3DTrackSim Detailed READ ME

Overall

Numerical simulations of fully 3D active-feedback single-particle tracking for comparisons between combined online Bayesian and windowed estimation of background and signal localization (COBWEBS) and Kalman mapped gaussian assumed density filter localization.

For additional conceptual information see 2D algorithm development, https://doi.org/10.1063/5.0118317 and 3D communication conference proceeding,

Contents

Tracking Simulation Core Functions

To simulate a single diffusive trajectory use one of these 3 functions depending on your desired position estimation strategy.

Simulate a single trajectory using fully Kalman tracking.

Function: track_Kalman_3D(D,s,bofr,r,ki,kiz,N,tau,varargin)

Required Inputs:

D: Desired diffusion coefficient of particle in meters squared per second.

- Typical values for this system are on the order of 10^{-12}.

s: max particle count rate in counts per second (cps).

- Typical values (2.8×10^5, 7.0×10^7) cps which correspond to observed particle intensities of (1.2×10^4, 3.0×10^6)

bofr: background intensity in cps on each side of any defined radii.

- Typical values (0, 1×10^6).

r: radial position of background intensity changes in meters relative to initial particle position.

- To pass even background tracking environments use inputs:

bofr	Desired b in cps
r	[]

- To pass a dot environment where particle intensity goes from 1000 cps in the center to 2000 cps outside 2 μm from its initial position use inputs:

bofr	[1000,2000]
r	[2e-6]

ki: integral feedback control parameter along X/Y.

kiz: integral feedback control parameter along Z.

N: number of steps desired to run given trajectories.

- On our computers trajectory computation duration was ~14 seconds per 1 second of simulated data at 20 us bins.

Tau: Bin time in seconds.

- Recommended value: 20×10^{-6} s.

Optional inputs:

Pass as name value pairs at end of function inputs in place of varagin.

- Ex. track_Kalman_3D(D,s,bofr,r,ki,kiz,N,tau,'send',1000)

'send': If not defined value defaults to s and particle intensity is constant for the entire trajectory.

If enabled by passing a non-zero lower value than the initial intensity, the s value for each step of the trajectory is defined to exponentially decay to the desired value by the end of the trajectory using the equation where sinit is the input s value, send is the final s value, and k is the given step index.

$$s(k) = s_{init} * (\frac{s_{end}}{s_{init}})^{k/N}$$

'Dest': is the D estimate used in calculation of particle positions. Units remain m^2/s. If not specified, Dest defaults to the true value used to generate particle positions.

Outputs:

trkerr: (xy, z, xyz) absolute difference between particle position and stage position along the specified axes.

$$tracking\ error\ =\ \sqrt{(Stg_x - P_x)^2 + \left(Stg_y - P_y\right)^2 + (Stg_z - P_z)^2}$$

len: length of trajectory in steps

tau: final trajectory bin time after any coercion

mleerr: absolute difference between particle position and estimated particle positions. Where Pest along a given axis is the sum of stg position and particle position.

$$position\ estimation\ error\ =\ \sqrt{\left(P_{x,Est.} - P_x\right)^2 + \left(P_{y,Est.} - P_y\right)^2 + \left(P_{z,Est.} - P_z\right)^2}$$

photons: observed photons for each bin

stg_out: x, y, z stage position/laser scan center position in m

part_out: x, y, z particle positions in m

kalm_out: x, y, z position estimate relative to stage center

N: max number of steps in trajectory

sb: count rate of signal, and background photon for each trajectory bin.

aoe: axis of escape coding. Reports (if applicable) along which axis particle position was more than 0.5 μm from the edge of the laser scan area. 1 for x, 2 for y, 3 for z, 4 for particle reached full duration.

Simulate a single diffusive trajectory using COBWEBS tracking along XY paired with Kalman tracking along Z

Function: `track_XYBayesZKalm(D,s,bofr,r,ki,kiz,N,ogtau,varargin)`

Required Inputs:

D: Desired diffusion coefficient of particle in meters squared per second.
s: max particle count rate in counts per second (cps).
bofr: background intensity in cps on each side of any defined radii.
r: radial position of background intensity changes in meters relative to initial particle position.
ki: integral feedback control parameter along X/Y.
kiz: integral feedback control parameter along Z.
N: number of steps desired to run given trajectories.

- On our computers trajectory computation duration was ~22 seconds per 1 second of simulated data at 20 us bins.

ogtau: Bin time in seconds prior to coercion.

Optional inputs:

Pass as name value pairs at end of function inputs in place of varagin.

'sbest': Defaults to true. To turn off COBWEBS signal and background estimation and give particle localization perfect information about signal and background intensities, pass false.
'send': If not defined value defaults to s and particle intensity is constant for the entire trajectory.

If enabled by passing a non-zero lower value than the initial intensity, the s value for each step of the trajectory is defined to exponentially decay to the desired value by the end of the trajectory as described in Kalman tracking documentation.
'taucoerce': Defaults to true. Disabling turns of bin time coercion and forces bin time during tracking simulation to remain at the initially defined value.
'Destxy': is the D estimate used in calculation of particle positions along X/Y. Units remain m^2/s. If not specified, Dest defaults to the true value used to generate particle positions.
'Destz': is the D estimate used in calculation of particle positions along Z. Units remain m^2/s. If not specified, Dest defaults to the true value used to generate particle positions.

Outputs:

trkerr: (xy, z, xyz) absolute difference between particle position and stage position along the specified axes.
len: length of trajectory in steps
tau: final trajectory bin time after any coercion
mleerr: absolute difference between particle position and estimated particle positions. Where Pest along a given axis is the sum of stg position and particle position.
photons: observed photons for each bin
stg_out: x, y, z stage position/laser scan center position in m
part_out: x, y, z particle positions in m
posest_out: x, y, z position estimates relative to stage center.
N: max number of steps in trajectory
sbandests: signal, background, signal estimate, and background estimates for each bin.
aoe: axis of escape coding. Reports (if applicable) along which axis particle position was more than 0.5 μm from the edge of the laser scan area. 1 for x, 2 for y, 3 for z, 4 for particle reached full duration.

Simulate a single diffusive trajectory using COBWEBS tracking along XY paired with COBWEBS tracking along Z

Function: track_XYBayesZBayes(D,s,bofr,r,ki,kiz,N,ogtau,varargin)

Required Inputs:

D: Desired diffusion coefficient of particle in meters squared per second.
s: max particle count rate in counts per second (cps).
bofr: background intensity in cps on each side of any defined radii.
r: radial position of background intensity changes in meters relative to initial particle position.
ki: integral feedback control parameter along X/Y.
kiz: integral feedback control parameter along Z.
N: number of steps desired to run given trajectories.

- On our computers trajectory computation duration was ~48 seconds per 1 second of simulated data at 20 us bins.

ogtau: Bin time in seconds prior to coercion.

Optional inputs:

Pass as name value pairs at end of function inputs in place of varagin.

'sbest': Defaults to true. To turn off COBWEBS signal and background estimation and give particle localization perfect information about signal and background intensities, pass false.
'send': If not defined value defaults to s and particle intensity is constant for the entire trajectory.

If enabled by passing a non-zero lower value than the initial intensity, the s value for each step of the trajectory is defined to exponentially decay to the desired value by the end of the trajectory as described in Kalman tracking documentation.
'Dest': is the D estimate used in calculation of particle positions. Units remain m^2/s. If not specified, Dest defaults to the true value used to generate particle positions.

Outputs:

trkerr: (xy, z, xyz) absolute difference between particle position and stage position along the specified axes.
len: length of trajectory in steps
tau: final trajectory bin time after any coercion
mleerr: absolute difference between particle position and estimated particle positions. Where Pest along a given axis is the sum of stg position and particle position.
photons: observed photons for each bin
stg_out: x, y, z stage position/laser scan center position in m
part_out: x, y, z particle positions in m
posest_out: x, y, z position estimates relative to stage center.
N: max number of steps in trajectory
sbandests: signal, background, signal estimate, and background estimates for each bin.
aoe: axis of escape coding. Reports (if applicable) along which axis particle position was more than 0.5 μm from the edge of the laser scan area. 1 for x, 2 for y, 3 for z, 4 for particle reached full duration.

Dependencies for Tracking Core Functions

galvoimpulseresponse.mat experimentally measured step response function from galvo mirror along XY

Impulse_Response.txt experimentally measured step response from piezo along XY. Piezo response was assumed to be isotropic and performance extrapolated to Z here.

Statistics and Machine Learning Toolbox – used to generate photons observed in a given bin from pseudo-random numbers sampled from poisson distributions. Unavoidable dependency without rewriting photon generation to derive observed photon counts from built in uniform distribution pseudo random number generation.

Signal Processing Toolbox – used to resample measured step response functions to coerced tau. Will not apply to Kalman tracking at 20 us bin time. Can be avoided in COBWEBS Kalman tracking if needed by disabling bin coercion.

Batch Files for optimizing relevant parameters and comparing batches of multiple trajectories.

WEBS signal background estimation coefficient generation from calibration particle position distribution.

Script: WEBS_weightingconversiongen.m

To calculate the weighting coefficients associated with a given position calibration distribution, modify the values of sigmaxy and sigmaz and filename to the desired values then run the script. The variable weight defined by the script will export a 3x2 matrix. The top row is the proportion coefficients for the center pixel, the middle row is the proportion coefficients for the corner pixels, the bottom row is the overall maximum count rate of a given signal under knight's tour illumination.

Optimize Ki values along each axis for selected even background cases.

Generate Data

Script: explore3DKi.m

About: Simulate trajectories over a range of integral feedback control values. This code batches and sequentially runs a series of particle diffusive speeds and intensities for selected algorithm. Data is then aggregated and automatically saved.

How to use: Algorithm selection performed by redefining alg at top of script.

value of alg	Associated algorithm
1	Kalman xy, z
2	COBWEBS xy + Kalman z
3	COBWEBS xy + COBWEBS z

During initialization, code opens user interface to select desired save folder for data. Code generates a subfolder for each algorithm then saves the aggregated raw data. File name is controlled through algorithm selection/simulation datas combined with the manually defined datename as a prefix describing date of data generation to prevent accidental saving over data.

Visualize Data

Script: temp220830_individualcaseKiOptimizationplots.m

About: Visualize performance at tested ki X/Y and ki Z combinations for each individual particle explored.

How to use: To use code load saved data from explore3DKi for a given algorithm into workspace then run script. Uncomment title field to understand which plot corresponds to which particle parameters.

Example plot output: