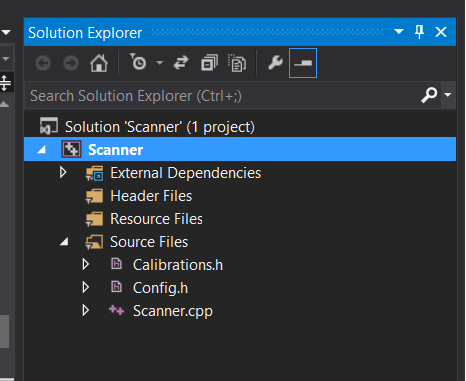
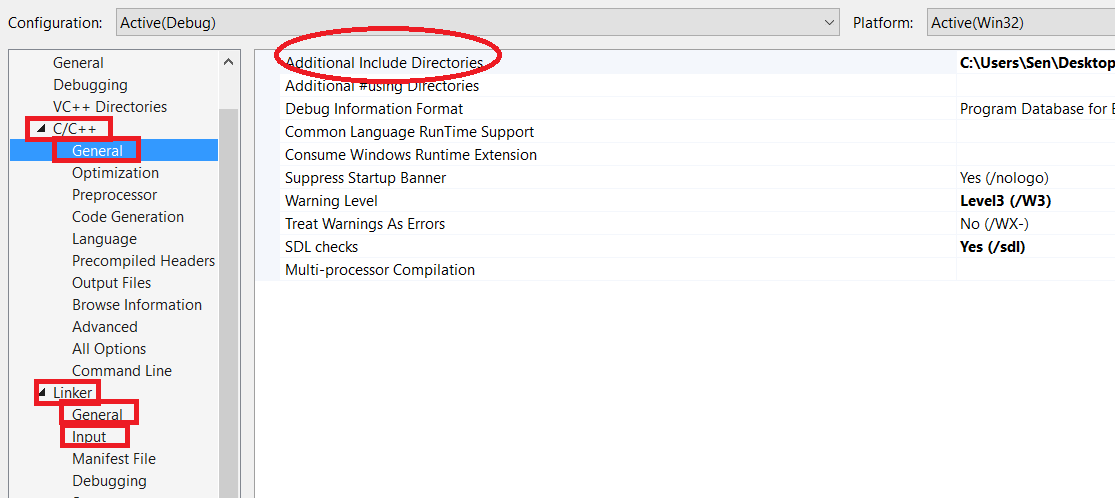
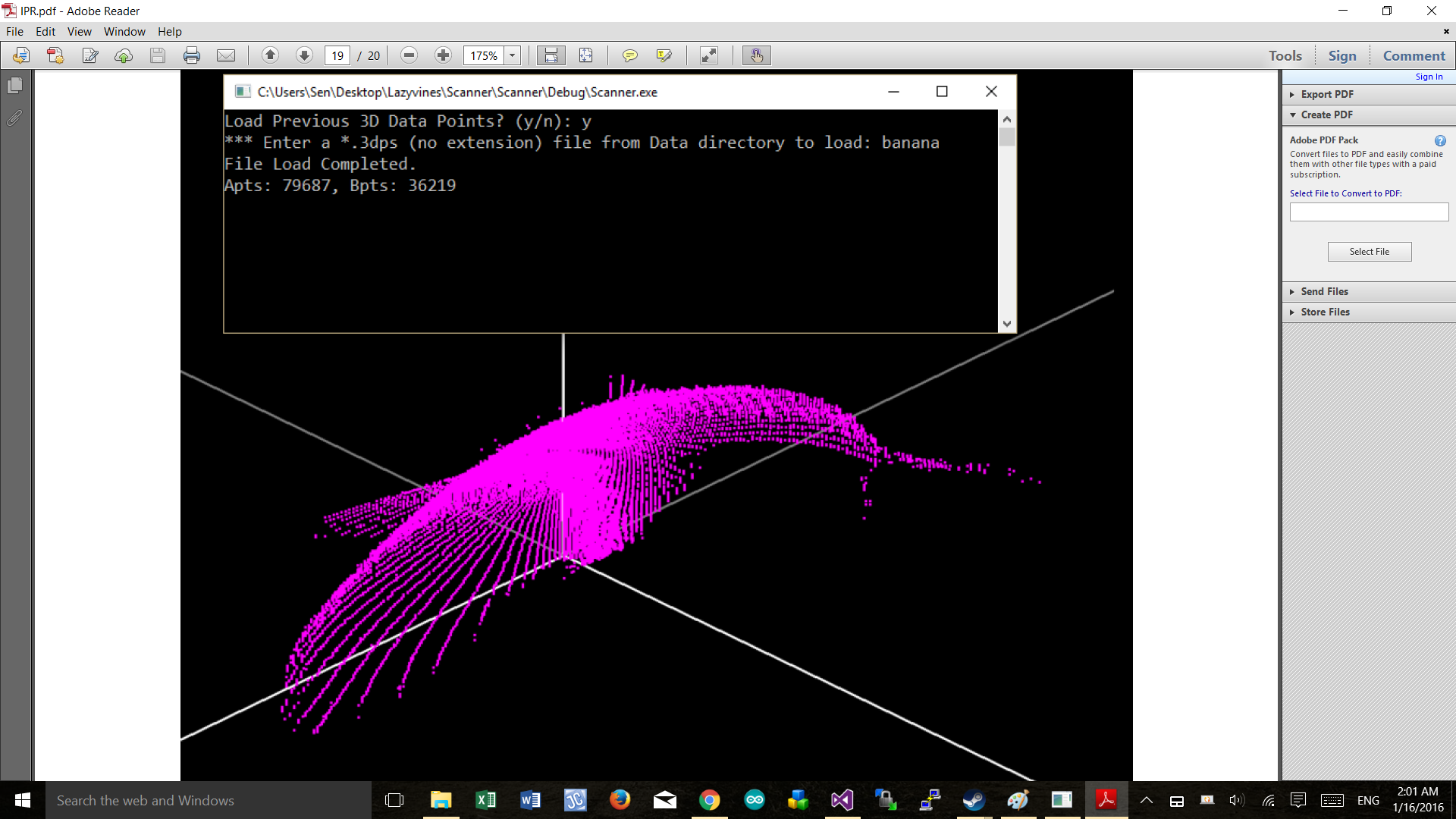
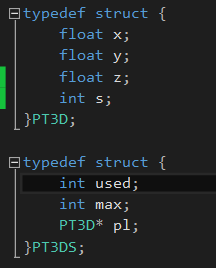
The accompanying Lazyvine folder contains the entire directory of the main executable application, last checked Jan. 16, 2016. The version is 4.1. This program is the mainframe for all scanner operations, and is compatible with only the latest version of scanner, the *Lazyvine*.

**INSTALLING** (If you can compile the code after step 3, skip this section).

1. Install MS Visual Studios.
2. Go to Lazyvines > Scanner > Scanner > Scanner and open the solution file Scanner.sln.
3. To the right side there are settings:
4. Just right click on that Scanner button, and you will see a drop down menu, click on “**Properties**”.
5. There is a list on the left. Go to C/C++ > Generals:
6. Configure the files in the Add’ Incl’ Dir’ so that the corresponding files can be relatively relocated. Do this for all the sections you see in the red boxes. Once you link these libraries, you should be able to compile and run the source code by pressing F5.

**RUNNING**

1. There will appear a prompt, which will ask if you would like to load a file:
2. Always respond “y”. If you select no, the application will automatically assume you want to scan a new object instead. If you do not have the hardware connected to your PC, the program will then exit in error. The files it uses are from the Data, with files that ends with \*.3dps. New \*.3dps files can be requested by telling Sen what object to scan.
3. If you load the file, these data will be read for you into a data structure. This data is global, but is designed as READ-ONLY. If you must manipulate the data, you must allocate your own memory and copy the data.   
     
   Notice P3D\_A and P3D\_B. These are separate, each corresponding to a camera. The PT3DS is a structure that has 3 fields:
   1. PT3DS.used := how many 3D points did this camera capture.
   2. PT3DS.max is the size of data allocated for the points, defined in config.h.
   3. PT3DS.pl := the pointer to the array of PT3D points, which contains *used* points.
4. Check out the structure definition in config.h of 3D pixels:
   1. Each PT3D file contains the x, y, z and s (step number) data. Note that x, y, z are normalized values from 0 to 1. And step number can be 0 to REV\_STEPS, which is 160 at this time. This value may be changed if we want to trade off some speed with precision, or vice versa.   
      

**EXTENDING THE PROGRAM**

1. Files:
   1. Calibration.h: Please do **not** make any modifications to this file. It is used for hardware calibration and will not be used if you are loading points.
   2. Config.h: You can toggle debug options, and add some new constants here if you wish. You can also modify/add/remove the values under *illustrator* to implement newer illustrators. Please do NOT touch the constants.
   3. Scanner.cpp:
      1. Please assume all global data is dynamic AND read-only.