**1, Evaluate Detection Performance**

* 1. **Evaluation Metric**

TP: True positives. Stars detected by the star tracker that belong to the groundtruth stars.

FP: False positives. Stars detected by the star tracker that do not belong to the groundtruth stars.

FN: False negatives. Groundtruth stars that are not detected by the star tracker.

Precision: Measure the ability to reject false detections.

A picture containing text

Description automatically generated  
Recall: Measure the ability to detect all stars.

A picture containing schematic

Description automatically generated

F1 score: Combine both precision and recall giving a comprehensive evaluation.

A picture containing text

Description automatically generated

***For our experiments, we want to calculate Precision, Recall, and F1 score for each case. Also remember to keep the raw data for future verification. Also beware not all the stars displayed on the screen are visible to the star tracker, some are outside of FOV.***

* 1. **Tune Baseline Before Data Collection**

We want to tune the detection parameters of our baseline methods to give them a fair chance to compete with our neural network method.

**(1)** mask = detection\_globalThreshold(img.copy(), factor=2, pixel\_area=3)

Larger “factor” -> pixels need to have higher intensity to be detected.

Large “pixel\_area” -> a region of bright pixels need to contain “pixel\_area” of pixels to be detected as a star.

(2) mask = detection\_WITM(img.copy(), delta=-0.02, DELTA = 0.6, pixel\_area=3)

“delta” -> weight coefficient. in the range of [-0.40, 0]. smaller delta => more star detection, but more noisy.

Large “pixel\_area” -> a region of bright pixels need to contain “pixel\_area” of pixels to be detected as a star.

(3) detection\_ST16(img.copy(), threshold=2, pixel\_area=4)

“threshold” -> a pixel needs to have intensity higher than background\_value + “threshold” to be detected.

Large “pixel\_area” -> a region of bright pixels need to contain “pixel\_area” of pixels to be detected as a star.

(4)detection\_erosion\_dilation(img.copy(), gaussian\_sigma=5, average\_window\_size=10, detection\_sigma=3, pixel\_area=3)

“detection\_sigma” -> a pixel needs to have intensity higher than background\_value + “detection\_sigma” to be detected.

Large “pixel\_area” -> a region of bright pixels need to contain “pixel\_area” of pixels to be detected as a star.

More information can be found (source papers etc) inside the definition of these functions.

***Before data collection, we set the camera exposure time to 150 ms, show an arbitrary star field on the screen, and tune these parameters until we get a reasonable number of stars detected.***

* 1. **Dynamic Detection Experiment**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | |  | | |  | | | |
| **Rec** | **Prec** | **F1** | **Rec** | **Prec** | **F1** | **Rec** | | **Prec** | **F1** |
| **Neural Net** |  |  |  |  |  |  |  |  | |  |
| **Global Threshold** |  |  |  |  |  |  |  |  | |  |
| **WITM** |  |  |  |  |  |  |  |  | |  |
| **ST16** |  |  |  |  |  |  |  |  | |  |
| **Erosion Dilation** |  |  |  |  |  |  |  |  | |  |

Maintain camera temperature to “Cold”.

Integration time is fixed to 150 ms.

Unit for angular velocity is deg/s. Just plug in x, y, z into the line below. They will get converted to radians for further processing.

 w = np.array([ radians(x), radians(y), radians(z) ])

For each case we let the starfield animation running, and for each method we capture 10 images for analysis.

Run starfield simulation and detection algorithm on two separate PCs.

**2, Evaluate Centroiding Performance**

**2.1 Evaluation Metric**

Angular Distance Error (ADE) = difference between theoretical angular distance and observed angular distance, which can be used to estimate the centroid accuracy directly.

We measure Root-Mean-Square (RMS) ADE:

Shape

Description automatically generated with low confidence

Where is the *i* th observed angular distance, is the *i* th theoretical angular distance.

RMS errors = accuracy at 1 sigma

***Use the best baseline detection method from the previous experiments.***

**2.2 Dynamic Centroiding Experiment**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| **ADE (arcsec)** | **ADE (arcsec)** | **ADE (arcsec)** |
| **Neural Net** |  |  |  |
| **Gaussian Grid** |  |  |  |
| **Center of Mass** |  |  |  |

Maintain camera temperature to “Cold”.

Integration time is fixed to 150 ms.

Unit for angular velocity is deg/s. Just plug in x, y, z into the line below. They will get converted to radians for further processing.

 w = np.array([ radians(x), radians(y), radians(z) ])

Run starfield simulation and detection algorithm on two separate PCs.