UNIVERSITY OF HAIFA

אוניברסיטת חיפה جامعة حيفا

RT Systems Lab

Weekly Report number (2)

Contact Us

Julian Ewaied 215543497 Namir Ballan 326165156

8/6/2023

Contents

This Week's Tasks

- Running on a full room (360 degrees)
- Testing lower fps rates.
- Testing different height approximation functions.
- Testing different maximal heights
- Moving to Real Time setup.

Tasks Done

Running on full room

We started by writing the path in python, which we done using the existing wrappers Bar offered us. We got 6 h264 files, 6 csv height files, and 6 npy files. To ease our mission, we updated the python code to automatically convert whole .npy files directory into a new directory of the files with the same names. Next up, we wrote the main function as it follows:

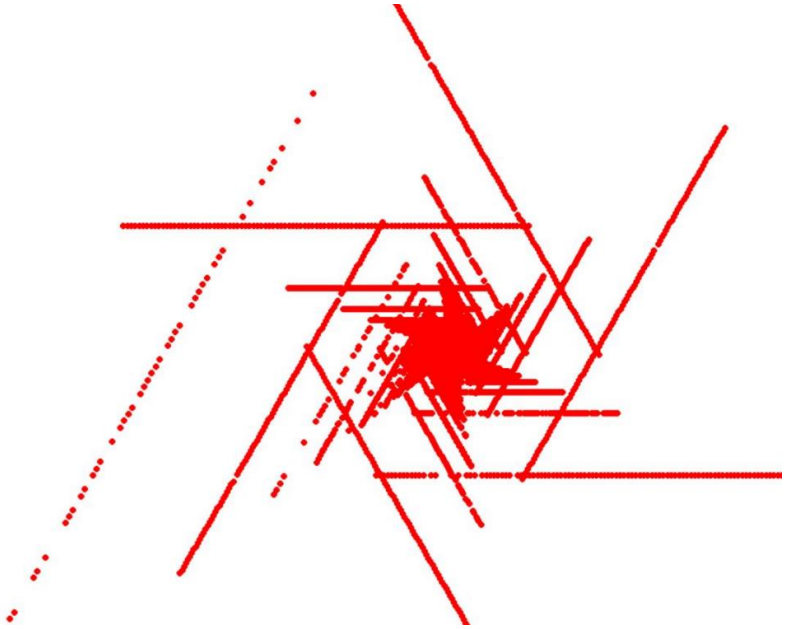
```
void continuize(vector<double>& heights)
{
    int i = 1, j = 0;
    while (i < heights.size())
    {
        while (i < heights.size() && heights[i] == heights[j]) i++;
        if (i == heights.size()) break;
        double d = heights[i] - heights[j];
        double diff = d / (i - j);
        for (int k = j + 1; k < i;k++)
            heights[k] = heights[k - 1] + diff;
        j = i;
    }
}

vector<Eigen::Vector3d> extractPoints(string path, string heights_path,int angle)
{
    auto motionVectors = importMV(path);
    CSVFile height_file(heights_path, NUM_FRM);
    height_file.openFile();
    auto heights = height_file.readColumn();
    auto centers = getCenters();
    Analyzer analyzer(fx, fy, cx, cy);
    vector<Eigen::Vector3d> points;
    // continuize the heights function.
    continuize(heights);
    differences(heights);
    for (int i = 0;i < motionVectors.size();i++)
    {
        vector<Eigen::Vector3d> tmp = analyzer.mapPoints(centers,
motionVectors[i], heights[i]);
        points.insert(points.end(), tmp.begin(), tmp.end());
    }
    Analyzer::rotatePoints(points,angle);
    return points;
}
```

```
void showTD(vector<Eigen::Vector3d> points)
{
    string window_name = "Room Map";
    PointDisplayer displayer(window_name);
    displayer.topDownView(points);
}
int BuildTDView(vector<string> mvFiles, vector<string> heightFiles)
{

    if (mvFiles.size() != heightFiles.size()) throw "Invalid sizes in
BuildTDView";
    vector<Eigen::Vector3d> points;
    for (int i = 0;i < mvFiles.size();i++)
    {
        auto tmp = extractPoints(mvFiles[i], heightFiles[i], 60*i);
        std::cout << "Processing Angle : " << 60 * i << std::endl;
        points.insert(points.end(), tmp.begin(), tmp.end());
    }
    showTD(points);
    return 0;
}
```

We got the following results:

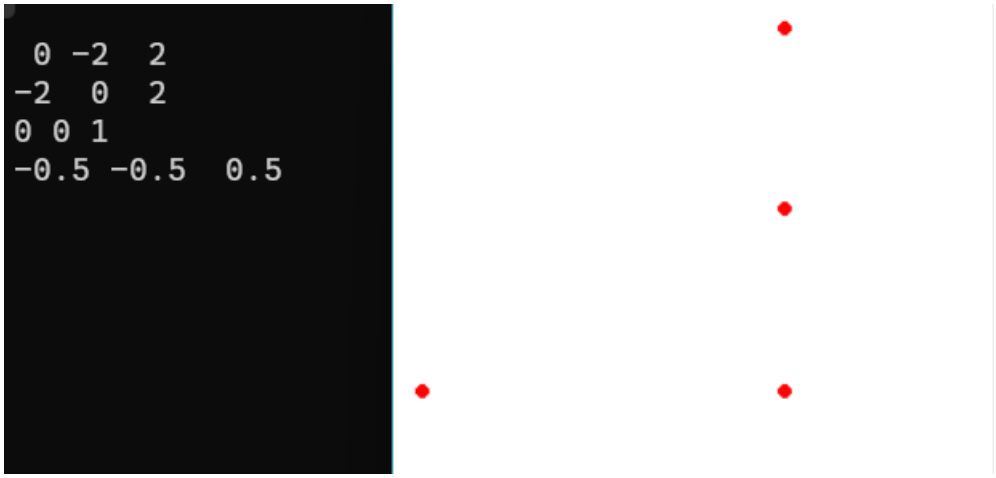


The results seemed incorrect since we expect them to be round, so we decided to recheck our code. We ran on checks sanity check:

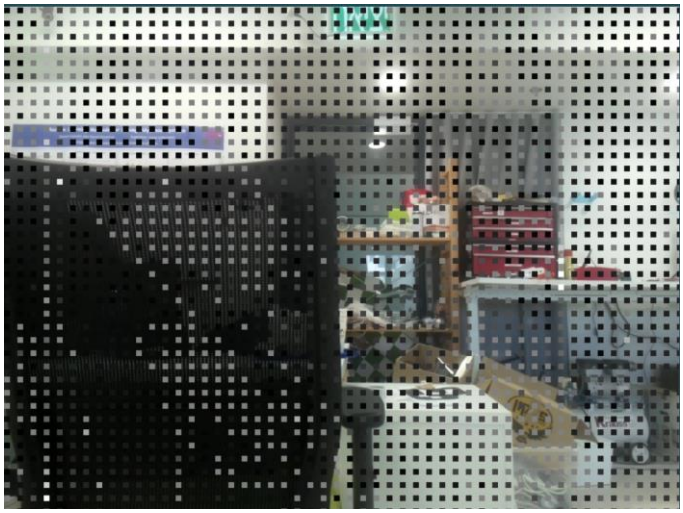
```
   vector<Eigen::Vector2d> motionVectors {

        Eigen::Vector2d(0,1), Eigen::Vector2d(0,1),
        Eigen::Vector2d(0,2), Eigen::Vector2d(0,4)
    };
    double height = 2;
    vector<Eigen::Vector2d> centers{
        Eigen::Vector2d(1,0), Eigen::Vector2d(0,1),
        Eigen::Vector2d(1,1), Eigen::Vector2d(0,0)};
```

We ran the code and got the expected points:



We realized the data is well processed, but we don't know the top-down view shape, meaning that it's not a useful way to check if we're doing the right calculations, so we implemented depth map, and got the following results:

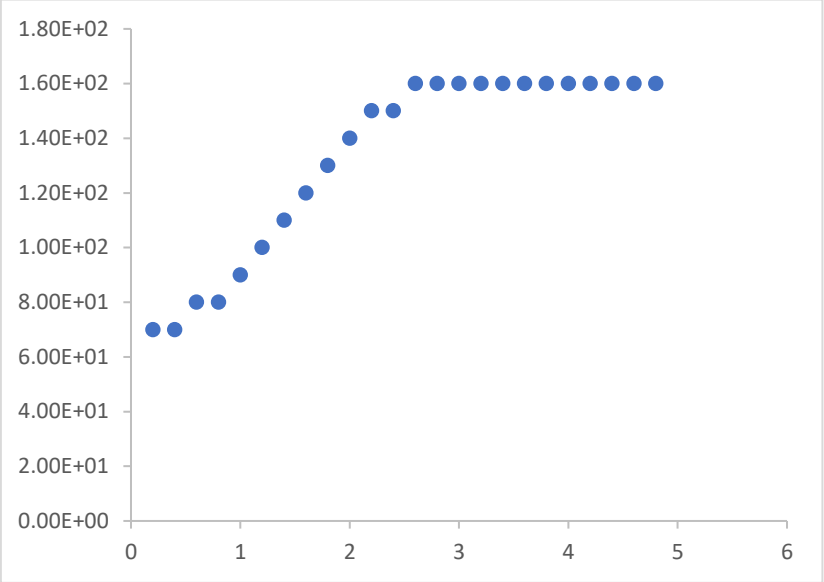


we can see that closer points are distinguished from further points. This could indicate that maybe our results are somewhat good. We are not sure this is what we need yet, therefore we will be running more sanity checks next week to debug our code.

We must mention that we used 5 fps rate, and moved up to 90 cm (height range of 70-160 cm), which allowed better data values distribution (less 0 motion vectors), and did give better results.

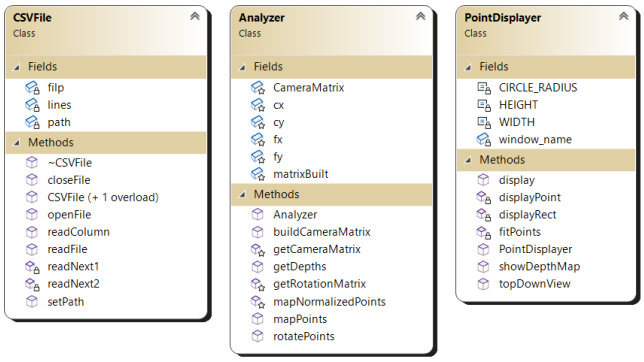
Choosing Optimal Height Function

We looked at the height as a function of time, and got the following function:



Which indicates that the velocity over time is almost constant, meaning that the best approximation would be $\Delta H\ i \to j\ = \frac{h_j - h_i}{j - i}$.

At the end, we are showing a class diagram of our project for futuristic indications.



Next Week's Tasks

- ☞ Using safe-environment presentation of points cloud (MAYA or MS 3D Painter) with obj files.
- ☞ Building more interactive depth map.
- ☞ Further debugging of the code.
- ☞ Matching between two representations of the same point in two different frames
- ☞ Making the code more elegant
- ☞ Making the code more efficient