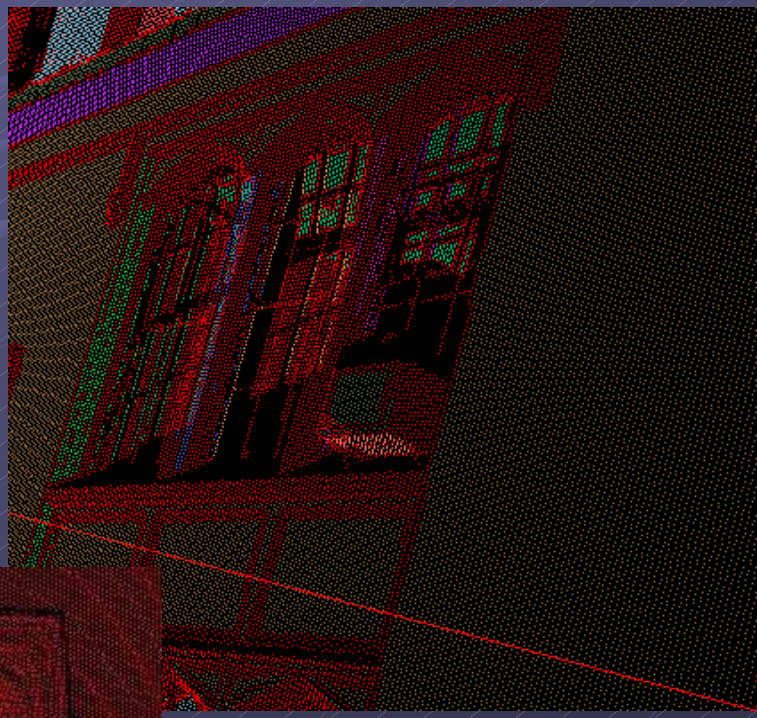
Window Filling and Junk Data Removal

Hadi Fadaifard

- The goal of this project was to fill windows/holes that are present in the 3D scans of large buildings.
  - It lead to realization of the need to remove junk data:
    - Points behind the windows
    - Points from other objects not related to the building:
      - Trees
      - Traffic lights
      - Flags

Some Failed Techniques

- ***Linear Interpolation:*** Fill the holes by interpolating in the place of missing points
  - ***Problems:***
    - Junk data inside the windows affect and distort the generated points.
    - Edges and occlusion make the result of interpolation unpleasant.

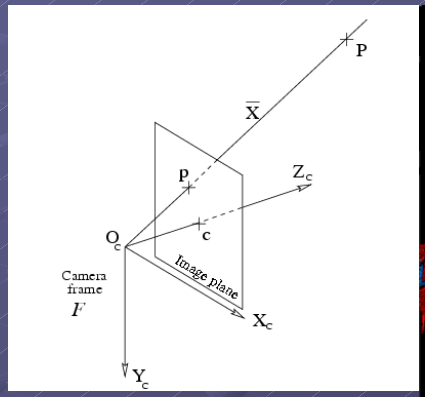


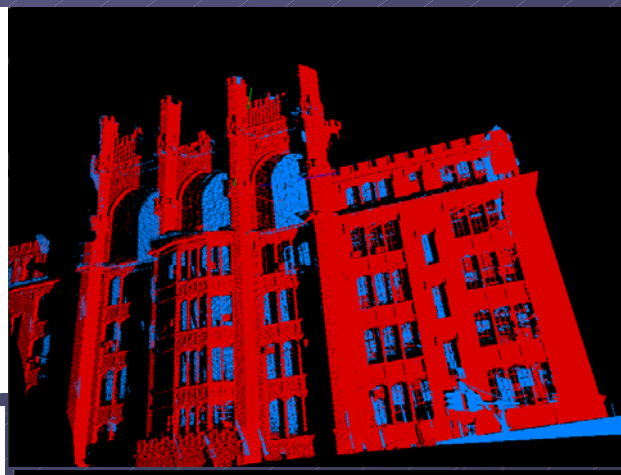
2D to 3D projection

- Fill Holes in the 2D image of the building and project the results back in 3D:
  - Do segmentation of the points in the scan and find the planes each point belongs to.
  - For each hole region, find the plane its neighbors belong to.

2D to 3D projection

- Find the intrinsic parameters of the scanner. (I assumed it could be modeled with a pinhole camera.)
- Do reverse perspective projection from 2D into 3D.

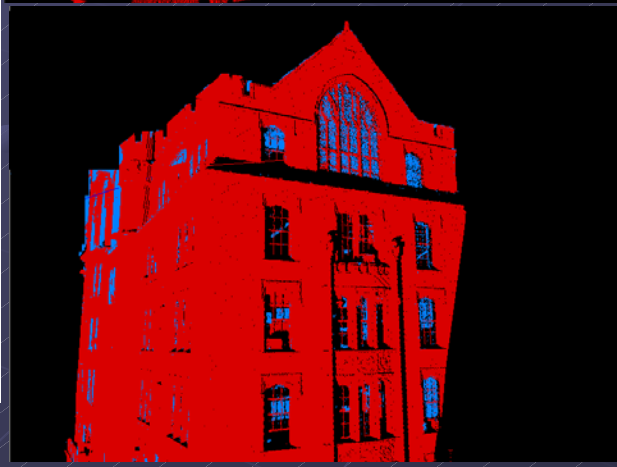




$$x = C_x + \frac{f_x X}{Z}$$

$$y = C_y + \frac{f_y Y}{Z}$$

$$\begin{bmatrix} X_0 & 0 & Z_0 & 0 \\ 0 & Y_0 & 0 & Z_0 \\ X_1 & 0 & Z_1 & 0 \\ 0 & Y_1 & 0 & Z_1 \\ \dots & \dots & \dots & \dots \end{bmatrix} \begin{bmatrix} f_x \\ f_y \\ C_x \\ C_y \end{bmatrix} = \begin{bmatrix} x_0 z_0 \\ y_0 z_0 \\ x_1 z_1 \\ y_1 z_1 \\ \dots \end{bmatrix}$$



2D to 3D projection

- The results were better than the previous techniques
- Correct projection was not obtained due to:
  - Inaccuracies in finding the destination plane in 3D because of junk data.
  - Did not take into account the large radial distortion of the scanner.

Removing Junk Data

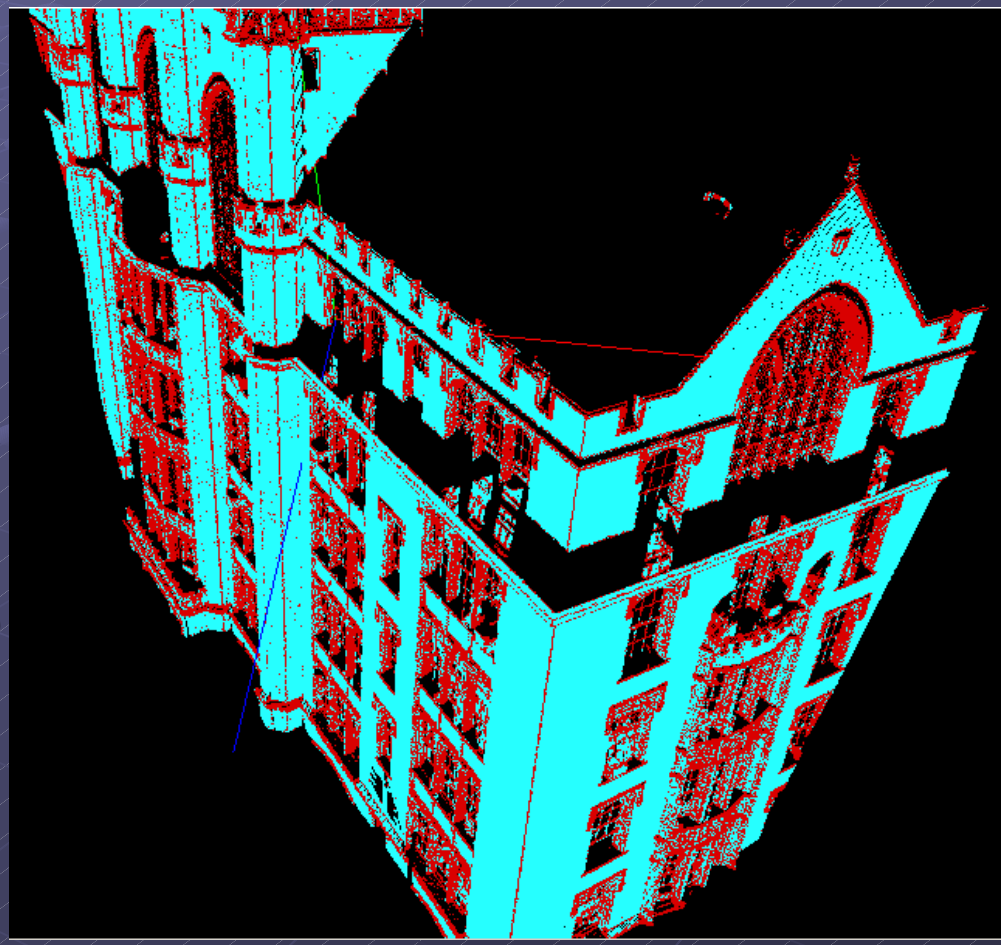
- Junk data always affect the results of window filling algorithms.
- Remove as much junk data as possible before hole filling.

Junk Removal

- ***Method I:***
  - Find the top 10-20 planes that define the building.
  - Determine the plane each point belongs to.
  - Consider the points that are not in any plane as junk. Remove them

Junk Removal

- Problems with this Method:
  - It would consider some junk data as legitimate (e.g. ceilings)
  - It would remove lots of useful data (e.g. around the window edges)



Junk Removal

- Find top 4-5 planes and remove all points that don’t belong any of the found planes AND have are behind the planes.
  - ***Problem:*** it would still regard some junk as useful data.

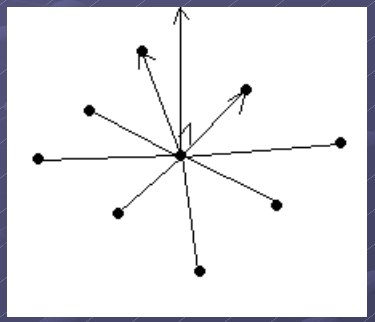
A Solution that works

- ***Algorithm:***

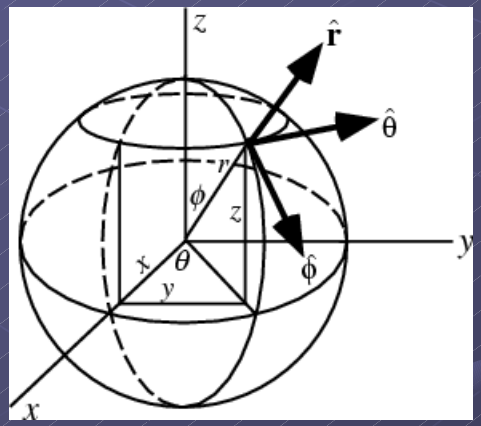
1. Find the planes that define the building
2. Do orthographic projection of all points near the plane (<t).
3. Generate an image of the projected points.
4. Use connected component algorithm to find and fill the windows in 2D.
5. Transform the filled windows back to the original 3D coordinate system.

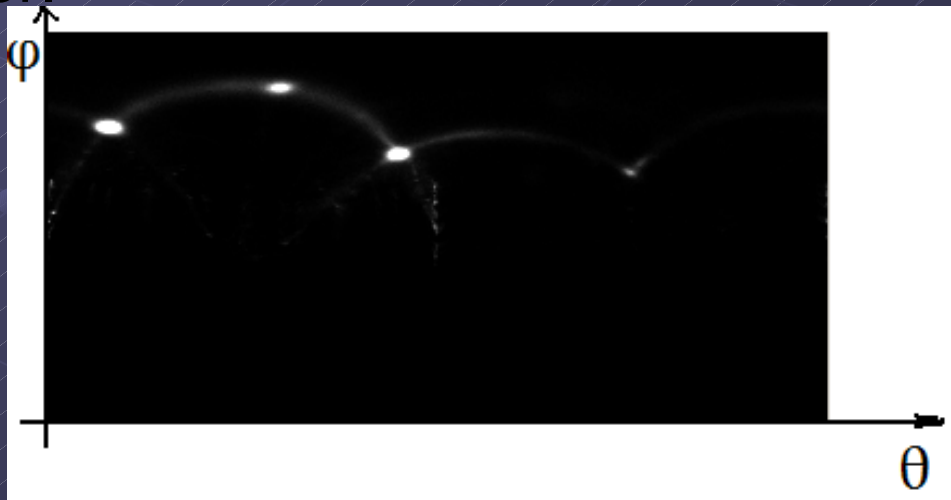
Finding the planes

- Hough transform (or Accumulator?):
  - Find the normal at each point:



  - Transform the normal to spherical coordinate system and store the θ, φ in a 2D accumulator. Generate an image of the accumulator.

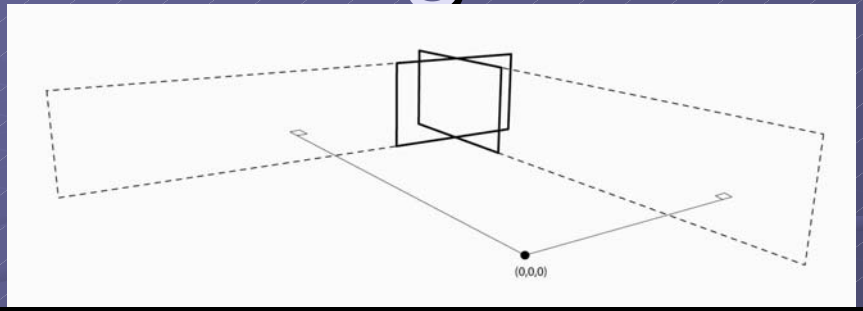


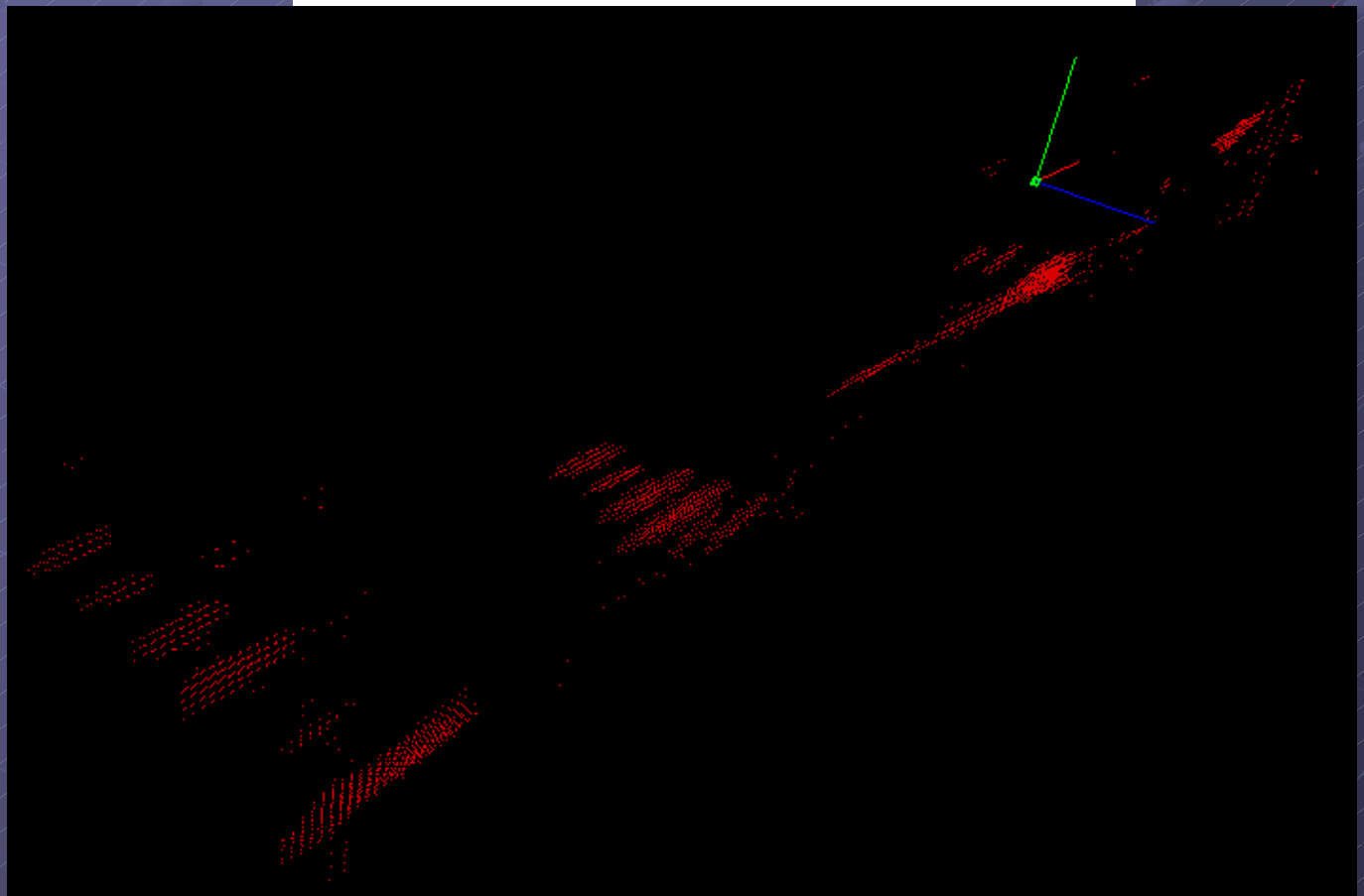


Finding Planes

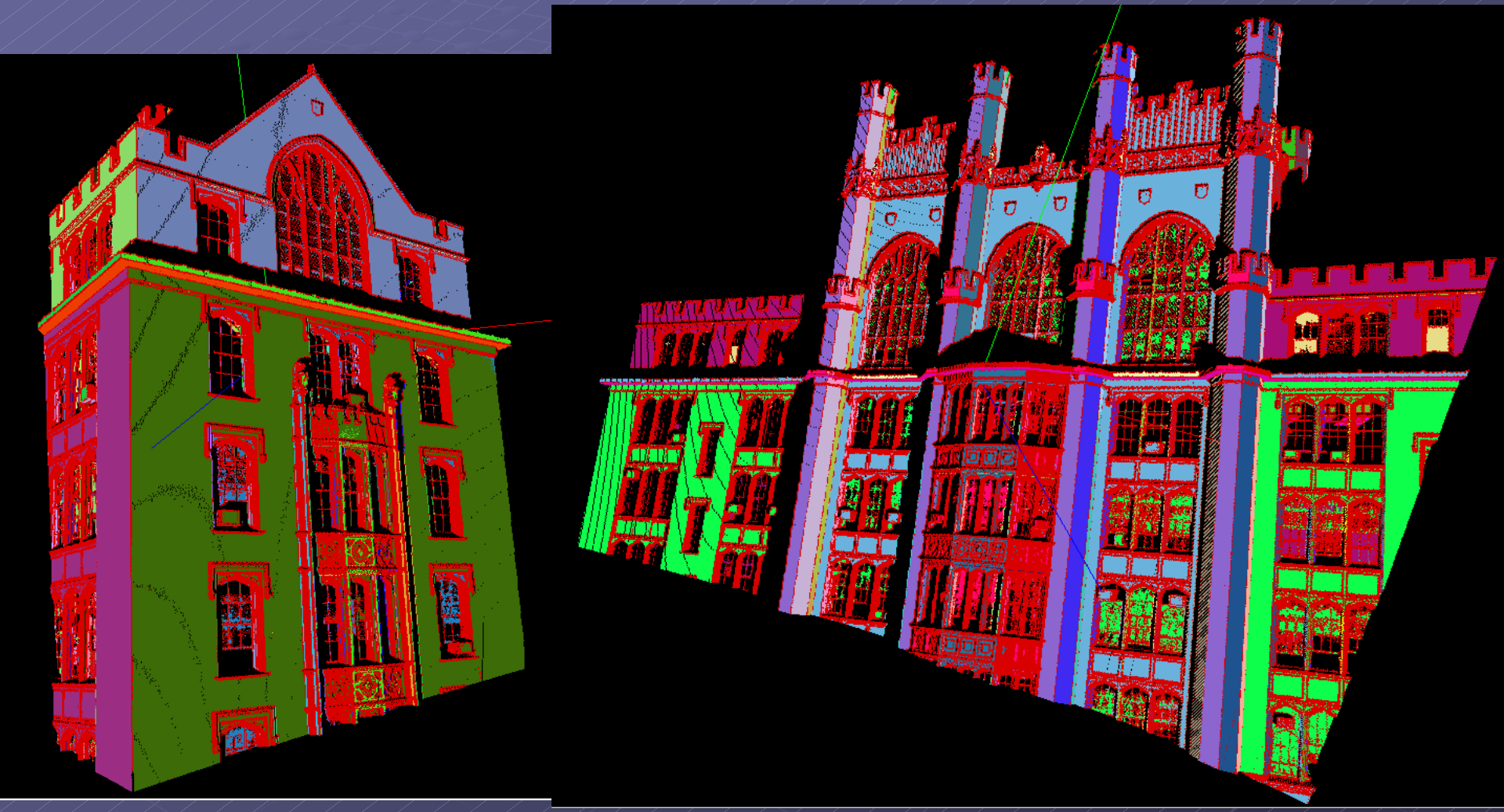
- Use connected component algorithm to find the regions. Pick top 4-5 regions. Consider the center each region as the normal you are looking for.
- Classify 3D points using the normals found above.
- Reclassify points that have the same normal to more clusters based on their distance from the origin in the direction of the normal.

Finding Planes



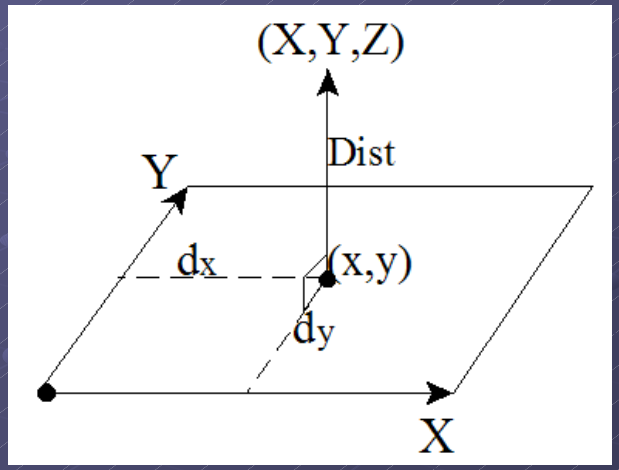


Finding Planes



Filling Windows

- Find 4 corners for each plane.
- Project points onto the bounded plane
  - Choose the resolution properly.

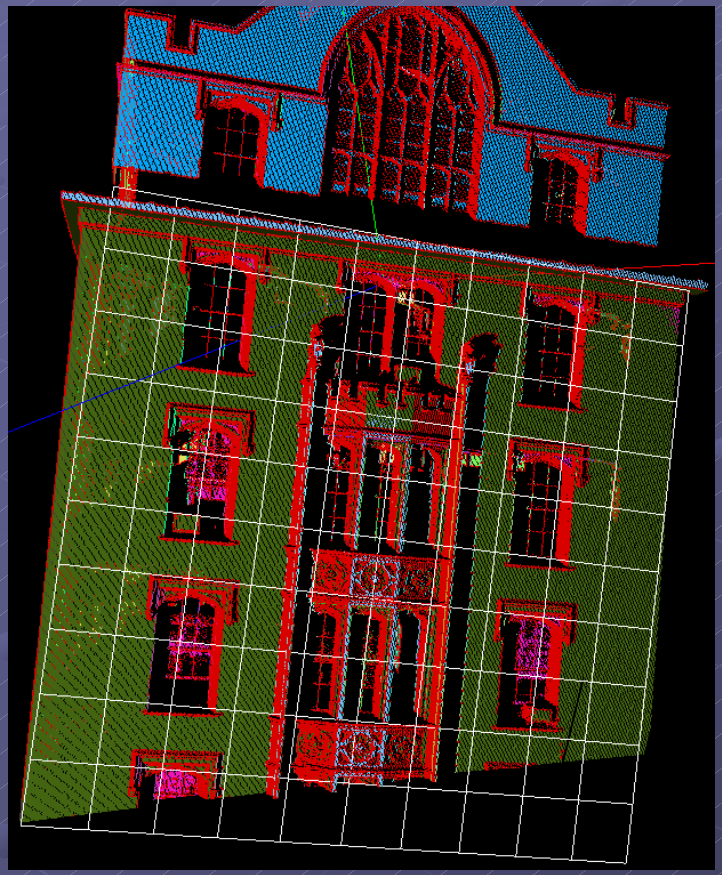


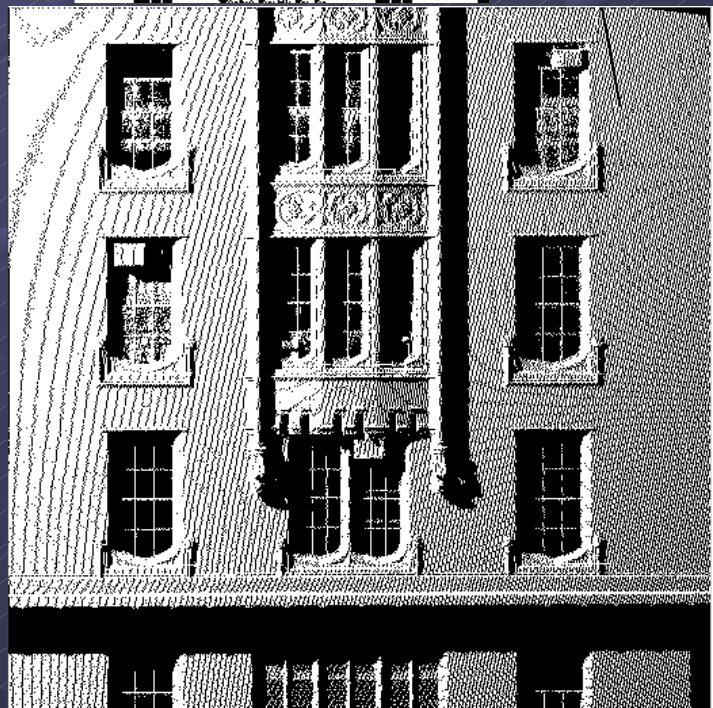
$$xScale = iWidth / fWidth$$
$$yScale = iHeight / fHeight$$

$$\begin{bmatrix} X_p \\ Y_p \\ Z_p \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} - dist \times \begin{bmatrix} N_x \\ N_y \\ N_z \end{bmatrix}$$

$$x = d_x \cdot xScale$$
$$y = d_y \cdot yScale$$

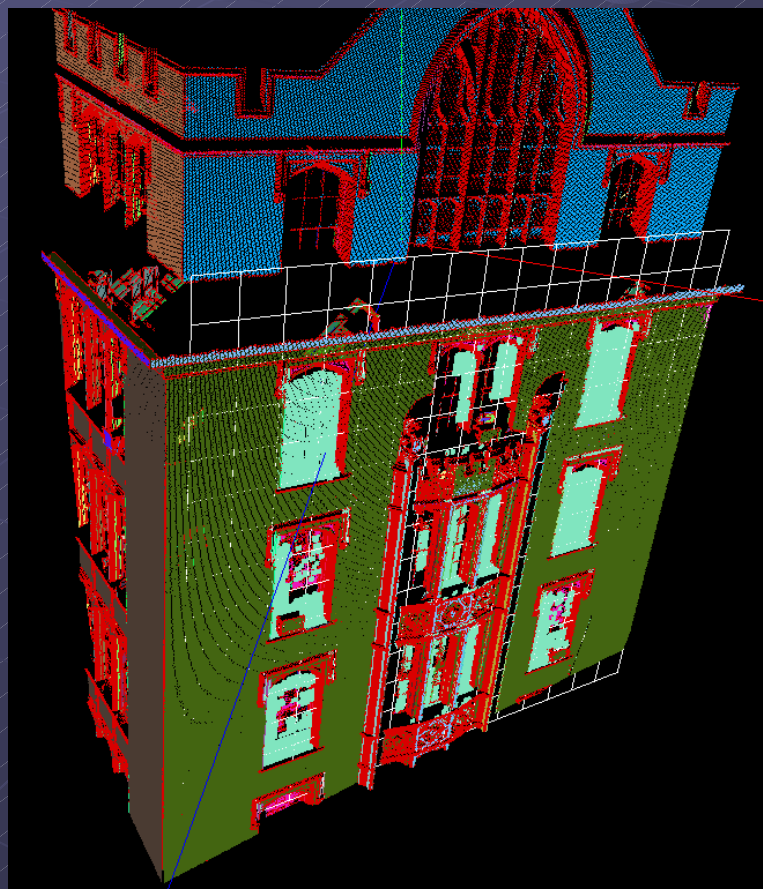
Filling Windows



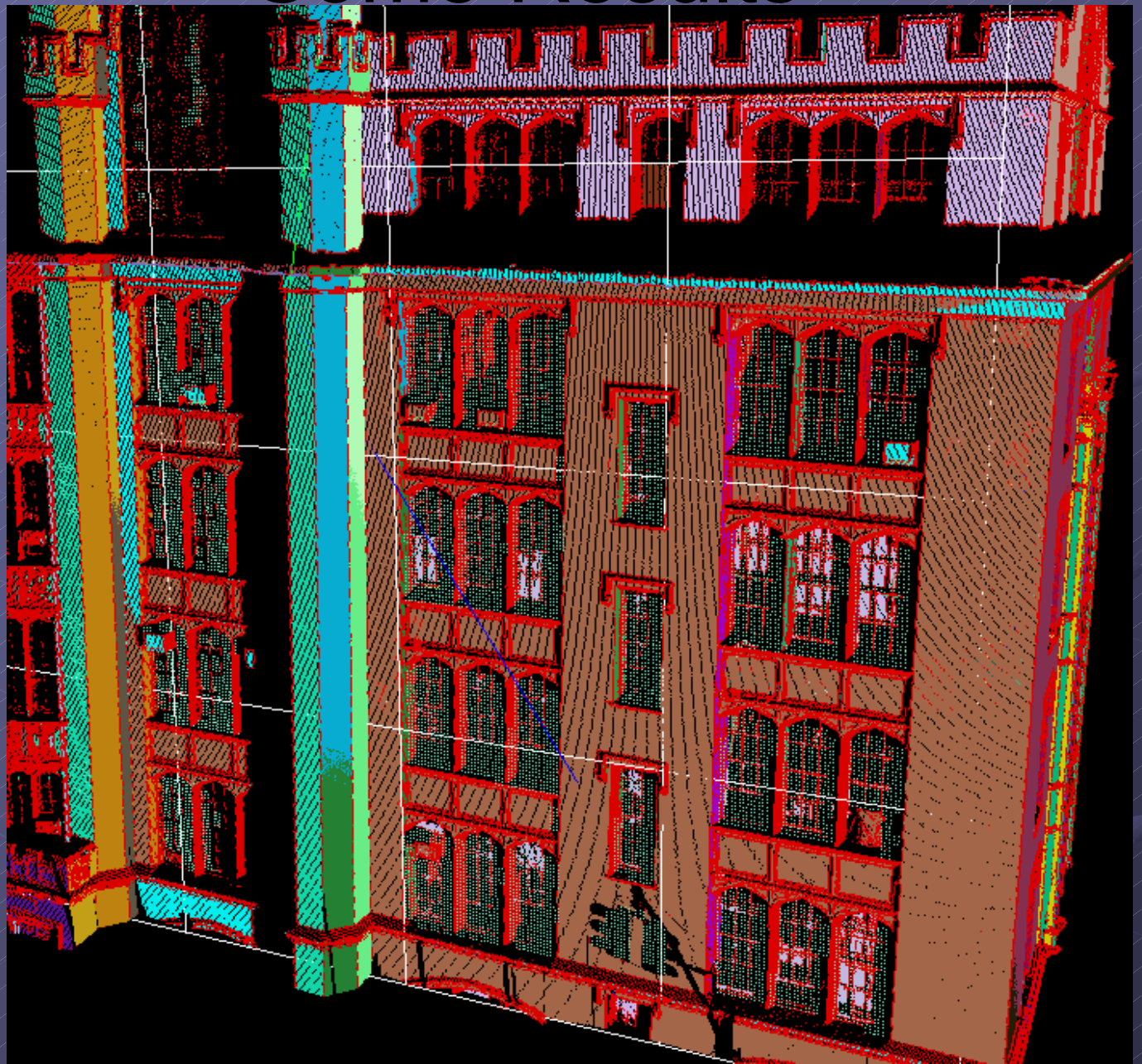


Filling Windows

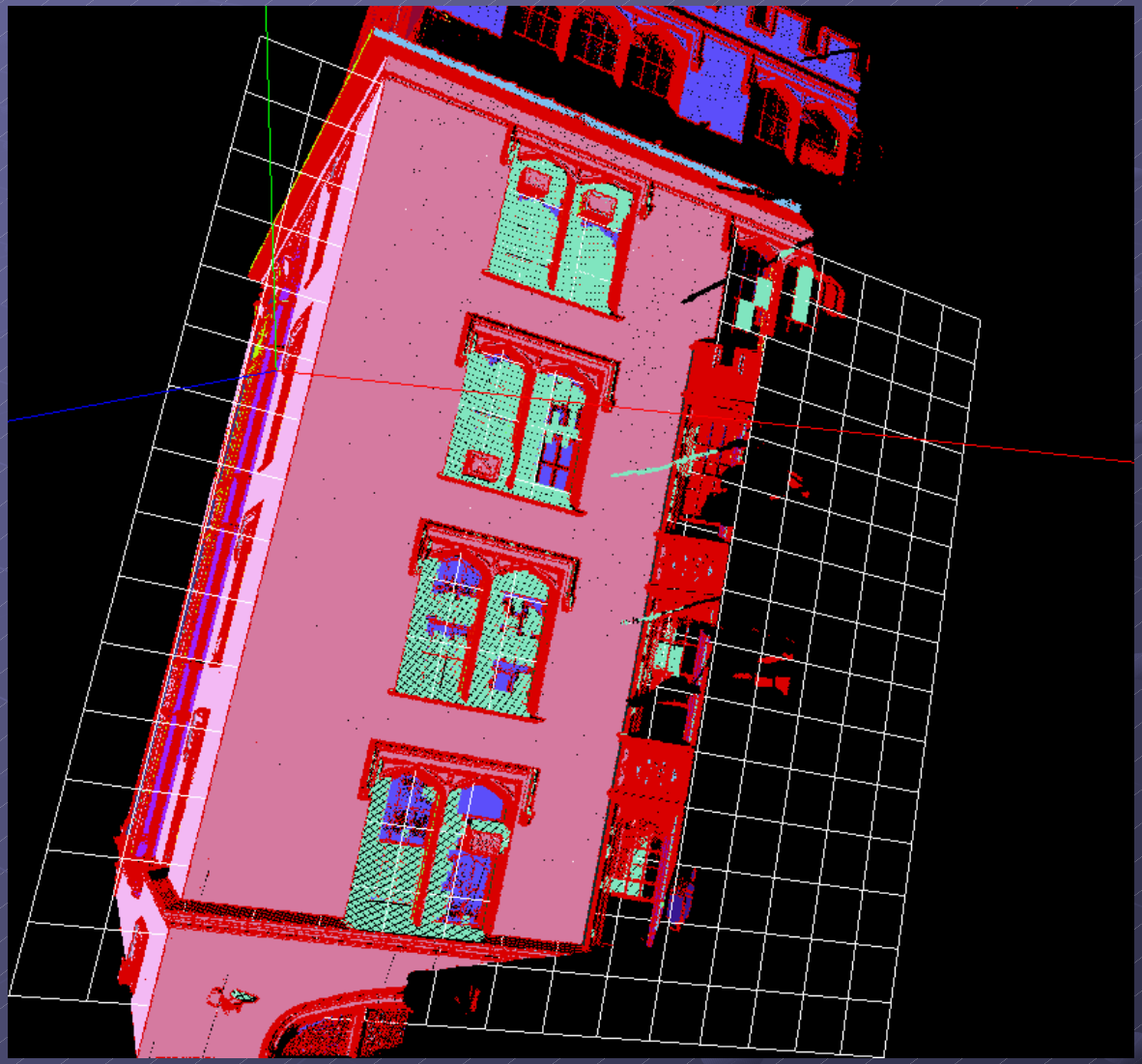
- Find the windows in the image
- Transform the new points back to 3D.



Some Results



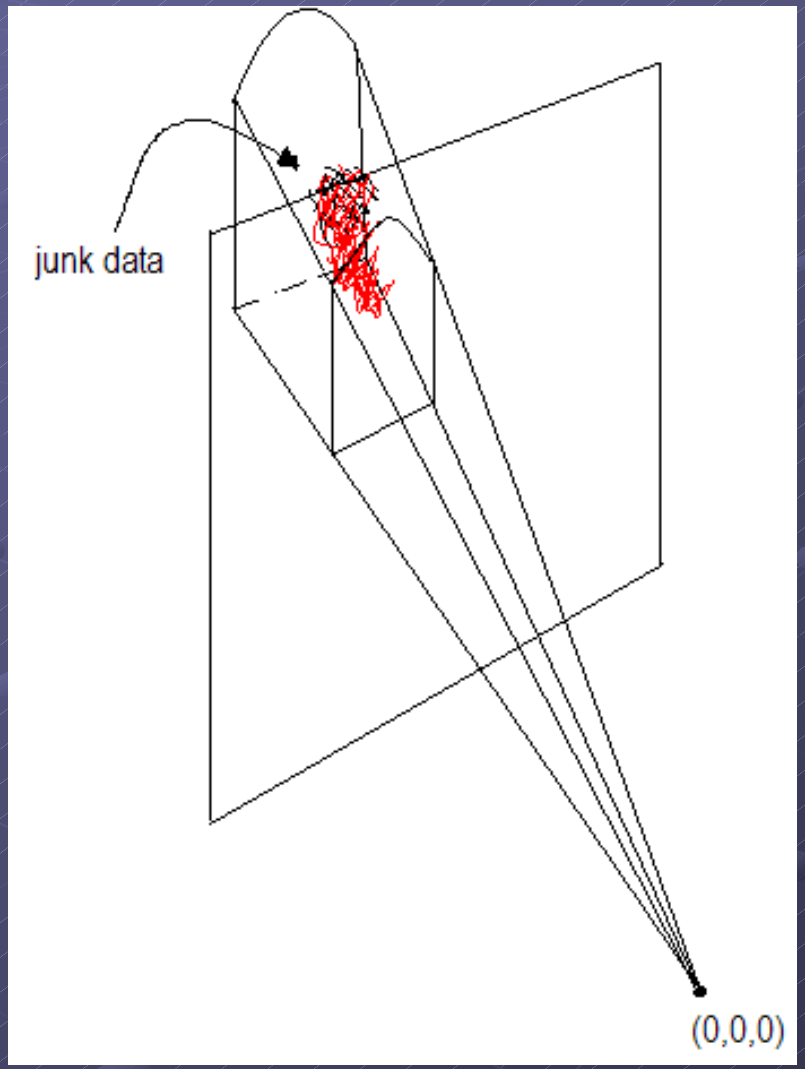
Some Results



Future Work

- Removing junk data behind the windows:
  - Find the edges of windows in the orthographic image.
  - Create a frustum with its apex at the scanner origin and its base being the boundaries of a window.
  - Remove all the data that is inside the frustum and behind the window plane.

Future Work



The Interface

- Hough pick 3D
- Window fill

Thanks to

- Professor Wolberg
- Professor Stamos
- Siavash Zokai
- Gene Yu