Matlab Code for computation of all Intrinsic and Exrinsic Parameters

For the purpose of demonstration of the 11 parameters, this live script was written to explain each step in a detailed format.

Loading the image

```
I = imread('test_image.bmp');
```

loading the 2D image data

```
load observe.dat
```

loading the 3D image data

```
load model.dat
```

The below two for loops give out the size of the observe and model data files, so that thier values can be used as argumets for other loops

```
[On,Ot] = size(observe);
for i=1:On
    ic_x(i,1) = observe(i,1);

    ic_y(i,1) = observe(i,2);
end
[Oq,Ow,Oe] = size(model);
for i=1:Oq
    wc_x(i,1) = model(i,1);

    wc_y(i,1) = model(i,2);
    wc_z(i,1) = model(i,3);
end
```

We can now use the variable obtained from calculating the length of the observe data

```
n = On; %Generalize the code so that it can run for any given observation data
```

Step 1 of the parameter estimation process.

```
Q(1:2*n,1:12) = 0; % Initilizing a matrix Q of the required dimensions
```

```
j=1;  %Populating the matrix with the required elements

for i=1:2:(2*n)
    Q(i,1) = wc_x(j);
    Q(i,2) = wc_y(j);
    Q(i,3) = wc_z(j);
    Q(i,4) = 1;
    Q(i+1,5) = wc_x(j);
    Q(i+1,6) = wc_y(j);
    Q(i+1,7) = wc_z(j);
    Q(i+1,8) = 1;
    Q(i,9:12) = Q(i,1:4) * -1 * ic_x(j);
    Q(i+1, 9:12) = Q(i,1:4) * -1 * ic_y(j);
    j = j+1;
end
```

We can use the SVD function to solve for the eigen value problem.

```
[~,S,V] = svd(Q);

[~, min_index] = min(diag(S(1:12,1:12)));

m = V(1:12,min_index);

%After solving svd, we have to normalise such that the 3rd rotation vector becomes 1

norm_rt = norm(m(9:11));

m_normazlised = m / norm_rt;

M(1,1:4) = m_normazlised(1:4);
M(2,1:4) = m_normazlised(5:8);
M(3,1:4) = m_normazlised(9:12);
m3 = M(3,1:4);

%assigning variable names accordingly
a1 = M(1,1:3);
a2 = M(2,1:3);
a3 = M(3,1:3);
b = M(1:3,4);
r3 = a3;

rho = -1/norm(a3);
```

Computation of Intrinsic and Extrinsic Parameters

```
u_o = rho^2 * (dot(a1,a3))
```

```
u_o = 2.0592e+04
```

```
v_o = rho^2 * (dot(a2,a3))
```

```
v_o = 5.1169e+03
```

```
crossa1a3 = cross(a1,a3);
crossa2a3 = cross(a2,a3);

theta = acos(-1 * dot(crossa1a3,crossa2a3)/(norm(crossa1a3)*norm(crossa2a3)))
```

```
theta = 1.4840
```

```
alpha = norm(crossa1a3) * sin(theta)
```

```
alpha = 4.1409e+04
```

```
beta = norm(crossa2a3) * sin(theta)
```

```
beta = 2.7811e+04
```

```
r1 = crossa2a3/norm(crossa2a3);
r2 = cross(r3,r1);

K = [alpha, -1*alpha*cot(theta), u_o;
    0, beta/sin(theta), v_o;
    0,0,1]
```

```
K =
   1.0e+04 *
    4.1409   -0.3603    2.0592
         0    2.7916    0.5117
         0         0    0.0001
```

```
trnnls_vector = inv(K) * b
```

```
trnnls_vector =
   1.0e+03
    0.5573
    0.1849
   -1.1053
```

```
R(1,1:3) = r1;
R(2,1:3) = r2;
R(3,1:3) = r3;
r1
```

```
r1 =
    0.4221   -0.8482    0.3200
```

```
r2
```

```
r2 =
    0.6274    0.5281    0.5723
```

```
r3
```

```
r3 =
   -0.6544   -0.0408    0.7550
```

```
A1 = M(1,1:4);
A2 = M(2,1:4);
A3 = M(3,1:4);
```