Advanced Vision Practical 2

Bob Fisher
School of Informatics

March 2014

Abstract

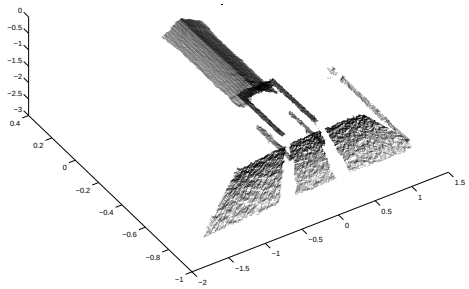
This describes the second assignment for assessment on Advanced Vision. The main goal is to extract 3D points from a set of range images obtained from a Kinect sensor and assemble the points into a more complete 3D object. The assignment is due: **4pm Thursday 20 March**. You must do this practical in teams of 2, and submit 1 PDF report only.

Task Background

At the URL:

`http://homepages.inf.ed.ac.uk/rbf/AVDATA/AV214DATA/`

you will find a matlab file `kinect_recyclebox_20frames.mat` (42 Mb) containing image data from 20 views as captured by a Kinect sensor. The Kinect captures (after a little processing) two registered 640x480 images, one of which is an RGB view of the scene and the other is an XYZ image of the observed data points. You can see the RGB intensity image and a rotated view of the XYZ values here:



After you load the data file (`load kinect_recyclebox_20frames`) there is a frame data structure in memory. There are 20 frames. You can access the i^{th} frame using `xyzrgb=kinect_recyclebox_20frames{i}`. (Note: this command uses "{}", not "()".) The colour image is in the array `xyzrgb(1:480,1:640,4:6)` and the range image is in the array `xyzrgb(1:480,1:640,1:3)`. *I.e.* `xyzrgb(r,c,4:6)` is the RGB value at pixel (r, c) and `xyzrgb(r,c,1:3)` is the XYZ position of the corresponding surface point. The colour image shown above was generated using `imshow(xyzrgb(:,:,4:6)/255)`.

Note that the XYZ data might not be complete, *e.g.* because of occlusions, surface too slanted, surface too dark, etc. In this case, the XYZ value will be the Matlab value `NaN`. To display the 3D data as a point cloud, the following code works:

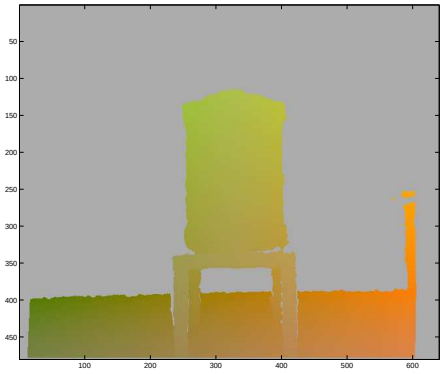
```
xyz = reshape(xyzrgb(:,:,1:3),640*480,3);  % make a list of (x,y,z) points
xyzc = xyz(~any(isnan(xyz),2),:);          % remove any entries that have a NaN
```

```
plot3(xyzc(:,1),xyzc(:,2),xyzc(:,3),'k.','MarkerSize',0.1);
```

If you want to see a false coloured depth image, use:

```
figure(3)
frgb=zeros(480,640,3);
for r = 1 : 480
 for c = 1 : 640
  if ~isnan(xyzrgb(r,c,2))
     frgb(r,c,:) = xyzrgb(r,c,1:3);
   end
  end
end
mn = min(min(min(frgb)));
mx = max(max(max(frgb)));
imagesc((frgb-mn)/(mx-mn))
```

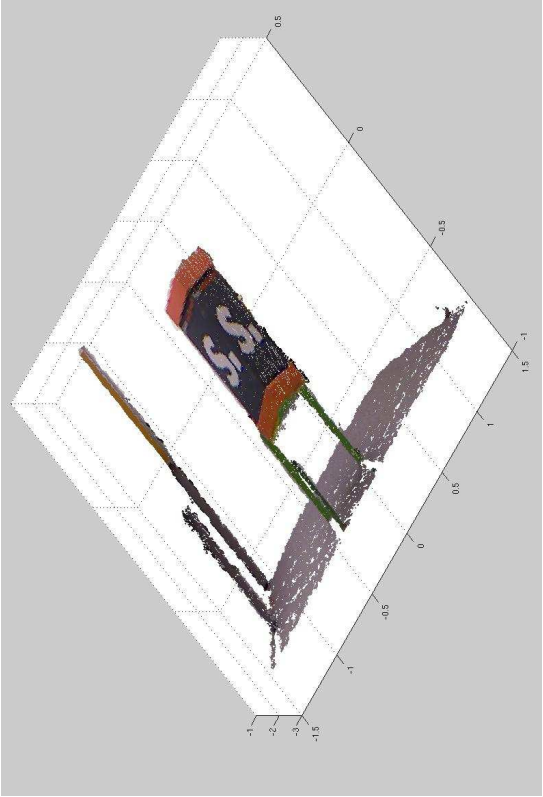
which looks like



You will want to plot texture mapped point clouds, and example code to do this can be found at:

`http://homepages.inf.ed.ac.uk/rbf/AVDATA/AV214DATA/scatter3_kinectxyzrgb.m`

You will have to adapt the code to suit your implementation. This will produce a 3D point sloud that looks like this:



The Practical

The main practical goal is to extract the red and black recycling bin from the data and fuse the 20 views together to make a better model of the bin. Because the kinect sensor is zooming towards the bin, you will need to use both the intensity and range data.

The key idea is to use the 3D data to register each image to the next, and then combine all of the range data into a single dataset. You will need to develop algorithms to:

1. Isolate the 3D and 2D coloured points of the bin in each image
2. Register the points of the individual frames together into one 3D point set
3. Estimate the equations of the two planes and compute the angle between the 2 planes

More details are given in the sections below.

To start with, you have to choose one of the frames as the "Foundation" frame. This frame will be at the identity transformation and all other image data is registered to this frame. A good frame to choose is the middle frame of the sequence.

3D Point Isolation

The original images and point sets include NaN's and other structure besides the bin. The NaN's are easily detected using the sample drawing code given above. Points with NaN's can be ignored. The bin can be further isolated using the registered colour image, because all points either belong to the red edges or the black and white surfaces that lie between the two large red regions.

Hint: Save the isolated XYZ/RGB points after you have processed all of the frames so you don't need to do this each time that you run the program.

3D Point Registration

This is the more difficult aspect of this practical. The difficulty arises because the bin surfaces are planes, and so registation algorithms that only consider 3D shape can lead to an large number of solutions where the pair of planes slide along the fold in the middle. To overcome this issue, we'll exploit the registered colour image.

The colour image has edges around the large printed text and between the black and red patches. These edges can be used in an ICP (Iterated Closest Point) 3D registration algorithm: Use a 2D edge detector (possibly exploiting colour) to find the edge points and then find the corresponding 3D points that are registered with the edge points. These 3D points are more distinctive and can be used in an ICP algorithm.

One can combine the full set of 3D data with the 3D data from the edges as well: 1) do an ICP iteration with the 'edge' data, leading to a pose estimation matrix M_e. 2) do an ICP iteration with the full set of 3D data, leading to a pose estimation matrix M_f. 3) Combine the two pose estimation matrices with a weighting that favours the 'edge data', *e.g.* $\mathrm{M}_f + 10 * \mathrm{M}_e$. 4) Estimate the new alignment transformation and repeat 1-4 in the usual ICP manner until the alignment transformation hardly changes in an iteration.

Other possible approaches are to include multiple copies of the edge points in the point set, or only do the registration with the edges, especially for the initial approximate alignment.

You will need a close initial alignment to get ICP to work well. One can use the colour information to produce an approximate 2D and thereafter 3D registration.

See: `en.wikipedia.org/wiki/Iterative_closest_point` and
`www.mathworks.com/matlabcentral/fileexchange/27804-iterative-closest-point`

Hint: save the estimated transformations after you have processed all of the frames so you don't need to do this each time that you run the program.

Plane Equation and Angle

This can be done using the methods introduced in the videos.

Because the registration may not be very precise, plane fitting would probably work best if you separate the data for the 2 planes first and then estimate the plane of each them independently. There are several possibilities: 1) Use RANSAC on the merged dataset, 2) Use RANSAC on each image independently and then do a second plane fitting once you have all the points from all the images for that plane, 3) use plane growing in each image independently and then do a second plane fitting once you have all the points from all the images for that plane, etc. The goal is to eventually estimate only 2 planes.

Once you have the 2 planes, you can: a) estimate the angle between the planes and b) texture map the plane by projecting the original 3D points onto the plane. The projected points can be coloured with the RGB values of the original unprojected points.

Evaluation

There is no ground truth here, so you will have to mainly depend on your visual judgement.

- Produce images of the bin after each dataset is merged. After each merging, produce one image for just the range points, and a second image for the points coloured using their original colour. Do this for both the frontal view and also the top view of the box (which should ideally look like a "V" with an interior angle of about 90 degrees. Repeat this after merging each new image to obtain 20 incrementally denser views of the bin 3D points, the coloured bin points and the top view.

- Produce a zoomed image of the leftmost white square and nearby black pixels on the left side of the bin after each dataset is merged. Rotate the surface containing the white square so that the surface is parallel to the image plane, ie. the surface normal points to the viewer.

Repeat this after merging each new image to obtain 20 incrementally denser views of the square.

- Extract the surface normal of each of the rightward facing planes from the 20 images. Invert if necessary to make sure that they are all facing towards the viewer. After applying the registration transformation to each, compute the angle between each vector and the 'Foundation' image's vector. Report the average and standard deviation of the angles.
- After 3D points from each frame have been merged into the point set, one can compute a measure of how well the data points are aligned. This will be based on plane fitting a point set: 1) After each merging, create a bounding box around the rightmost merged white S and square. 2) Extract the 3D points from that merged set. 3) Fit a plane to the point set using `www.inf.ed.ac.uk/teaching/courses/av/MATLAB/TASK3/fitplane.m`.
  4) Compute the average absolute value of the distance of each datapoint from that plane.
  5) Plot these average distances as a function of the number of planes added.

Your Report

Write a report that describes:

- The algorithm used for bin extraction.
- The algorithm used for data point registration.
- The algorithm used for data fusion.
- The algorithm used for plane equation extraction.
- Show example images of each processing stage.
- Show the images requested above of the fused bin points after each new point set is added.
- Show examples of unsuccessful extractions and fusions, and discuss why they happened.
- Your code. Do not include code that was downloaded from the AV, IVR or external web sites.

Other Comments

1. You can use the lecture example code from:
   `http://www.inf.ed.ac.uk/teaching/courses/av/MATLAB/`
2. Because there are a limited number of MATLAB Image Processing library licenses available, use alternative MATLAB functions from
   `http://www.inf.ed.ac.uk/teaching/courses/av/MATLAB/UTILITIES/`

Assignment Submission

Submit your report in PDF online by 4pm Thursday March 20. The online submission line is:

```
submit av 2 FILE
```

where `FILE` is the name of your file.

The assignment is estimated to take 10 hours coding/test and 5 hours report writing per person, resulting in a 5-10 page report plus the code appendix. You must do this assignment in teams of 2. You must find your partner and email Bob Fisher (rbf@inf.ed.ac.uk) the name of your partner. A single, joint, report is to be submitted. Split the work so that each partner does some independently (*i.e.* share the work rather than duplicate it).

Live Demonstration

There will also be a demonstration session on the morning of Friday March 21, where you will have to demonstrate your code executing on the same dataset. We'll email you about the location and schedule.

You will need your matlab program to show (add a pause after each frame):

1. The original data images
2. The bin 2D points extracted from each image.
3. The bin 3D points extracted from each image.
4. A 3D view with all current extracted 3D points registered and coloured.

Marking

The assignment will be marked as follows:

Issue	Percentage
1. Clear description of algorithms used	25%
2. Performance on supplied image data	25%
3. Clear Matlab code	10%
4. Discussion of result quality and causes of any failures	20%
5. Performance on supplied demonstration image data	20%

Publication of Solutions

We will not publish a solution set of code. You may make public your solution **but only 2 weeks after the submission date**. Making the solutions public before then will create suspicions about why you made them public.

Plagiarism Avoidance Advice

You are expected to write the document in your own words. Short quotations (with proper, explicit attribution) are allowed, but the bulk of the submission should be your own work. Use proper citation style for all citations, whether traditional paper resources or web-based materials.

If you use small amounts of code from another student or the web, you must acknowledge the original source and make clear what portions of the code were yours and what were obtained elsewhere. You can ignore this condition for the AV lecture examples, which can be used freely.

The school has a robust policy on plagiarism that can be viewed here:

`http://www.inf.ed.ac.uk/teaching/plagiarism.html`

The school uses various techniques to detect plagiarism, included automated tools and comparison against on-line repositories. *Remember: a weak assignment is not a ruined career (and may not reduce your final average more than 1%), but getting caught at plagiarism could ruin it.*

LATE SUBMISSION POLICY

See:
`http://www.inf.ed.ac.uk/student-services/teaching-organisation/...`
`for-taught-students/coursework-and-projects/late-coursework-submission`

ACADEMIC MISCONDUCT POLICY

Our policy on academic conduct:
http://www.ed.ac.uk/schools-departments/academic-services/students/...
postgraduate-taught/discipline/academic-misconduct