Week 11

Exercise 11.1

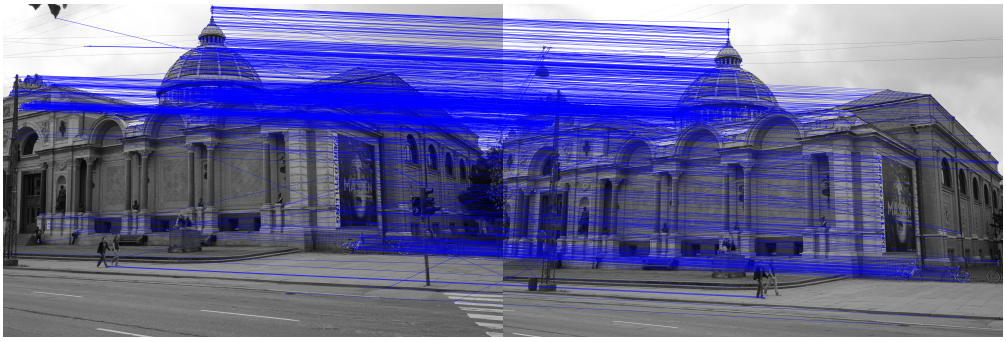

Figure 1: Matches between im0 and im1

Out of the 2000 keypoints initially detected using SIFT, 957 keypoints were remaining after keeping those that were matched.

Exercise 11.2

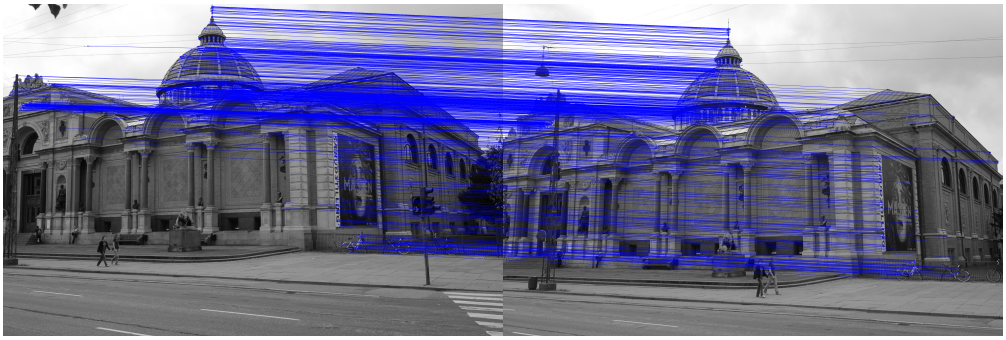

Figure 2: Matches between im0 and im1 after filtering out outliers using the masks from the functions cv2.findEssentialMat and cv2.recoverPose

After having filtered out the outliers using the masks, 644 keypoints were remaining out of the original 2000 keypoints detected in the images.

Exercise 11.4

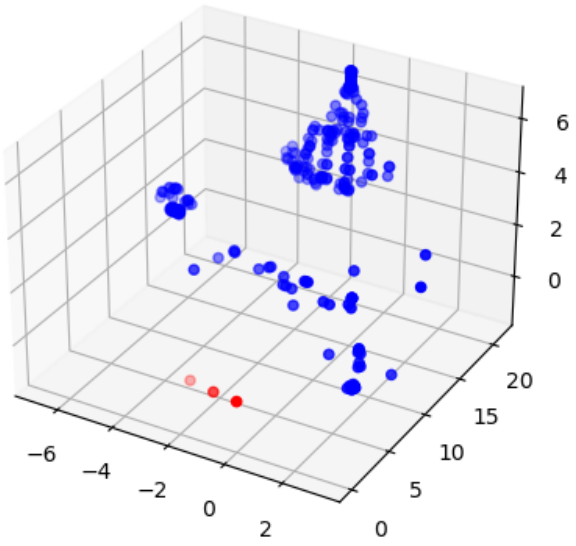

Figure 3: Camera position and inliers 3d points plot for the first three frames.

If you want an interactive view, you can use the code in the appendix below:

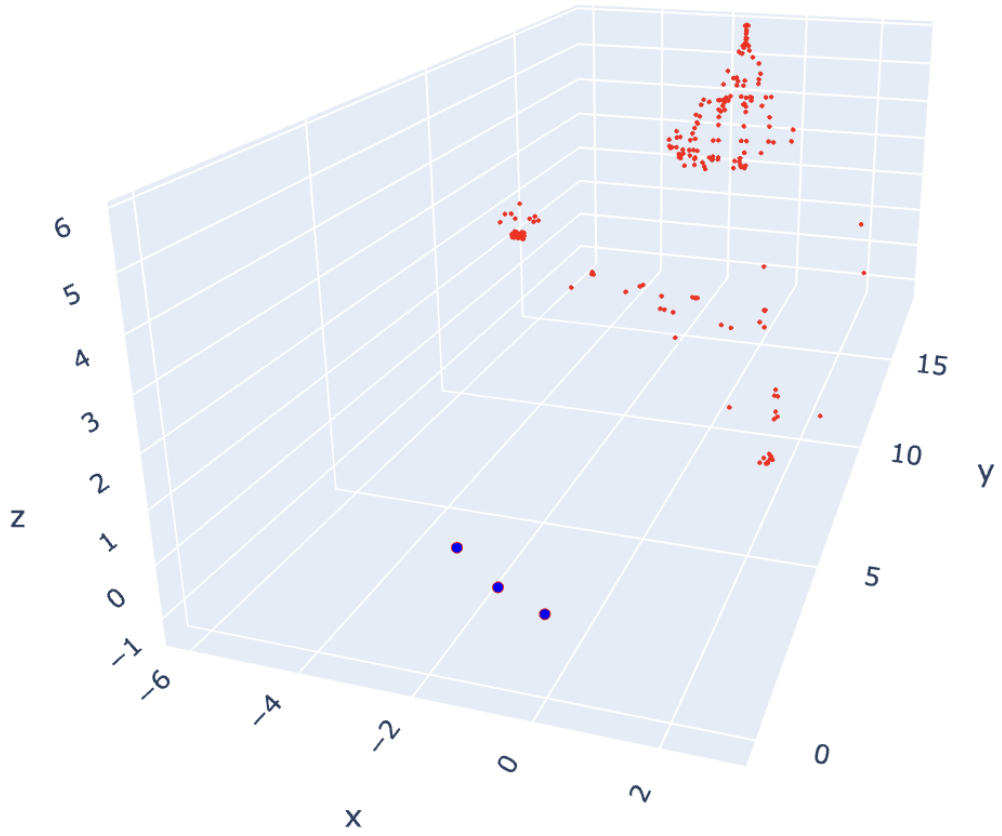

Figure 4: Camera position and inliers 3d points plot for the first three frames.

Exercise 11.5

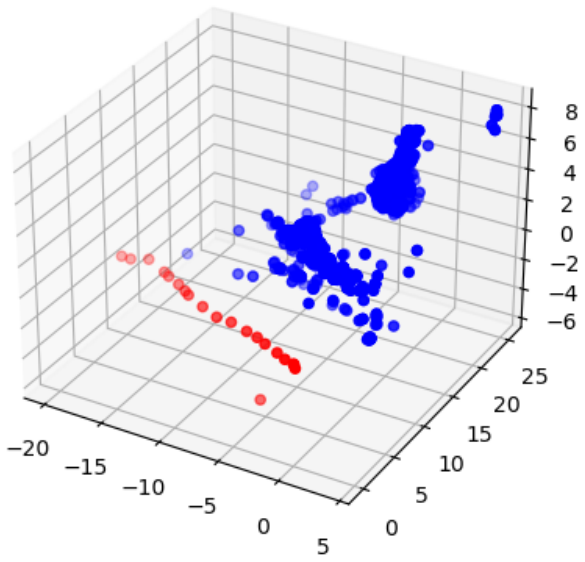

Figure 5: Camera position and inliers 3d points plot for all frames.

The feature matching lost track from img 17, where no point pairs are considered as inliers for solving PnP.

Appendix

```
import plotly.offline as py
import plotly.graph_objs as go

def display_points(P0, P1):
    # P0: The 3D points that you have triangulated.
    # A numpy array of size Nx3, where N is number of points
    # P1: The camera poses in the world coordinate system.
    # Also a Kx3 numpy array
    points_3d = go.Scatter3d(
        x = P0[:,0],
        y = P0[:,2],
        z = -P0[:,1],
        mode = 'markers',
        marker=dict(
        color='rgb(255, 0, 0)',
        size=1,
        symbol='circle',
```

```
        line=dict(
            color='rgb(255, 0, 0)',
            width=1
        )
    ),

    )
    cam_pos = go.Scatter3d(
        x = P1[:,0],
        y = P1[:,2],
        z = P1[:,1],
        mode = 'markers',
        marker=dict(
        color='rgb(0, 0, 255)',
        size=3,
        symbol='circle',
        line=dict(
            color='rgb(255, 0, 0)',
            width=1
        )
    ),
    )
    data = [points_3d, cam_pos]
    layout = go.Layout(
        margin=dict(
        l=0,
        r=0,
        b=0,
        t=0
        ),
        scene=dict(
            aspectmode='data'
        )
    )
    fig = go.Figure(data=data, layout=layout)

    py.iplot(fig, filename='Annotation Points')
```