Assignment A1: Camera Models

CS 5320/6320
Spring 2016

Assigned: 11 January 2016

Due: 27 January 2016 (use handin on CADE and turn in hardcopy of report by start of class)

For this problem, handin a lab report pdf named A1.pdf (include name, date, assignment and class number in pdf) which adheres to the specified lab format and which studies the camera model perspective projection. You should handin the report pdf as well as the Matlab source code used in the study. The code should conform to the style requested in the class materials. You need to pose some questions to be answered in the lab; e.g., how sensitive is the transform to parameter values (both intrinsic and extrinsic)?

Your assignment is to:

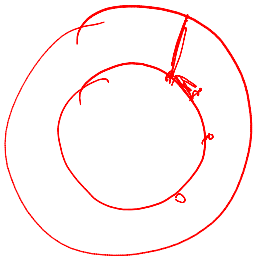
1. Implement CS5320_camera function as described below.
2. Implement CS5320_gen_R function as described below.
3. Implement CS5320_gen_cube as described below.
4. Implement CS5320_gen_sphere as described below.
5. Use CS5320_movie_trans to generate a movie using your camera code. Name the avi file A1_trans.avi.
6. Use CS5320_movie_rotate to generate a movie using your camera code. Name the avi file A1_rotate.avi.

Write a lab report in the format (please do not deviate from this format!) described in the course materials.

```
function im = CS5320_camera(scene,alpha,beta,theta,x0,y0,R,t)
% CS5320_camera - produce camera model perspective projection
% On input:
%     scene (4xk array): 3D homogeneous coordinate world points
%     alpha (float): pixel scale parameter in x (intrinsic)
%     beta (float): pixel scale parameter in y (intrinsic)
%     theta (float): image frame skew angle (intrinsic)
%     x0 (float): image plane center offset (intrinsic)
%     y0 (float): image plane center offset (intrinsic)
%     R (3x3 array): rotation array (extrinsic)
%     t (3x1 vector): translation vector world origin to camera
%     (extrinsic)
% On output:
%     im (3xk array): homogeneous coordinates for points on image
%     plane
%      row 1: X values
%      row 2: Y values
%      row 3: 1's (homogeneous coordinate)
% Call:
%     im = CS5320_camera(cube,1,1,pi/2,0,0,eye(3,3),[0;0;0]);
% Author:
%     <Your name>
%     UU
%     Spring 2016
%


function R = CS5320_gen_R(u,theta)
% CS5320_gen_R - generate a rotation matrix about an arbitrary vector
% On input:
%     u (3x1 vector): unit vector about which to rotate
%     theta (float): amount to rotate (in radians)
% On output:
%     R (3x3 matrix): rotation matrix
% Call:
%     R = CS5320_gen_R([0;0;1],pi/2);
% Author:
%     T. Henderson
%     UU
%     Spring 2016
%
```

```
function sphere = CS5320_gen_sphere(C,radius,del_x,del_p)
% CS5320_gen_sphere - generate 3D points on the surface of a sphere
% On input:
%     C (3x1 vector): center of sphere
%     radius (float): radius of sphere
%     del_x (float): step size along major axis to generate points
%     del_p (float): distance between points on the sphere
% On output:
%     sphere (4xk array): homogeneous coordinates for points on the
%     sphere
%      row 1: X values
%      row 2: Y values
%      row 3: Z values
%      row 4: 1's (homogeneous coordinate)
% Call:
%     sphere = CS5320_gen_sphere([0;0;0],2,0.1,0.1);
% Author:
%     <Your name>
%     UU
%     Spring 2016
%
```



```
function cube = CS5320_gen_cube(C,del_x,S)
% CS5320_gen_cube - generate 3D points on the edges of a cube
% On input:
%     C (3x1 vector): center of sphere
%     del_x (float): step size along edges to generate points
%     S (float): length of side of cube
% On output:
%     cube (4xk array): homogeneous coordinates for points on the cube
%      row 1: X values
%      row 2: Y values
%      row 3: Z values
%      row 4: 1's (homogeneous coordinate)
% Call:
%     cube = CS5320_gen_cube([0;0;0],0.01,1);
% Author:
%     <Your name>
%     UU
%     Spring 2016
%
```