Exercise Sheet 2

Anselm Coogan

November 11, 2019

1 Exercise 1: Camera Models

```
// gets the camera specific parameters (cx, cy, fx, fy, etc.)
CamT cam = CamT::getTestProjections();
// iterate through 40 pts where x and y are whole numbers in [-10, 10]
for (int x = -10; x <= 10; x++) {
    for (int y = -10; y <= 10; y++) {
      // creates a vector give x and y, at constant distance z
      Vec3 p(x, y, 5);

      // normalizes said vector
      Vec3 p_normalized = p.normalized();
      // projects vector onto the image plane for the given camera
      Vec2 res = cam.project(p);
      // unprojects point on the image plane (2D) onto 3D
      // However, depth information is lost in the process (only ray direction is preserved)
      Vec3 p_uproj = cam.unproject(res);

      // Hence, we can only compare the normalized vectors
      // and confirm that the camera model preserves the original directions
      ASSERT_TRUE(p_normalized.isApprox(p_uproj))
          << "p_normalized " << p_normalized.transpose() << " p_uproj "
          << p_uproj.transpose();
  }
}
```

2 Exercise 2: Optimization

The difference is that the first example assumes that the underlying data points are sampled from an exponential function + Gaussian noise; the second example, on the other hand, can deal with data points that are outside of the assumed distribution (i.e. outliers). Using an added loss function in the residual, the second example is able to better neglect outliers and hence becomes more robust for "real" data.

3 Exercise 3: Camera Calibration

The command line arguments are show-gui, dataset-path, and cam-model.

- show-gui: (optional) when this option is added to the command, the GUI is displayed.

- dataset-path: (required) the path where to find the image(s)
- cam-model: (optional) the camera model to use: pinhole, ds, eucm, kb4. Defaults to ds.

3.1 Cam model comparison

Model	Minimizer Iterations	Cost change	Final cost	Time (s)
Pinhole	16	1.78e7	1.57e5	1.16
KannalaBrandt4	8	5.79e6	1.62e2	0.67
Extended Unified	7	5.35e6	1.63e2	0.52
Double Sphere	15	5.35e6	1.63e2	0.96

The final cost is the most obvious way to compare the different models and as can be seen from above table, all camera models but the pinhole camera achieve basically the same final cost. The pinhole camera, being the simplest model, has a cost a thousandfold larger than the other cameras. A second metric to compare the models by is the time the optimization takes, and in this metric the Extended Unified camera is the winner, taking roughly half a second. It beats the Double Sphere camera by a factor of two and is about 150ms faster than the Kannala-Brandt model.