Optical-camera-for-space-navigation

1. Scope

This simulator aims to evaluate the operative range of an optical camera sensor for space relative navigation. You have to provide true relative range of your follower spacecraft relative to the leader spacecraft, then the simulator tell you if your real optical camera would be able to compute position for each point in your relative range data. It is very easy to provide relative range data for this simulator purposes, you can do it manually. You don't need to import complex trajectories by mission analysis tools.

2. What you have to do

- Run initialization camera.m,
- Enjoy results,
- Modify the following parameters:
    - Specifications of optical camera,
    - Image processing capabilities,
    - Pattern specifications,
    - Relative range.
- Repeat.

3. Definitions

Marker: a corner-cube reflector mounted on the leader spacecraft that reflects light back in the direction of the source.

Pattern: the scheme composed by markers. An elliptical pattern is considered in this work (figure 1). It is made up by 4 markers.

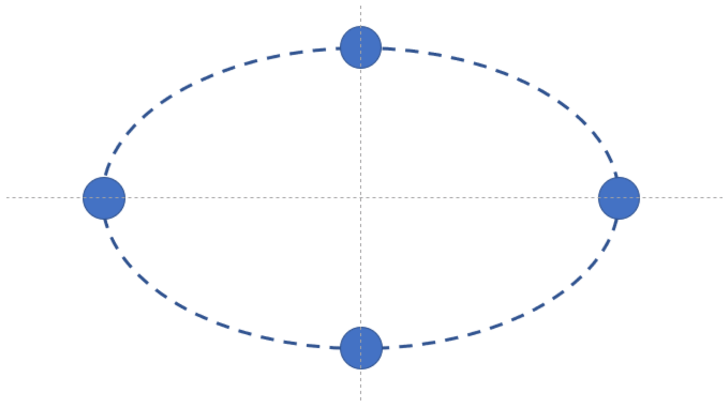

Figure 1: considered pattern

4. Real algorithm: pattern evaluation algorithm

It is the on-board algorithm used by the follower spacecraft to estimate its position relative to the leader. It is included in the simulator in camera_model.slxn (figure 2).

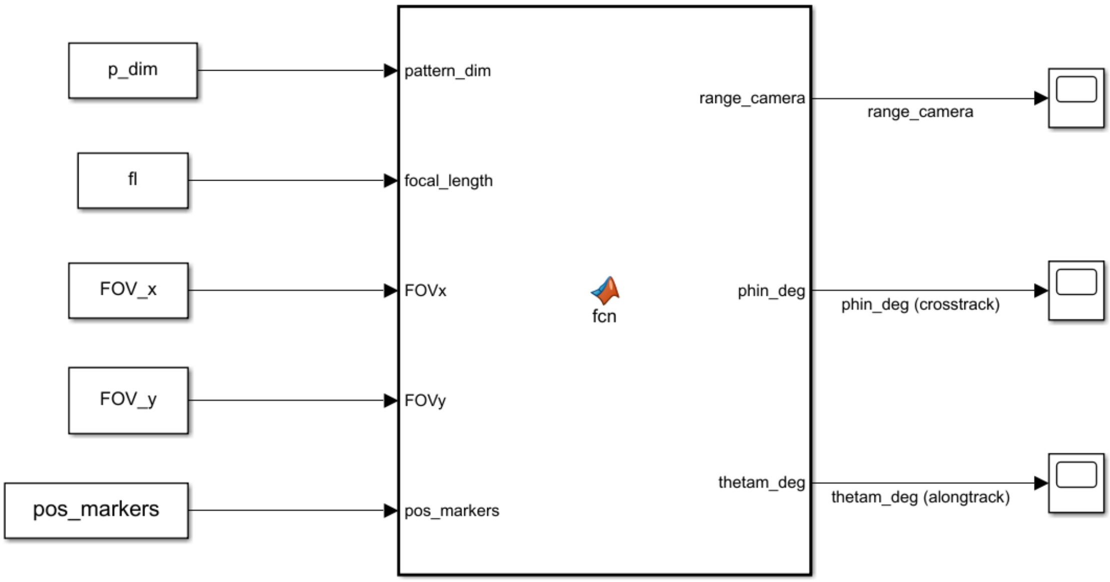

Figure 2: pattern evaluation algorithm, input and output

Inputs of pattern evaluation algorithm:

1. marker coordinates in the focal plane of the camera (markers are represented as material points by couples (x, y)),
2. real pattern characteristic length on the leader spacecraft (major axis of the ellipse is chosen in this code),
3. camera specifications: focal length (fl) and FOV angles (FOV_x, FOV_y).

On orbit, marker coordinates are computed by applying image processing techniques to real photos of the pattern.

Outputs of pattern evaluation algorithm:

1. relative range estimation,
2. LOS cross-track angle estimation,
3. LOS along-track angle estimation.

Relative position is uniquely determined by these three values [1].

The algorithm is made up by the following steps [1].

1. **Determination of the centre of the ellipse in the focal plane $C = (x_c, y_c)$**

The origin of the reference system is in the centre of the focal plane, the positive x axis points to the right, the positive y axis points upwards with reference to the focal plane (figure 3).

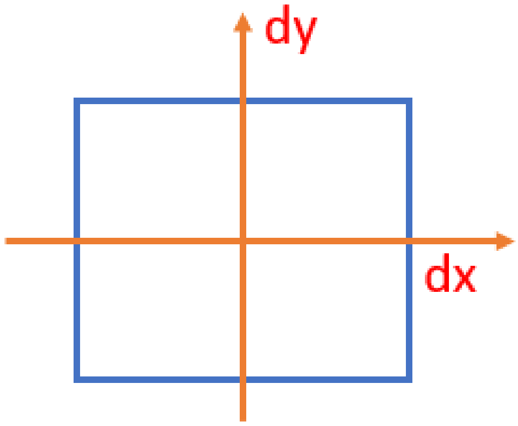

Figure 3: reference system for markers coordinates in the focal plane

The coordinates of the center are:

$$x_c = \frac{\max(x) + \min(x)}{2}$$

$$y_c = \frac{\max(y) + \min(y)}{2}.$$

where x and y are vectors containing x-coordinates and y-coordinates of all markers respectively:

$$x = [x_1, x_2, x_3, x_4]$$

$$y = [y_1, y_2, y_3, y_4].$$

2. Determination of the length of the two axes (a_1, a_2) and selection of the major axis

This step includes a simple *for* cycle that allows the algorithm to compute the length of the major axis a_{max} independently of the order of coordinates in the input array.

$$a_1 = \sqrt{(\min(x) - \max(x))^2 + (\mathrm{y}(\min(x)) - \mathrm{y}(\max(x)))^2}$$

$$a_2 = \sqrt{(x(\min(y)) - x(\max(y))^2 + (\min(y) - \max(y))^2}$$

$$a_{max} = \max(a_1, a_2)$$

where $y(\min(x))$ indicates the x coordinate of the marker that has the minimum x and similarly for the other markers.

3. Relative range determination (direct law)

The relative range r_{camera} of the camera is computed as follow:

$$r_{camera} = p_{dim} \cdot \frac{fl}{a_{max}}$$

where p_{dim} is the real length of the major axis of the pattern and fl is the focal length of the camera sensor.

4. **LOS determination:**

$$\phi = \frac{FOV_x \cdot x_c}{x_{fp}}$$

$$\theta = \frac{FOV_y \cdot y_c}{y_{fp}}$$

where ϕ is cross-track and θ is along-track, $x_{fp} = fl \cdot \tan FOV_X$ is the dimension of the focal plane cross-track and $y_{fp} = fl \cdot \tan FOV_y$ is the dimension of the focal plane along-track.

5. Simulation algorithm

This algorithm is implemented in Matlab/Simulink to study performances (operating range) of the optical sensor during rendezvous and docking. It includes the pattern evaluation algorithm.

Simulation inputs:

1. camera specifications (focal length, FOV angles, GSD factor),
2. image processing capabilities (markers must be composed of a minimum number of pixels px_{min} to allow computation of marker coordinates),
3. real pattern dimensions (major axis),
4. real marker size (characteristic length m_{dim} depending on the shape of markers),
5. true relative range.

Outputs:

1. relative range during the simulation time (see it in Simulink scope);
2. minimum computable range → it depends on dimensions of the real pattern composed by markers on the leader spacecraft. The pattern formed by markers exceeds focal plane dimensions under this range making impossible to compute relative range,
3. maximum computable range: it depends on image processing capabilities and real marker dimensions. Image processing algorithm is not able to compute marker coordinates from photos over this limit. Markers needs to be represented by a minimum number of pixels. Under this number, image processing is not able to distinguish them and calculate their center coordinates.

Figure 4: photos by navigation camera: over maximum range (sx), operative range (centre), under minimum range (dx)

The simulation algorithm is made up by the following steps.

1. **Computation of marker coordinates in the focal plane**
   Then the major axis of the elliptical pattern is calculated from the true relative range by using the following proportional inverse law:

$$a_{max} = \frac{p_{dim} \cdot fl}{r_{true}}.$$

   Then, marker coordinates in the focal plane are computed using a_{max} through the following simplifying hypothesis:

$$M_1 = (x_1, y_1) = \left(\frac{a_{max}}{2}, 0\right)$$

$$M_2 = (x_2, y_2) = \left(-\frac{a_{max}}{2}, 0\right)$$

$$M_3 = (x_3, y_3) = \left(0, \frac{a_{max}}{2}\right)$$

$$M_4 = (x_4, y_4) = \left(0, -\frac{a_{max}}{2}\right).$$

   This hypothesis means that the chaser spacecraft is supposed to perform final rendezvous and docking without pointing error, i.e. null LOS angles are imposed during the whole simulation time. This is not realistic, but very simple and does not affect the outputs we are interested in (operating range determination).

2. **Estimation of relative range and LOS angles**
   Relative range and LOS angles are estimated via the pattern evaluation algorithm (previous section). LOS estimation works only as a check: due to the simplifying hypothesis they must be null.

3. **Evaluation of operating range**

   The minimum computable range is the minimum relative range for which the following condition is true:

$$a_{max} > \min(2x_{fp}, 2y_{fp}).$$

   The maximum computable range is the maximum relative range for which the following condition is true:

$$\frac{m_{dim}}{GSD} < px_{min},$$

where $GSD = GSD\ factor \cdot r_{camera}$.
Illumination of the whole FOV is supposed.

The operating range is the range between minimum and maximum computable ranges.

This simulator can be improved by adding noise to marker coordinates. This can be a way to better simulate image processing capabilities. In my code, I simply set a threshold using the minimum number of pixels per marker needed to compute marker coordinates. This is a sort of Boolean condition:

- If (px per marker > min px per marker): marker coordinates are computed by image processing exactly whatever is the margin of the number of pixels relative to the threshold,

- If (px per marker < min px per marker): relative range estimation is impossible because image processing is not able to compute marker coordinates.

It would be more realistic to introduce a random noise proportional to the relative range on marker coordinates.

6. References

[1] W. Fehse, Automated Rendezvous and Docking of Spacecraft, Springer, 2003 (cap. 7)