Department of
NUCLEAR ENGINEERING
University of California, Berkeley
Thermal Hydraulics Laboratory

XPETS Module 2
Summary and User's Manual

By Grant Buster

May 27, 2015

Fluoride Salt Cooled High Temperature Reactor

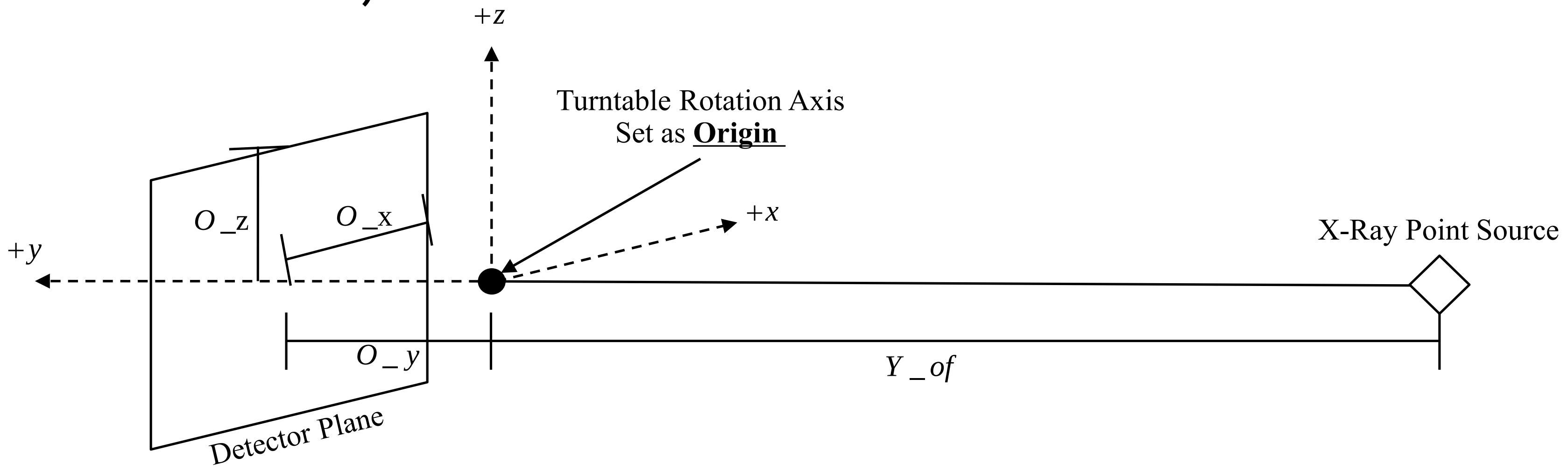
Nuclear Energy University Programs
U.S. Department of Energy

XPETS Module 2 Calibration

Geometry Solver

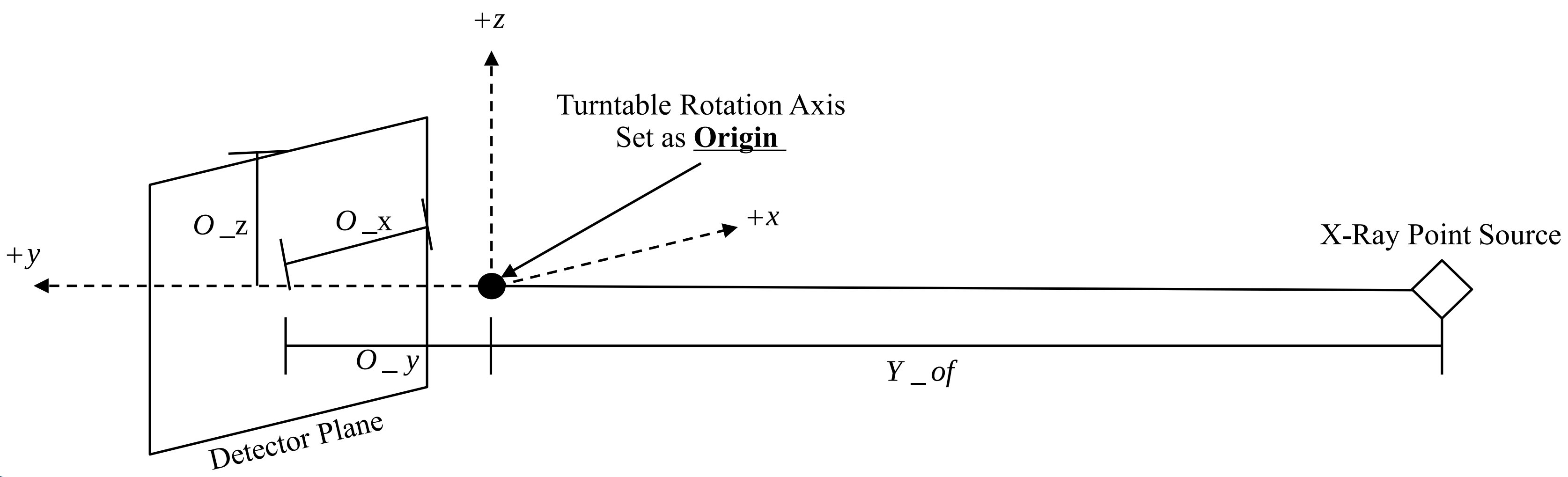
Geometry Assumptions

- Your system geometry is only defined by your x-ray point source, detector and turntable, NOT the test section
- The detector is assumed to be a perfectly vertical plane
- The detector is assumed not to be rotated in the xz plane (LEVEL IT!)
- The origin is set as (0,0,0)
  - The xy origin is set at the turntable rotational axis
  - The z origin coordinate is set at the plane intersecting the x-ray point source and perpendicular to the detector plane (the z-origin plane is horizontal)



Importance of Geometry

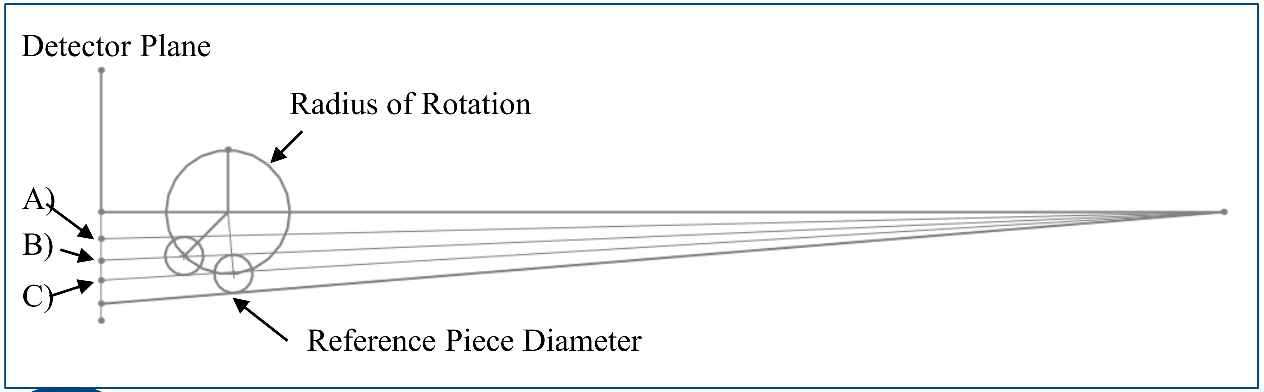
- An accurate 3D reconstruction is impossible without a good geometry solution
- What does this mean?
- 4 key geometric parameters need to be defined:
  - Y_of (y axis distance from origin to focal/source point)
  - O_x (x axis distance from right hand side of detector to origin)
  - O_y (y axis distance from detector plane to origin)
  - O_z (z axis distance from top of detector to origin)



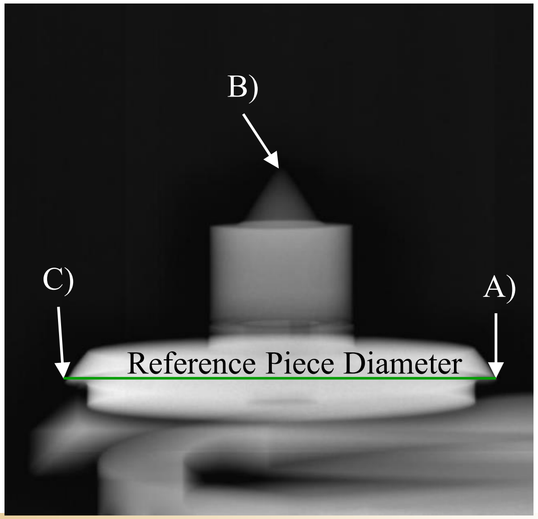
O_y and Y_of

- O_y and Y_of have been defined using a machined reference geometry and Solidworks geometry solver
  - O_y = 0.2283 (m)
  - Y_of = 1.8453 (m)
- These values should be considered constant unless:
  - The turntable is moved
  - The columnator or detector are moved in the y-direction (but vertical shifts should not affect O_y and Y_of)

SolidWorks Geo Solver (Top Down View, XY Plane)



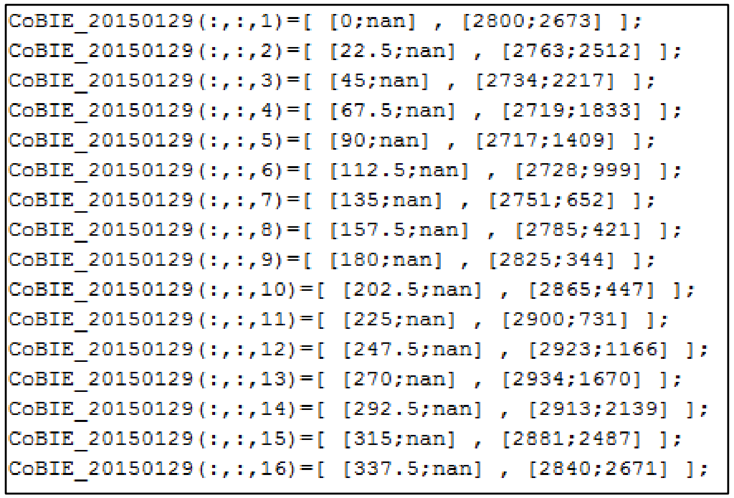
Machined Reference Piece



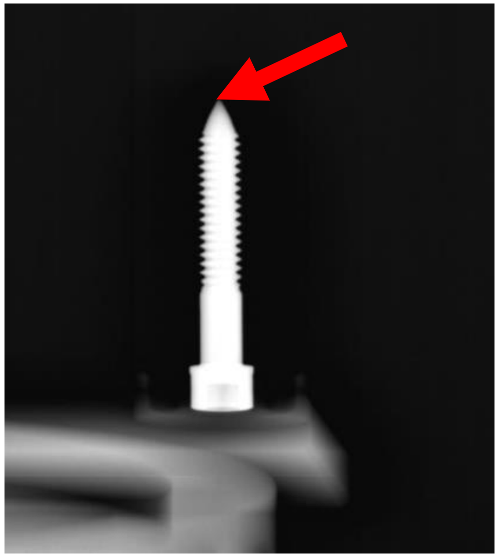
Defining O_x

- **We define O_x using an x-ray image set of a single point (usually a thumb screw) rotated through 360 degrees. NOTE: the screw should be placed as far off-center as possible (away from the center of the detector in z and x directions).**
- **The angle and the i,j tip location (in pixels) (i=column#, j=row#) for each image need to be entered in the format:**
  - Ox_DATA(:,:,1)=[[angle;nan] , [j ; i]];
- **Into the code:** ***XPREX_20141029_Full_Geo_Solver.m***
- **Example:**

XPREX_20141029_Full_Geo_Solver.m

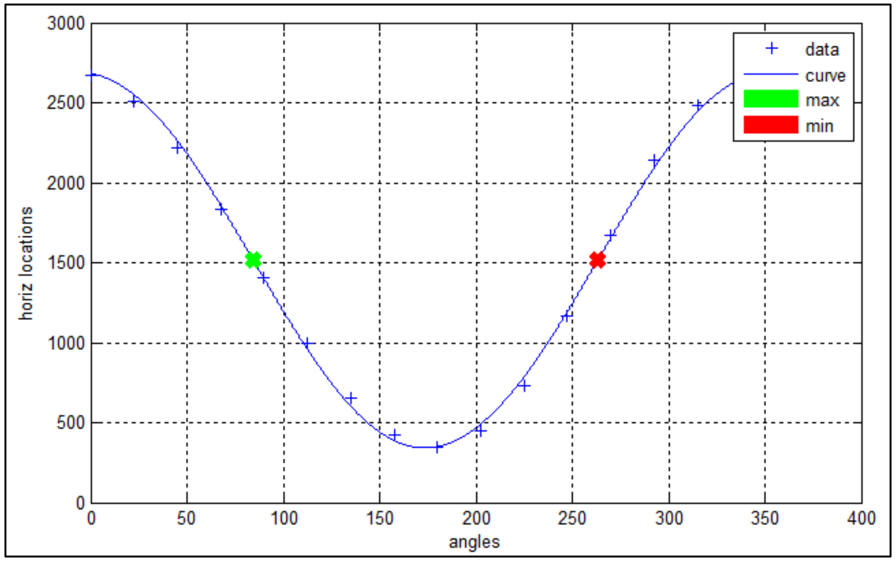


```
CoBIE_20150129(:,:,1)=[ [0;nan] , [2800;2673] ];
CoBIE_20150129(:,:,2)=[ [22.5;nan] , [2763;2512] ];
CoBIE_20150129(:,:,3)=[ [45;nan] , [2734;2217] ];
CoBIE_20150129(:,:,4)=[ [67.5;nan] , [2719;1833] ];
CoBIE_20150129(:,:,5)=[ [90;nan] , [2717;1409] ];
CoBIE_20150129(:,:,6)=[ [112.5;nan] , [2728;999] ];
CoBIE_20150129(:,:,7)=[ [135;nan] , [2751;652] ];
CoBIE_20150129(:,:,8)=[ [157.5;nan] , [2785;421] ];
CoBIE_20150129(:,:,9)=[ [180;nan] , [2825;344] ];
CoBIE_20150129(:,:,10)=[ [202.5;nan] , [2865;447] ];
CoBIE_20150129(:,:,11)=[ [225;nan] , [2900;731] ];
CoBIE_20150129(:,:,12)=[ [247.5;nan] , [2923;1166] ];
CoBIE_20150129(:,:,13)=[ [270;nan] , [2934;1670] ];
CoBIE_20150129(:,:,14)=[ [292.5;nan] , [2913;2139] ];
CoBIE_20150129(:,:,15)=[ [315;nan] , [2881;2487] ];
CoBIE_20150129(:,:,16)=[ [337.5;nan] , [2840;2671] ];
```



Defining O_x

- *XPREX_20141029_Full_Geo_Solver.m* then utilizes the fact that the horizontal location of the screw tip will move in a sinusoidal path:



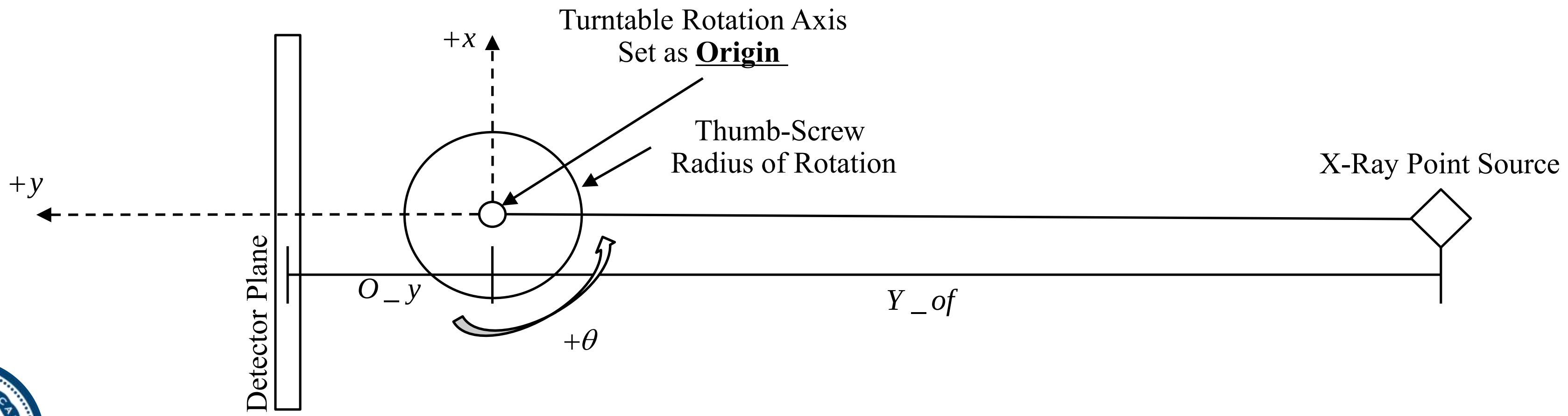
- *XPREX_20141029_Full_Geo_Solver.m* will automatically calculate any angle offset and O_x
  - Angle offset is caused by the screw not being placed perfectly at 0-degees when the turntable is at 0-degrees. This is accounted for and it is okay to place the screw at a random angle.

Defining O_z

- This is the (algebraically) difficult part
- Start from a top-down view, in order to calculate the radius of rotation of the thumb screw tip
- Radius of rotation is calculated by the following equation:

$$\boxed{R_{rot} = \frac{Y\_of \times abs\left(i - O\_x\right)}{\left(Y\_of + O\_y\right)\cos\left(\theta\right) - abs\left(i - O\_x\right)\sin\left(\theta\right)}}$$

- Where i is the horizontal pixel location (column index) of the thumbscrew tip in one image. This is usually taken when the screw is far right or far left. Theta in the above equation is theta of the turntable plus the theta offset



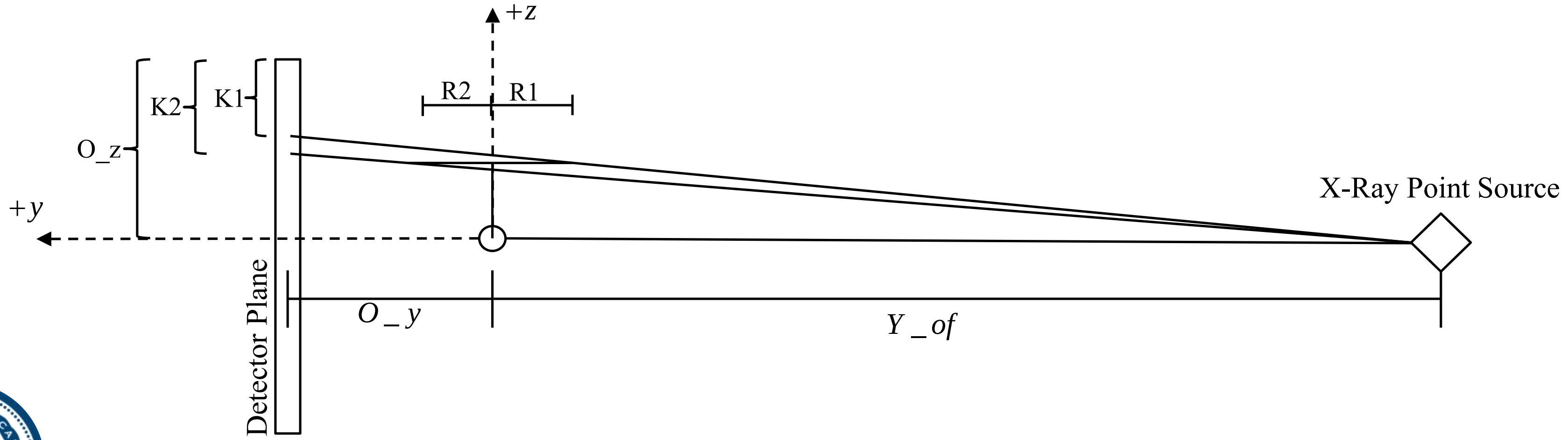
Defining O_z

- We move to a side-view, in order to calculate O_z
- Two images are picked, usually the two where the thumbscrew tip is highest and lowest in the image
- R1 and R2 are calculated using R_rot*sin(theta), such that:

$$R1 < 0$$
$$R2 > 0$$

- O_z is calculated as follows:

$$\boxed{O\_z = \frac{(K2)(Y\_of) + (K2)(R2) - (K1)(Y\_of) - (K1)(R1)}{R2 - R1}}$$

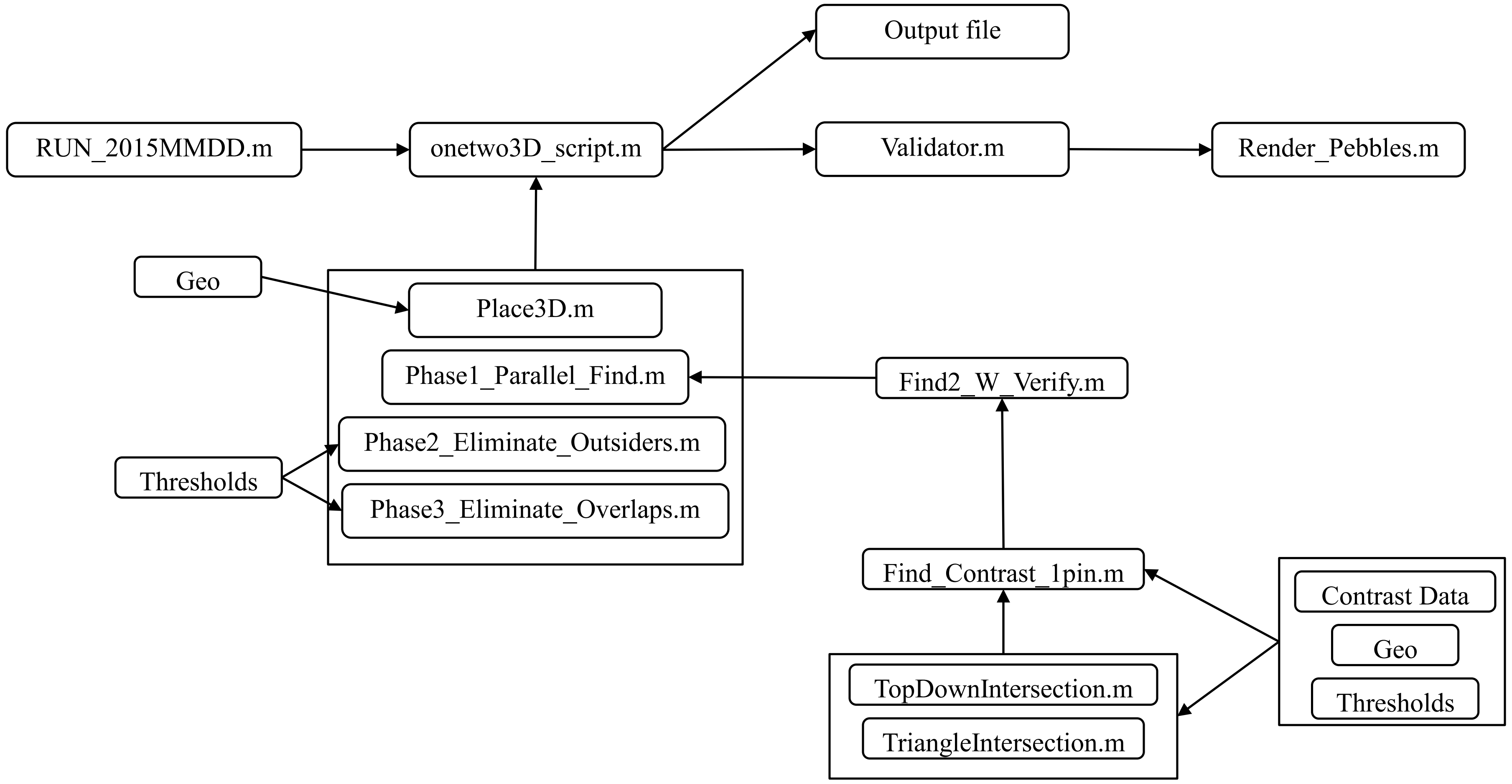


Geometry Solved!

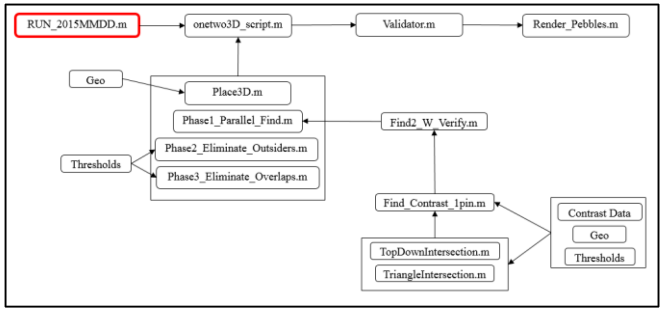
- The geo.mat file should now be good to go!
- Two other important parameters:
  - The detector has 6993 pixels per meter (PPM)
  - A pin/pebble diameter is 0.01257 (m)
- The geo.mat file should be placed in the active directory, and loaded before module2 starts to run

XPETS Module 2
3D Bed Reconstruction

Module 2 Code Architecture



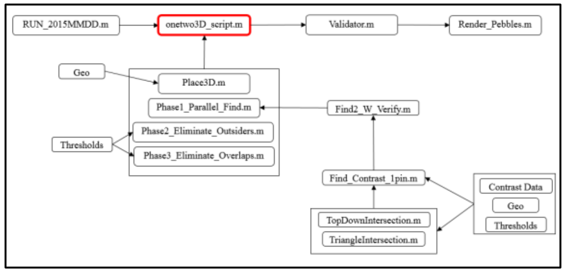
RUN_2015MMDD.m



- This file will execute the following:
  - Setup parallel toolbox
  - Iterate through study timesteps
    - » Import module1 data
    - » Take user selected threshold values
    - » Load geometry file
    - » Save output file
  - Iterate through post processing
    - » Validator
    - » Render

- This file is intended to <u>hold all of the user inputs and setup parameters</u>. High level organization.

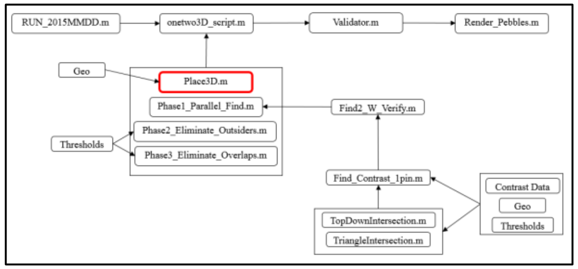
onetwo3D_script.m



- This file will execute the following:
  - (Optional) pre select a region of module1 data
    - » This is good for prototyping code
    - » Selects a small region of interest for faster runtime
  - Place3D.m
  - Phase1_Parallel_Find.m
  - Phase2_Eliminate_Outsiders.m
  - Phase3_Eliminate_Overlaps.m

- This file is largely organizational in nature
- See details on each phase in later slides

Place3D.m



- This file is responsible for taking module 1 data and placing the endpoints on their respective detector planes in 3-dimensional space
- Highly geometry dependent
- Input (module1) (in pixels):

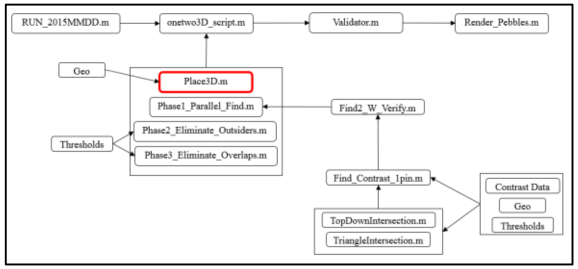
$$X(:,:,n)=\begin{pmatrix} \text{RowIndex\_1} & \text{RowIndex\_2} \\ \text{ColIndex\_1} & \text{ColIndex\_2} \end{pmatrix}$$

- Output (3D in meters):

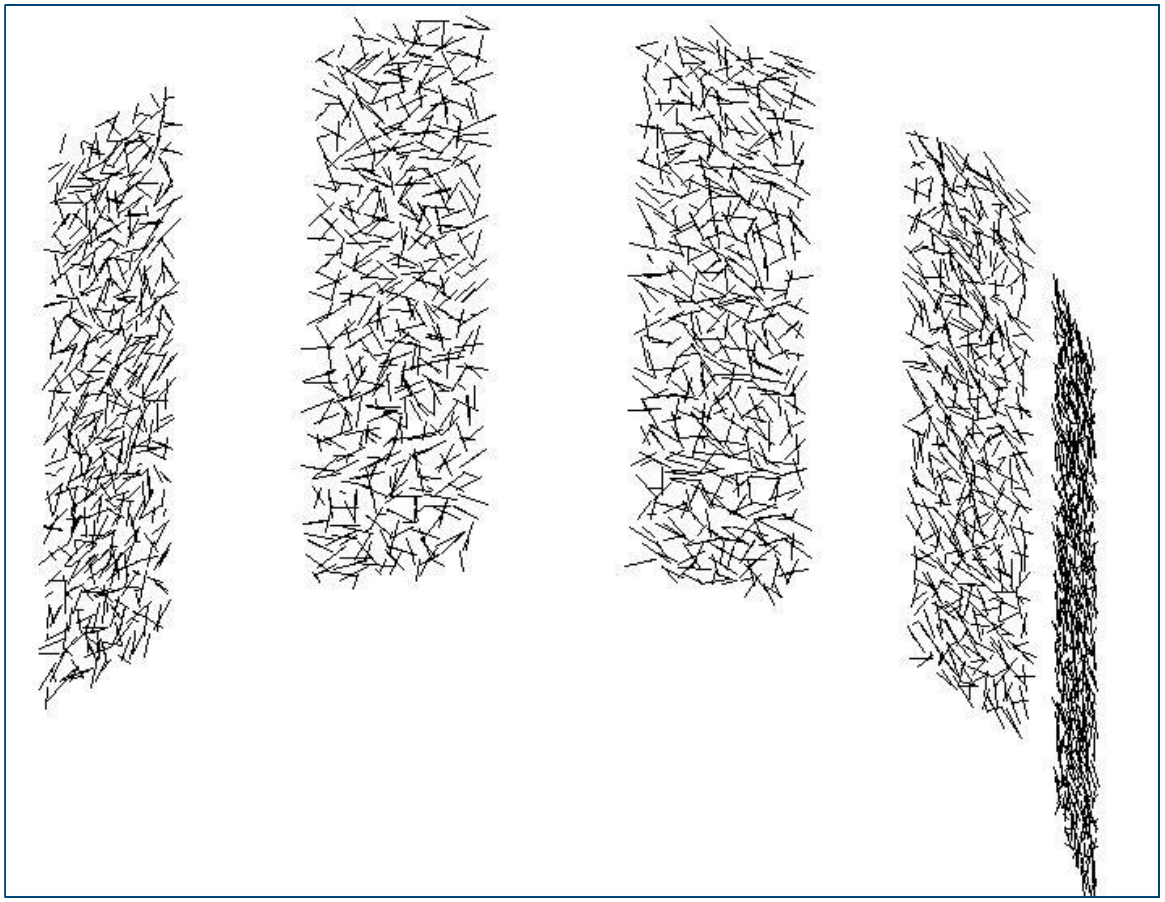
$$X(:,:,n)=\begin{pmatrix} \text{point1\_x} & \text{point2\_x} & \text{source\_x} \\ \text{point1\_y} & \text{point2\_y} & \text{source\_y} \\ \text{point1\_z} & \text{point2\_z} & \text{source\_z} \end{pmatrix}$$

- Where point 1 and point 2 should be the endpoints of a pin in an x-ray image projected onto the detector plane in 3D space

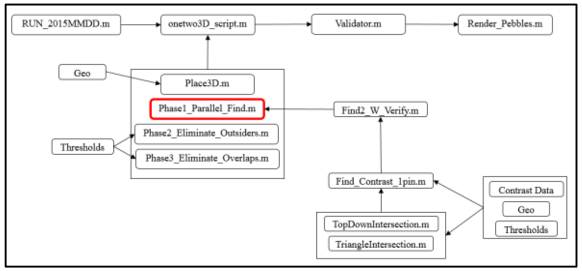
Place3D.m



- Looks like this:

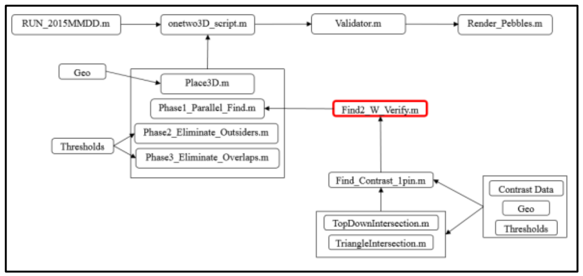


Phase1_Parallel_Find.m



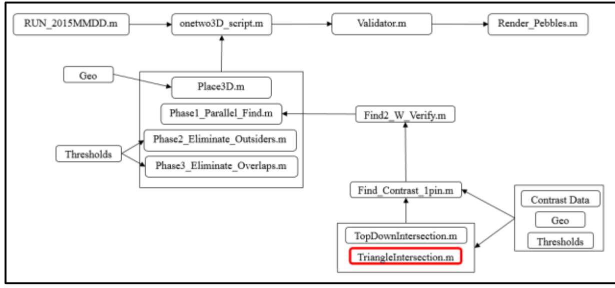
- This phase sets up the parallel processing inputs in a way that MATLAB will accept
- This phase iterates through possible comparisons of module 1 data, running the Find2_W_Verify.m algorithm
  - For example, module 1 will return endpoints for 5 different detector angles
    - » The find2_w_verify algorithm will take data sets from 2 detector angles and compare them for possible 3D results
    - » Phase1_parallel_find.m iterates through possible input pairs for find2_w_verify.m
- Phase1_Parallel_Find.m is organizational and is mostly meant to setup the parallel processing structure
- Parallel processing can be easily swapped with serial by replacing the PARFOR loop with a FOR loop
  - Everything else should be kept the same… setup is for parallel but can also work for serial, not the other way around

Find2_W_Verify.m



- This is the main reconstruction algorithm
- The function takes in two sets of pairs of endpoints
- The function iterates through both sets, trying all possible comparisons between two pairs of endpoints
- An initial midpoint vector calculation is done to see if they are at all close vertically (easy way to eliminate possibilities)
- If pairs of endpoints are close, run intersection algorithms
  - TriangleIntersection.m
  - TopDownIntersection.m
- If intersection algorithms find a solution (within set thresholds) use a contrast verification algorithm to check solution against contrast data for all five images
- If contrast data is verified, add solution to output

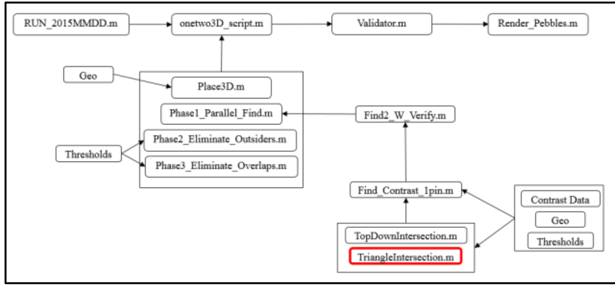
TriangleIntersection.m

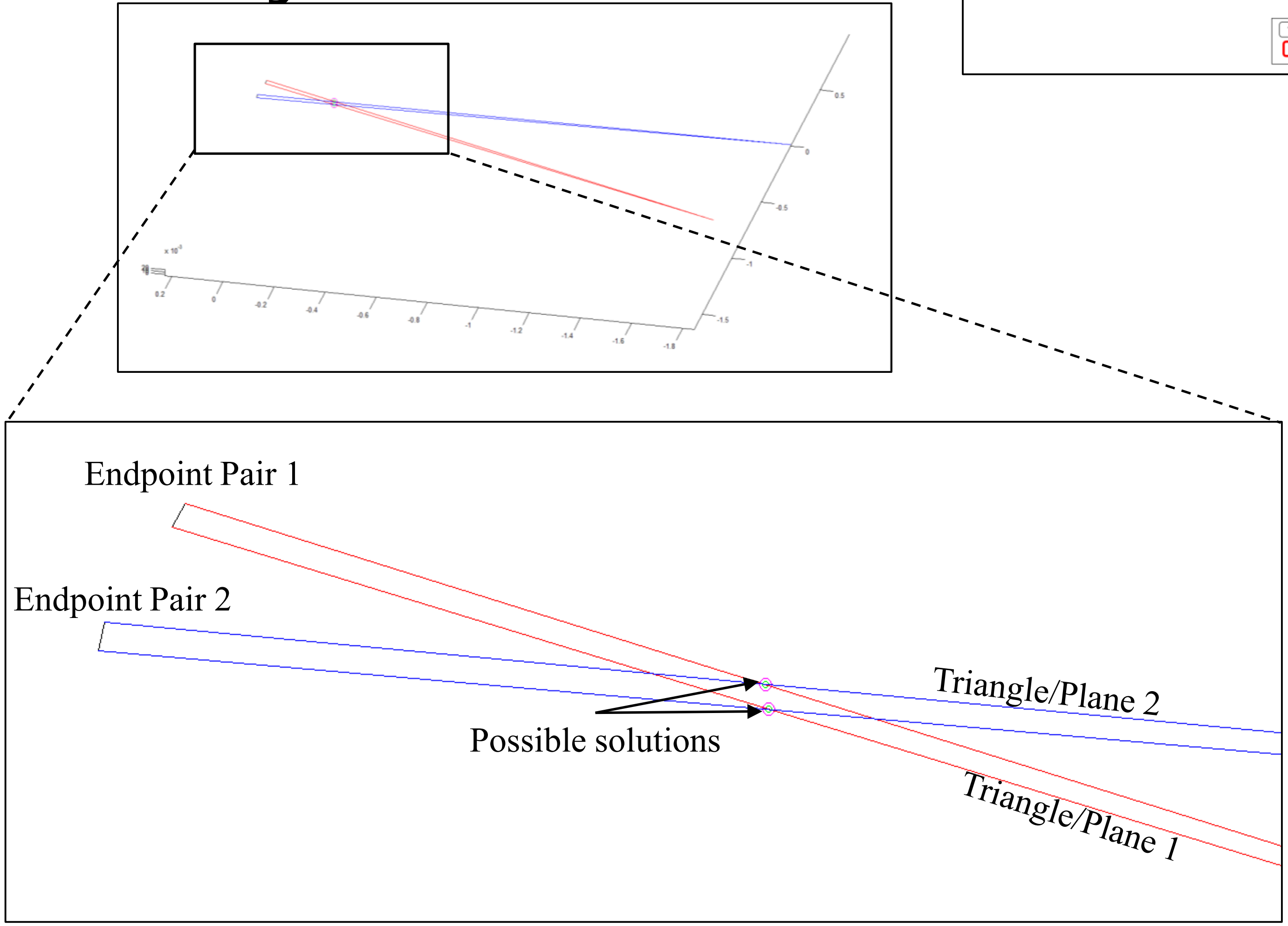


- A endpoints belonging to the same pin in an image form a triangle with the third point being the x-ray point source
- Vector calculus can be used to compare two triangles and find their linear intersection
- A good vector calculus reference can be found here: http://geomalgorithms.com/a06-_intersect-2.html
- The algorithm results in two line segment solutions:
  - Endpoints1 as a triangle intersecting endpoints2 as a plane
  - Endpoints2 as a triangle intersecting endpoints1 as a plane
- Thresholds used to eliminate:
  - Normalized length: are the linear solutions the same length as the pin's true length (pebble diameter)?
  - Metric spatial similarity: are the midpoints of the two linear solutions close together?
- The two linear solutions will always be perfectly parallel

TriangleIntersection.m

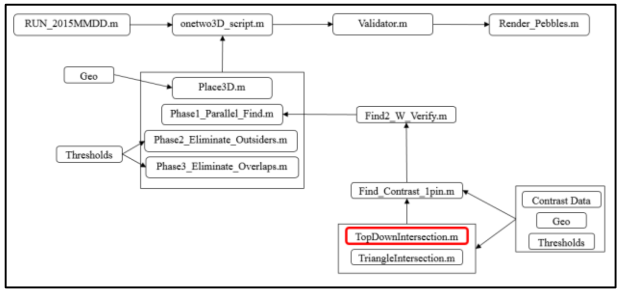
- **This is what a triangle solutions look like:**





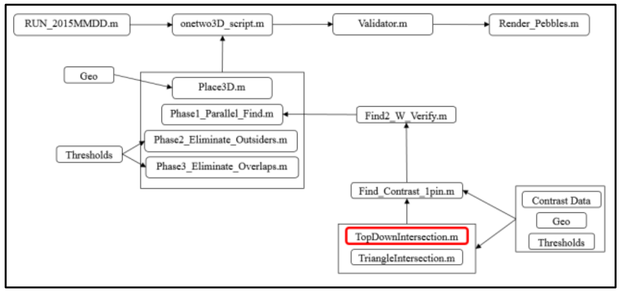
- **Note: In this case, the solution is valid so the two possible line solutions overlap perfectly**

TopDownIntersection.m

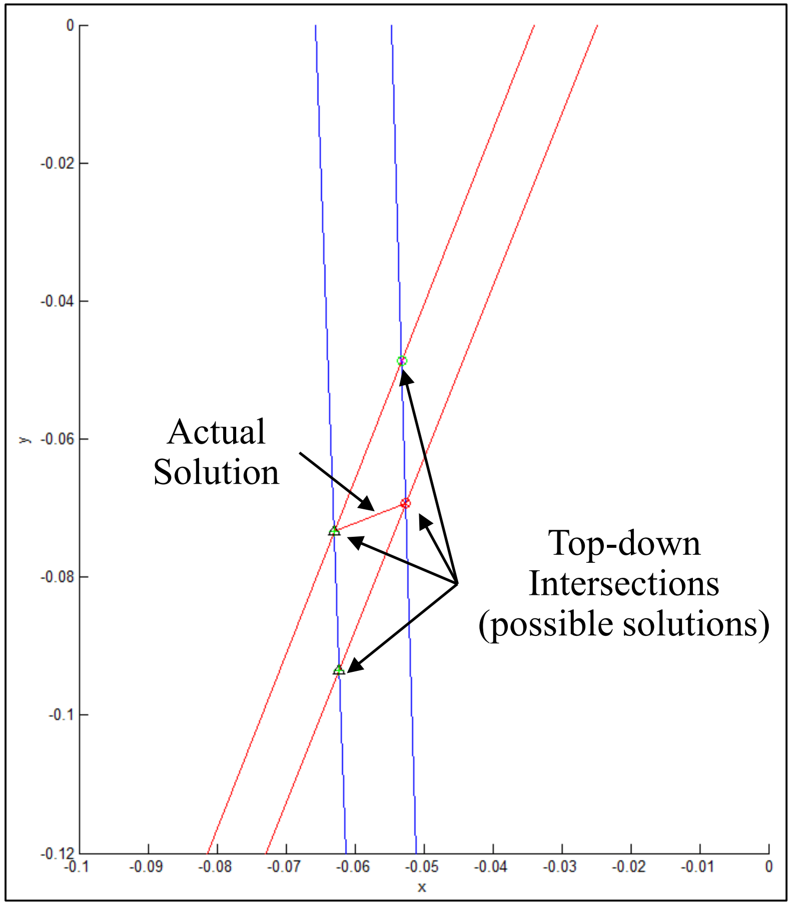


- As the pin approaches a horizontal orientation in the actual system, the triangle intersection method approaches massive uncertainty (think about the intersection of two near-horizontal planes)
- The top down method addresses this issue by examining the intersection of the XY vector components, and then examining where these projected intersections actually lie in the 3-space
- The thresholds are:
  - Normalized length: are the linear solutions the same length as the pin’s true length (pebble diameter)?
  - Metric spatial similarity: are there consistent solutions in 3-space?
- This method returns more possible results than the triangle intersection method (see next slide)

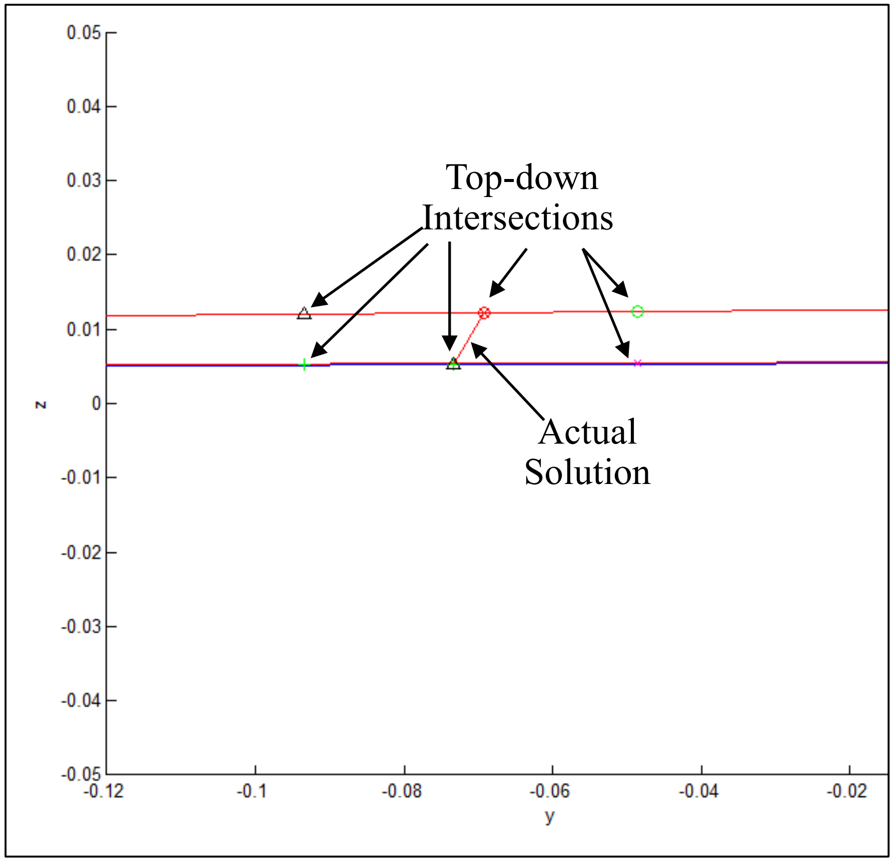
TopDownIntersection.m



- **The possible solutions look like this:**



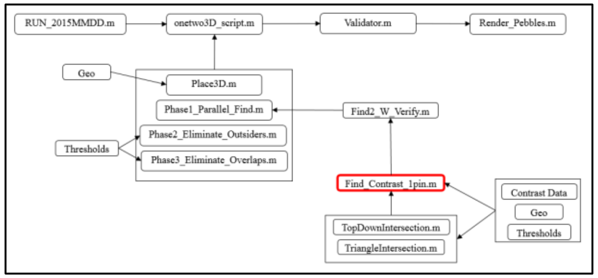
Top-Down View



Side View

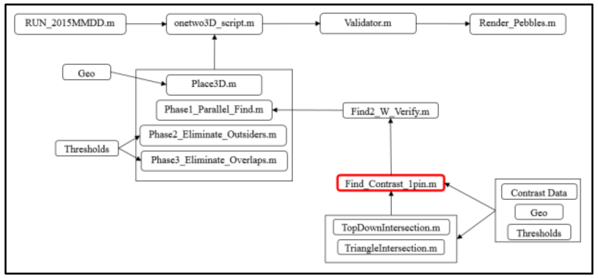
- **Note: in this example, the pin is likely found by the triangle intersection method as well.**

Find_Contrast_1pin.m

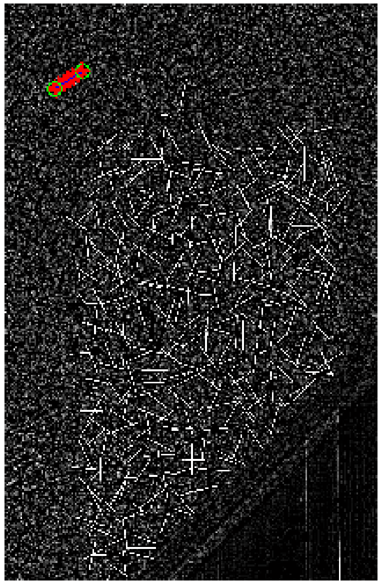


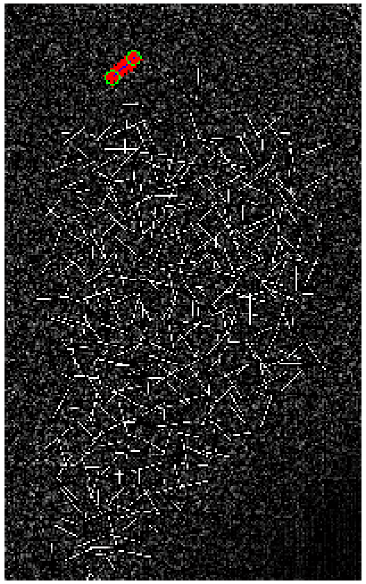
- Every possible solution is projected onto the x-ray contrast data for every detector angle (usually 5 angles) so that it can be verified that this is indeed a pin in each image
- Vector math is used for the projections
  - Similar to intersection algorithms
- An average contrast threshold is set for each image angle (set in the run file)
- 5 pixels of "slop" is considered and factored into an quantitative uncertainty (in the output)
- This function is called the most times, and is therefore the most computationally expensive function to run… It is, on the other hand, incredibly useful in making sure you actually found a solution

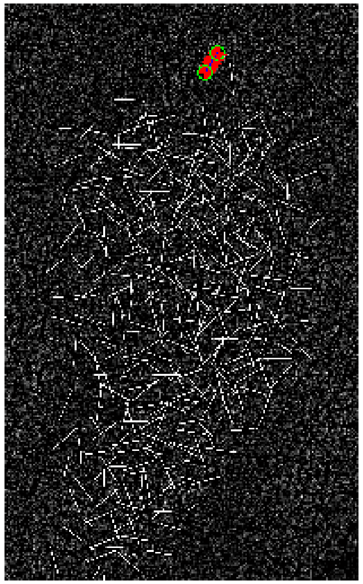
Find_Contrast_1pin.m

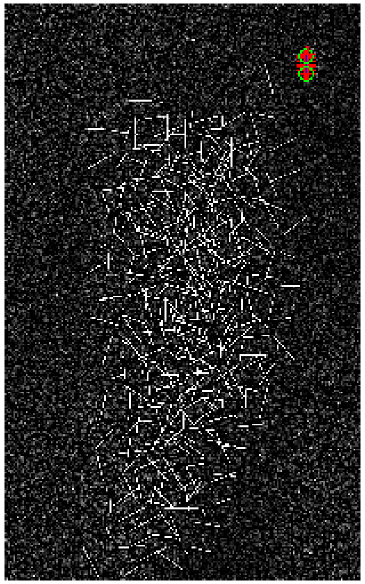


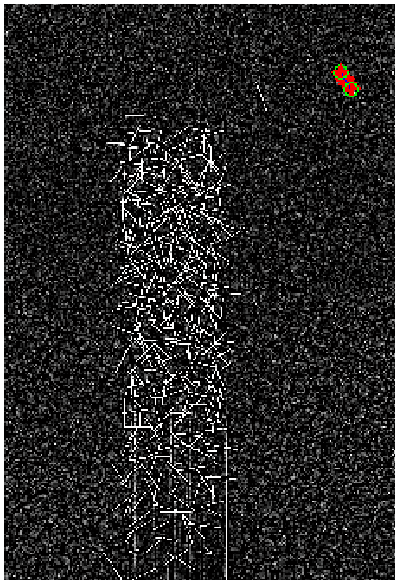
- The contrast projection looks like this:


0-degree


22.5-degree


45-degree

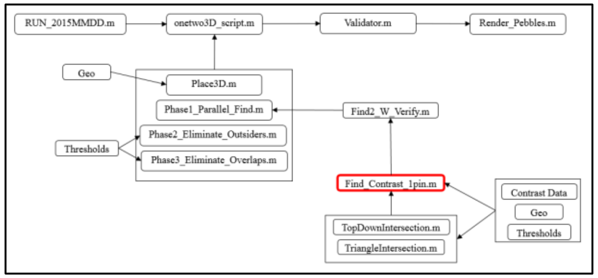

67.5-degree


90-degree

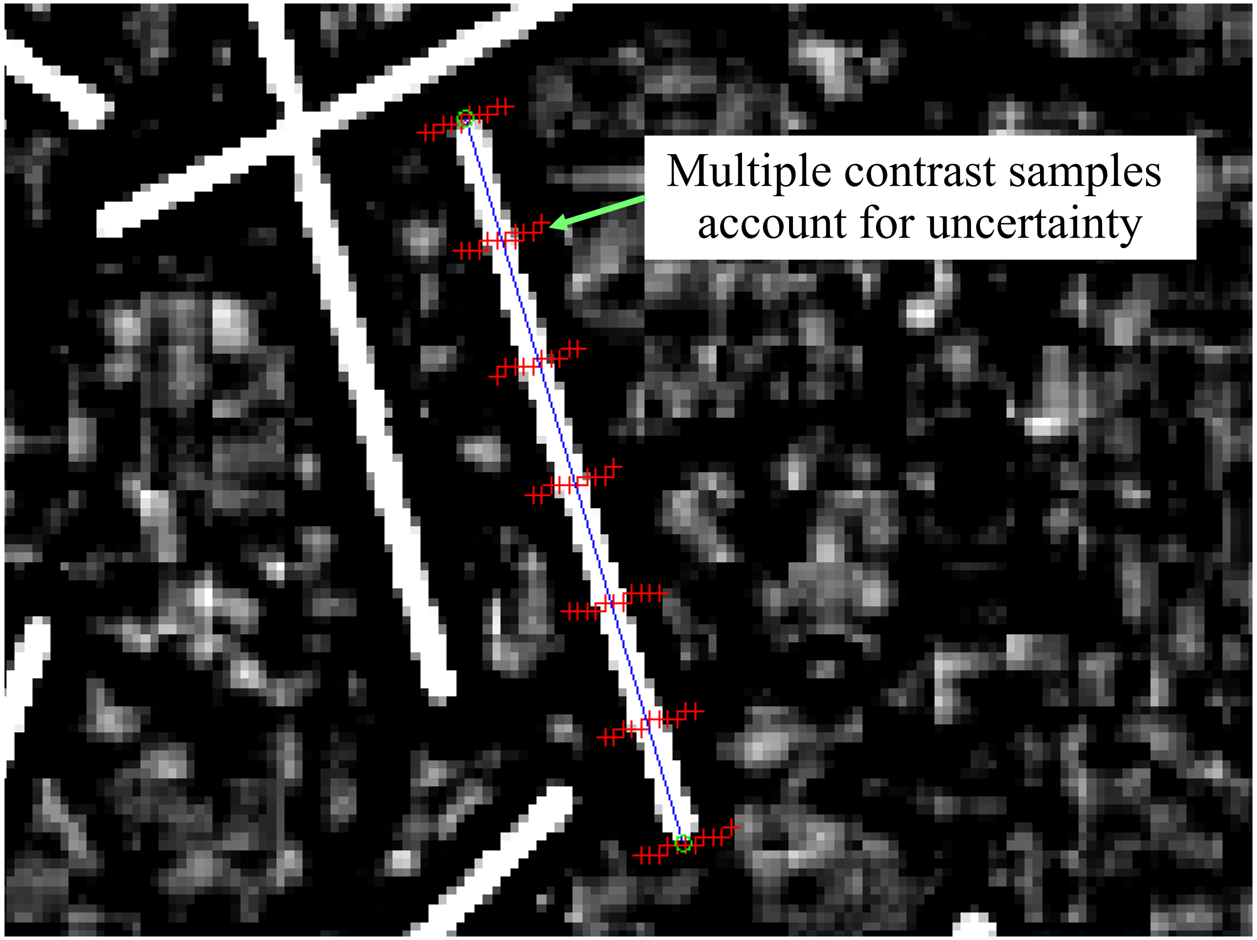
- The output contrast for the 5 images:

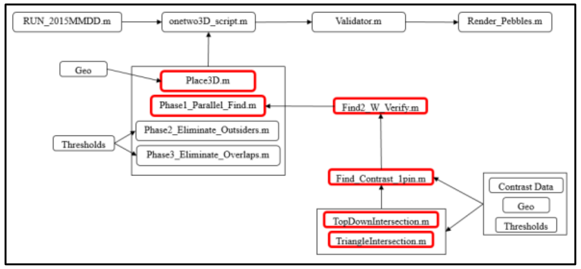
$$\text{Contrast}=\begin{pmatrix} 3.48 \\ 2.21 \\ 1.69 \\ 1.12 \\ 2.27 \end{pmatrix}$$

Find_Contrast_1pin.m



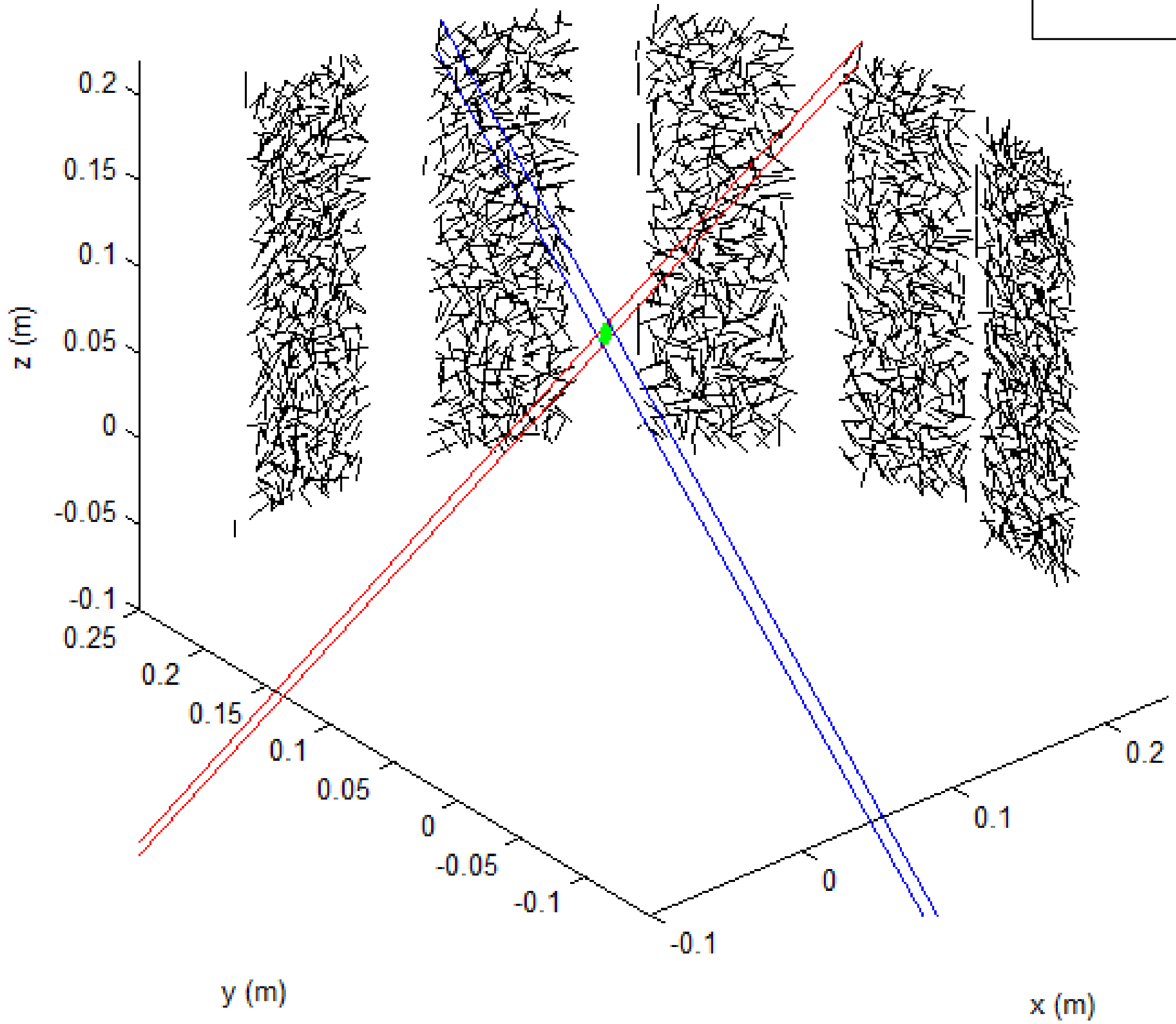
- **Detailed view:**



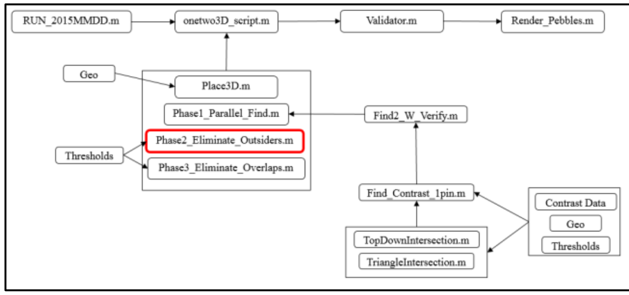


PHASE1 Summary

- In summary, phase 1 looks like this:

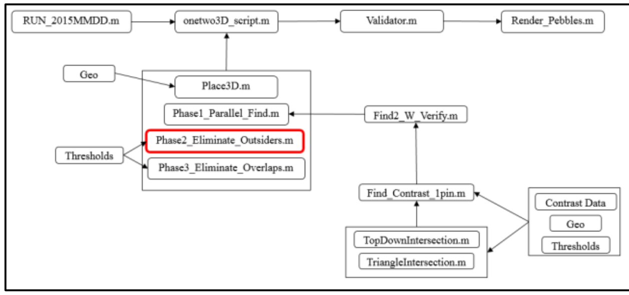


Phase2_Eliminate_Outsiders.m

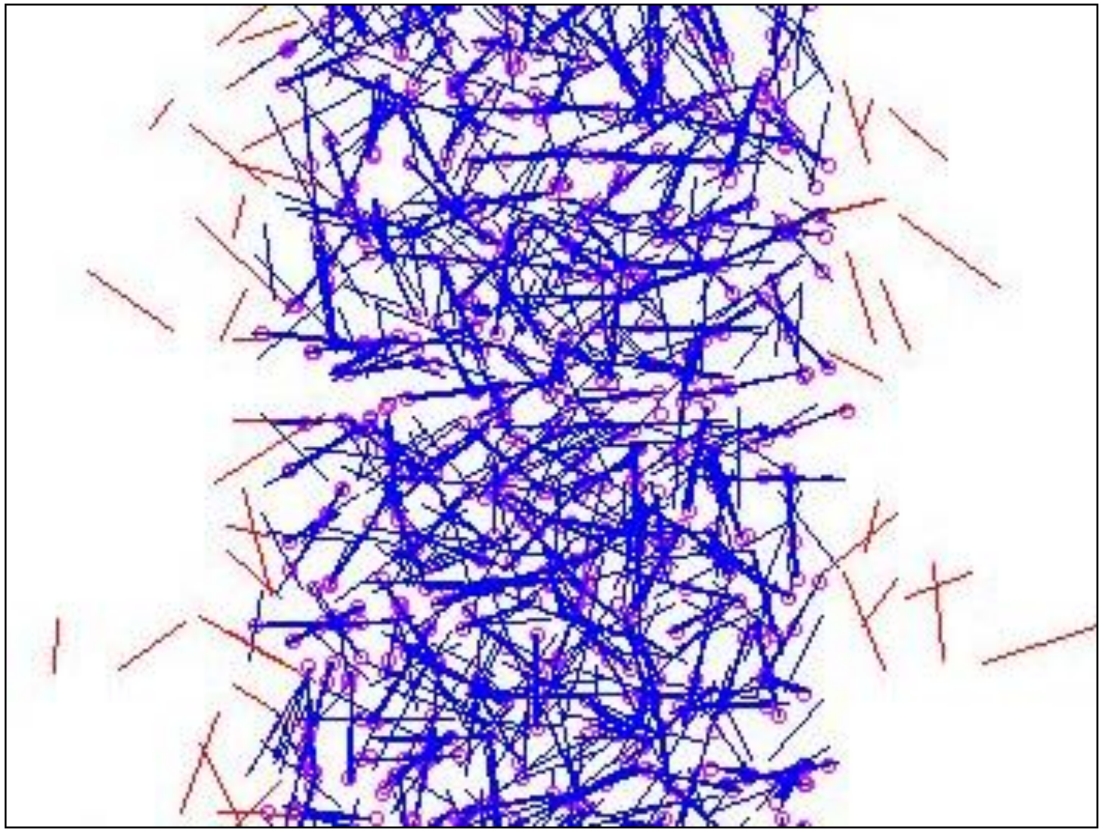


- This phase eliminates any possible solutions that may have been incorrectly identified as correct solutions outside of the physical silo
- It uses max and min endpoints from module 1 to set silo boundaries in the 5 different views
- It also uses two custom set silo boundaries at custom set angles (user input in the RUN.m file)
- It only eliminates horizontal outliers using vertical boundaries

Phase2_Eliminate_Outsiders.m

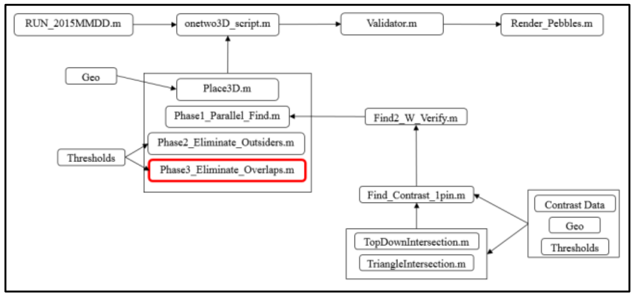


- Looks like this:



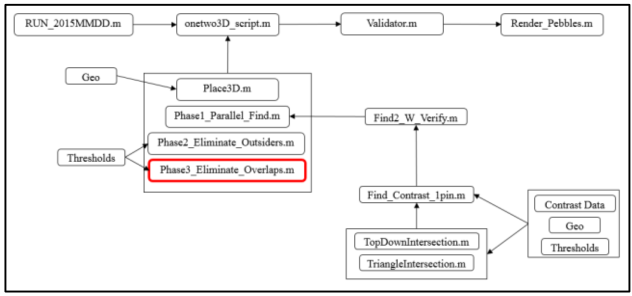
- Note: this image was created with very liberal thresholds and the result is therefore considerably more messy than it should be. Red lines are eliminated for being outside of the silo, blue lines are solutions, pink circles are module 1 results.

Phase3_Eliminate_Overlaps.m

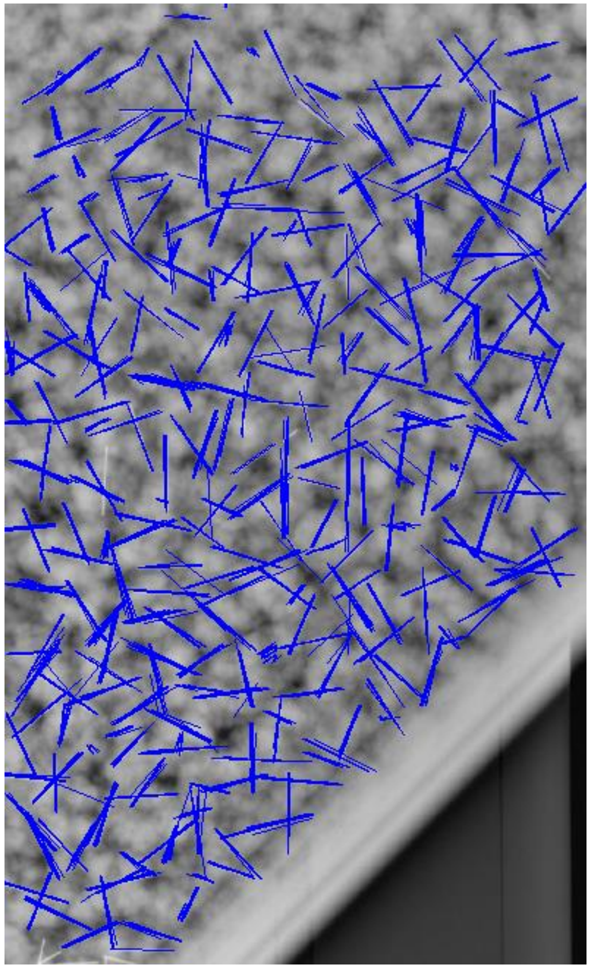


- This module uses the physical constraint that pebbles should contact but not overlap
- An overlap threshold is set (normalized to a pebble diameter) in the RUN.m file
- If any two solutions overlap more than the threshold, one is eliminated based on the uncertainty set forth by the find contrast function

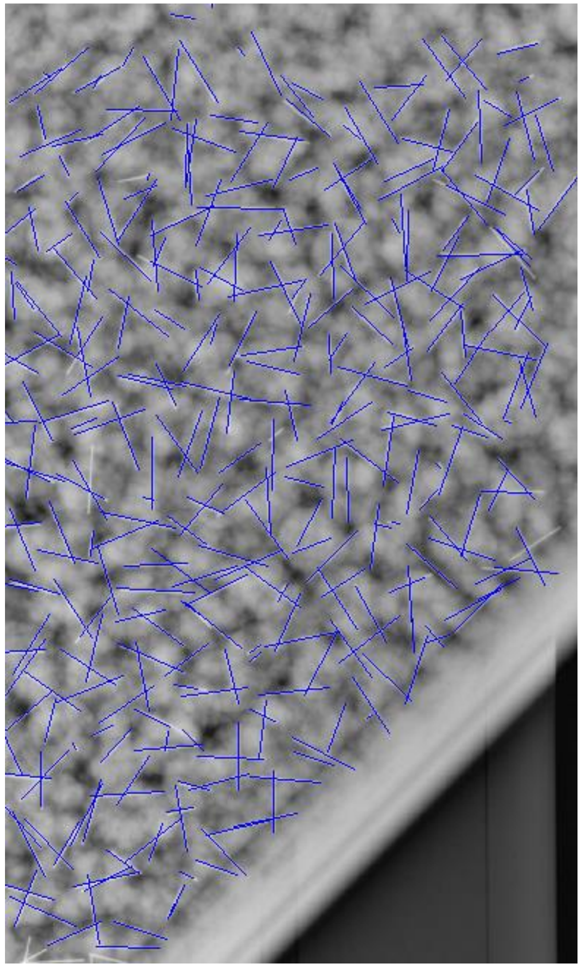
Phase3_Eliminate_Overlaps.m



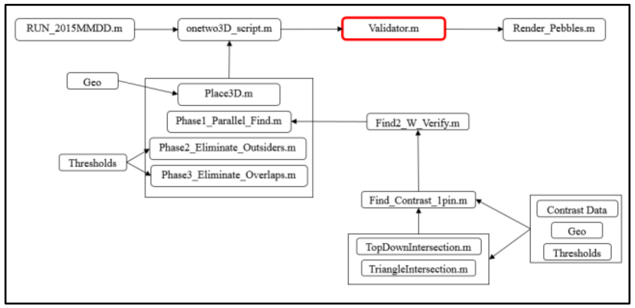
Before Overlap Elimination

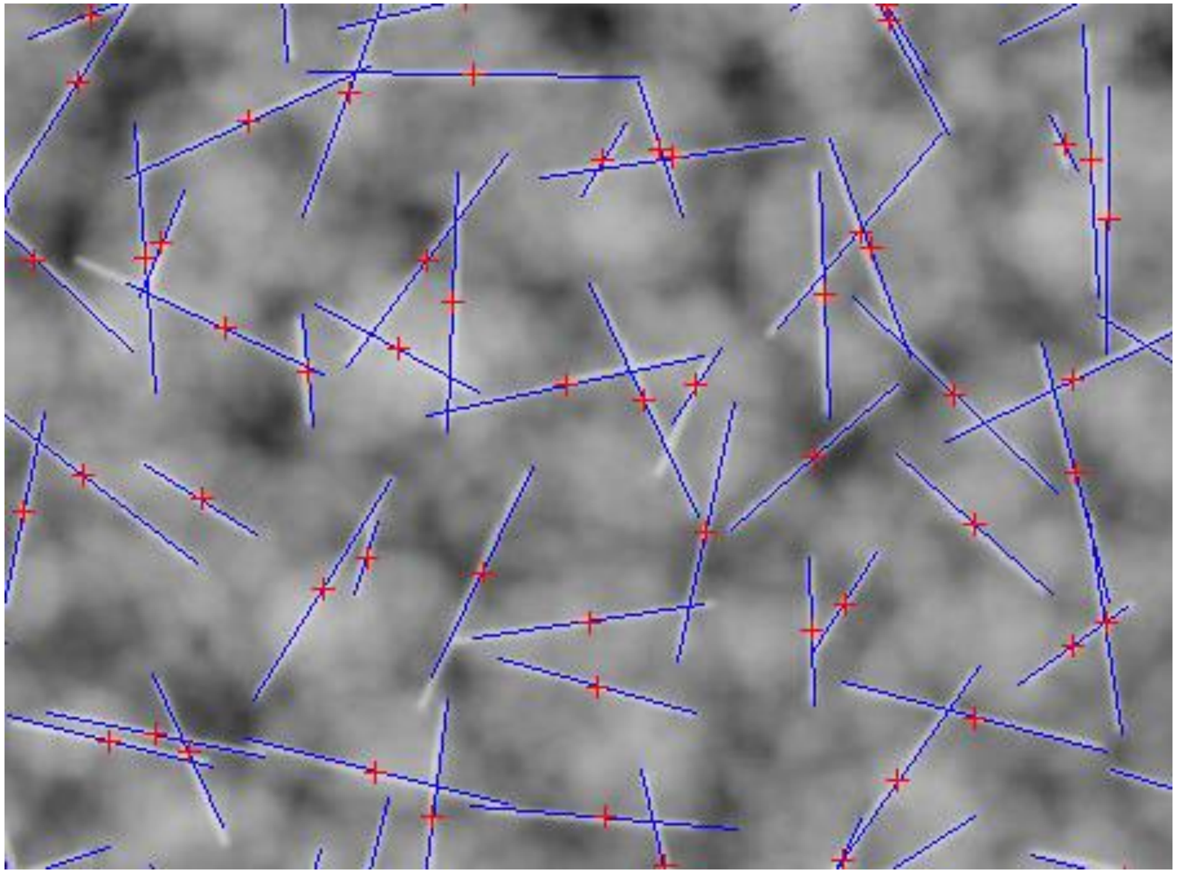


After Overlap Elimination



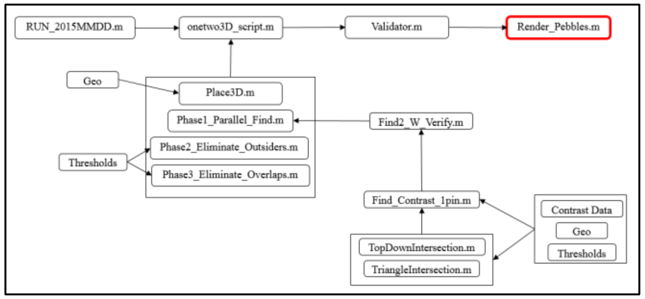
Validator.m



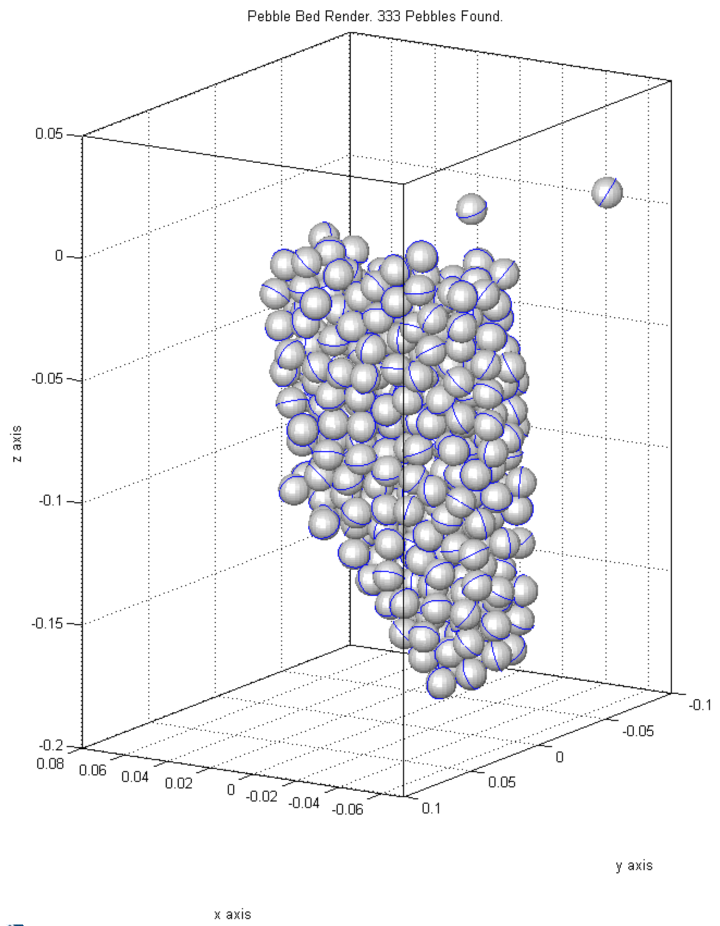


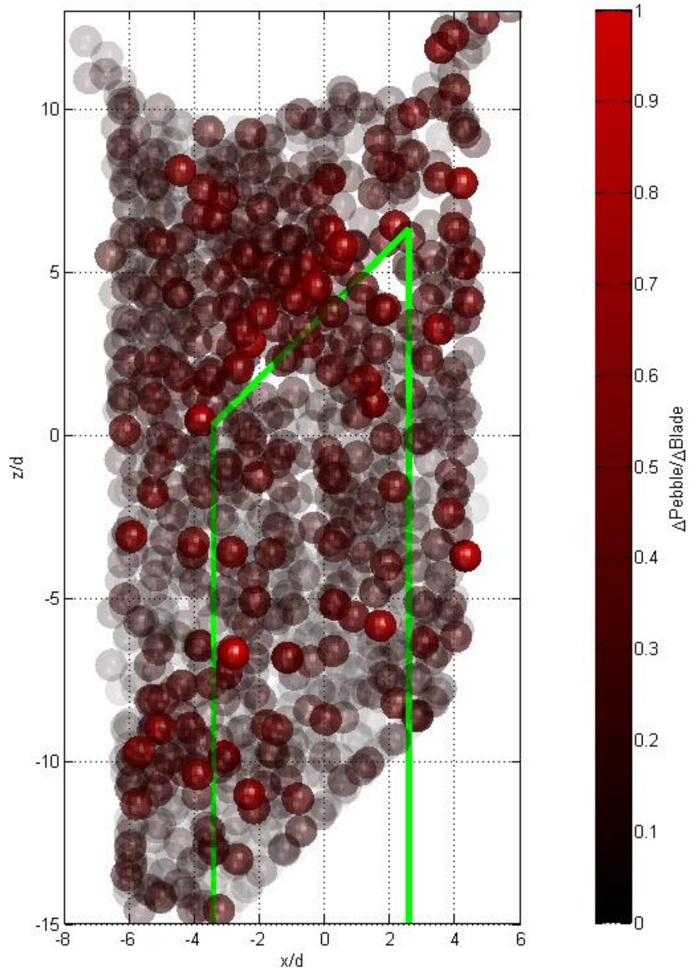
- This simply projects the solutions onto the x-ray image so the user can visually verify the results
- Various options can be found in the code for labels, image saving, module1 plotting and other "accessories"

Render_Pebbles.m



- This function plots pebbles with the option to plot the pebble equator, color and transparency
- Very helpful for post-processing modules





XPETS Module 2
Summary

User Input Summary

- Inputs found in the RUN.m file:
  - TriangleIntersection Thresholds
    - » Normalized Length
      - 0 to 1, how close to the true pin length is the possible solution? 1 is exactly the correct length
    - » Metric Spatial
      - In meters. How close are the possible solutions in 3D metric coordinates? Perfectly close together (threshold=0) is a better solution.
  - TopDownIntersection Thresholds (also uses normalized length)
    - » Similar Z (metric)
      - In meters. How close are possible solutions in the z-direction? Perfectly close together (threshold=0) is a better solution.

User Input Summary

- **Inputs found in the RUN.m file:**
  - Elimination Thresholds
    - » Silo Boundary (Theta, Max, Min)
      - If you were to take an x-ray image at the specified theta (degrees), the min and the max values (in pixels) specify left and right silo boundaries. Any solutions found to be left of min and right of max will be eliminated
    - » Contrast Threshold
      - Average contrast of solution projected onto the image must be greater than the contrast threshold
    - » Number of Valid images
      - The solution must have contrast greater than the contrast threshold in the specified number of images. 5 signifies that the solution must be validated in all 5 images
    - » Normalized overlap
      - Fraction of a pebble diameter that two pins can overlap. 1 means two pebbles cannot overlap at all (centroids must be 1 diameter apart). 0.5 means that centroids can be 1 radius apart and not be eliminated.

Output Summary

- Module 2 saves output files with outputs from each phase (1,2,3) as well as the related diagnostics files. Phase3 is the final output
- The output data structure is as follows:

$$\mathrm{PHASE3\_OUT}(:,:,n) = \begin{pmatrix} X1 & X2 \\ Y1 & Y2 \\ Z1 & Z2 \end{pmatrix}$$

- Where the 1 and 2 signify the two endpoints of a physical pin in 3-dimensions for n number of pins
- And the corresponding diagnostics data structure is:

$$\mathrm{PHASE3\_DIAGNOSTICS}(:,:,n) = \begin{pmatrix} \mathrm{Length} & P_1X_1 & P_1X_2 & S_1X & P_2X_1 & P_2X_2 & S_2X & C_1 & C_4 \\ \mathrm{Colinearity} & P_1Y_1 & P_1Y_2 & S_1Y & P_2Y_1 & P_2Y_2 & S_2Y & C_2 & C_5 \\ \mathrm{Spatial\_Similarity} & P_1Z_1 & P_1Z_2 & S_1Z & P_2Z_1 & P_2Z_2 & S_2Z & C_3 & U \end{pmatrix}$$

- Diagnostics definitions on the next slide

Output Diagnostics Format Details

$$\mathrm{PHASE3\_DIAGNOSTICS}(:,:,n)=\begin{pmatrix} \mathrm{Length} & P_1X_1 & P_1X_2 & S_1X & P_2X_1 & P_2X_2 & S_2X & C_1 & C_4 \\ \mathrm{Colinearity} & P_1Y_1 & P_1Y_2 & S_1Y & P_2Y_1 & P_2Y_2 & S_2Y & C_2 & C_5 \\ \mathrm{Spatial\_Similarity} & P_1Z_1 & P_1Z_2 & S_1Z & P_2Z_1 & P_2Z_2 & S_2Z & C_3 & U \end{pmatrix}$$

- **Length, colinearity and spatial similarity are all threshold-type variables set by the intersection algorithms**
- **P is the 3D module 1 point (from Place3D.m) that the solution originated from**
- **S is the 3D source point corresponding to P**
- **C is the average contrast of the solution in each image (1-5)**
- **U is an uncertainty parameter set by the find contrast algorithm**

DON’T PANIC

Always feel free to call 720-495-6245 or email grantbuster@gmail.com