This version of IRSAW is highly modified which a lot of changes.

The changes are listed below

1. Using Intel RealSense Cross Platform API (librealsense) to capture camera data instead of the DCM and the SDK, you don’t need to install anything to use the camera now.
2. Ported to Visual Studio 2013. librealsense is not compatible with VS 2012
3. Ported to Cinder 0.9.0 from 0.8.6
4. You don’t have to build a Vibration Actuator with a button. All the Vibration Actuators are made the same, but the code uploaded to every single Vibration actuator using the Particle Web interface is different (Just one variable has a different value for every single photon for identification purposes) The toggling of the motors can be done using the Laptop app UI or the TouchOSC app running on Android.
5. You don’t need to save the IP addresses for the Photons in the router. You only need to save the IP Address of the Laptop running the application in the Router.
6. Using an android phone (Also compatible with an iPhone, but not tested)to control the application (For turning specific motors on/off, changing the range of detection) We use an android app called Touch OSC <http://hexler.net/software/touchosc> <https://play.google.com/store/apps/details?id=net.hexler.touchosc_a>
7. Gradual and Pulsing Vibration - using the app you can change the type of vibration feedback you get on your body. It can be gradual or pulsing based on the distance of the object.

To get started, download the project from the git repository (which you already have if you are reading this), Download OpenCV 2.4.9 and Cinder 0.9.0. Set environment Variable for Cinder and OpenCV

Set the following Environment Variables

Tip: your Cinder root folder is where you Include folder is, for example in my case,

C:\Dev\cinder\_0.9.0\_vc2013, so that is the value of my CINDER\_ROOT variable

For Open CV it is C:\opencv\build, so that is the value of my OPENCV\_ROOT variable

CINDER\_ROOT = <Location of you cinder 0.9.0 directory>

OPENCV\_ROOT = <Location of your OpenCV directory>

librealsense is added as a submodule, so don’t forget to init and update

In the git bash, enter the following commands

git submodule init

git submodule update

These two commands will clone the librealsense repository in the thirdparty folder.

Open the project IRSAW.sln file in visual studio 2013. In the solution explorer, expand the thirdparty folder. Right click on the realsense-s project and open the project properties of the realsense-s project.

In the properties make the following changes

In debug configuration, go to C/C++ -> Code Generation, and change the Runtime Library to **Multi-threaded Debug(/MTd)**

In Release configuration, go to C/C++ -> Code Generation, and change the Runtime Library to **Multi-threaded (/MT)**

Apply and save these changes and then compile and run the application, things should run fine and the application window opens. Pressing ‘v’ on the keyboard toggles the visualizer.

**Setting up the Router**

The Laptop and the photons connect to Router.

Once you have setup up an SSID and Password to the router, connect the router to the internet from the router settings.

The Router is dual band, but it doesn’t matter if you connect the laptop to the 2.4GHz or the 5GHz band. In the router settings under WiFi Clients change the IP address of the Laptop to “192.168.0.102” Make sure you reserve this IP address for the Laptop and save it so the laptop always gets assigned with the same IP address. If you change the IP address of the laptop for some reason, you will have to change the Server IP address in the photon code to match the IP address of the laptop and also the IP address for the Touch OSC app on your phone.

No need to assign fixed IP addresses to the phone or the Photons on the router.

Try running ipconfig on the laptop on the command line and confirm the ip address of the Laptop. (Sometimes you have to manually reboot the router for the changes to take effect)

**Touch OSC app configuration**

Once you downloaded the touch osc app from the app store, go in Settings->OSC

Change the Host address to the IP address of the laptop “192.168.0.102”

Change the Port (Outgoing) to 3000

Change the Port (Incoming) to 3001

Now in the project repository, go in the PhoneOSC folder and open TouchOSCEditor.exe

Click on Open and navigate to the layouts folder in the PhoneOSC folder and open the SAW\_ControlsPulse.touchosc file in the editor

Make sure that your laptop and Android phone are on the same Wifi network (Preferably the router’s wifi) and then click “Sync” in the editor and it will open a window giving instructions.

Now on the phone, in the TouchOSC app’s settings, go to Layout->Add from Editor

Add the IP address of the Laptop where the Editor is running in the Host field and click download.

This will download the IRSAW TouchOSC UI

Make sure you select SAW\_ControlSPulse UI from the Layouts on the app and then tap on Done

This will show show you the TouchOSC app running the IRSAW UI to control the main application.

Now given that you phone and laptop are connected to the same WiFi and the application is running on the laptop, you will be able to control the Main application from your phone. You can toggle motors, individually change the detection range for the Top, Middle, Bottom motors from 1 meter to 2 meters. It is at 1.5 meters by default.

**Setting up the photons**

The repository has a folder called Particle Photon Code in the file called Photon Code.txt

Important changes in the code

1. Photons can now have a gradual or a pulsing change of vibration controlled from the touchosc app on the phone
2. Every photon identifies itself to the server application running on the laptop and that Identification is stored in the char variable called “myName”. The Value in this variable changes for every photon (the values range from 1 through 8 for 8 vibration actuators)

//char myName = 1; // Top Left

//char myName = 2; // Top Center

//char myName = 3; // Top Right

//char myName = 4; // Middle Left

//char myName = 5; // Middle Center

//char myName = 6; // Middle Right

//char myName = 7; // Bottom Left

//char myName = 8; // Bottom Right

So change the variable value depending on which Vibration Actuator you want to upload the code to.

1. There is no vibration actuator with a physical button to toggle the motors On and Off.

Go to build.particle.io log in, then Create New App and copy and paste the code In the app. The follow the process of flashing the code to photons stated on the particle website

<https://docs.particle.io/guide/getting-started/intro/photon/>

When you flash this code to the photons, the photons will disconnect from the Particle Cloud, so to update the photons again, you will have to put the photons to safe mode and follow the process of uploading the code to the photon again. Instructions on putting the photons to safe mode can be found at docs.particle.io <https://docs.particle.io/guide/getting-started/modes/photon/#safe-mode> Putting the photon in safe mode will also let you update the code on the photons without factory resetting it.

Once your have flashed the code on all photons, you can now check if the photons connect to the Server application running on the laptop.

**Known Issues**

The vibration actuators freeze on closing the application and then go in deep sleep mode. Cause is yet to be found. But solution is just press the reset button on the photon to bring the photon back to life.

If the application is not running already, the photons will go to deep sleep and you will be required to wake them up by resetting them by pressing the reset button.