AV Assignment 2
"3D Image Analysis"

s0949775 and s1330128

17th March 2014

1 Introduction

This report covers the construction of a 3D model from a sequence of RGB and depth frames from the Kinect. The algorithms we chose and explored are discussed for each phase of processing:

- extracting the bin from the RGB images and corresponding 3D points
- aligning the extracted bin points from each frame in a single coordinate frame

2 Methods

Registering data points into one combined dataset and extracting planes of the sides of the box involved a few steps:

1. Extract points that belong only to the box
2. Align the box points with the ones in foundation frame
3. Add the points to combined dataset
4. Fit the planes to current dataset

Each of the steps are described below.

2.1 Box extraction

To extract box we used the fact that it has two orange regions. We found bottom orange region of the box and then took all of the pixels above it. After extracting these points from image, we intersected them with depth information - we removed part of pixels that are above top orange region using the fact that above the box all of the depth values are NaN.

For actual region lookup, basic color thresholding was used since orange region had very distinguishable color. Before thresholding we converted regular

RGB colors to chromaticity colors to reduce impact of lighting and applied gaussian smoothing on pixels of the frame. After these stops two big regions of orange were found and we took the biggest one since bottom region is always the biggest (later it even gets truncated because box gets out of frame).

Colors for region thresholding we fine-tuned manually.

2.2 Point alignment

As was suggested in the assignment we took middle frame from the sequence as our foundation frame. Against it all other frame we compared.

To align points for different frames we used ICP algorithm that was given in the assignment. As actual data that was supplied to ICP we used edge points. Current box's image was converted to greyscale and MATLAB's canny was used to find edges. Using edges, we selected only those depth points that we under the edges. We tuned edge detector to detect only long edges that have bigger intensity changes thus finding few but high quality edges. This allowed ICP to run fast and produce alignments of high quality.

While in the assignment there was suggested to use weighted transformation matrices we did not use that. Utilising transformation from full point set of the box only produced worse results compared to transformation matrices that used points from the edges.

2.3 Plane fitting

For plane extraction we chose to implement modified K-means algorithm. It uses plane's normal vector $\vec{n}$ and data point (x, y, z) dot product as a distance measure. In maximization step it refits plane on assigned data points for each plane using SVD (as given in "fitplane.m" algorithm"). Code can be found in appendix A.

This approach easily allowed to fit data points to any of number of planes that was required though it is prone (as regular K-mean is) to initialization. To cope with this, thresholds were introduced to find cases when initialization was definitely wrong.

3 Results

3.1 Dataset merging

3.2 Alignment accuracy

3.3 Plane fitting

4 Discussion

ICP and plane fitting algorithms in most cases worked without problems. K-means sometimes suffered from poor initialization but that is easy to fix giving

just a few checks of total absolute error. The trickiest part is initial box region detection. Finetuning color thresholds is tedious process and always captures some artefacts around the box that might have impact in later stages.

One of possible techniques to try for box detection would be a probabilistic modelling - it might be possible to detect background from whole series of frames and then use it on each frame individually. We haven't tried this and just sticked with easy to implement approach that works good enough.

References

[1] ADVANCED VISION - TASK 4 MATLAB CODE `http://www.inf.ed.ac.uk/teaching/courses/av/MATLAB/TASK4/`

A Code