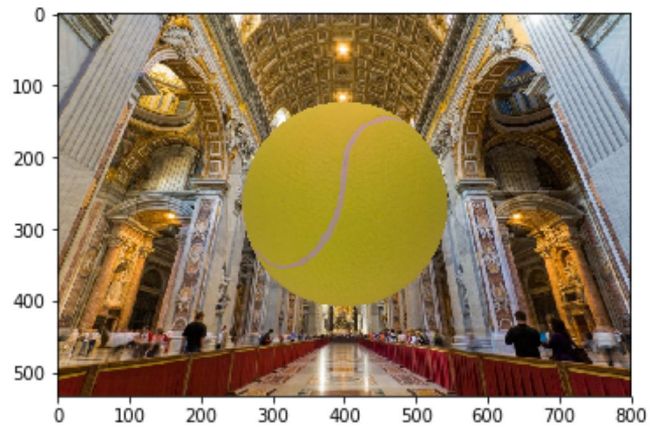
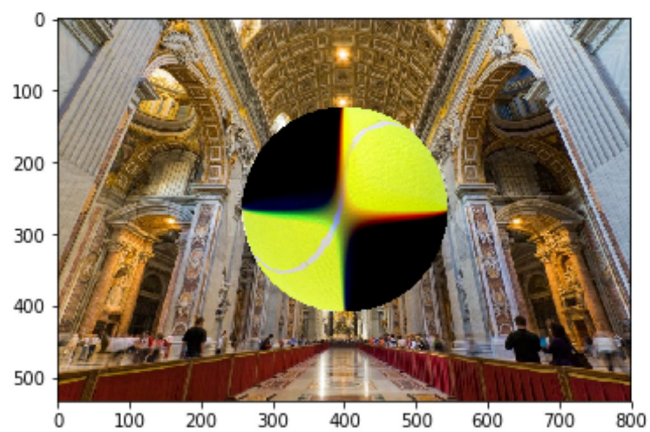
In [4]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.misc import imread,imsave

imFile = 'stpeters_probe_small.png'
compositeFile = 'tennis.png'
targetFile = 'interior.jpg'

# This loads and returns all of the images needed for the problem
# data - the image of the spherical mirror
# tennis - the image of the tennis ball that we will relight
# target - the image that we will paste the tennis ball onto
def loadImages():
    imFile = 'stpeters_probe_small.png'
    compositeFile = 'tennis.png'
    targetFile = 'interior.jpg'

    data = imread(imFile).astype('float')*1.5
    tennis = imread(compositeFile).astype('float')
    target = imread(targetFile).astype('float')/255

    return data, tennis, target


# This function takes as input a square image of size m x m x c
# where c is the number of color channels in the image.  We
# assume that the image contains a scphere and that the edges
# of the sphere touch the edge of the image.
# The output is a tuple (ns, vs) where ns is an n x 3 matrix
# where each row is a unit vector of the direction of incoming light
# vs is an n x c vector where the ith row corresponds with the
# image intensity of incoming light from the corresponding row in ns
def extractNormals(img):

    # Assumes the image is square
    d = img.shape[0]
    r = d / 2
    ns = []
    vs = []
    for i in range(d):
        for j in range(d):

            # Determine if the pixel is on the sphere
            x = j - r
            y = i - r
            if x*x + y*y > r*r-100:
                continue

            # Figure out the normal vector at the point
            # We assume that the image is an orthographic projection
            z = np.sqrt(r*r-x*x-y*y)
            n = np.asarray([x,y,z])
            n = n / np.sqrt(np.sum(np.square(n)))
            view = np.asarray([0,0,-1])
            n = 2*n*(np.sum(n*view))-view
            ns.append(n)
            vs.append(img[i,j])

    return np.asarray(ns), np.asarray(vs)

# This function renders a diffuse sphere of radius r
# using the spherical harmonic coefficients given in
```

```
Question 5c
Coefficients:
[[ 202.31845431  162.41956802  149.07075034]
 [ -27.66555164  -17.88905339  -12.92356688]
 [  -5.15203925   -4.51375871   -4.24262639]
 [  -1.08629293    0.42947012    1.15475569]
 [  -3.14053107   -3.70269907   -3.74382934]
 [  23.67671768   23.15698002   21.94638397]
 [  -3.82167171    0.57606634    1.81637483]
 [   4.7346737     1.4677692    -1.12253649]
 [  -9.72739616   -5.75691108   -4.8395598 ]]
```



```
Question 5d
Coefficients:
[[  2.13318421e+02   1.70780299e+02   1.57126297e+02]
 [ -3.23046362e+01  -2.02975310e+01  -1.45516114e+01]
 [ -4.31689131e+00  -3.80778081e+00  -4.83616306e+00]
 [ -4.89811386e+00  -3.37684058e+00  -1.14207091e+00]
 [ -7.05901066e+03  -7.39934207e+03  -4.26448732e+03]
 [ -3.05378224e+02  -1.56329401e+02   3.50285345e+02]
 [ -9.76079364e+00  -5.33182216e+00  -1.55699782e+00]
 [  7.30792588e+02   3.52130316e+02  -6.11683200e+02]
 [ -9.08887079e+00  -3.84309477e+00  -4.16456437e+00]]
```



```
Question 5e
Coefficients:
[[ 209.38212459  169.03666402  155.36677288]
 [ -30.26805402  -20.30443706  -15.20472049]
 [  -5.753416     -5.07881542   -4.78144904]
 [  -1.05630713    0.46377951    1.19195587]
 [  -7.90569522   -8.20316831   -8.05137623]
 [  54.96251667   52.62398401   50.09265545]
 [  -3.8491927     0.55663535    1.80236903]
 [   7.32655583    3.83064183    1.07500107]
 [ -10.90665749   -6.8522162    -5.87526417]]
```