# **Topic 1: Software needs to install for testing in UT:**

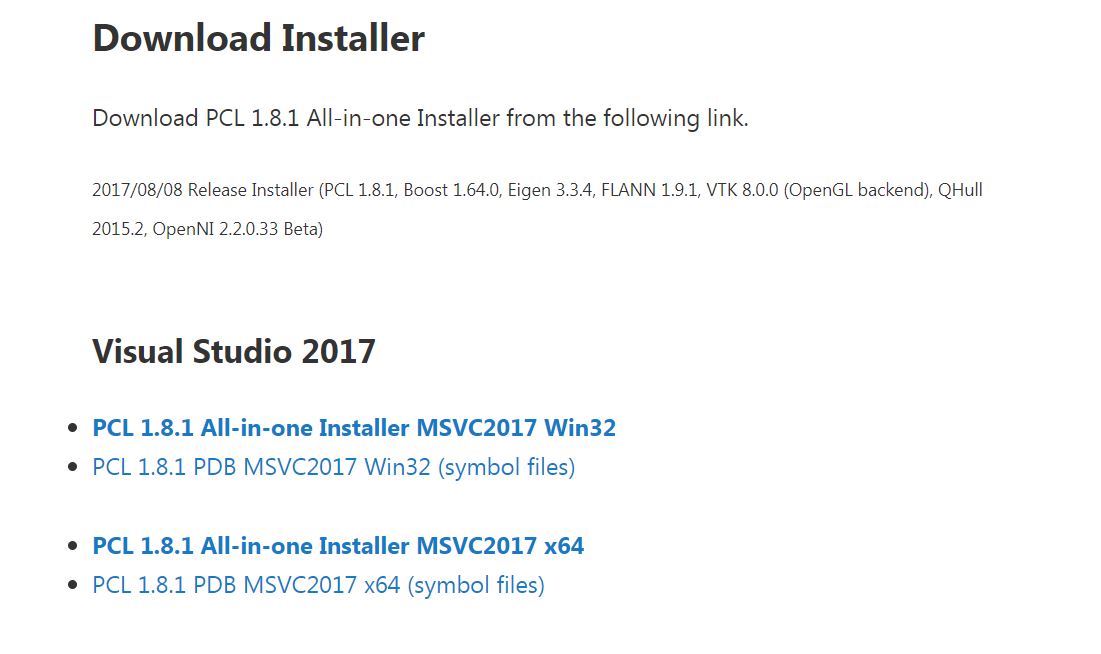
1. **Software development Kit(SDK)**

Download and Install Microsoft visual studio 2017 for windows 7 (x64) from the link- <https://www.visualstudio.com/downloads/>

1. **Libraries need to be installed:**
2. **PCL 1.8.1 (PointCloudLibrary)**

PCL has dependencies on third party libraries like Boost, Eigen, etc. These need to be installed as pre- requisite. PCL is packaged along with its dependencies and is available for easy installation on this link<http://unanancyowen.com/pcl181> for Visual Studio 2015 and 2017.

Refer the screenshot, Click on the PCL 1.8.1 All-in-one-Installer MSVC2017 x64. It contains PCL 1.8.1 and 3rd Party libraries also.



1. **Opencv 3.2.0**

Download and install opencv 3.2.0 from the below link: <https://opencv.org/releases.html>.

# Visual Studio Setting for this running the project:

**1. Include Header Directories:**

**Right Click on the project -> Properties -> C++ -> General -> Additional Include Directories**

Opencv Installed Directory: C:\Pactris\_Pro\opencv\build\include

Pcl Installed directory :

1. C:\Program Files\PCL 1.8.1\include\pcl-1.8

2. C:\Program Files\PCL 1.8.1\3rdParty\Boost\include\boost-1\_64

3. C:\Program Files\PCL 1.8.1\3rdParty\Eigen\eigen3

4. C:\Program Files\PCL 1.8.1\3rdParty\FLANN\include

5. C:\Pactris\_Pro\Approach1\Approach1\dist\json

dist folder contains the header file as well as source file which needs for json parser.

I will copy the dist folder in the project folder, just add the path as your project directory + \dist\json

You can change the directory path as per your system configuration.

**2. Add the library path**

**Right click on the project ->Properties ->Linker ->General -> Add Library Directories**

1. C:\Program Files\PCL 1.8.1\lib (PCL Library)

2. C:\Program Files\PCL 1.8.1\3rdParty\Boost\lib(Boost)

3. C:\Pactris\_Pro\opencv\build\x64\vc14\lib(opencv)

You can change the directory path as per your system configuration.

**3. Add the library**

**Right click on the project ->Properties ->Linker ->Input ->Additional dependencies**

1. pcl\_common\_debug.lib(Print console Function)

2. pcl\_io\_debug.lib(OuputArray,Gaussian Blur)

3. opencv\_world320d.lib(Opencv Related Function)

4. pcl\_segmentation\_debug.lib(Segmentation)

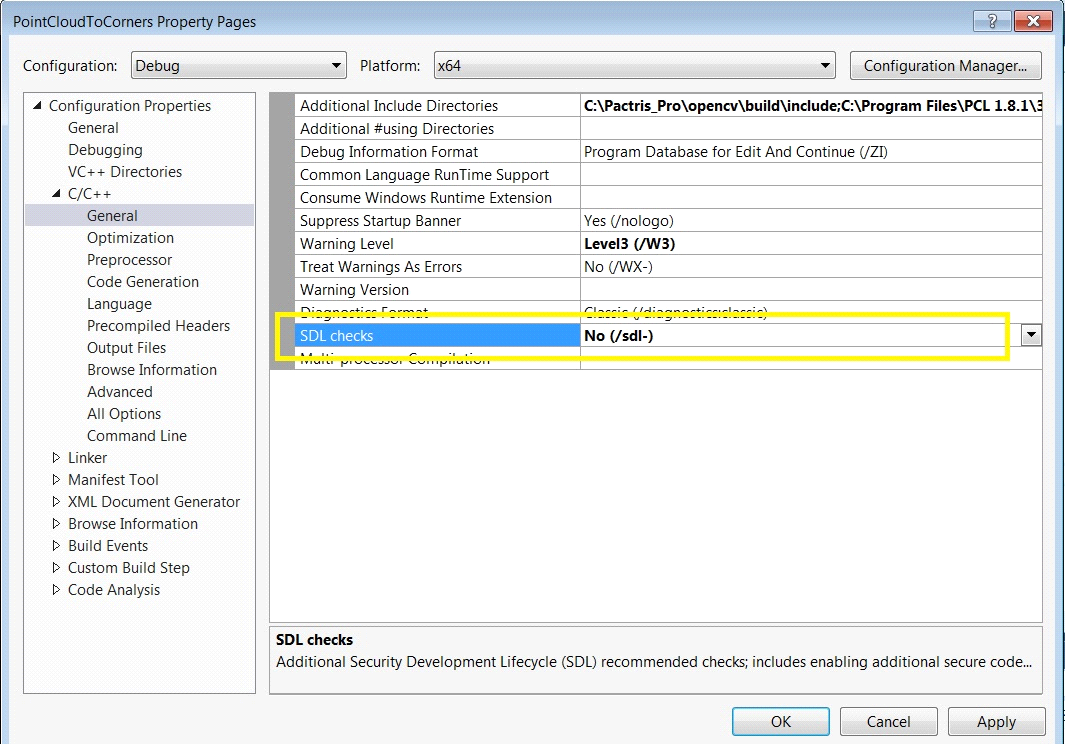
5. pcl\_search\_debug.lib(Search)

6.pcl\_surface\_debug.lib(Convex Hull)

7. pcl\_filters\_debug.lib(ExtractIndices)

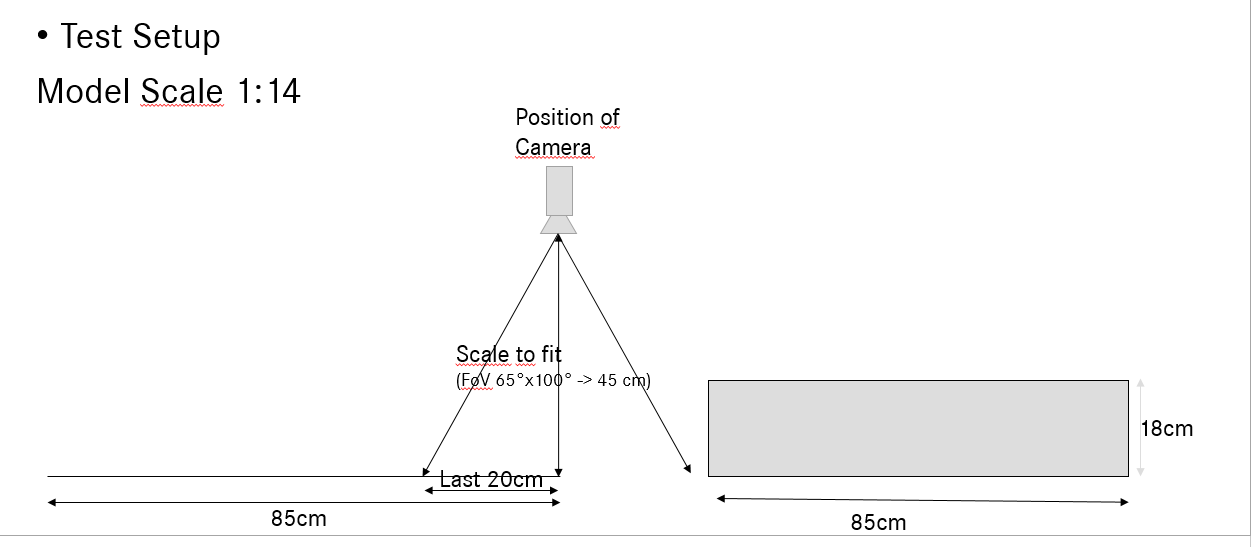
8. pcl\_octree\_debug.lib(octree change detection)

**4. To disable the warnings Right Click on the project->Properties -> C++-> General -> SDL Checks ->Change yes to No.**

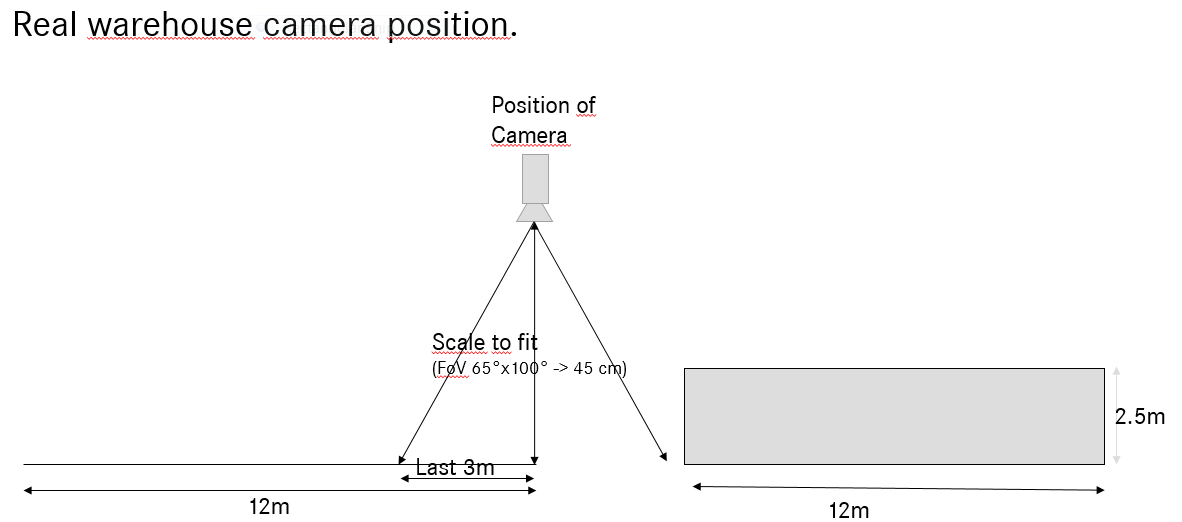


# Approach 1: Box Detection (Approach to find the dimension and real world coordinates of the load Carrier)

**Camera Placement for Approach 1 in Test Setup:**



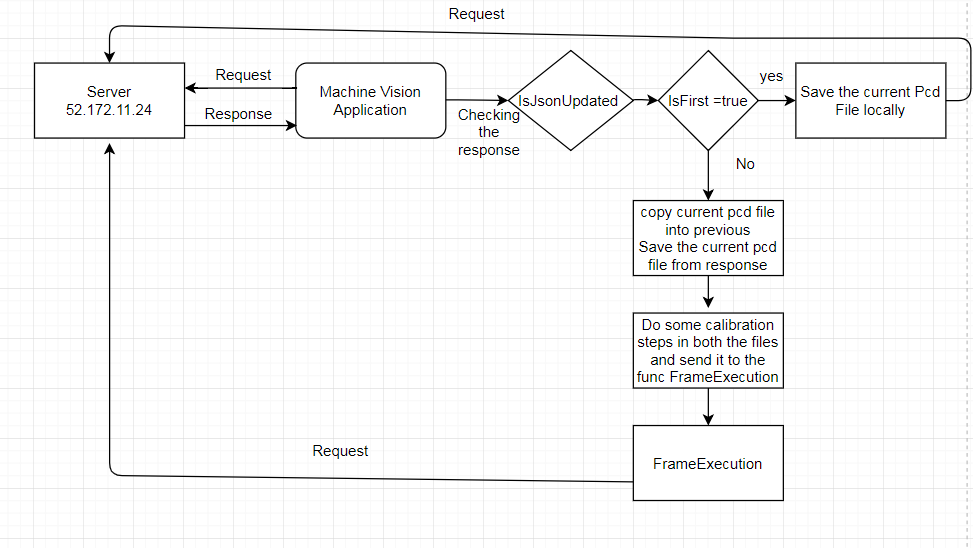
**Camera Placement for Approach 1 in Warehouse(1:1) Setup:**

****

Note: Camera should be placed parallel to the prestacking area

**Flowchart for approach1:**

1. **Pipeline of Machine Vision Part**

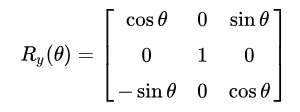
****

**Calibration steps:**

* 1. **Add rotation matrix to the pointcloud**

**Case 1: Speyer Camera Mounded Setup**

* **Rotation in Y(-90 degree)**

****

**0 0 -1**

**Ry(-90) = 0 1 0**

**1 0 0**

**2. Removing NAN Values from the pointcloud**

* **Step 1:**

**pcl::removeNaNFromPointCloud(InputCloud, OutputCloud, indices);**

**InputCloud – Cloud which contains NAN Values and Non NAN Values**

**OutputCloud – Cloud which holds only NAN Values**

**Indices – which contains index of Non NAN Values.**

* **Step 2:**

**Create and initialize the pointcloud with 0.**

**Copy the Points whose index are available in the Indices array to the newly created pointcloud.**

**Translation factor for the speyer usecase:**

**Translation in X = XValue + 1.10**

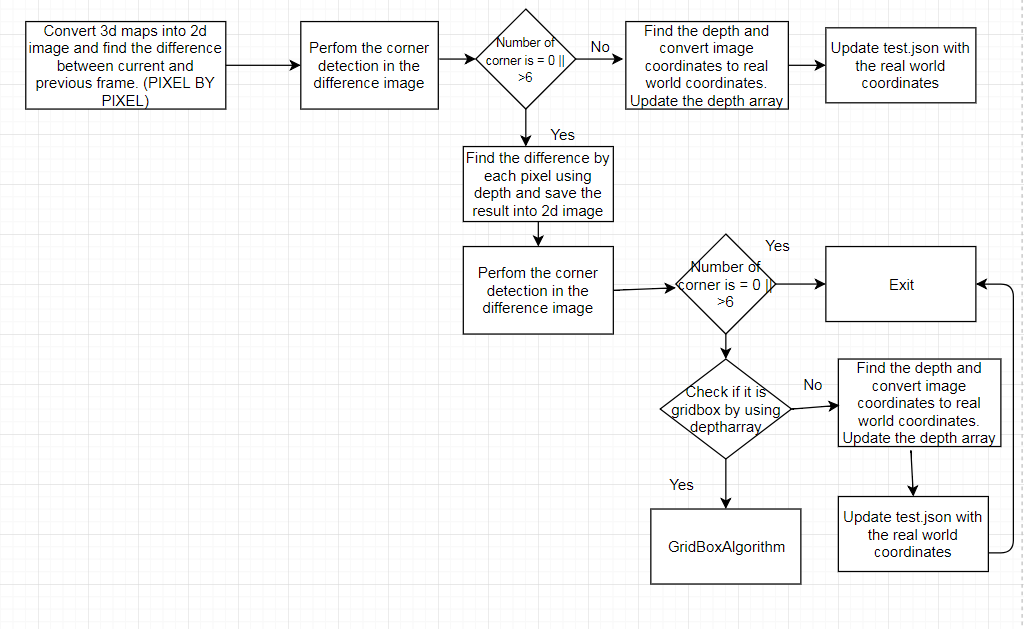
**Translation in Y = YValue + 5.93**

**Translation in Z = Max\_height – ZValue;**

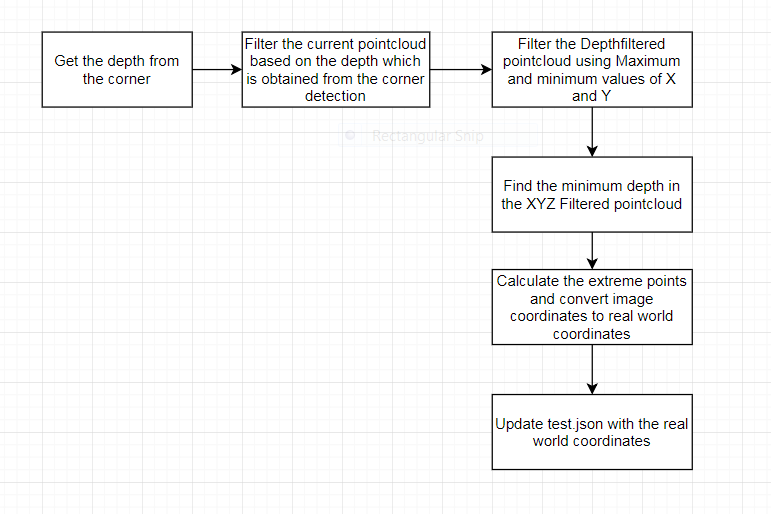
**Where**

**Max\_height = Camera Mounted height from the floor(In Meter)**

1. **Flowchart for Function Frame Execution**

****

1. **Grid Box Algorithm**

****

**Algorithm:**

1. Get the pcd files from the server and check whether is it updated one or not.
2. If is it updated one, then continue with the below the steps, otherwise send get request to the server and check it again.
3. Subtract Current and previous frame to get the difference(Single Box as ROI)
4. Perform corner detection in the difference image
5. If the corner count is more than 6 or less than 0, then do the grid box detection algorithm.

1. **Grid Box Algorithm:**
   1. Find the bottom layer of the box, find X and Y coordinates by doing corner detection.
   2. After getting the corners, needs to get the depth out of it (i.e. bottom point depth).
   3. Filter the pointcloud, which is having less depth than the bottom layer depth.
   4. Find the minimum depth which is available in the filtered Pointcloud.
   5. Construct the 3d Box Out of it and find the two extreme points.
   6. Convert image coordinates into real world coordinates.
   7. Update the json and send it to the server.
2. **Closed Box Detection:**
   1. Find the Top layer of the box, find X and Y coordinates by doing corner detection.
   2. After getting the corners, needs to get the depth out of it.(i.e. bottom point depth)
   3. Find the two extreme Points(one bottom and one top )
   4. Convert image coordinates into real world coordinates
   5. Update the json and send it to the server

**Output in the PlaneViewer:**

