CS284 HW5 Report: Surface Reconstruct

Hongchen Cao
2019533114
caohch1@shanghaitech.edu.cn

***Abstract*—This report covers a basic description of the implementation, performance, and observations on the results of HW5.**

I. Project

A. Description of the algorithm

The algorithm can be summarized as follow:

ALGORITHM 1: Compute TSDF

for $i = 0 : voxelNum$ **do**
  for $j = 0 : voxelNum$ **do**
    for $k = 0 : voxelNum$ **do**
      $x_g, y_g, z_g = voxel2glb(i, j, k)$ # Get global coordinate
      $x_c, y_c, z_c = glb2cam(x_g, y_g, z_g)$ # Get camera coordinate
      $u, v = cam2img(x_c, y_c, z_c)$ # Get image coordinate
      $\lambda = get\_lambda(u, v)$
      # Compute sdf
      $sdf = \frac{1}{\lambda} * ||[x_g, y_g, z_g] - camPose^{-1}[:3, 3]||_2 - rawDepth$
      # Compute tsdf
      if $sdf > 0$ **then**
        $tsdf = min(1.0, \frac{sdf}{truncationDis})$
      else
        $tsdf = max(-1.0, \frac{sdf}{truncationDis})$
      end if
      # Update
      $newTsdf = \frac{preWeight * preTsdf + currWeight * tsdf}{preWeight + currWeight}$
      $newWeight = preWeight + currWeight$
    end for
  end for
end for

ALGORITHM 2: get_lambda

$uv = [u, v, 1]$ # image coordinate
$\lambda = ||K^{-1} * uv.T||_2$
return λ

ALGORITHM 3: cam2img

$cam = [xc, yx, zc]$ # image coordinate
$vu = K^{-1} * cam.T$
$vu = (vu/vu[2])[:2]$
return $\lfloor vu \rfloor$

I use *python* to implement the algorithm and visualize the result.

B. Structure of the project

CS284_hw5.pdf is the introduction for HW5.
main.py is the implementation.
CS284_hw5_data contains raw data.
result.ply is the result.

C. Dependencies

python == 3.6.13
matplotlib == 3.3.4
numpy == 1.19.5
scikit-image == 0.17.2
opencv-python == 4.5.4.58

D. Instructions

Run `python main.py`

II. Performance

A. Efficieny

Voxel num: $20 \times 20 \times 20 = 8000$, time consumed: $48.57540s$.

Voxel num: $40 \times 40 \times 40 = 64000$, time consumed: $74.79588s$.

Voxel num: $100 \times 100 \times 100 = 1000000$, time consumed: $516.67956s$.

The time calculated includes the process of creating data, the main calculating part, printing, saving and so on.

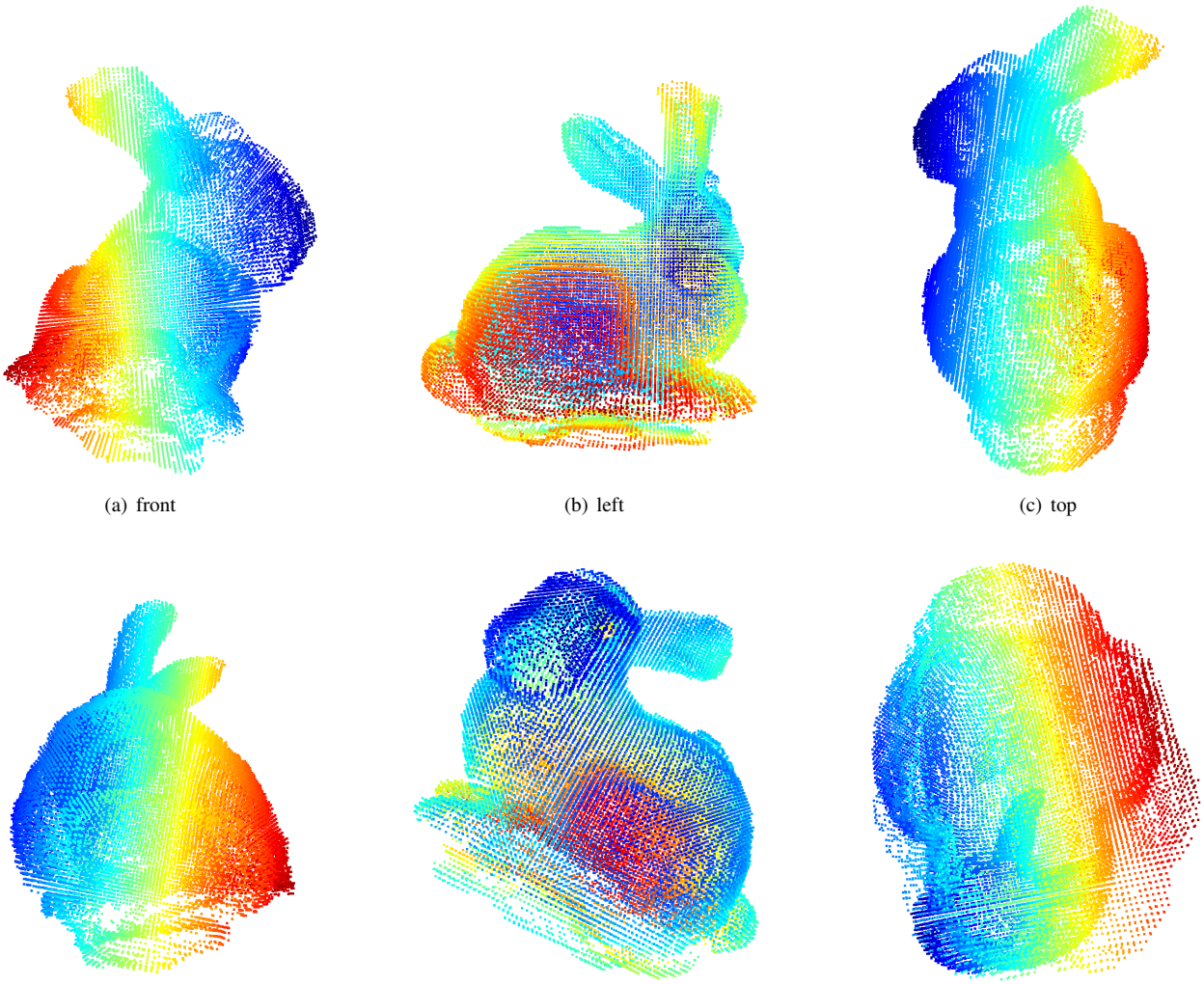
B. Visualization

See Fig. 1

III. Q&A

A. How to improve the results

To improve accuracy, we can increase the number of voxels or try more values of truncation distance and select the best one. To improve efficieny, we can decrease the number of voxels.



(a) front (b) left (c) top

(d) rear (e) right (f) bottom

Fig. 1: Visualization