**Report**

**Design Choices**:

* **DENSE**: Deciding the parameters for SGBM stereo (researched and tested). As well filterspeckles integrated into SGBM by specklewindowsize and specklerange. Preprocessing done to images to reduce photometric variations between the images.
* Tested wls\_filter against normal computing of disparity map (<http://timosam.com/python_opencv_depthimage>). Much less noise, it does left to right and right to left. Obtains a hole free depth image, unlike original disparity calculations which has some holes (seen as noise). So smoothed that distances for stationary objects vary wildly – Not recommended
* Dense box calculations:
* Use of adapative histogram equalisation <https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_histograms/py_histogram_equalization/py_histogram_equalization.html>
* Method of calculating distance, average disparities using a Gaussian filter
* **SPARSE:** Implementation of Sparse Stereo. Discuss feature matching, padding, etc. Describe quantitative/qualitative comparison with it and DENSE. Thinking show images with (no of objects detected/accuracy – accuracy calculated from google maps, etc).
* Changes to YOLO – sharpening image beforehand gives more objects detected at closer range (possibly better depending on application)
* Filtering of images before and after use

**Performance:**

* Quantitative: Show examples of it working
* Qualitative: How well does it work? (Use stationary objects, google maps for distances)
  + E.g. Based on representative subset of N examples, this approach worked XX^ of the time for the detection of Y regions
  + Show it working well for detecting objects, distances

**Comparison**:

* Comparison between Sparse Stereo and SGBM stereo
  + Comparison of speed/accuracy
  + Which is more correct
  + Do they both work to some degree
  + Quantitative and Qualitative

Structure:

* Design Choices – Pre processing, disparity calculation (Gaussian kernel), distance calculation
* Performance
* Comparison between two – 2 images each. Show distances based on Google Maps and compare their results (links heavily to performance)

Possible Heuristic:

* Ignore Glass in a car (remove top half of car) (DONE)

Still to do:

* Adjust parameters for ORB object and StereoSGBM object
* V1 = Somethings ignore objects with 0 disparity at the centre