preLab4.py

```
"""
CIVE 6374 - Optical Imaging Metrology
Professor: Dr. Craig Glennie
Author: Joshua Genova
Lab # 4
Description: Relative Orientation
Deadline: April 5, 2023 10:00 AM
"""

"""
Correct Find Model Space Images
1.) Affine Transformation into fiducial coordinates
2.) Principal Point Offset
3.) Radial Lens Distortion
4.) Decentering Lens Distortion
5.) Atmospheric Refraction
"""
import numpy as np
from numpy.linalg import inv, det
import math
from math import sin, cos

def transform_images(xr, yr, c, omega, phi, kappa):
    rot_mat = np.array([
        [cos(phi)*cos(kappa), cos(omega)*sin(kappa)+sin(omega)*sin(phi)*cos(kappa),
sin(omega)*sin(kappa)-cos(omega)*sin(phi)*cos(kappa)],
        [-cos(phi)*sin(kappa), cos(omega)*cos(kappa)-sin(omega)*sin(phi)*sin(kappa),
sin(omega)*cos(kappa)+cos(omega)*sin(phi)*sin(kappa)],
        [sin(phi), -sin(omega)*cos(phi), cos(omega)*cos(phi)]
    ])

    xr_t = np.zeros(len(xr))
    yr_t = np.zeros(len(yr))
    zr_t = np.zeros(len(xr))
    for i in range(len(xr)):
        vr = np.array([xr[i], yr[i], -c])
        xr_t[i], yr_t[i], zr_t[i] = np.dot(rot_mat.T, vr.T)
    return xr_t, yr_t, zr_t

def space_intersection(xl, yl, c, xr, yr, zr, bx, by, bz):
    scale = (bx*zr - bz*xr) / (xl*zr + c*xr)
    mu = (-bx*c - bz*xl) / (xl*zr + c*xr)

    model_Xl = scale*xl
    model_Yl = scale*yl
    model_Zl = -scale*c

    model_Xr = mu*xr + bx
    model_Yr = mu*yr + by
    model_Zr = mu*zr + bz

    model_L = np.transpose(np.array([model_Xl, (model_Yl + model_Yr)/2, model_Zl]))
    model_R = np.transpose(np.array([model_Xr, (model_Yl + model_Yr)/2, model_Zr]))
```

```
    pY = model_Yr - model_Yl

    return model_L, model_R, pY, scale, mu

# Task 1
def task_1():
    # from lab 3
    bX = 92.000
    bY = -1.421510121593803
    bZ = -1.2872527980970032
    w = -0.01706070025860212
    p = 0.004738042393395472
    k = -0.030192790615912662

    c = 153.358 # mm

    xl = np.array([-9.444, 18.919, 90.289])
    yl = np.array([96.236, -81.819, -91.049])
    xr = np.array([-105.378, -72.539, -1.405])
    yr = np.array([98.756, -79.786, -86.941])
    xr_t, yr_t, zr_t = transform_images(xr, yr, c, w, p, k)
    model_L, model_R, pY, scale_left, scale_right = space_intersection(xl, yl, c, xr_t, yr_t,
zr_t, bX, bY, bZ)
    return model_L, model_R

def task_5():
    # from lab 3
    bX = 92.000
    bY = -1.421510121593803
    bZ = -1.2872527980970032
    w = -0.01706070025860212
    p = 0.004738042393395472
    k = -0.030192790615912662

    c = 153.358 # mm

    xl = np.array([18.174, 44.681, -7.578, 52.736])
    yl = np.array([109.538, 7.483, -49.077, -93.140])
    xr = np.array([-77.840, -48.786, -98.814, -38.924])
    yr = np.array([113.375, 10.165, -48.039, -90.035])
    xr_t, yr_t, zr_t = transform_images(xr, yr, c, w, p, k)
    model_L, model_R, pY, scale_left, scale_right = space_intersection(xl, yl, c, xr_t, yr_t,
zr_t, bX, bY, bZ)
    return model_L, model_R
```