CS 6320: Computer Vision
Assignment 1

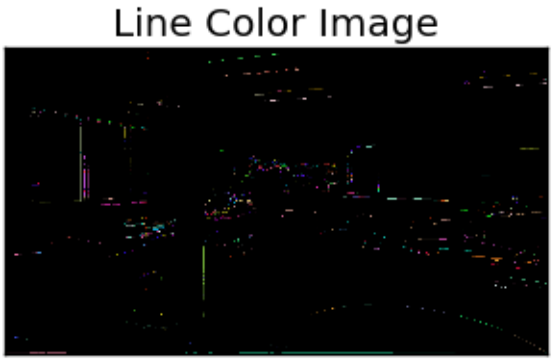
Abhinav Kumar, u1209853
(Abhinav.Kumar@utah.edu)

Q1) Line Detection (q1.py)
main function with argparse --

We randomly sample two edge pixels and compute a line equation that passes through these two points. For the chosen line equation, we check the total number of edge pixels that lie within a small threshold distance. Based on the number of points that lie on the chosen line equation, we decide to treat it as an image line or not. We iteratively find all the line segments in the image based on some threshold (i.e., each line should have at least some big edge pixels). After finding every line, we remove all the edge points lying on this line before finding the next line.

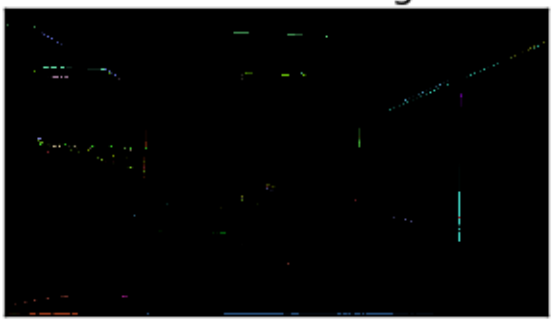
A small resolution image was also taken for the purpose of illustration. The results look much better in smaller image.

Larger Resolution images – different parameters used (since now a line corresponds to a greater number of pixels)

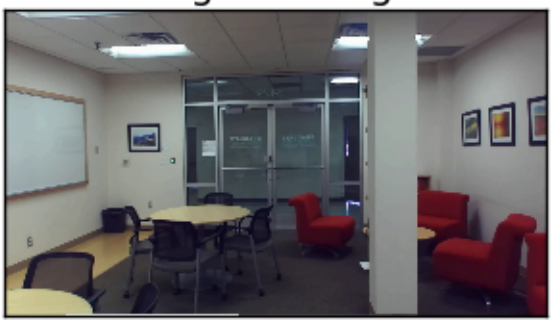


Original Image

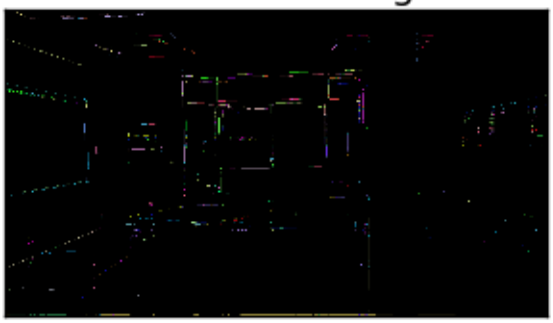
Line Color Image



Original Image



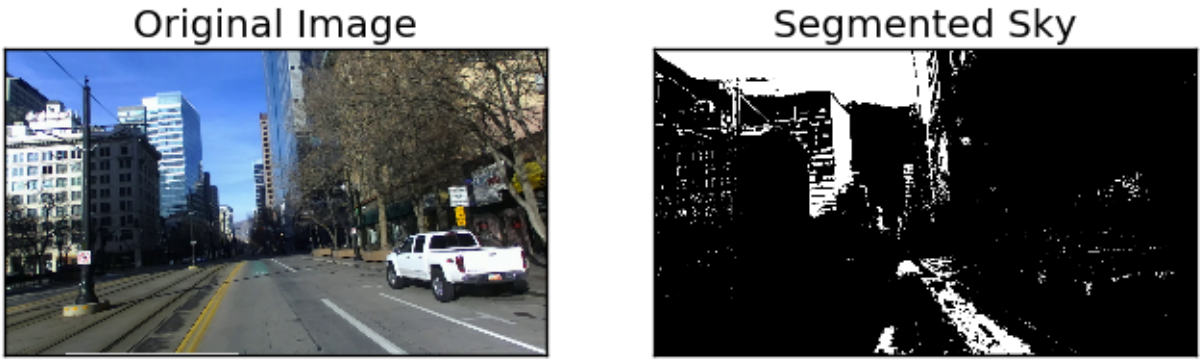
Line Color Image

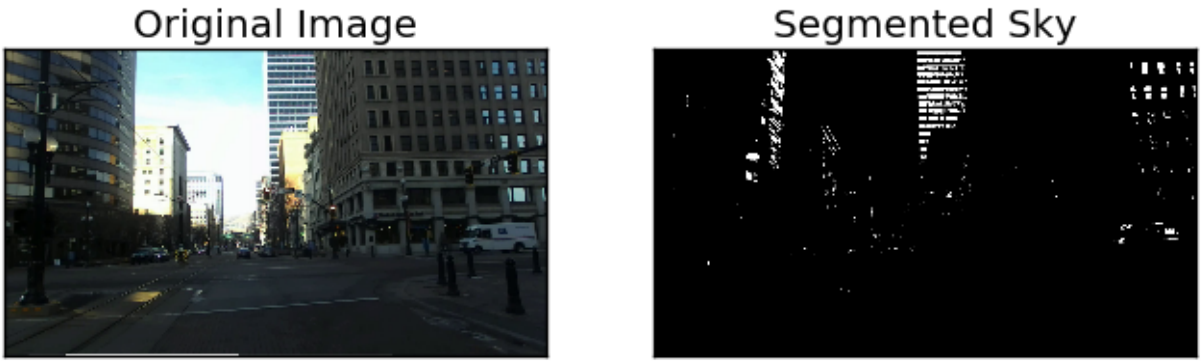


Q2) Sky Segmentation (q2.py)
main function with argparse --

Algorithm :
Iterate over all pixels

```
if ( r > R_MIN and r < R_MAX and g > G_MIN and g < G_MAX and b > B_MIN and b < B_MAX):
        #print("sky found")
        sky[i,j,0] = 255
        sky[i,j,1] = 255
        sky[i,j,2] = 255
```





The algorithm is not perfect as it sometimes segments out buildings instead of sky and their glasses because of the same color.

Q3) Disparity Map Calculation (q3.py)
main function with argparse --

Algorithm :
Iterate over all the pixels

Slide along the x axis of the right images and get different patches. The patch with displacement which gives the highest correlation is the best disparity for that pixel

WIN_SIZE for the first two images are kept at 5. The third image disparity map was very noisy since the images are really big and the certain patches are really dark. The WIN_SIZE was increased to 51 for the third image and then we get the correct disparity map.

Q4) 3D Reconstruction

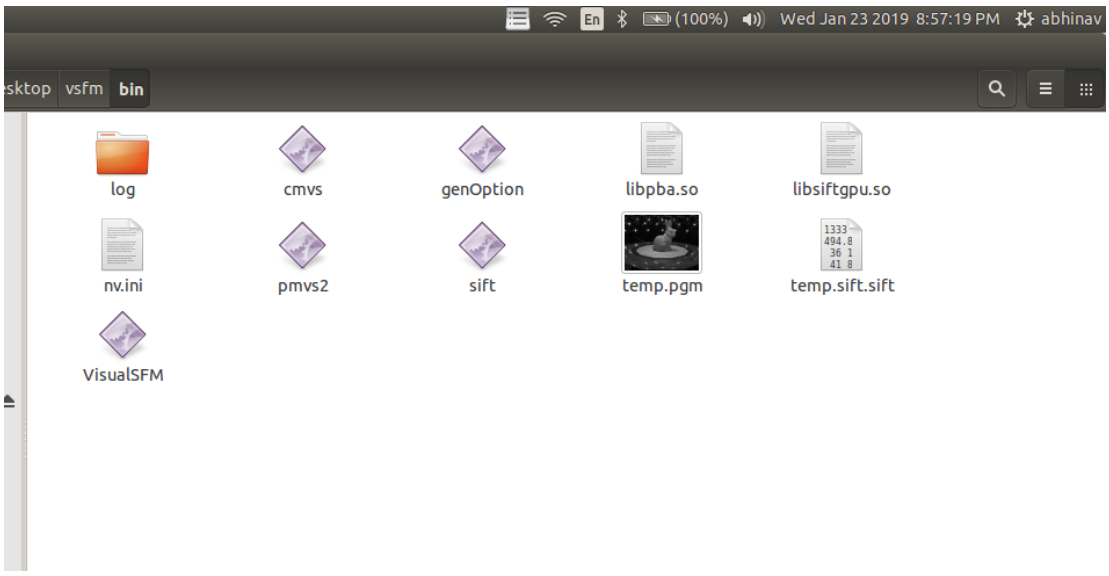
Installation of VSFM in Ubuntu 14.04

The installation proceeds through the following steps mentioned at
http://www.10flow.com/2012/08/15/building-visualsfm-on-ubuntu-12-04-precise-pangolin-desktop-64-bit/

```
sudo apt-get install libgtk2.0-dev libdevil-dev libboost-all-dev libatlas-cpp-0.6-dev
libatlas-dev imagemagick libatlas3gf-base libcminpack-dev libgfortran3 libmetis-edf-dev
libparmetis-dev freeglut3-dev libgsl0-dev
sudo apt-get install libglew-dev


# Install VSFM
# Download from http://ccwu.me/vsfm/download/VisualSFM_linux_64bit.zip
unzip VisualSFM_linux_64bit.zip
cd vsfm
make

############################################################################
# SIFTGPU
############################################################################
# SiftGPU is no longer hosted on UNC website
git clone https://github.com/pitzer/SiftGPU.git
cd SiftGPU/
make
cp bin/libsiftgpu.so ~/Desktop/vsfm/bin

############################################################################
# If SIFT GPU is not working otherwise skip to Multicore BA or PBA section
############################################################################
# Download http://www.cs.ubc.ca/~lowe/keypoints/siftDemoV4.zip file
# Unzip to get the SIFT binary and copy it into vsfm folder
cp ~/Desktop/siftDemoV4/sift ~/Desktop/vsfm/bin/

############################################################################
# Multicore Bundle Adjustment or PBA
############################################################################
Download and unzip pba_v1.0.4.zip
In “pba/src/pba”, edit “SparseBundleCU.h” and “pba.h” by adding this one line to the top
of each file and add
#include <stdlib.h>

make in “~/vsfm/pba” directory.

# if GPU

# else copy the no_gpu so and rename to gpa
cp ~/Desktop/pba/bin/libpba_no_gpu.so ~/Desktop/vsfm/bin/libpba_no_gpu.so
mv ~/Desktop/vsfm/bin/libpba_no_gpu.so ~/Desktop/vsfm/bin/libpba.so


############################################################################
# Multicore Bundle Adjustment or PBA
############################################################################
tar xf pmvs-2.tar.gz
cd pmvs-2/program/main/
cp mylapack.o mylapack.o.backup
make clean
cp mylapack.o.backup mylapack.o
make depend
make

############################################################################
# Graculus Installation
```

```
###############################################################################
# Download graclus1.2.tar.gz from
http://www.cs.utexas.edu/users/dml/Software/graclus1.2.tar.gz
# edit “Makefile.in” to set “-DNUMBITS=64"
make

###############################################################################
# CMVS Installation
###############################################################################
tar xf cmvs-fix2.tar.gz
cp pmvs-2/program/main/mylapack.o cmvs/program/main/

# Edit “cmvs/program/base/cmvs/bundle.cc” by adding these includes at the top of the
file:
  #include <vector>
  #include <numeric>

# Now edit “cmvs/program/main/genOption.cc” by adding this include statement at the top:

#include <stdlib.h>

# Edit the make file of cmvs

# Ensure a hash is there each Your * *
#Your INCLUDE path (e.g., -I/usr/include)

YOUR_INCLUDE_PATH =

#Your metis directory (contains header files under graclus1.2/metisLib/)
YOUR_INCLUDE_METIS_PATH = -I/home/scott/vsfm/graclus1.2/metisLib

#Your LDLIBRARY path (e.g., -L/usr/lib)
YOUR_LDLIB_PATH = -L/home/scott/vsfm/graclus1.2

make

cp cmvs /home/abhinav/Desktop/vsfm/bin/
cp pmvs2 /home/abhinav/Desktop/vsfm/bin/
cp genOption /home/abhinav/Desktop/vsfm/bin/
```

In total there should be following folders in the vsfm/bin/directory
- cmvs
- genOption
- libpba.so
- libsiftgpu.so / sift
- pmvs2
- VisualSFM



Launching VSFM

```
export PATH=$PATH:/home/abhinav/Desktop/vsfm/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/abhinav/Desktop/vsfm/bin
/home/abhinav/Desktop/vsfm/bin/VisualSFM
```

Choose the disable GPU option everytime you open vsfm (in case you do not have GPU cards) by

Tools --> Enable GPU ---> disable siftGPU

File --> Open+ Multiple Images --> Choose the images

Click on Compute 3D Reconstruction Icon (blue and red arrow) on the standard toolbar

```
#sift count:  5479

28: im_0020
image_size:  1280x1024
#sift count:  5381

Loading image pixel data ...done in 2s

Run full 3D reconstruction, begin...
170 pairs have two-view models
226 pairs have fundamental matrices
Initialize with 0028 and 0027
185 3D points initialized from two images
PBA library version = 105
PBA: 185 3D pts, 2 cams and 370 projs...
PBA: 0.121 -> 0.058 (1 LMs in 0.02sec)
Focal Length : [1228.800]->[1223.752]
Focal Length : [1228.800]->[1229.161]

############################
#3: [0026] sees 223 (+194) 3D points
Estimated Focal Length [1550][1.51N]
# 467 projs (165 pts and 0 merges)
PBA: 350 3D pts, 3 cams and 837 projs...
PBA: 1.187 -> 0.250 (13 LMs in 0.06sec)
```

You should see a model being reconstructed. Once the reconstruction has been done, the same would be shown in the log window.

Saving Model

SfM --> Save NView Match --> your_model_name.ply

ply file could be viewed in other softwares such as meshlab

For saving a view of the model

Tools --> Save Current View --> your_model_name_view.jpg

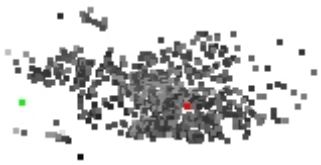
Datasets used -

- Cat from http://www.aoki.ecei.tohoku.ac.jp/mvs/cat_images.zip
- Bunny from https://vision.in.tum.de/old/data/bunny_data.tar.gz

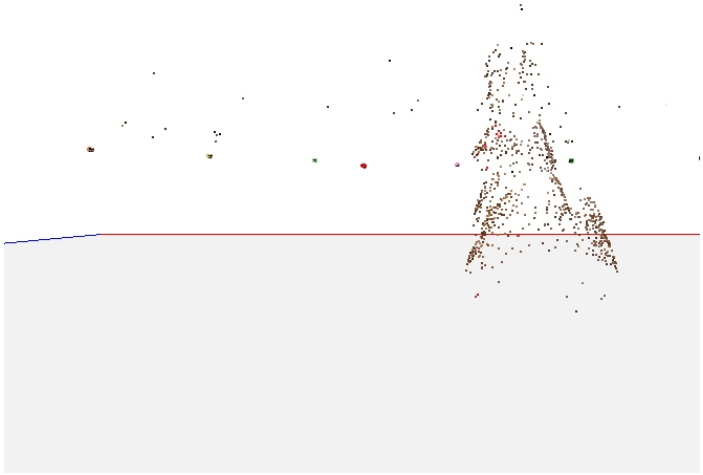
Results -

We get better results when we get more images

View of cat



View of bunny



The outputs were written to ply file and the same were visualised in meshlab.