Homework 4 Errata & Addendum

General

images_test/0003.jpg is hard to get SIFT matches on. It’s okay if it doesn’t work on this one—you can ignore it. (If you want to try a different one, extra_test/0015.jpg should be doable).

In a real application, if you wanted to fix this, there are several things you might try:

• Process images at a higher resolution (I resized them to make processing more manageable).

• Use more than one training (template) image per object, especially when viewing objects from very different viewpoints. SIFT accounts for scale and rotation, but severe perspective effects will affect performance (as will large changes in lighting conditions).

• Consider a library of background features when matching. Having SIFT features of points seen in the background (e.g., plant, chair) can avoid spurious matches in these areas, getting you a better inlier-to-outlier ratio.

You can’t really do any of these for this assignment, so you can just ignore this image.

Problem 4

Q4.6 The function header should be [R,t]=approxRtFromH(K,H) (you will need K to compute R and t).

Some people asked for a reference. This paper has a slightly more general derivation: [1] and includes an “optimal” solution (please cite if you use it).


Drawing

For the purposes of this assignment, you should just be able to use what is in displaybunny.m: that is, project the 3D vertices or points using the projection equation (with projective division), and use the negative z associated with each 3D point after applying
\[ p = K[R|t](x \ y \ z \ 1)^T \] (that is, \(z_{\text{coords}}=-p(3,:);\)) as the z-coordinate for any Matlab drawing functions.

The explanation (optional, more computer graphics than computer vision)

A somewhat long explanation for the reasons follows. In computer graphics, 3D models are most commonly drawn by approximating the 3D surface of the object using many polygons defined by their vertices. The polygons are known as the "faces" of the object, and are most commonly triangles (because only 3 vertices define a triangle and are always coplanar and convex). "Quads" or quadrilateral polygons are also common—the teapot model is defined by quads—but less so, because 4 vertices are not guaranteed to be coplanar.

To draw the 3D figure correctly into an image, we just need to project the 3D vertices into the corresponding image locations (Matlab will take care of drawing the faces, see `displaybunny.m`). There is an additional problem though: the object has volume, and each camera ray (for each pixel) can intersect more than one polygon (one at the front of the bunny, and one directly behind, which should be occluded).

There are a number of solutions to this: we can draw the faces in order from back to front, or use a z-buffer. Matlab uses a z-buffer: it draws all the polygons in arbitrary order, but checks an internal z-buffer (for each pixel) before drawing. The z-buffer stores the depth of what we have drawn up to now at that pixel, and we only draw (and update the z-buffer) if the location we are drawing is closer to the camera than the previous value in the z-buffer. This way, after drawing all triangles, each pixel will have the color corresponding to the closest point in 3D, and Matlab takes care of all of this internally.

Left: Wireframe drawing of triangles on the bunny model. The faces are transparent here, you can see the overlap. Right: Triangle faces are drawn in white, with z-buffering turned on to handle occlusion of the underside of the bunny.

The reason we have to flip the z-coordinate and make it negative \(-p(3,:);\) is because the convention about whether the positive z-axis points into the screen (more depth is more positive) or out of the screen (more depth is more negative) is somewhat arbitrary.\(^1\)

\(^1\)Another related technique is called back-face culling, where only polygons oriented toward the camera are drawn.
and computer graphics usually follows the opposite approach of that followed in vision. (This is related to the convention of whether you use a left-handed coordinate system or a right-handed one).

It is okay to be somewhat confused by this, you really have to sit down with a paper and draw things before all the coordinate systems start making sense, but that shouldn’t be necessary for this assignment.

References