CS117 Final Project Progress Report for Reverse Engineer for Reconstructing 3D Game Model

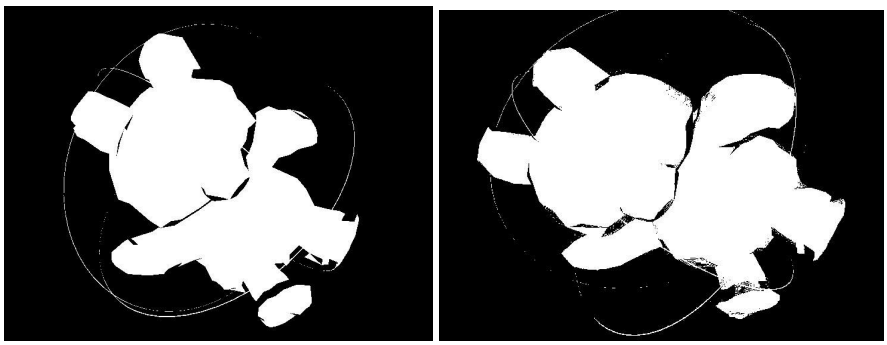
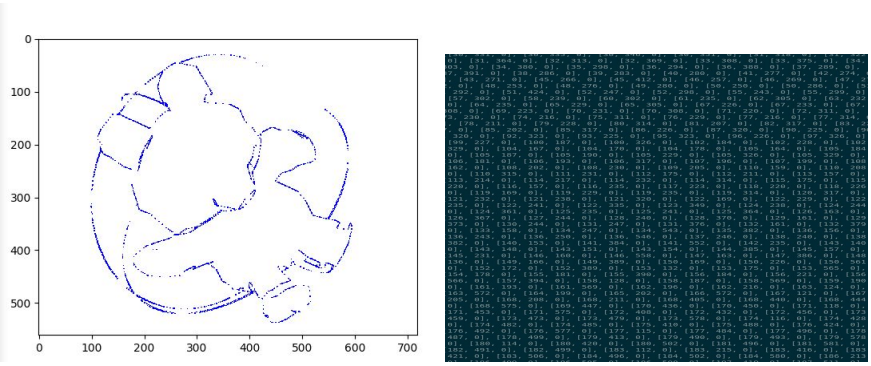
Written by: Kai Yan #24684490

The project contains three essential milestone:

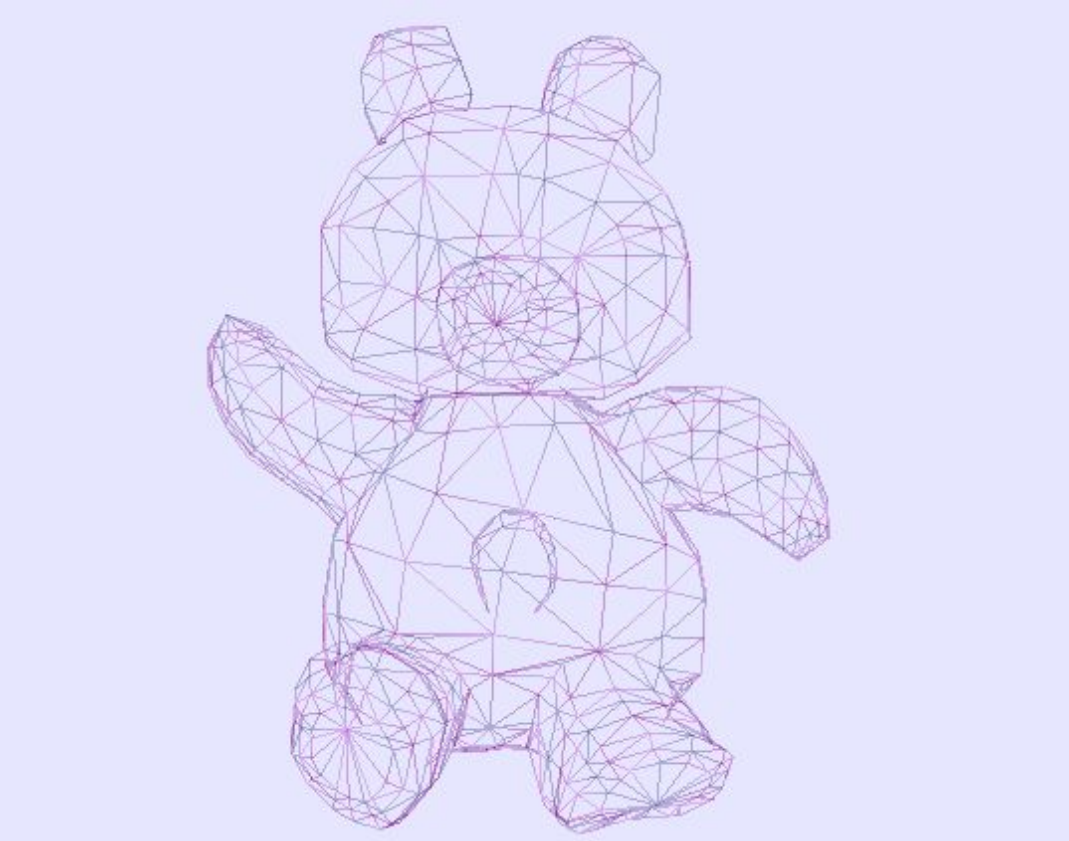
1. A python based program to do image understanding and calibration using numerous 2d image to return an data object that contains the location of every necessary 3D points.
2. Using the object that contains 3D points for meshing, denoising, merge sub meshes
3. Advanced method to optimise the mesh and coloring. Build user interface that allows user to manually edit the mesh or 3D model.
4. After effects such as shading and lighting for debugging and presentation purpose.

Current Progress:

- **Complete: milestone 1**
    - This step is one of the hardest step in the project. The three main tasks in this step are initial image loading, data detection and 3D Reconstruction.
    - The majority part of this step used python helper library such as numpy, matplotlib and cv2 for demonstrating data and files management.
    - The function for loading, showing and loading image are written in load.py, assist by cv2.imread() cv2.imshow and cv2.imwrite. The Image from PIL also contains Image.open() and Image.save()
    - For demonstration and debugging the processed image, the program used matplotlib.
    - The image loading step contains image optimization, the related files are;
        - purify.py: Use numpy do the heavy lifting for converting pixels to pure black or white. The function take a color value to determine how bright that a pixel can be white.
        - imgDenoise.py: The algorthm call fastNlMeansDenoising to deniose the image for a clear and sharp corner and edge. This increase the correctness of purify and corner detection
        - calibration.py and camera.py: camera.py contains a camera class that include every necessary parameter and matrix for camera calibration. The Calibration.py uses calibration method to return the actual calibrated image points.
    - The data detection step contains, Corner detection the related files are;
        - HarrisCorner.py: Harris Corner detection. This algorthm is the most important method to retrieve the useful point from the image. The file also include an optimation to delete the overlap point that have minor distance call fuse()

        - ShiTomasi.py: another way for corner detection and algorithm based on paper Good Features to Track. Helping to find the strongest points. This function can be used when the image is more geomatic.
        - houghT.py: Find imperfect instances of objects within a certain class of shapes by a voting procedure, represent that shape in mathematical form.
        - SIFT.py: points match between different picture and SIFT descriptors with FLANN based matcher and ratio test. (This step is in progress)
    - The difficulty is SIFT and point match, which required more optimation on code
    - **Preliminary results samples:**
    - 
    - 
    -
- **In progress: milestone 2**
    - This step use the array contains all the actual model points from milestone 1 to an 3d model. The file format is .obj
    - For construction and load the 3d model, the logic is written in uiconstruction.py
    - The user interface used pygame framework.
    - The steps of removing holes, denoising and merging the various model has not been start.
    - One of the difficulty is if the model is concave, there should be more optimization since some of the meshing step build 3d convex hull.

- **Preliminary results samples:**



- **Not Started: milestone 3 and 4**
  - These milestone will optimize the mesh by hole fixing and denoising. It will also color the final mesh.
  - The difficulty is coloring by retrieving the data from the original image. We need to find out what pixel on the original image is related to the certain point and color the model.