or may not know about. For the gigapixel viewer, we're only concerned with the "dzc_output_images" folder that's highlighted in this picture. Doubleclick on that.


Settings

Output type

[✓ Images] Silverlight Deep Zoom Seadragon Ajax

Name

GaleCrater

Location

C:\Users\Jesse\Documents\Expression\Deep Zc

Export options

Export as a composition (single image) [✓ Export as a collection (multiple images)]

☐ Enable Smooth Streaming Support

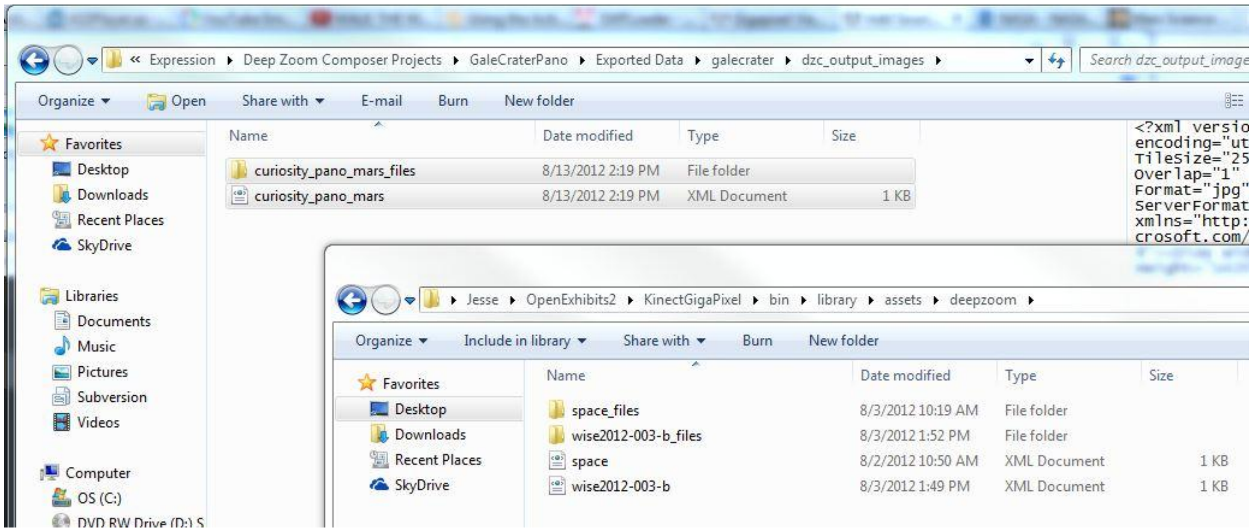
Image settings

Format JPEG

Quality 95

Image Width 800

You should now be in a folder with a file, and another folder. The file should be an XML Document, and the folder should have a matching name to it with all your image files. If you bother looking in, you'll just see a bunch of folders labeled with numbers, those are the "levels" of your gigapixel image. You don't need to mess with those.

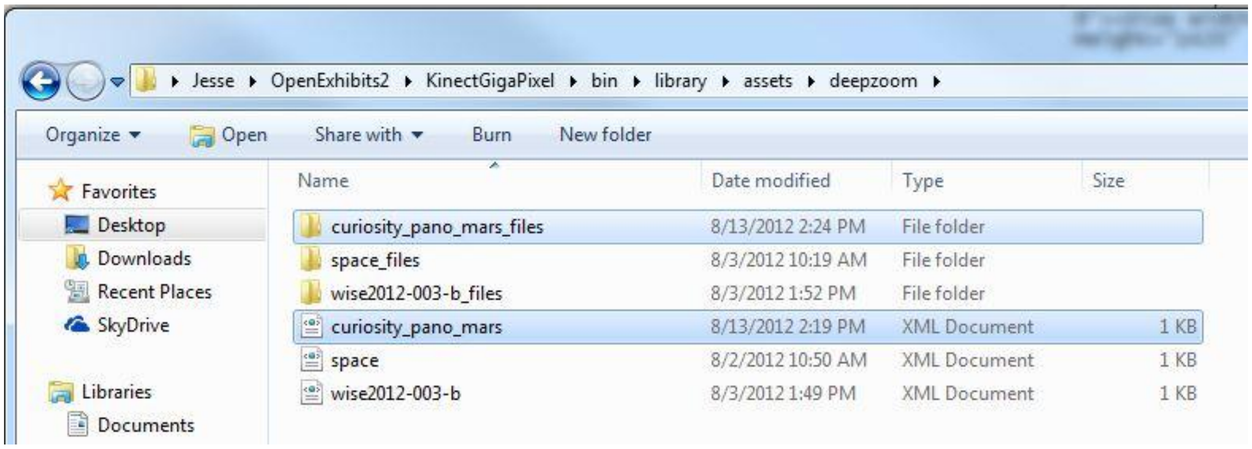


Name	Date modified	Type	Size
curiosity_pano_mars_files	8/13/2012 2:19 PM	File folder	
curiosity_pano_mars	8/13/2012 2:19 PM	XML Document	1 KB

```
<?xml versio
encoding="ut
TileSize="25
Overlap="1"
Format="jpg"
ServerFormat
xmlns="http:
crosoft.com/
```

Name	Date modified	Type	Size
space_files	8/3/2012 10:19 AM	File folder	
wise2012-003-b_files	8/3/2012 1:52 PM	File folder	
space	8/2/2012 10:50 AM	XML Document	1 KB
wise2012-003-b	8/3/2012 1:49 PM	XML Document	1 KB

Instead, what you want to do is take your XML file, and its matching folder full of image folders, and paste them into the "bin/library/assets/deepzoom" folder of your KinectGigaPixel folder, wherever you saved that. As you'll notice, there are two already present with your download. Once you paste the XML and the folder in, you'll have a third. The XML file's name is all you need to reference in your CML now to load your new gigapixel image.

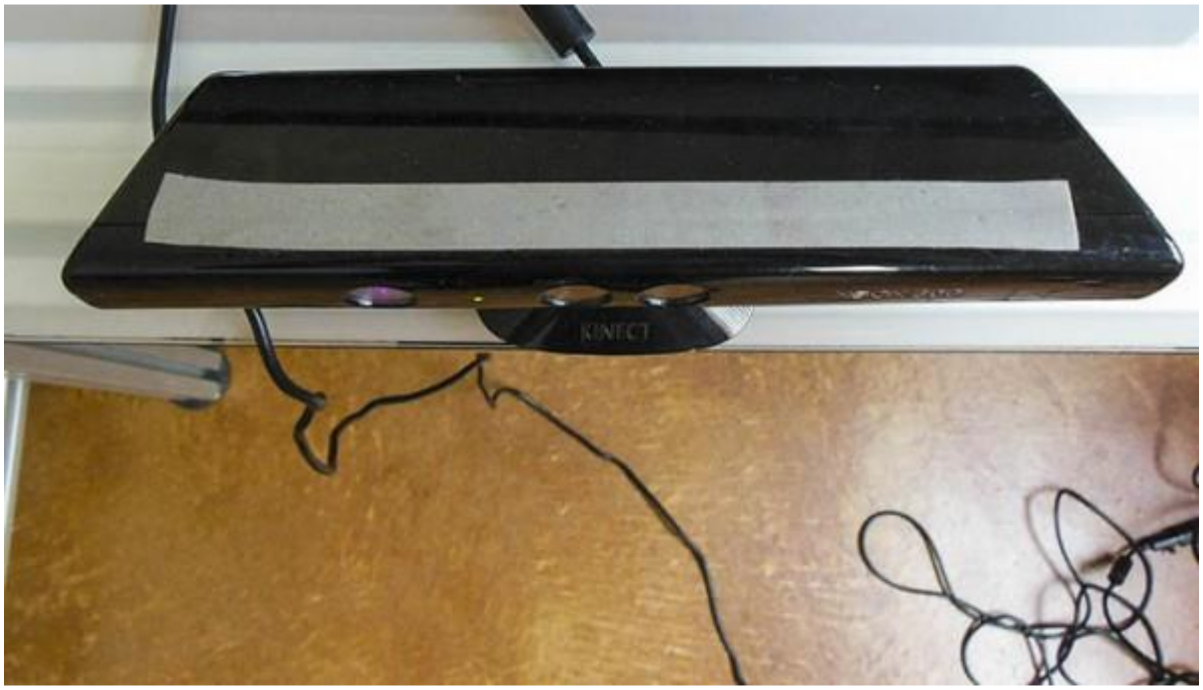


Name	Date modified	Type	Size
curiosity_pano_mars_files	8/13/2012 2:24 PM	File folder	
space_files	8/3/2012 10:19 AM	File folder	
wise2012-003-b_files	8/3/2012 1:52 PM	File folder	
curiosity_pano_mars	8/13/2012 2:19 PM	XML Document	1 KB
space	8/2/2012 10:50 AM	XML Document	1 KB
wise2012-003-b	8/3/2012 1:49 PM	XML Document	1 KB

Part 3: Physically Using the Kinect

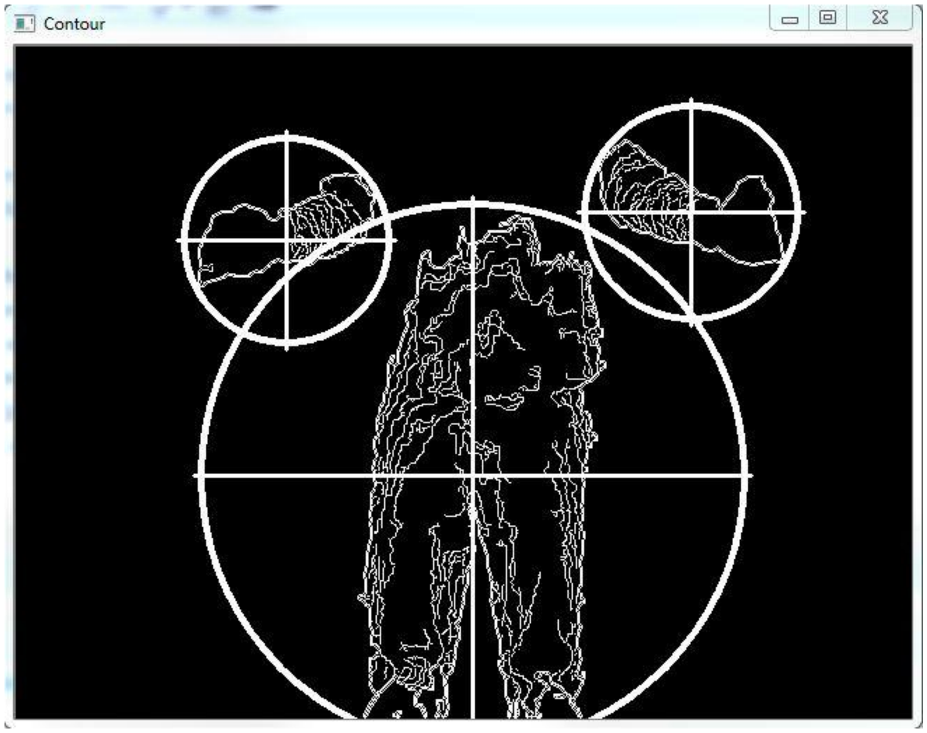
The Kinect is unique and extremely hygienic in the fact that it allows a user to manipulate digital items without touching anything, though arguably the part that most people enjoy more involves the part that makes them feel like they have telekinesis. This tutorial has focused on setting up the Kinect enabled Gigapixel viewer. Part 1 focused primarily on software setup and some physical space setup, Part 2 focused on using Microsoft Deep Zoom Composer to turn a high-res image into your own gigapixel image, and this section will focus on the physical setup and operation of the Kinect so you know what to expect.

There's one very important thing I want to stress about setting up the Kinect: put it on the edge of your desk/table/shelf/whatever you have it resting on.



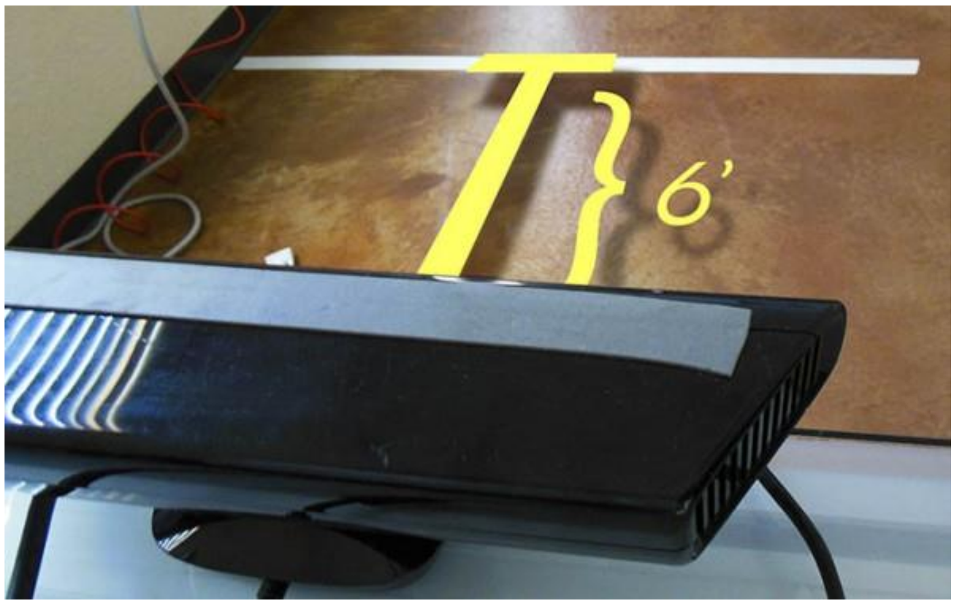
The reason for this is you want an accurate depth picture, and you don't want the infrared beams and sensor cut off by a huge chunk of desk in the bottom of its field of view.

The Kinect works by broadcasting what's basically an infrared checker pattern. The placement of its sensors helps to interpret the depth, as well as the way the checker pattern warps around curved surfaces, such as a person's body. The TUIO Kinect Complete package hooks you up with a blob tracker, which looks for significant blobs and translates them into touch points in the OpenExhibits Framework. This means you can use these touch points to track drags, touches, pinches and zooms.



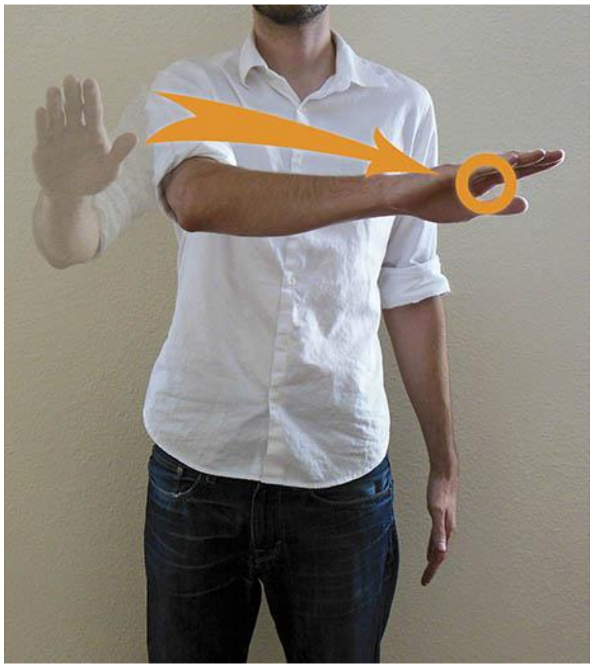
The TUIO Kinect Complete Contour software comes already set up, you shouldn't have to worry about adjusting or calibrating your depth, though you will probably need to experience for yourself when it starts to really track your body parts. As the TUIO Kinect Complete uses a blob tracker and not a skeletal tracker, any part of your body can become a touch point at the right distances.

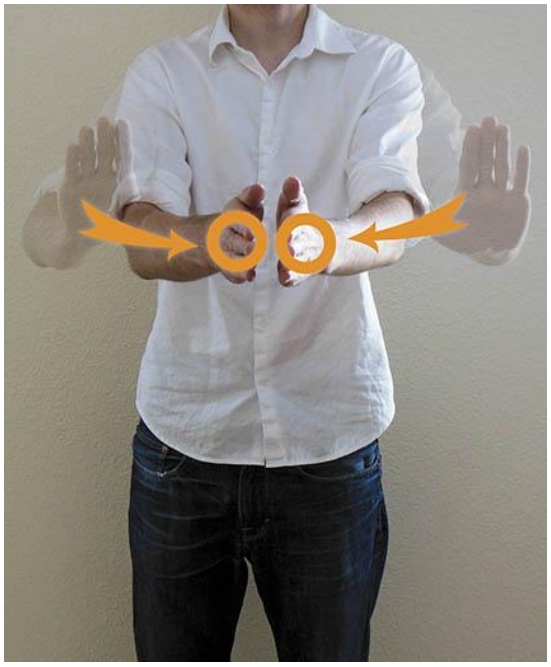
For our setup here at Ideum, the ideal threshold is about six feet away. This means once a person reaches their hand within six feet, the Kinect starts registering it as a touch point. This is something to keep in mind when preparing your Kinect, it's nice to give a marker of some sort to note where this threshold of interaction is.

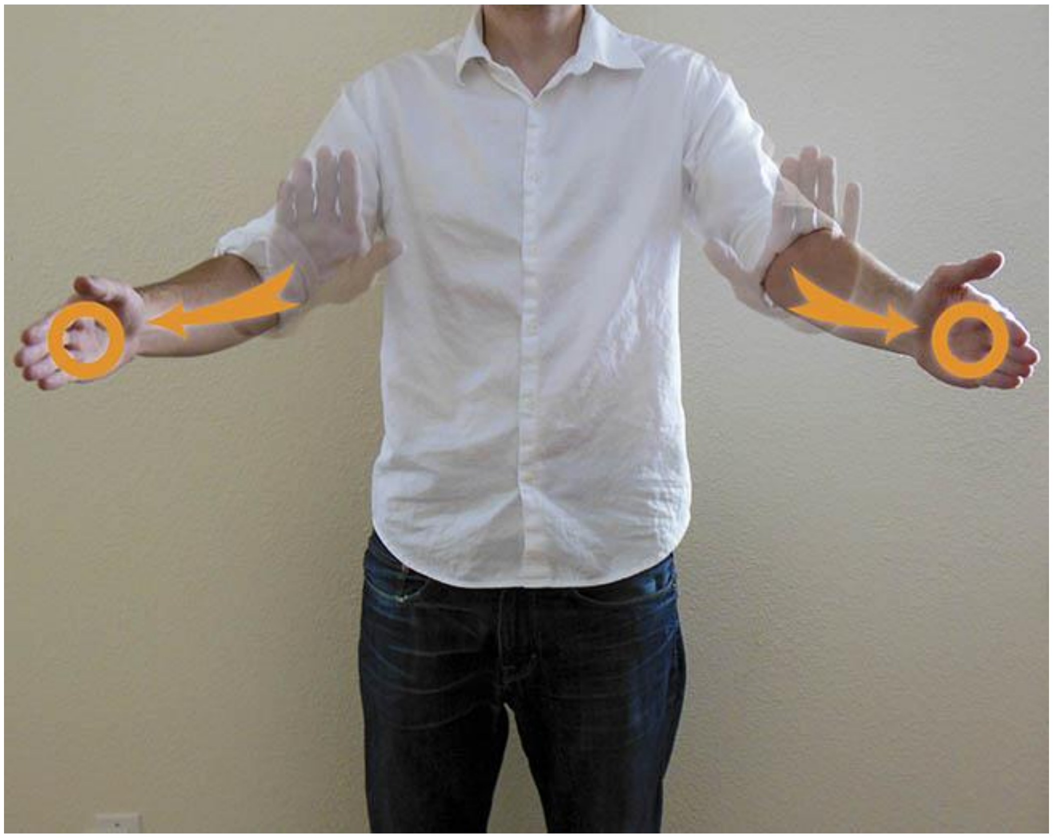


Joshua Hicks, September 6, 2012

The next thing to remember is a user's hands, or body parts are being tracked as touch points. This means you primarily want to imagine one-point-drag, two points to pinch and zoom.


Pan: move the image around with one hand.


Pinch: Zoom out on the image.


Zoom: made by spreading two points apart.

Joshua Hicks, September 6, 2012

You will also want to pay attention to your threshold of interaction. It's better to push your hand(s) in, manipulate the image, and pull them back out of range, reset your hands, and repeat. This gives you swimming and paddling motions to zoom in, or pan across. If you find the viewer becoming unresponsive, try dropping your hands down to clear your touch points, then reach back up.

And that's the end of this three-part tutorial session on how to manipulate the Gigapixel Viewer using what appears to everyone else to be Jedi mind tricks. Hopefully you've been able to get your Kinect set up, installed, and working with the Kinect Gigapixel Viewer, as well as make your own branched, gigapixel images. Or at least you know how to go about making your own gigapixel images even if you don't have any you need to make.

Joshua Hicks, September 6, 2012