Assignment 3

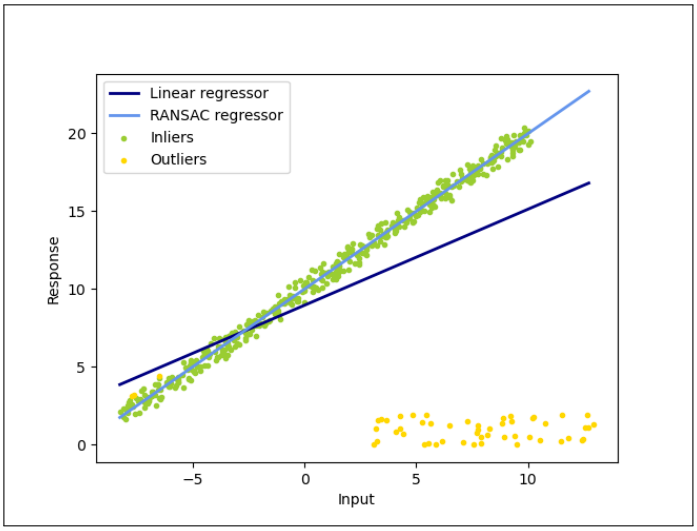
Computer Vision ETH, Fall 2021

Quentin Guignard

3th December, 2021

2 Fitting

Figure 1: Result of applying the RANSAC and the least square algorithms to the data



2.1.3 Results *For k, b, write down the ground truth, estimation from least-squares and estimation from RANSAC in the report.*

```
Estimated coefficients (true, linear regression, RANSAC):
1 10 0.6159656578755459 8.96172714144364 0.9972406479640572 10.002577853989651
```

3 Multi view stereo

3.2.2 Differentiable Warping *For k, b, write down the ground truth, estimation from least-squares and estimation from RANSAC in the report.*

The equation as described in the PatchmatchNet article[1]:

$$p_{i,j} = K_i \cdot (R_{0,i} \cdot (K_0^{-1} \cdot p \cdot d_j) + t_{0,i})$$

Which can be rewritten as:

$$p_{i,j} = K_i \cdot (R_i R_0^{-1} \cdot (K_0^{-1} \cdot p \cdot d_j) + t_{0,i})$$

or

$$p_{i,j} = (K_i \cdot R_i)(R_0^{-1} \cdot K_0^{-1}) \cdot p \cdot d_j + t_{0,i})$$

and

$$p_{i,j} = (K_i \cdot R_i)(K_0 R_0)^{-1} \cdot p \cdot d_j + t_{0,i})$$

Which intuitively means that given $p = (x, y, 1)$, we scale it to by dj which is equivalent at setting the point in the image space at depth dj and then projecting back the point to the camera space by K_0^{-1} to finally rotate back the point to the origin (I did not developed for t_0. We then transform the pixel (the other way around) to the image plane of the source i. In the assignment, as I understood, the projections $P_j = K_j[R_j|t_j]$ are multiplied already so we are left to apply the following:

$$p_{i,j} = R \cdot p_{x,y,1} \cdot d_j + t$$

Where R and t produces the transformation as defined above.

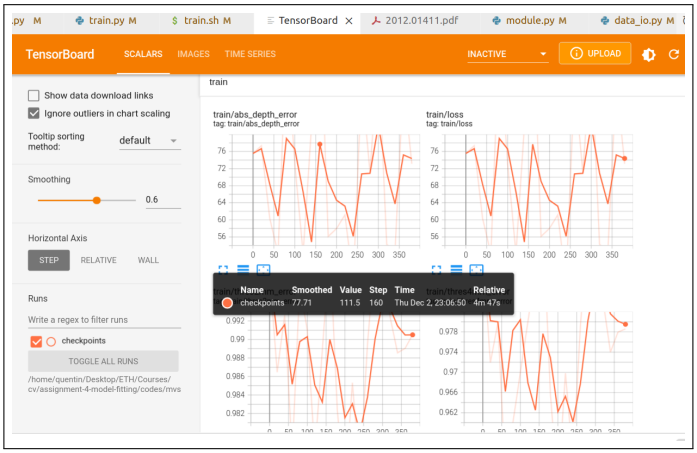
3.3 Training

The training did not went well. First, I have only a weak GPU which obligated me to lower the batch size to 1 because of memory limitations (still it would takes hours to computed as with my CPU). Secondly, there is certainly an error in the code as the loss was fluctuating and the output images where white (or slightly greyish).

We therefore don't have the results of the MVS for this part.

[1] Fangjinhua Wang, Silvano Galliani, Christoph Vogel, Pablo Speciale, Marc Pollefeys, In *PatchmatchNet: Learned Multi-View Patchmatch Stereo*

Figure 2: Tensor board screen shot of the learning try



3.4.1 Test *Explain what geometric consistency filtering is doing in the report*

Geometric consistency filtering is a method to get remove points that do not have a consistent depth matching for a source image regarding to the reference. As I understood the code, we take a point (pixel) in the reference image plane and use the estimated depth of the reference to transform the point into one of the source image. From here, we apply the same transformation backward, we re-project the point to the reference image plane using the source image estimated depth. We then compare the original point and the re-projected point. If they match enough we keep them otherwise we mask them. This is a way of getting rid of inconsistent depth relations through the images.

3.4.2 Test *For all the scenes, visualize (visualize ply.py) and take screenshots of the point clouds in Open3D.*

3.5.1 Questions *In our method, we sample depth values, dj* $D_{j=1}^{d_j}$*, that are uniformly distributed in the range [DEPTH MIN, DEPTH MAX]. We can also sample depth values that are uniformly distributed in the inverse range [1/DEPTH MAX, 1/DEPTH MIN]. Which do you think is more suitable for large-scale scenes?*

The range [DEPTH MIN, DEPTH MAX] is more suitable since the sampled depth values are more far to the camera as opposed to [1/DEPTH MAX, 1/DEPTH MIN] which would yield sampled depth values very close to the camera. If we try to reconstruct a large scene with depth samples very close to the

camera, it is possible that the depth on differents views will not correlate anymore because the features of the images could be at higher depths and would not match.

3.5.2 Questions *In our method, we take the average while integrating the matching similarity from several source views. Do you think it is robust to some challenging situations such as occlusions ?*

Yes, because if we didn't averaged, the occluded parts would impact the learning as there would be a unexpected gap between the target and some of the views. Averaging allows weighting this gap and still get results in the occulded parts.