OpenARK CMAKE build instructions for Windows:

Prerequisites:

We recommend using Visual Studio 2015 and 64-bit Windows for OpenARK. When given the option please select Visual Studio 2015 (also known as vc14) and x64 for all prerequisite software.

1. Install Visual Studio 2015. Select custom installation and make sure to install the C++ build tools under Programming Languages.

2. Install cmake: https://cmake.org/download/
   Make sure to select "add cmake to system PATH" either for current user or all users.

3. Install PCL1.8, easiest way is to use the all-in-one installer made by a kind PCL user:
   http://unanancyowen.com/en/pcl18/#Download
   Make sure to select "add PCL to system PATH" either for current user or all users.
   OpenNI2 will be installed as part of this installer. You will need to add the OpenNI2 dlls to the system path manually. The default location these are installed to is C:\Program Files\OpenNI2\Tools. See Step 4 in the opencv install instructions below for more details on modifying system variables.

General CMAKE Build Instructions: (Use an Administrator command prompt)
Most of the following files will follow the general CMAKE build steps shown below:

1. make a build directory to store build files:
   ```
   mkdir build
   ```

2. Generate a Visual Studio solution:
   ```
   cd build
   cmake -G"Visual Studio 14 2015 Win64" ..
   ```

3. Build and install:
   ```
   cmake --build . --config Release --target install
   ```

Install opencv 3.4 and opencv_contrib from source:

1. Download opencv_contrib 3.4.8 source from https://github.com/opencv/opencv_contrib/releases/tag/3.4.8. Extract the zip folder.
2. Download OpenCV 3.4.8 source from https://github.com/opencv/opencv/releases/tag/3.4.8. Extract the zip folder and cd to the extracted directory
3. Follow CMAKE steps above, EXCEPT replace `cmake -G"Visual Studio 14 2015 Win64" ..` with
   ```
   cmake -G"Visual Studio 14 2015 Win64" -
   DOPENCV_EXTRA_MODULES_PATH=<opencv_contrib>/modules -DWITH_MSMF=OFF ..
   ```
   Where: `<opencv_contrib>` is the location you extracted the opencv_contrib zip file to.

4. Set the OpenCV enviroment variable and add it to the systems path. You can edit the environment variables using an interface by going to "Control Panel->System and Security->System->Advanced system settings->Environment Variables".

   We will need to modify two system variables in order to use OpenCV. First we want to add the system variable OpenCV_DIR and set it to "<extraction_directory>\opencv\Build\install" where <extraction_directory> is the directory you extracted OpenCV to.

   You will also need to add the bin directory to the PATH variable. Select the variable "Path" and add a line with the value "%OpenCV_DIR%\x64\vc14\bin". This allows your computer to find the proper library files when running programs using OpenCV.

   Hit "OK" to close the Environment Variables dialog box. The variables will not be set until you do this. You will also have to reopen any open command prompt windows for the variables to be updated for that window.

Install Eigen 3:

1. Download source from http://eigen.tuxfamily.org/. Extract zip folder and cd to the extracted directory.
2. Follow CMAKE steps above.

Install Glog:

1. Download source from https://github.com/google/glog. Extract zip folder and cd to the extracted directory.
2. Follow CMAKE steps above.

Install SuiteSparse:

1. Download source from https://github.com/joemenke/suitesparse-metis-for-windows. Extract zip folder and cd to the extracted directory. (Note this is our forked version that builds with VS2015).
2. Follow CMAKE steps above.
3. You will need to add the "SuiteSparse_DIR" environment variable and set it to "<build dir from step 2>\install"
4. You will also need to add the lapack install location to your Path. Open the Path environment variable and add a new row and set it to "<build dir from step 2>\install\lib64\lapack_blas_windows"

Install Ceres Solver:

1. Download source from https://github.com/joemenke/ceres-solver. Extract zip folder and cd to the extracted directory. (Note: this is simply a fork from the original git repository to make install on windows easier)
2. Follow CMAKE steps above.

Install Boost:

1. Download source from https://www.boost.org/users/history/version_1_72_0.html. Extract zip folder and cd to the extracted directory.
2. Set up:
   `bootstrap`
3. Install:
   `.\b2 --toolset=msvc-14.0 --build-type=complete --prefix=C:\Boost architecture=x86 address-model=64 install`
   Note: You may get an error about the msvc version not matching. If so you need to modify project-config.jam with "using msvc : 14.0 ;"
4. Boost doesn't officially support Visual Studio 2015 (though it works fine), you may need to modify $(boost install location)/include/boost-1_XX/boost/config/auto_link.hpp and change:
   #eleif defined(BOOST_MSVC)
   #define BOOST_LIB_TOOLSET "vc120"
   to
   #eleif defined(BOOST_MSVC)
   #define BOOST_LIB_TOOLSET "vc140"

Install OpenGV:

1. Download source from https://github.com/joemenke/opengv. Extract zip folder and cd to the extracted directory. (Note: this is simply a fork from the original git repository to make install on windows easier)
2. Follow CMAKE steps above.
3. You will need to add the environment variable "OpenGV_DIR" and set it to the install location of OpenGV (probably C:\Program Files\opengv)

Install Brisk:

1. Download source from https://github.com/joemenke/brisk. Extract zip folder and cd to the extracted directory. (Note: this is simply the brisk originally included with okvis, modified to make install on windows easier).
2. Follow CMAKE steps above.

Install DBoW2:

1. Download source from https://github.com/joemenke/DBoW2_Mod. Extract zip folder and cd to the extracted directory. (Note: this is a fork from the original git repository to make install on windows easier and add support for Brisk descriptors).
2. Follow CMAKE steps above.
3. You will need to add the environment variable "DBoW2_ROOT_DIR" and set it to the install location of DBoW2 (probably C:\Program Files\DBoW2)

Install DLoopDetector:

1. Download source from https://github.com/joemenke/DLoopDetector. Extract zip folder and cd to the extracted directory. (Note: this is a fork from the original git repository with some minor modifications to support OpenARK integration).
2. Follow CMAKE steps above.
3. You will need to add the environment variable "DLoopDetector_INCLUDE_DIRS" and set it to the install location include directory of DLoopDetector (probably C:\Program Files\DLoopDetector\include\DLoopDetector)

Install Okvis+:

1. Download source from https://github.com/joemenke/okvis. Extract zip folder and cd to the extracted directory.
2. Follow CMAKE steps above.

Install Open3D:

1. Download source from https://github.com/adamchang2000/Open3D. Extract the zip folder and cd to the extracted directory. (Note: this is a fork from the original git repository with additional functionality to support OpenARK real-time reconstruction).
2. Follow CMAKE steps above, ADD "–parallel <number of cores>" to step 2 of the CMAKE build instructions. This will improve Open3D's performance.

Setting up the Intel Realsense D435i:

1. Follow the instructions here: https://github.com/IntelRealSense/librealsense/blob/master/doc/installation_windows.md to install librealsense. Bet sure to follow the "Enabling metadata on Windows" section as the timestamps coming from the device will be incorrect without it.
2. Update the device firmware using the windows firmware update tool: https://downloadcenter.intel.com/download/27514/Windows-Device-Firmware-Update-Tool-for-Intel-RealSense-D400-Product-Family
3. Using the "realsense-viewer" tool (located in build/tools/realsense-viewer), ensure that all sensors are able to stream data.

Building OpenARK:

1. Open a VS2015 x64 Native Tools Command Prompt

2. cd to the directory to which you have downloaded the OpenARK source code

3. make a build directory to store build files:
   ```
   mkdir build
   ```

4. Generate a Visual Studio solution:
   ```
   cd build
   ```

```
cmake -G"Visual Studio 14 2015 Win64" ..
```

5. You can now either open the Visual Studio Solution generated in the build directory labeled “OpenARK.sln” or continue to build using the command prompt.

Building via Visual Studio:

Right click on the OpenARK project in the solution explorer and select “Set as Startup Project”. Build and run as usual.

Building via Command Prompt:

```
cmake --build . --config Release
```

To run the OpenARK hand demo:

```
cd Release
OpenARK_hand_demo.exe
```

To run the OpenARK SLAM demo:

```
cd Release
OpenARK_SLAM_demo.exe
```

To run the OpenARK 3D Reconstruction demo:

```
cd Release
3DRecon_Data_Recording.exe
```