Kinect > OpenNI > NITE > OSCeleton > Max/MSP/Jitter on Windows

This is a short tutorial explaining how to get your Xbox Kinect working with Cycling '74 Max on a Windows machine. Along with the installation instructions, I will give some simplified explanations of how the various hardware and software layers interact. There are a fair amount of tutorials out there for this process, but no single one is comprehensive enough in my opinion. The installation process was tricky and very frustrating at times, so I've tried to write a description that makes it as painless as possible for others. In particular, I have been meticulous to document the exact versions of the software used as to make things as transparent as possible. I'm pretty new to using Max/Kinect/OpenNI, so I definitely encourage any constructive criticism you may have. Feel free to email me at njt3c@virginia.edu.

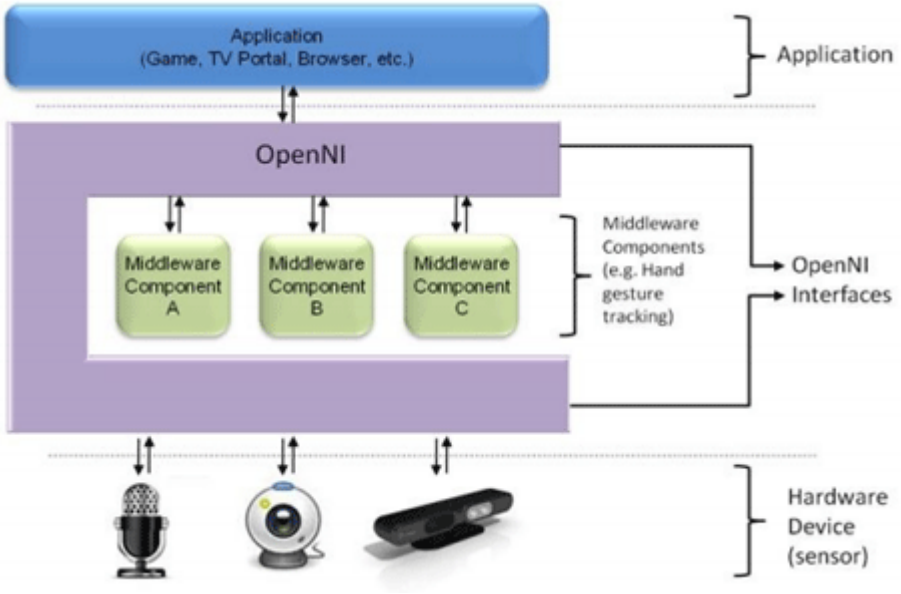
Software and hardware:

- Xbox Kinect
    - Note that this is for the Xbox Kinect model 1414, not the Windows version.
- Windows 7 64-bit
    - I have only tested with this version of windows. I cannot guarantee compatibility with 32-bit or other versions of Windows.
- Visual Studios
    - I tested with the full version. I have read elsewhere that it should work fine with the (free) Express C version, but haven't verified that myself.
- Max/MSP/Jitter 6.1 32-bit
    - As of July 2013, OSC messaging, and thus, OSCeleton, does NOT work with the 64-bit version of Max. If you currently own a 64-bit version of Max, that same license entitles you to a download of the 32-bit version.
    - While I haven't tested, I do believe that Max 5.0+ versions should work fine.
- OpenNI 1.3.4
    - While more recent versions of OpenNI exist (OpenNI2.0) as of July 2013, this version was installed because it was contained in a bundled download. I will elaborate more on this below. OpenNI claims to have full backwards compatibility, so a newer version should theoretically work fine.
- Primesense NITE 1.4.2
    - There is also a newer version of NITE, but this was used as it was also contained in the bundled download.
- Primesense Kinect Sensor Driver
- OSCeleton
- CNMAT (Berkeley) OSC package 1.0-39 Windows version (released June 7, 2013)

Basic Component Description:

So, what is OpenNI, NITE, and this other stuff? Most simply, they are a collection of software necessary to get a peripheral device, such as the Kinect, to talk to an application on your computer, such as Max. To begin, OpenNI (Open Natural Interaction) is an open source framework that was partly developed by PrimeSense (one of the creators of Kinect). OpenNI provides standard application programming interfaces (API) which allow developers to write applications based on natural interactions. One of the main advantages to OpenNI, as opposed to the official MS KinectSDK, is that it can interface with a wide variety of peripheral devices and not just the Kinect.

Provided by OpenNI, the image below gives a depiction of how the various layers of software and hardware interact. In our case, the device is a Kinect and the middleware is NITE and OSCeleton. The Primesense Kinect driver is the driver that allows the raw depth and color data from Kinect to be sent to OpenNI. OpenNI then passes on this information to the NITE middleware which is software specifically designed to take raw Kinect data and turn it into skeleton tracking. The skeleton information is then passed on to OSCeleton, another middleware component, who translates the skeletal joint information from NITE into OSC messages. Finally, the CNMAT Max external, OSCroute, is used to bring the OSCeleton messages into Max/MSP/Jitter. I highly recommend reading the first couple pages of OpenNI's User Guide PDF if you are unfamiliar with OpenNI. The PDF can be found here: https://github.com/OpenNI/OpenNI/blob/master/Documentation/OpenNI_UserGuide.pdf.



Software architecture

Installation:

Assuming you already have the Windows, Visual Studios, Kinect, and Max versions I described above installed, I will walk you through getting the rest working.

1. Download and unzip the CNMAT externals package from here: http://cnmat.berkeley.edu/downloads
   1.1. Place that folder in C:/program files (x86)/Cycling '74/Max6.1/max-externals. I don't think it has to go here, but this works for me.

2. Download OpenNI, NITE, and the Kinect Driver from here: http://developkinect.com/resource/package-installer/zigfu-package-installer. This is the bundled download I was referring to above. This allows you to easily install each software package "without any command line wizardry." The manual Kinect driver installation process requires you to execute some python scripts, so this method is much easier if you are not familiar with that process. If for some reason you want to see this process, see https://github.com/avin2/SensorKinect. Once this bundle has finished installing, you should see two new folders in your Program Files (x86): OpenNI and Primesense. The Primesense folder contains both the NITE and Kinect sensor driver software.
    2.1. Test to see if these have been installed properly by running a demo. To do this, you will need to navigate to Program Files (x86)>Primesense>NITE>Samples>Bin>Release>Sample-StickFigure. Run the StickFigure application (be patient – it can take ~10 seconds to load), and assume the position. No, not that position. Assume the calibration position which looks like the American football symbol for touchdown. Be sure there is nothing blocking the Kinect's view and stand at least 7-8 feet back. This allows NITE to pick up on your skeleton and you should see a stick figure of yourself on the screen. There are also other demos for you to play around within the NITE and OpenNI folders.
3. Download then extract OSCeleton from here: https://github.com/Sensebloom/OSCeleton
    3.1. Open the folder and browse for the OSCeleton.sln file. Open this file with Visual Studios. As mentioned above, this may work fine with the free version Visual Express C++. Generate an executable of the contents by clicking Build>Build Solution. This executable will be located within a new folder called Debug.
    3.2. If your computer did not produce any errors during the build, then lucky you. Skip to step 3.4.
    3.3. ***If your computer gave an error about not being able to transfer a glut32.dll file, then you will have to do it manually. This seems to be a common problem. To check, navigate to OSCeleton-master>Debug. If you don't see the file, then go to OSCeleton-master>src>lib and copy the glut32.dll file. Paste this into OSCeleton-master>Debug. This may not be the only way to fix this problem, but it is what worked for me.
    3.4. Now you should be able to run the OSCeleton executable found in OSCeleton-master>Debug. To make sure everything worked smoothly, verify by running with the –h option. This will require you to navigate through the Windows command prompt. To do this, open the command prompt. Type "cd C:\Program Files (x86)\OSCeleton-master\Debug" or wherever your file is located and press enter. Then run OSCeleton by typing "OSCeleton.exe –h" and pressing enter. You should see a menu list pop up. If so, you're done!
4. You're finished! Go make something cool.