STReAKS: Synthetic sTreak Rendering for sAtellite Kinematics and Surveillance

1st Léonard Flückiger
Physics MSc student
EPFL
Lausanne, Switzerland
leonard.fluckiger@epfl.ch

***Abstract*—This written report precedes the oral presentation of the project from *Optional research project in computer science II*, CS-596, from EPFL, supervised by Stephan Hellmich and Andrew Price.**

***Index Terms*—streak, synthetic, Langevin dynamics, FITS**

I. Introduction

This semester project is adapted from a proposed master project called STReAKS. It inserts itself in the objective of determining the physical properties of the space debris population. Synthetic streaks, integrated in astronomical images, can be used to train and test streak detection algorithms. The goals for this project have changed over the course of the semester, but turn out to be the following:

- Understanding of the FITS file format, the representation of astronomical images and the physical propreties of the observation stored in the metadata, rendering of FITS images.
- Elaboration of a physical model for the effect of atmospherical scintillation on observation.
- Creation of a code pipeline to insert a synthetic streak in a FITS image, whose properties are consistent with the physical properties of the observation. The langage is python, using the astropy.io package to open FITS files.

The code pipeline can be found in the EPFL Space Center Github.

II. FITS format and observation metadata

Using a FITS image from an observation from ESO and the documentation of OmegaCAM as source material, I made a number of assumptions for the final code pipeline. They are about the conventions used for the creation of the FITS files. I assumed the FITS file to be composed of a primary HDU of index 0, and then of images indexed from 1, whose data are 2D arrays of uint16. There are 32 of those images, ordered in the OmegaCam layout [1]. Using the ordering, naming the horizonzal axis x and vertical y and the *(y,x)* sorting of elements in the data array, I deduced the origin of the data array to be at the lower right (in order for visible structures to be continuous in-between ordered individual images, see Fig. 1). Useful header cards in the primary HDU are the exposure time, the seeing FWHM at start and end of the exposition (more in Sec. III), windspeed at the telescope and the filter name. An useful header card in the image HDU is the pixel to arcsec conversion. A rendering of an image can be seen in Fig. 2.

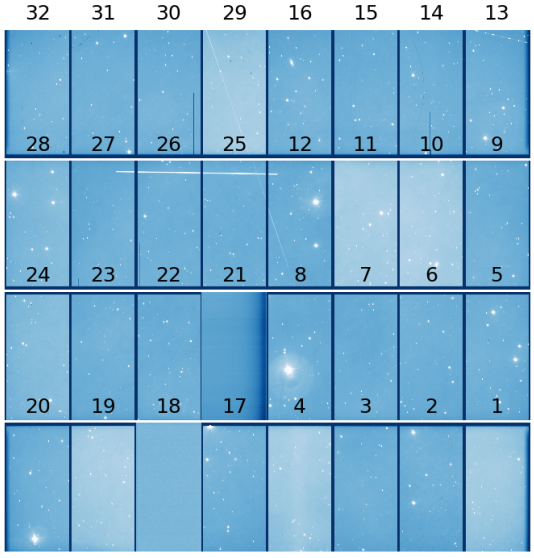

Fig. 1. Rendering of the 32 images in the ESO observation FITS file, rendered using the ZscaleInterval from astropy.io.

III. Physical model for scintillation

A. Moffat distribution for a PSF

Seen from a telescope, a star does not look like a point source, but is modelized as a Moffat distribution, with probability density function [2]

$$f(x, y; \alpha, \beta) = \frac{\beta - 1}{\pi \alpha^2} \left[1 + \left(\frac{x^2 + y^2}{\alpha^2} \right) \right]^{-\beta}, \qquad (1)$$

where α and β are parameters that basically change how wide and how steep the distribution is[1].

Atmospherical scintillation is a phenomenon that impacts observation. It can be seen by eye when watching a small light source in the distance at night, the light seems to flicker. For an astronomical image with a long exposure time, taken from the ground, this effect will turn a certain distribution,

[1]The astropy convention uses γ instead of α and α instead of β.

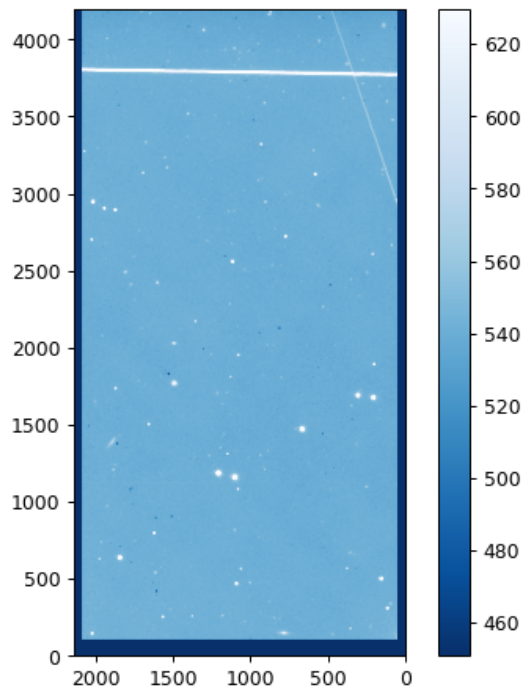

Fig. 2. Rendering of an image in the FITS file using the ZscaleInterval from astropy.io. Index image number 25 from Fig. 1. The origin is at the lower right. There are two visible streaks on the image.

like the Moffat distribution, into a Gaussian distribution over time. Such an image uses stars as fixed points for the image, and thus the Gaussian distribution seen from those images is used to extract a parameter characterizing the observation, that is the "seeing FWHM" (where FWHM stands for full width at half maximum) of this Gaussian. To have an idea of what this quantity represents, it can be pictured for a 1D distribution as the distance on the x-axis between the two points where the height of the distribution is half of its maximal value (this quantity only makes sense for a distribution that monotoneously decreases away from its maximal value). In the case of a Gaussian distribution, it is related to the standard deviation σ by the relation [3]

$$FWHM = 2\sqrt{2\ln 2}\sigma. \tag{2}$$

B. Langevin dynamics

In order to model a behaviour for the scintillation, thus the time evolution of an observed point source, it is necessary that the position histogram of this time evolution be distributed like a Gaussian. The path a scintillating point source takes appears to be random, except that it stays around its "real" position. I have therefore adapted Langevin dynamics as a model for the observation scintillation. The Langevin dynamics, and its resulting properties, were seen in the Biophysics class [4]. The Langevin dynamics in 1D is the evolution of the position $x(t)$ in time, given by the differential equation

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \partial_x V(x) + \eta(t), \tag{3}$$

where ∂_x represents the partial derivative in x (could be total derivative but this way allows for higher dimension generalization), $V(x)$ is a potential term and $\eta(t) : \mathbb{R} \to \mathbb{R}$ is the "noise" term, a random variable that has properties

$$\langle \eta(t) \rangle = 0 \qquad \langle \eta(t)\eta(t') \rangle = 2D\delta(t - t'), \tag{4}$$

where D $[m^2/s]$ is the diffusion coefficient, and $\delta(x)$ the Dirac delta, with $\delta(x \neq 0) = 0$ and $\int_{\mathbb{R}} \delta(x)\mathrm{d}x = 1$. This means $\eta(t)$ is a normally distributed random variable of expectancy 0, no correlation over time, and variance $2D$. For Langevin dynamics with $t \to \infty$, the distribution of the position of $x(t)$ becomes proportional to

$$\exp \frac{-V(x)}{D}. \tag{5}$$

C. Langevin dynamics for Gaussian seeing distribution

Therefore, for a certain potential $V(x)$ that allows the distribution position in time to be Gaussian, and assuming the behaviour of scintillation to be a random walk within a potential well, Langevin dynamics can be such a model.

To adapt Eq. 5 to a Gaussian, we need to use a harmonic potential well for $V(x)$

$$\exp \frac{-V(x)}{D} \equiv \exp \frac{-x^2}{2\sigma^2}, \tag{6}$$

hence

$$V(x) = D\frac{x^2}{2\sigma^2}. \tag{7}$$

Therefore, σ and D become parameters of the model. There will be need later on to link D to a physical parameter. The 2D extention of those equations is simply

$$\frac{\mathrm{d}\vec{x}}{\mathrm{d}t} = -\vec{\nabla}V(\vec{x}) + \vec{\eta}(t), \tag{8}$$

with a diagonal constant covariance matrix σ, the notation stays the same.

A discrete and finite-time simulation of this process is represented in Fig. 3, with method described in Sec. IV.

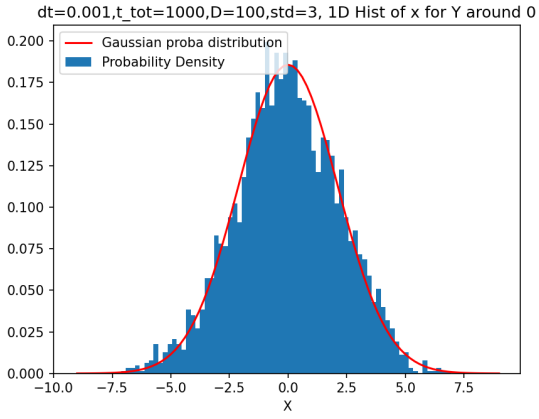

Fig. 3. Simulation of the time evolution of Langevin dynamics inside a two-dimensional harmonic potential well, with given parameters in the figure. A slice of the position histogram around the y-axis resembles very closely to a Gaussian (unnormalized).

IV. Python implementation of the model

A. Integration process

To implement the time evolution of $x(t)$ in Python, there is need to discretize the process over time steps $\mathrm{d}t$. The gaussian random variable thus takes as its properties

$$\left\langle \eta(\vec{t}) \right\rangle = \vec{0}, \qquad \left\langle \eta(\vec{t})\eta(\vec{t'}) \right\rangle = \frac{2D}{\mathrm{d}t}\delta_{t,t'}, \tag{9}$$

with $\delta_{\alpha,\beta} = 0 \quad \forall \alpha \neq \beta$ and $\delta_{\alpha,\alpha} = 1$ the Kronecker delta. For defining a normally distributed random variable in Python in two dimensions with a variance $2D$, one can simply generate a one-dimensional normal distribution, taken from the package numpy, of variance D in both dimensions.

To solve the differential equation, Euler integration would be straightforward

$$\mathrm{d}\vec{x} = \left(-\vec{\nabla}V(x) + \eta(\vec{t})\right)\mathrm{d}t \tag{10}$$

but leads usually to more instabilities than fourth order Runge-Kutta over the same time step. The RK4 implementation looks like this over the x axis

$$\begin{aligned}
x_{n+1} &= x_n + \frac{\mathrm{d}t}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) \\
f_x(t,x) &= -V_x'(x) + \eta_x(t) \\
k_1 &= f(t_n, x_n) \\
k_2 &= f\left(t_n + \frac{\mathrm{d}t}{2}, x_n + \mathrm{d}t\frac{k_1}{2}\right) \\
k_3 &= f\left(t_n + \frac{\mathrm{d}t}{2}, x_n + \mathrm{d}t\frac{k_2}{2}\right) \\
k_4 &= f(t_n + \mathrm{d}t, x_n + \mathrm{d}t k_3),
\end{aligned}$$

and similarly over the y axis. For evaluation of the "noise" term η over "half time-steps" like seen in k_2 and k_3, one needs simply use $\mathrm{d}t/2$ instead of $\mathrm{d}t$ in the normal random variable.

B. Behaviour

This integration process answers a simple quality check, scale consistency of its behaviour. Aspects of its general behaviour are its stability and its exploration of the space. Instability is characterized by very fast divergence, as in Fig. 4, and tends to happen for greater $\mathrm{d}t$ and greater D. Instability will be detected by checking whether a value is greater than 100 times σ. This proves sufficient in practice as divergence happens very fast and breaks through that threshold in only a couple of steps.

Exploration of the space can rather intuitively be explained as the tendency of the simulation to be able to walk around enough for its position histogram to be more or less gaussian. If it doesn't have enough time to explore, or if the diffusion coefficient is too small for the given time, the simulation might not explore the space at its disposal "enough" and its variance will not be consistent with the expected parameters.

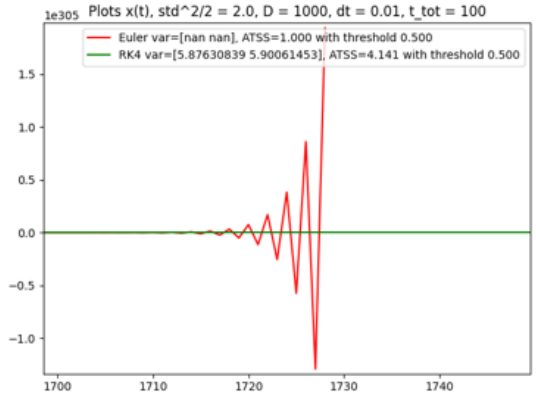

Fig. 4. Representation of a simulation for the Euler and RK4 integrators for a certain set of parameters over the x axis only. RK4 stays stable whereas Euler diverges, for the same run of random variables. For RK4, the variance over the x and y dimensions are greater than the value of 2.0 (std^2/2), which hints to unwanted behaviour, even though stable. The meaning of ATSS and threshold will be explained later on.

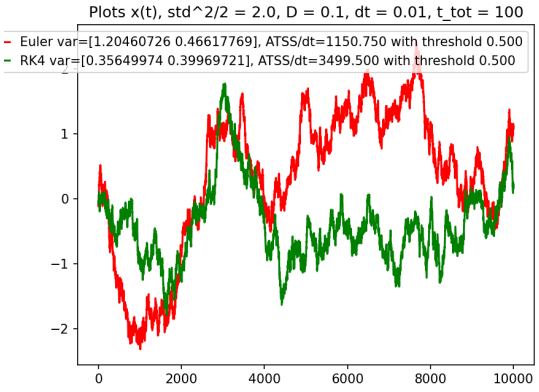

Fig. 5. Representation of a simulation for the Euler and RK4 integrators for a certain set of parameters over the x axis only. For both integrators, the variance over the x and y dimensions are smaller than the value of 2.0 (std^2/2), meaning it could not explore the space at its disposal enough. The meaning of ATSS and threshold will be explained later on.

C. D and scintillation timescale

The forementioned scale consistency means that the simulation should not be able to distinguish between different input parameters as long as they follow a certain transformation. The behaviour of the simulation is its stability (whether it diverges or not) and filling of the space (whether the variance of the simulation is around the value expected from the input). In general, for different runs with the same parameters, the behaviour stays the same. And it does too when the relationships $t_{tot}/\mathrm{d}t$ and $t_{tot} \cdot D$ are conserved. The first relationship is the number of steps done by the simulation (no unit), and the second one is the capacity of the walk to diffuse $[m^2]$. Indeed, the physical unit of D is $[m^2/s]$, multiplying it by the total time of the simulation preserves the units.

All the variables in the simulation process have a physical equivalent in an astronomical image, in a real streak, except for D. This parameter was introduced in the template Langevin dynamics, as a link between the variance of the "noise" and

the steepness of the resulting histogram. There is however another characteristic in a streak that can be matched to the diffusion coefficent D, that was called "scintillation timescale" in a paper: *"In some cases, however, the length scale of the wobbles can be used to estimate the angular velocity of the streak. For example, we expect that the wobble timescale will be of the order of the scintillation timescale (e.g., the telescope diameter divided by the wind speed)."* [5].

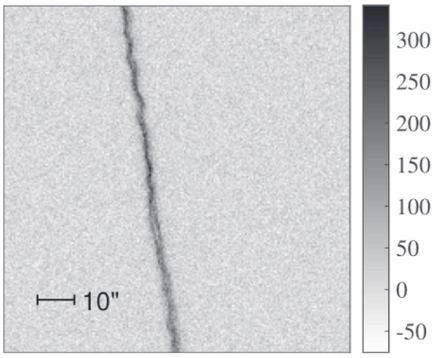

Figure 11. The same streak as shown in Figure 10(b) enlarged around the center of the streak. The wobbles in the streak are likely caused by tracking errors and more importantly by the turbulent atmosphere (astrometric scintillation). The wobble length scale is about 10 arcsec. Assuming the atmospheric scintillation timescale of about 10 ms, the expected angular speed of the object is on the order of 1000 arcsec s^{-1}. This is close to the actual angular speed of 788 arcsec s^{-1}, measured from the length of the streak (in Figure 10(b)) divided by the exposure time.

Fig. 6. The quantity called "wobble lengthscale" seems to be the length of the greatest visible oscillation cycle. [5]

As the "scintillation timescale" can be determined in an image using the assumption that it is the telescope diameter divided by the wind speed, it can be used to determine the according value of D needed for a simulation (if there is a correlation between the two, which there is). As the link between scintillation timescale and D can only be determined in the other way, that is running simulations with a different D, measuring the scintillation timescales, and choosing the value of D whose simulation gave a scintillation timescale closest to the target value. Now arises the question on how to measure a scintillation timescale for a given simulated streak. For both methods presented below, the analysis is done only on one axis of the two-dimensional motion, assumed equivalent to any radial projection by symmetry. It is also assumed that the behaviour of the axial projection is sufficient to represent the oscillation periods, in the sense that the perpendicular motion cancels out in average.

One approach can be to do a Fast Fourier Transform (FFT) of the streak. The position of the highest peak in Fourier space would correspond to the main oscillation frequency, thus period, thus oscillation timescale. This analysis however, probably due to the extremely noisy behaviour of the simulation, proves to be extremely inconsistent. Different simulations with the same parameters give way to main frequencies with orders of magnitude of difference. This method also often seems to produce results that are way off the oscillation timescale one would deduce by eye.

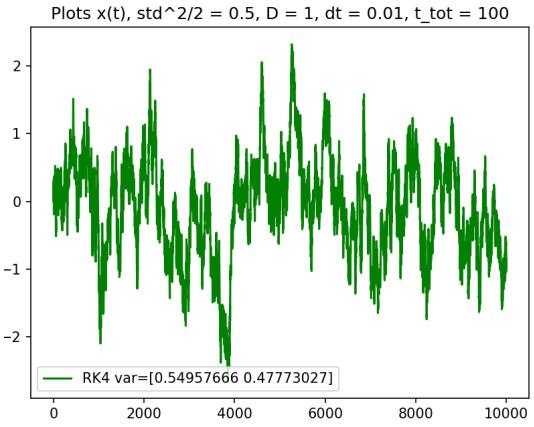

Fig. 7. Langevin dynamics of parameters in the figure. The highest peak in Fourier space from Fig. 7 would correspond to a main period of 3333.667 on the x-axis of this graph, meaning 3 main periods in this whole figure.

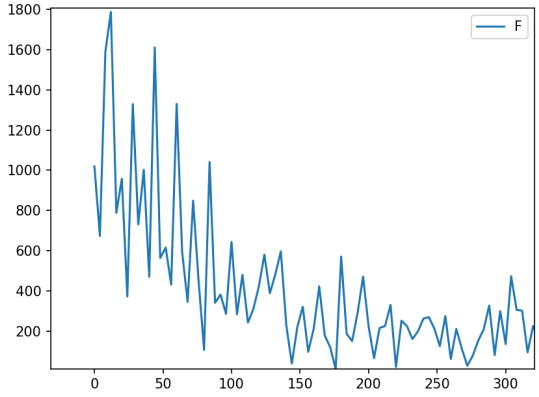

Fig. 8. FFT of the motion represented in Fig. 7. In-between runs of the same parameters, many peaks compete to be the highest, sometimes slightly, corresponding to sometimes very different frequencies.

D. Estimation process for D

The second method, that I implemented in the final pipeline, was to extract a value that I called ATSS (average time on same side). To extract it, compute all times where the simulation stays in the positive (or negative) values before taking a negative (or positive) value, and take the average of those times. Twice that value would be the estimation for scintillation timescale. However, it appears that due to the very noisy nature of the simulations, this resulting value becomes way smaller than expected. Adding a threshold that the simulation must pass before the "crossing" is acknowledged can make it less susceptible to be perturbated by very small oscillations. The threshold I chose is $\sigma/3$, which proved a good match for various simulations between the estimated value - that was consistent - and by-eye approximation. This means if the simulation is in the negative values, the crossing will not be

counted until it reaches at least $\sigma/3$, and $-\sigma/3$ in the positive case.

The forementioned correlation between D and ATSS holds most of the time: for other parameters the same, an increase in D decreases ATSS and vice-versa. The estimation process for D, for a given scintillation timescale, is as follows

- Start with a certain D, and a certain $\mathrm{d}t$. For any run, if it diverges, rerun it with a smaller $\mathrm{d}t$ ($\mathrm{d}t$/2 in my implementation) until it converges. Store (D,ATSS), as well as the $\mathrm{d}t$ value used.
- If the smallest stored ATSS is greater than the target value, rerun with greater value of D as the associated D (10·D in my implementation). And similarly if the greatest stored ATSS is smaller than the target value. Special case to be treated first, if all ATSS are infinity, run with greater D as the greatest D.
- If the ATSS value of a simulation falls between two stored ATSS values, do a simulation with D the mean of the two associated D values.
- Stop after a fixed number of steps, output the D value that has an ATSS the closest to the target value. Also output the used $\mathrm{d}t$.

This way, for an input scintillation timescale (twice the value of associated ATSS), there is an estimation of D and a value of $\mathrm{d}t$ that makes the simulation stable.

E. Pipeline inputs

The required inputs for our streak creation pipeline are its brightness [magnitude], apparent velocity [rad/s], starting coordinates [pixels,pixels] and angle [rad] in the trigonometrical convention, as well as values for α and β. The streak will be drawn in a total number of steps $t_{tot}/\mathrm{d}t+1$, each step printing on the image a spot in the shape of a Moffat distribution. To determine where the center of the Moffat is at a certain step, it is the addition of a linear term and the Langevin dynamics. The first term is given by the time sampling of a line, from starting point and given angle and length deduced from the apparent velocity. The second one is simply the value of the Langevin dynamics at that time step. Adding those two "vectors" gives the position of the center at a certain time step. There only remains to determine what the intensity of those Moffat spots should be.

To determine the number of pixel values to add to each Moffat spot, one needs to find a way to convert the input brightness. The number of electron counts for a value of 1 in a pixel is 2.5. The conversion from apparent magnitude to electron flux N_* is [6]

$$N_* = N_{20} 10^{-m_{app}-20/2.5}, \tag{11}$$

with N_{20} the flux of a mag20 source, and m_{app} the apparent magnitude. To determine N_{20}, the OmegaCAM Exposure Time Calculator [7] is used for a time of 1 second, given a certain filter, with the assumption that the airmass is 1.5. A table of values is then stored in order to output N_{20} for an input filter (from ESO), found in the FITS metadata. This way, one can find the values that each Moffat dots add to the image. To vastly increase efficiency, the Moffat dots are cropped in squares. Only the values greater than $1/(2 \cdot (t_{tot}/\mathrm{d}t + 1))$ are kept (threshold of significance, if all spots would add that value then it would result in 0.5), the rest is cropped out.

Values for α and β are left as parameters, for real streaks they can be extracted if one needs to create a synthetic streak with those same parameters.

The maximal value of $\mathrm{d}t$ used for a streak drawing is such that in average, there are $precision$ Moffat centers for a distance of one pixel in the image, with $precision$ defaulting to 50. In the end, $\mathrm{d}t$ can be smaller if the simulation with the original $\mathrm{d}t$ diverges.

F. Running the code

The main different assumptions that were made for the synthetic streak drawing pipeline are

- Constant arcsec to pixel conversion, that the conversion matrix is a constant times Id2.
- The seeing FWHM used is constant in the drawing, and the mean of the extracted values in the metadata (start FWHM and end FWHM).
- Stability of simulated Langevin dynamics is only dependent on its input values and not on chance.

This whole process gives a code pipeline, that for a couple of input parameters allows to draw a synthetic streak in a FITS image directly in the file. Fig. 9 and 10 show a synthetic streak inserted in the same FITS image than the previous Fig. 2, with the streak creation parameters detailes in legend of Fig. 9.

Looking at Fig. 1 and estimating by eye the parameters of the streak spanning through images 27, 26, 25 and 12 (most notably its length), I use those parameters to create a synthetic streak close to it that should in principle look similar, in Fig. 11. The artificial streak however seems to wobble more than the real one.

V. Discussion

This streak drawing pipeline is functional to modify FITS images, extracting meaningful data and using it to draw an artificial streak, all that in reasonable time (on my 2.3 GHz CPU it took around 1mn for the first example, half of the time being the simulations made to find the better value of D. For an example with wider drawing, the runtime can go quite higher, around 10mn for the second example). The numerical simulation of the Langevin dynamics turns out to be stable, having sufficient precision and still being computed in a reasonable number of steps. The by-eye comparaison between an artificial streak and a real one that should have the same characteristics shows that the process can clearly be improved. The most crude approximation made in the process is the definition and estimation of the scintillation timescale. The way to extract this information from a streak as well as the estimation process from the observation parameters (telescope diameter divided by windspeed) can clearly be improved.

Concerning the code aspect, there are certain improvements that could easily be made to the pipeline:

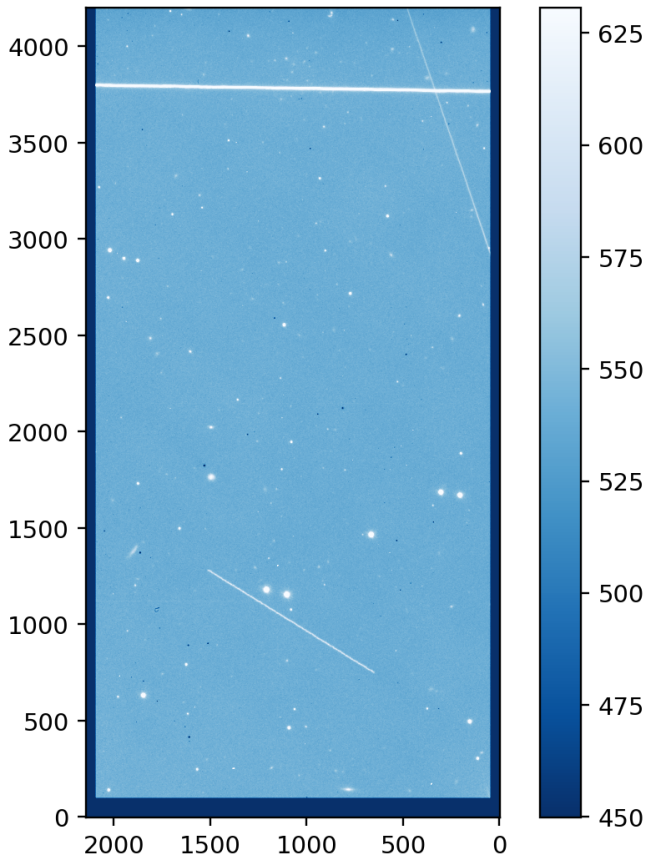

Fig. 9. Artificial streak generated on the same FITS image than Fig. 2 (the bottom streak is the artificial) with input parameters (as they are named in the code) brightness=15, appVel=1.4e-5, start=(650,750), angle=np.pi·3/17, alpha=3, beta=2, apertureDiam=2.6, drawingPrecision=20. The seeingStd (obtained through seeing FWHM) is 1.452, exposure time 75s, thus the used values for the simulation were around D=7.9, $\mathrm{d}t = 0.003$ corresponding to ATSS=0.424, for target ATSS being 0.429.

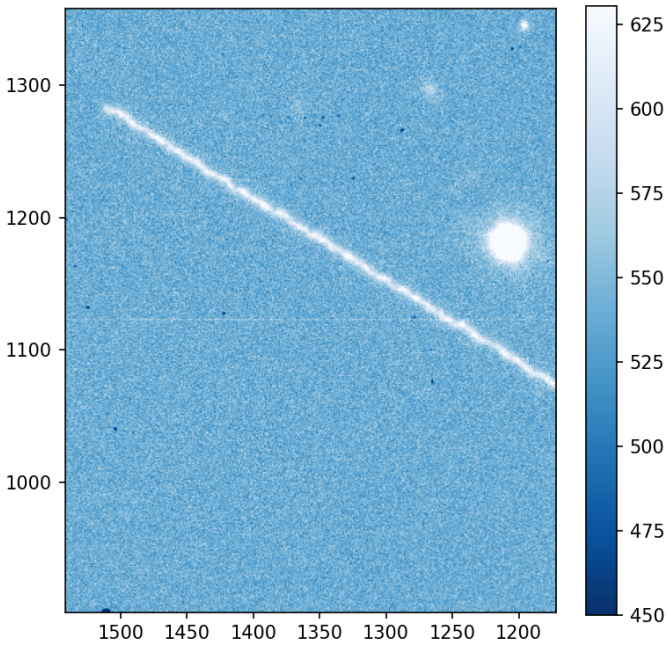

Fig. 10. Zoom on the artificial streak from Fig. 9.

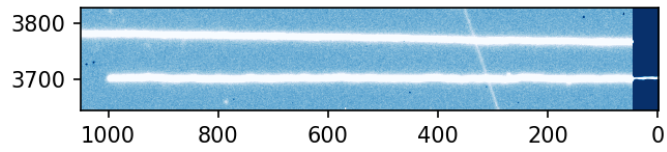

Fig. 11. The parameters of the artificial horizontal streak (the bottom one) are brightness=11, appVel=7e-5, start=(1000,3700), angle=np.pi, alpha=4, beta=2, apertureDiam=2.6, drawingPrecision=10. The seeingStd (obtained through seeing FWHM) is 1.452, exposure time 75s, thus the used values for the simulation were around D=7.19, $\mathrm{d}t = 0.001$ corresponding to ATSS=0.427, for target ATSS being 0.429.

- Betterment of the integration instability criterion. An idea would be to run for 100 (for example) additional "invisible" steps and check if the threshold is exceeded at that stage.
- If not implementing the previous point, simply checking in the final drawing function if there was a divergence and redo the drawing process with higher precision if it was the case. Currently, this is only done to find parameters that should lead to a stable drawing, but there is no check on the actual final drawing.
- A simple addition would be to accept a modulation in intensity over time, in the form of a function (or an array as long as the number of steps), that for a certain time give a modulation of the intensity (between 0 and 1), that could be used to simulate the evolution of a rotating body.
- Instead of using a constant seeing FHWM that is the mean of start and end of the observation, there could be a liner evolution of FHWM between start and end, which should be a better approximation. The stability tests should be done with the smallest FWHM value.

An actual follow-up of this project, with the time evolution of intensity option added, would be to actually analyze synthetic streaks to compare them to real streak images, and find out how good the Langevin dynamics process is to simulate scintillation.

References

[1] K. Kuijken, OmegaCAM user manual. OmegaCAM Instrument Consortium, 2011.
[2] Moffat distribution, Wikipedia, consulted on June 27, 2024.
[3] FWHM, Wikipedia, consulted on June 27, 2024.
[4] S. J. Rahi, Class Biophysics: physics of biological systems (PHYS-302), EPFL Lausanne, 2023.
[5] G. Nir, B. Zackay, E. O. Ofek, "Optimal and Efficient Streak Detection in Astronomical Images", page 9, 2018.
[6] S. Hellmich, E. Rachith, B. Y. Irureta-Goyena Chang, J-P. Kneib, "Harvesting large astronomical data archives for space debris observations", Laboratory of astrophysics, EPFL, 2023.
[7] OmegaCAM Exposure Time Calculator, consulted on June 27, 2024.