UDC 519.21+623.466 doi: 10.20535/2411-2976.12024.55-65

MODEL OF RANDOM-LIKE PLANAR TRAJECTORIES WITH INTERSECTIONS

[1] Vadim V. Romanuke, [2] Michał P. Pabich

[1] Vinnytsia Institute of Trade and Economics of State University of Trade and Economics, Ukraine
[2] ENTRAST, Sp. z o.o., Poland

Background. Recently the task of detecting and identifying trajectories of objects whose genuine purposes are uncertain or strike threatening has become extremely important. The known approaches produce insufficiently smooth trajectories.

Objective. The purpose of the paper is to build a model of generating random-like planar trajectories, which would have sufficiently smooth curves. A trajectory may have self-intersections and may intersect other trajectories.

Methods. Preliminarily two starting points on a plane are generated. The distance and angle between these points are calculated, which then are successively updated to calculate new trajectory points using the polar coordinate system. A trajectory of N points is generated using $4N-4$ values of normally distributed random variables with zero mean and unit variance and four values of $(0;1)$ -uniformly distributed random variables.

Results. The random-like trajectory generator has the same time complexity as its predecessors, including the direction randomization generator and its modifications. Exemplary trajectories appear very realistic. Self-intersections are important to manoeuvre and confuse the opponent side. The trajectory has four parameters to adjust its heading, scattering of points, and intensity of turns and twists. These parameters serve as magnitudes to amplify the respective properties. The highest influence has the angle-scattering parameter. Four simple conditions can be embedded to fit the trajectory within a rectangular domain.

Conclusions. The suggested model should serve either for generating trajectory datasets to train manoeuvring-object detectors on them or for masking reconnaissance. The model allows balancing the trajectory smoothness and randomness.

Keywords: *object observation; random trajectory; random path; heading; polar coordinate system; manoeuvrability.*

1. Planar trajectory modelling

Detecting and identifying trajectories of objects whose genuine purposes are uncertain or strike threatening is quite an important task. Until recently this task had been not so crucial as it became since newly developing air threats [1], [2]. Obtaining more information about new-discovered non-static objects is crucially needful to maintain confidence and safety.

Before detection and identification [3], [4], the trajectory should be modelled in order to study its specificities and probable bottlenecks of the detector and identifier [2], [5]. The trajectory can be planar or three-dimensional. The latter is usually obtained with more enhanced radar systems [6], [7].

Both types of the trajectory left by objects, which are intended and generally designed to move as more non-revealed as possible, appear to be random-like [1], [4], [8]. From the first glance, this could have been called pseudorandomness [9], [10], but the object trajectory may heavily fluctuate due to human intervention rather than to truly random influences (like, e. g., from weather conditions) [11], [12]. That is why such trajectories ought to be called random-like.

Obviously, the planar trajectory is far simpler to study. However, the two main principles to model random-like trajectories are the same for both planar and three-dimensional case. First, the neighbouring points (which are the object positions registered) are not equally distanced. Second, any heading changes of a real-world object trajectory are not truly abrupt [13]. They only may seem abrupt due to a disadvantageous view presentation or insufficiently frequent registrations of the object observation while it is tracked.

There are a few known approaches to model random-like planar trajectories or paths. They include (in order of improving smoothness): correlated random walk [14], [15], segment models [16], [17], random utility inverse reinforcement learning [18], probabilistic approaches for connecting two points [19], non-smooth discrete element method [11], manoeuvring modelling group model [20], time series segmentation and clustering analysis [17], etc. However, none of these approaches provide sufficient smoothness and controllability of trajectory generation.

2. Goals and tasks to achieve it

Given a starting planar point and a number of oncoming planar trajectory points, the goal is to build a model of generating random-like planar trajectories, which would have sufficiently smooth curves. A trajectory may have self-intersections and may intersect other trajectories. To achieve the goal, the approach with randomizing direction or heading is formalized first. Then, motivated by the lack of smoothness, another approach to produce much smoother and controllable trajectories is to be formalized. The respective computer simulations should be discussed

and an appropriate conclusion is to be made.

3. Direction randomization

The approach with direction randomization relies on using values of normally distributed random variables (NDRVs) and values of $(0;1)$-uniformly distributed random variables (UDRVs). If the total number of trajectory points is N, then values

$$\{\xi_1, \xi_2\},\ \{\zeta_1, \zeta_2\},\ \{\theta_i\}_{i=2}^{N-1},\ \{\vartheta_i\}_{i=2}^{N-1} \tag{1}$$

of four independent NDRVs with zero mean and unit variance (SNDRVs) are used, and values

$$\{\eta_1, \eta_2\},\ \{\mu_1, \mu_2\},\ \{\rho_i\}_{i=2}^{N-1},\ \{\upsilon_i\}_{i=2}^{N-1} \tag{2}$$

of four independent UDRVs are used. Denote the set of points of the trajectory by

$$\{\mathbf{P}_i\}_{i=1}^{N} = \{[x_i \quad y_i]\}_{i=1}^{N} \subset \mathbb{R}^2, \tag{3}$$

where point

$$[x_i \quad y_i]$$

follows point

$$[x_{i-1} \quad y_{i-1}] \text{ for } i = \overline{2, N}.$$

First, the starting point of the trajectory $[x_1 \quad y_1]$ is generated as

$$x_1 = a\xi_1 + s\eta_1,\ y_1 = a\zeta_1 + s\mu_1, \tag{4}$$

where $a > 0$ is a magnitude of the normal noise intended to scatter points. The scattering is severer as a is set to a greater value. Value $s \in \mathbb{R}$ is used to amplify the random offset. The following point is generated as

$$x_2 = a\xi_2 + s\eta_2,\ y_2 = a\zeta_2 + s\mu_2. \tag{5}$$

The initial direction along horizontal axis is determined as

$$d_x = \operatorname{sign}(x_2 - x_1). \tag{6}$$

The initial direction along vertical axis is determined in the same way:

$$d_y = \operatorname{sign}(y_2 - y_1). \tag{7}$$

Then, using a constant probability p of the direction change, the following steps are executed for $i = \overline{2, N-1}$ with already determined points (4) and (5): if $\rho_i < 1 - p$ then

$$d_x = -\operatorname{sign}(x_i - x_{i-1}); \tag{8}$$

else if $\upsilon_i < 1 - p$ then

$$d_y = -\operatorname{sign}(y_i - y_{i-1}). \tag{9}$$

Then

$$x_{i+1} = x_i + ad_x \cdot |\theta_i|,\ y_{i+1} = y_i + ad_y \cdot |\vartheta_i| \quad \text{for } i = \overline{2, N-1}. \tag{10}$$

The direction randomization approach is adjusted with its three parameters

$$a > 0,\ s \in \mathbb{R},\ p \in (0;1). \tag{11}$$

Nevertheless, the factual qualitative influence is made by magnitude a and probability p. An example of the planar trajectory of 102 points by $p = 0.1$ and $a = 0.52$ is presented in Fig. 1 (here and below the starting point is marked with square). Another trajectory of 102 points generated by the twice-lower direction change probability is shown in Fig. 2 (the pseudorandom number generator seed is different), where the trajectory appears to be less random than that in Fig. 1. As the direction change probability is set twice lower again, the trajectory appears to be far less random (Fig. 3).

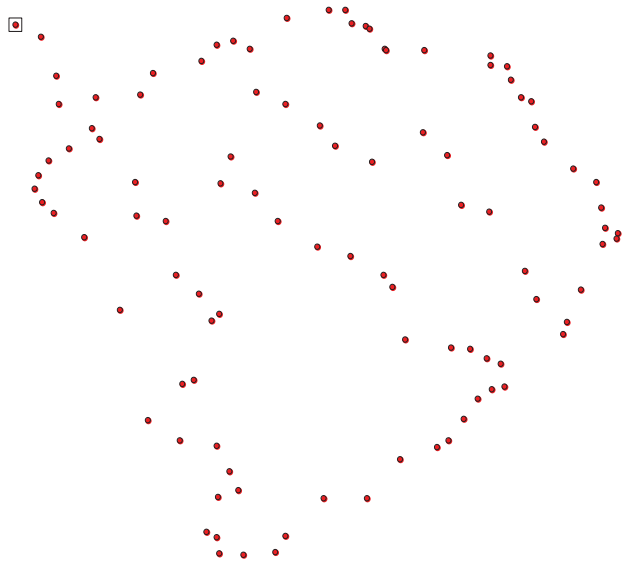

Fig. 1. A planar trajectory of 102 points generated by (1) — (10) and a relatively high probability of the direction change ($a = 0.52$, $s = 5$, $p = 0.1$)

As the number of points generated by (1) — (10) is increased (i. e., the trajectory is intended to be made longer), the trajectory appears overly randomized (Fig. 4). However, as the probability of the direction change is set to a lower value, a longer trajectory has distinct changes of direction either at the same angle or just oppositely. An example of this is presented in

Fig. 5, which can be compared to Fig. 3 as they are peers. The random-like trajectory in Fig. 5 either changes its direction at about angle 90° or changes in opposite direction. Therefore, despite having multiple distinct changes of direction, such a random-like trajectory is predictable to some extent. The only "true" randomness here is provided by the values of SNDRVs in (10), by which the distance between the neighbouring points is randomized.

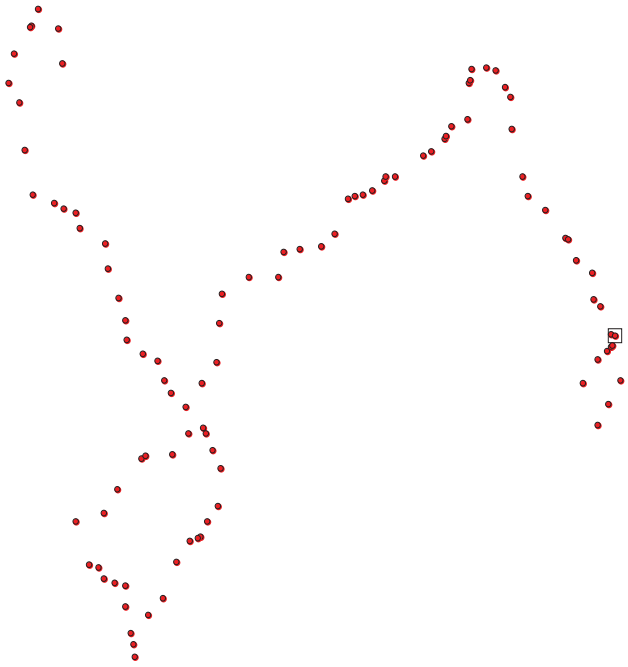

Fig. 2. A less-random-appearing planar trajectory of 102 points generated by (1) — (10) and the twice lower probability of the direction change compared to Fig. 1 ($a=0.52$, $s=5$, $p=0.05$)

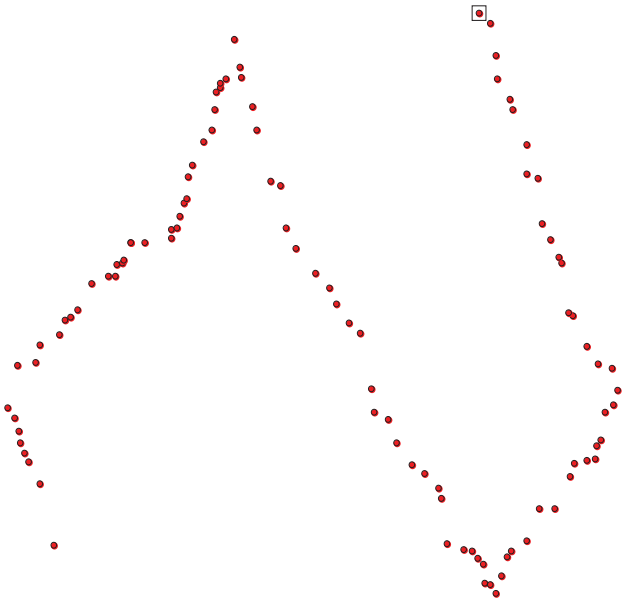

Fig. 3. A far-less-random-appearing planar trajectory of 102 points generated by (1) — (10) and the twice lower probability of the direction change compared to Fig. 2 ($a=0.52$, $s=5$, $p=0.025$); the trajectory has four distinct changes of direction at seemingly the same angle

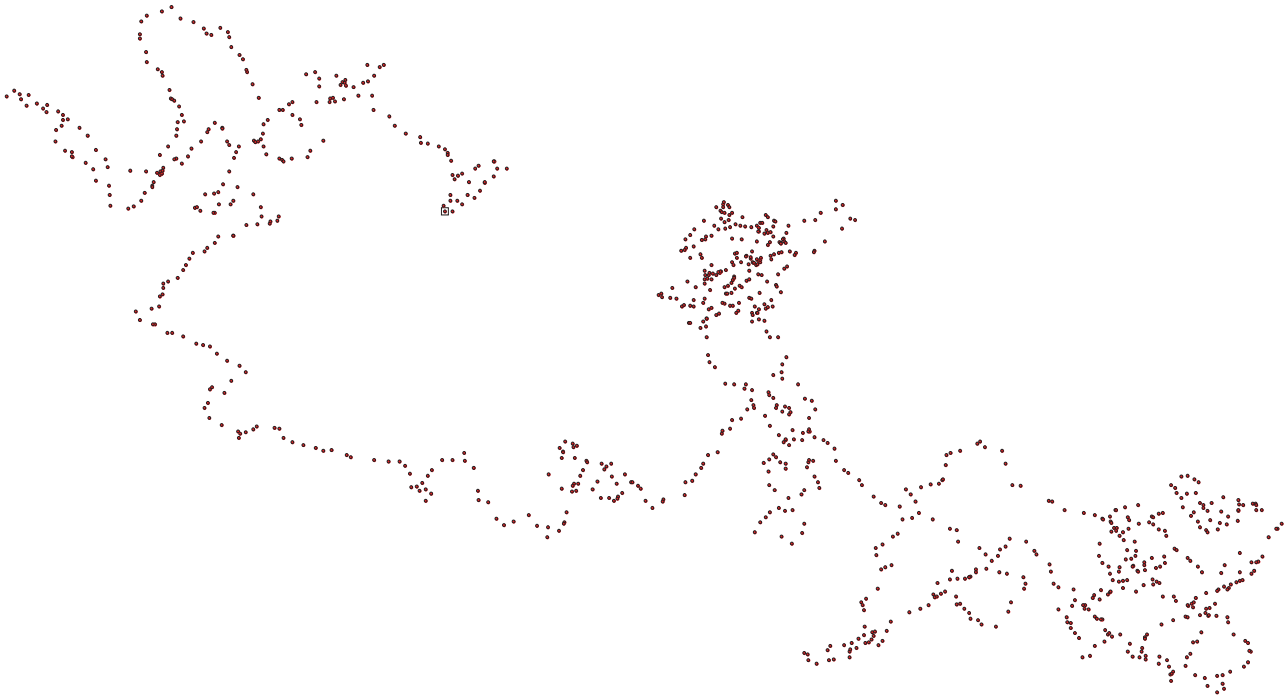

Fig. 4. An overly randomized planar trajectory of 1002 points generated by (1) — (10) and a relatively high probability of the direction change ($a=0.52$, $s=5$, $p=0.1$)

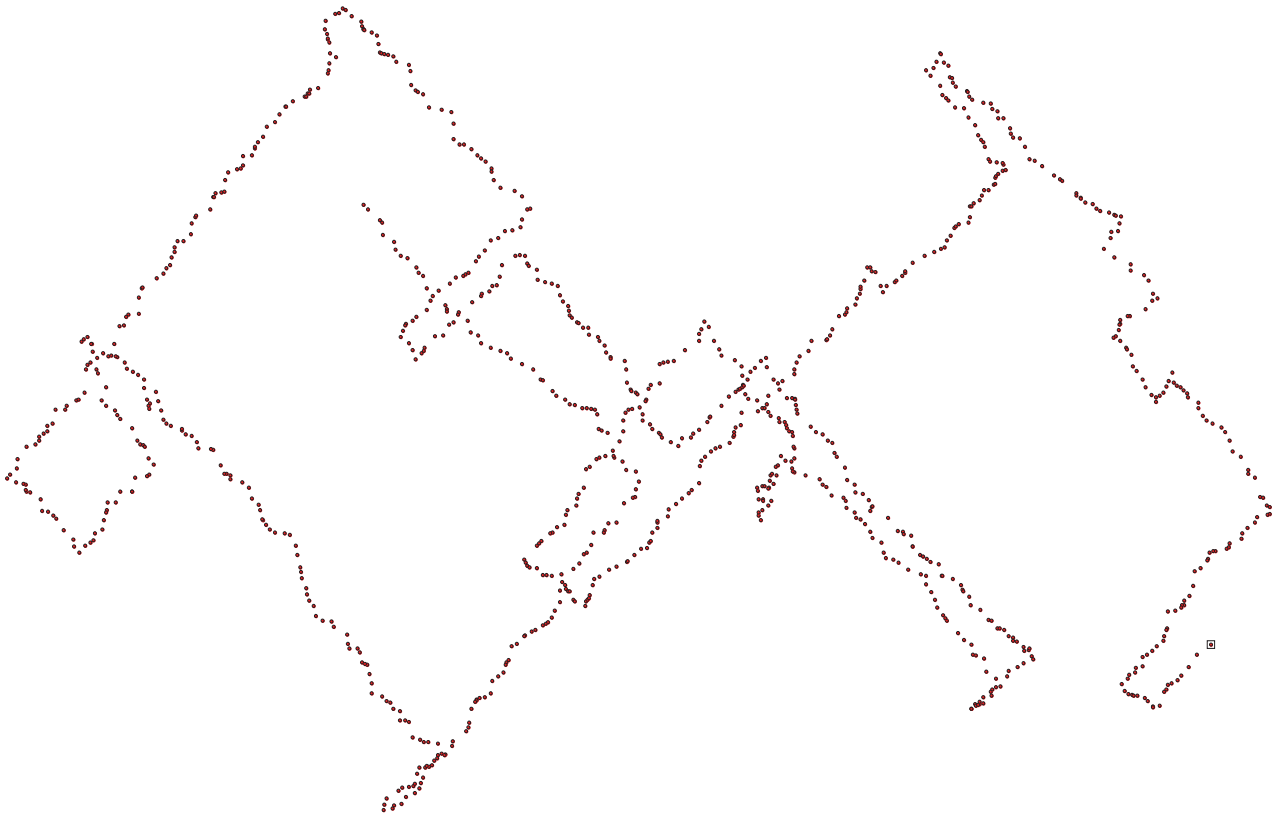

Fig. 5. A planar trajectory of 1002 points generated by (1) — (10) and $a = 0.52$, $s = 5$, $p = 0.025$; the trajectory has multiple distinct changes of direction, which either have roughly the same angle or are made oppositely

The approach with direction randomization produces either too chaotic trajectory or rectangularish trajectory, but there is an obvious lack of smoothness. Any manipulation with magnitude a and probability p cannot produce realistic trajectories. Sufficient smoothness can be imparted by sinusoidal functions, though.

4. Polar coordinate system

Along with absolute coordinates, planar trajectories can be described by polar coordinates, wherein sinusoidal functions are used. The approach with polar coordinates also relies on using values (1) of four independent SNDRVs and the first four values $\{\eta_1, \eta_2\}$, $\{\mu_1, \mu_2\}$ in (2) of two independent UDRVs, but two more SNDRVs are used additionally. The starting first and second points are generated as (4) and (5). The distance between these points is

$$r_{21} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{12}$$

and the respective angle is

$$\beta_2 = \arctan \frac{y_2}{x_2}. \tag{13}$$

The following steps are executed for $i = \overline{2, N-1}$ with already determined points (4) and (5): distance

$$r_{i+1,i} = \left| r_{i,i-1} + a\theta_i \right| \tag{14}$$

and angle

$$\beta_{i+1} = \beta_i + b\vartheta_i \tag{15}$$

are calculated for a magnitude $a > 0$ of the normal noise to scatter distances and for a magnitude $b > 0$ of the normal noise to scatter angles. Then

$$\begin{gathered} x_{i+1} = x_i + r_{i+1,i} \cos\beta_{i+1} + h_x r_{i+1,i} \omega_i^{(x)}, \\ y_{i+1} = y_i + r_{i+1,i} \sin\beta_{i+1} + h_y r_{i+1,i} \omega_i^{(y)} \\ \text{for } i = \overline{2, N-1} \end{gathered} \tag{16}$$

for magnitudes $h_x > 0$ and $h_y > 0$ to additionally scatter points by values

$$\left\{\omega_i^{(x)}\right\}_{i=2}^{N-1}, \left\{\omega_i^{(y)}\right\}_{i=2}^{N-1} \tag{17}$$

of two independent SNDRVs.

The polar coordinate system approach is adjusted with its four parameters:

$$a>0,\ b>0,\ h_x>0,\ h_y>0. \qquad (18)$$

Apart from the random offset parameter $s\in\mathbb{R}$ in the starting first and second points (4) and (5), parameters a, h_x, h_y serve to scatter points — magnitude a amplifies the distance between two neighbouring points, while parameters h_x and h_y amplify scattering along horizontal and vertical axes, respectively. Parameter b amplifies stochasticity of changes in the trajectory heading, while its smoothness is maintained quite satisfactory. Indeed, a tiled plot of 15 trajectories of 250 points by setting parameters (18) to 0.1 is shown in Fig. 6, and the plot trajectories are much smoother now than those in Fig. 2, 3, 5 (the trajectories in Fig. 1 and 4 remind clouds of randomly scattered points rather than random-like trajectories or paths). Trajectories in Fig. 6 do not have much changes (turns and twists).

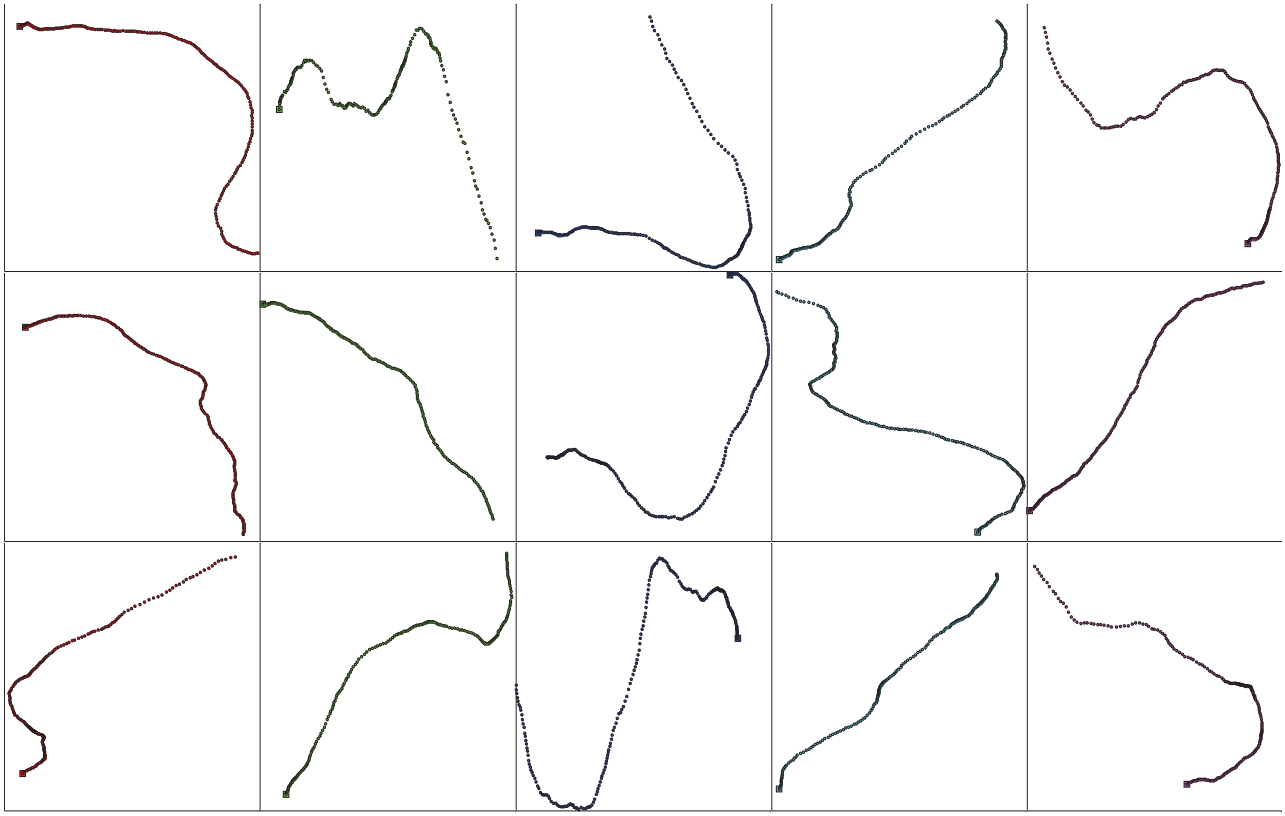

Fig. 6. A collection of 15 random-like trajectories of 250 points generated by (12) — (18) with $a=b=h_x=h_y=0.1$

As parameter b is set to 0.15 (i. e., it is 50 % increased), the 15 trajectories generated by the same pseudorandom number generator seed become more twisty (Fig. 7). The same pseudorandom number generator seed means that the starting first and second points are the same for each respective subplot in Fig. 6 and 7. Nevertheless, some trajectories in Fig. 7 still remain resembling to those in Fig. 6. These are trajectories ## 1, 4 in the first row, trajectories ## 2, 4, 5 in the second row, and trajectories ## 1, 2, 4 in the third row. Trajectories in Fig. 7, despite having more turns and twists compared to trajectories in Fig. 6, still do not have any self-intersections. Then, as parameter b is set to 0.2 (i. e., it is twice as increased compared to trajectories in Fig. 6), three out of the 15 trajectories generated by the same pseudorandom number generator seed have self-intersections (Fig. 8). Namely, these are trajectory #3 in the first row and trajectories ## 2, 5 in the third row. Amazingly enough, some trajectories in Fig. 8 still remain resembling to those in Fig. 6, although having much more turns and twists. These are trajectory #5 in the second row and trajectories ## 1, 4 in the third row. The trajectories in Fig. 7 and 8 are more comparable. Thus, trajectories ## 1, 2 in the first row, trajectories ## 4, 5 in the second row, and trajectories ## 1, 2, 4 in the third row are pretty resembling in the two tiled plots. The most resembling trajectories are trajectory #5 in the second row of Fig. 7 and 8, and trajectory #1 in the third row of Fig. 7 and 8. The latter trajectory in Fig. 8 repeats even small turns of the respective trajectory in Fig. 7 (which can be easily spotted by zooming in).

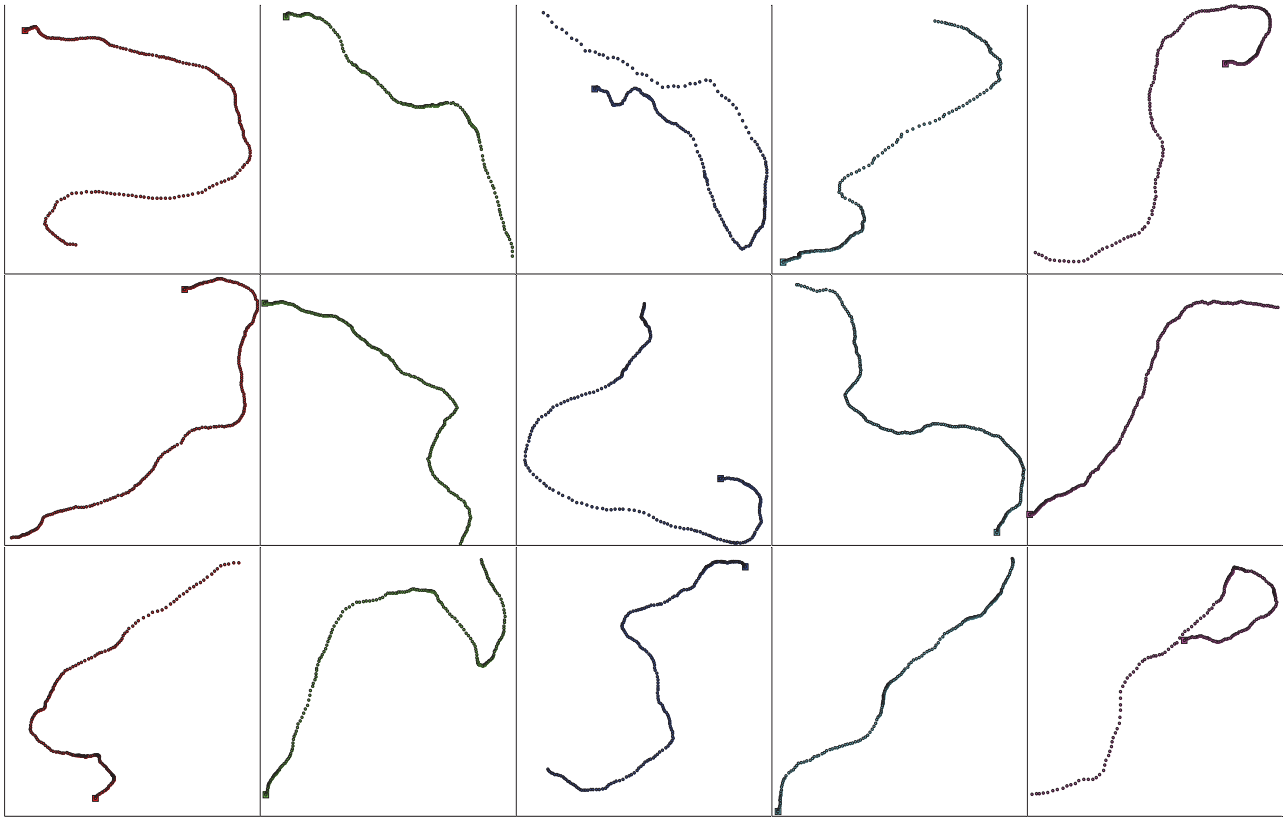

Fig. 7. A collection of 15 random-like trajectories of 250 points generated by (12) — (18) with $a = h_x = h_y = 0.1$, $b = 0.15$, **where the pseudorandom number generator seed is the same as for the collection in Fig. 6**

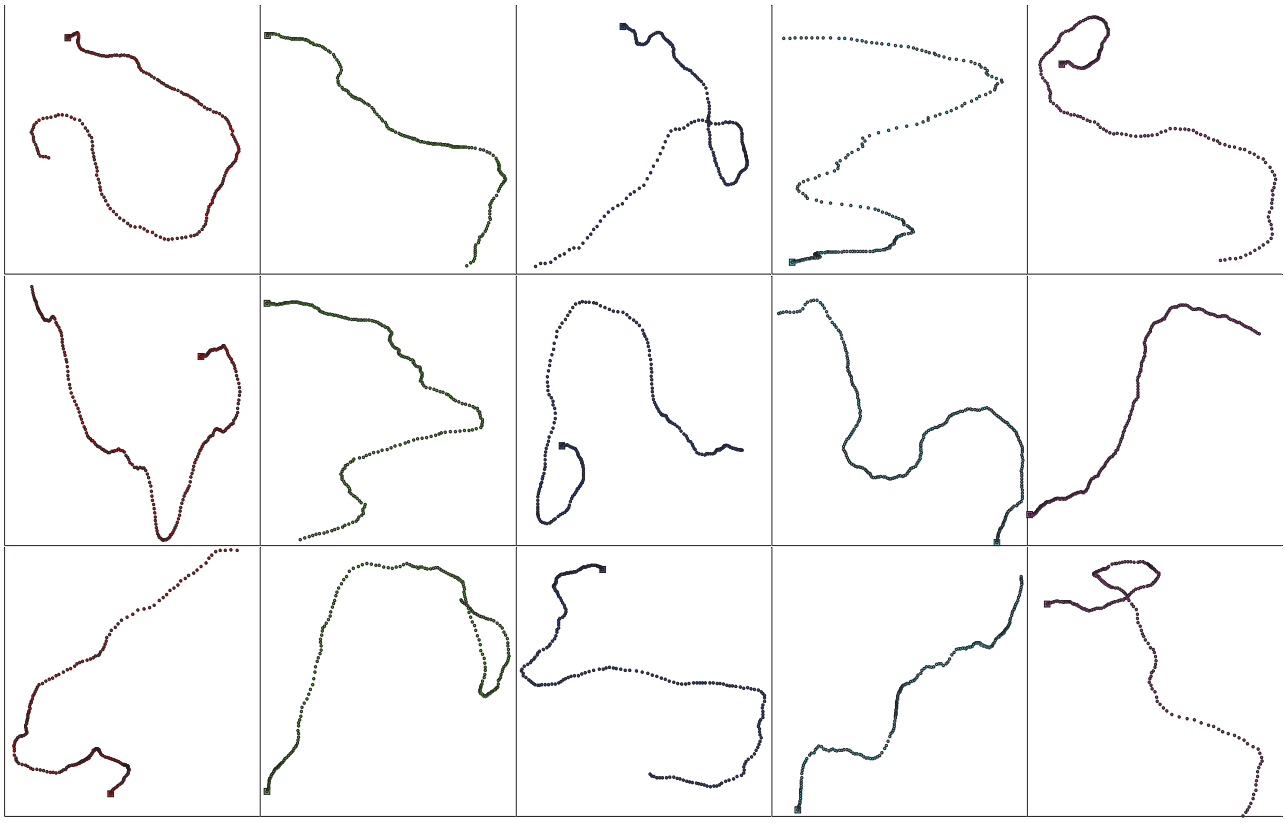

Fig. 8. Random-like trajectories of 250 points generated by (12) — (18) with $a = h_x = h_y = 0.1$, $b = 0.2$

Herein, it is noteworthy that setting angle-scattering parameter b to a lower value does not imply non-self-intersecting trajectories if such ones are generated for a fewer points. Indeed, none of the 15 trajectories in Fig. 6 intersects itself, but it may be due to a fewer points of the trajectory, rather than a too low value of the angle-scattering parameter. Thus, if to increase the number of trajectory points up to 4500, the respective collection of 15 random-like trajectories, generated by the same parameters (18) all set to 0.1 and by the same pseudorandom number generator seed used for the collection in Fig. 6, appear far more twisty (Fig. 9). Besides, now there are just three out of 15 trajectories without self-intersections: trajectory #3 in the first row and trajectories ## 4, 5 in the second row. Each of the other 12 trajectories has at least one self-intersection. Some of these self-intersections remind knots — simple (e. g., like that one in trajectory #1 in the first row, the knot in trajectory #4 in the first row, the simple knots in trajectories ## 2, 3 in the second row, the simple knots in trajectories ## 1, 4, 5 in the third row) and more sophisticated (e. g., like that one in trajectory #5 in the first row, the trickier knots in trajectories ## 1, 3 in the second row, and the not-the-easiest-to-unravel knots in trajectories ## 2, 3 in the third row). The size and structure of the knots vary severely — from tiny to almost gigantic, with respect to the trajectory size and volume (in the number of points).

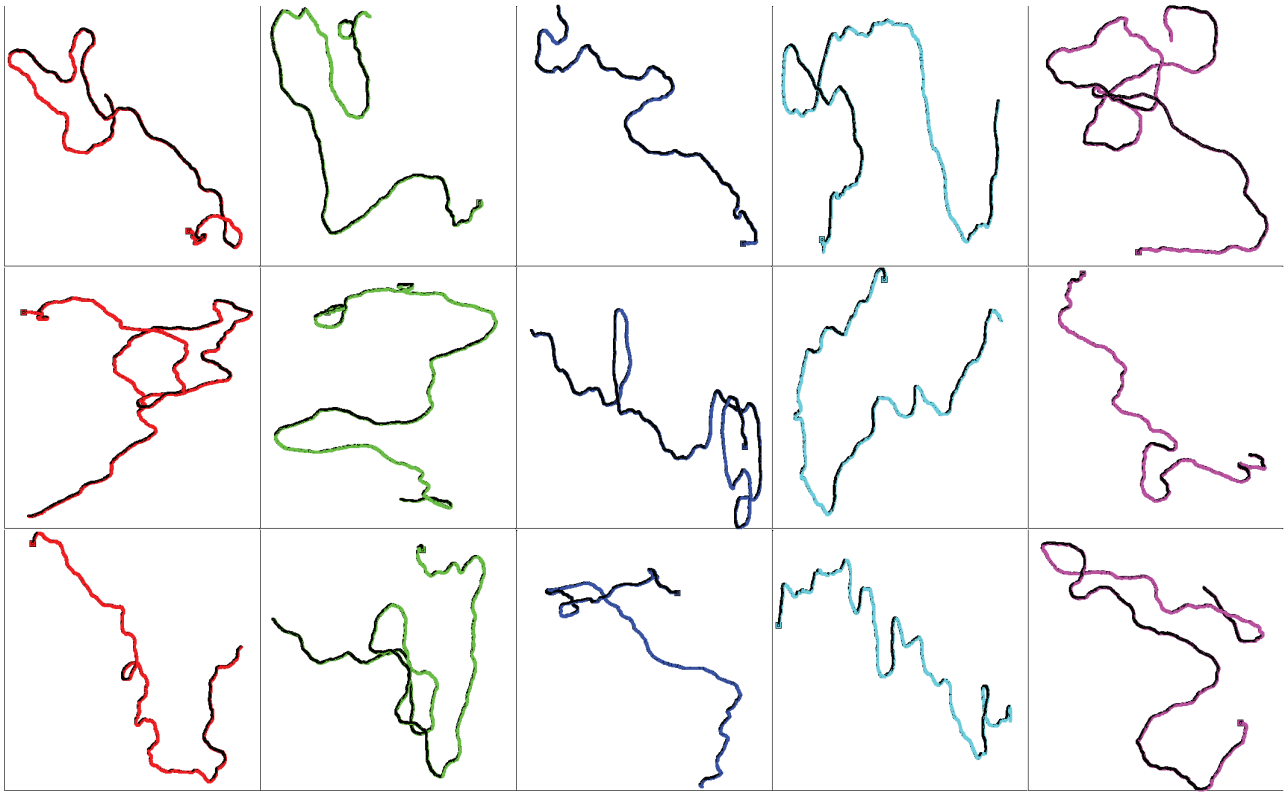

Fig. 9. A collection of 15 random-like trajectories of 4500 points generated by (12) — (18) with $a = b = h_x = h_y = 0.1$, where the pseudorandom number generator seed is the same as for the collection in Fig. 6 (obviously, every subplot here and in any other collection visualized has its own scaling and axes aspect ratio)

Could we make a voluminous trajectory (not a lengthy one, by the way) not so twisty as in Fig. 9 by changing only one parameter (or, speaking for it with a stricter style, by changing the minimum possible number of parameters)? The answer to this question is given in Fig. 10 whose subplots are trajectories generated by the same pseudorandom number generator seed as for the collection in Fig. 9, but with

$$a = h_x = h_y = 0.1,\ b = 0.025,$$

i. e. the angle-scattering parameter is set four times lower compared to that for Fig. 9. It is clearly seen that all the subplots in Fig. 10 are much smoother than those in Fig. 9, having no self-intersections or twisty parts. Another noticeable effect is that some trajectories have almost radically changed their heading. Trajectory #5 in the first row is one of such examples.

Next question is how do multiple trajectories appear in the same region (planar domain)? An example is presented in Fig. 11, where 14 trajectories of 2000

points each are generated by (12) — (18) with

$$a = h_x = h_y = 0.1,\ b = 0.075,$$

and shifting the random offset horizontally to the right for every new trajectory. The variety of trajectories is quite ample. The visible length of the trajectory varies significantly. There are three pairs of intersecting trajectories. One trajectory intersects itself. There is a great deal of turns and twists that simulate realistic manoeuvring of skulking objects.

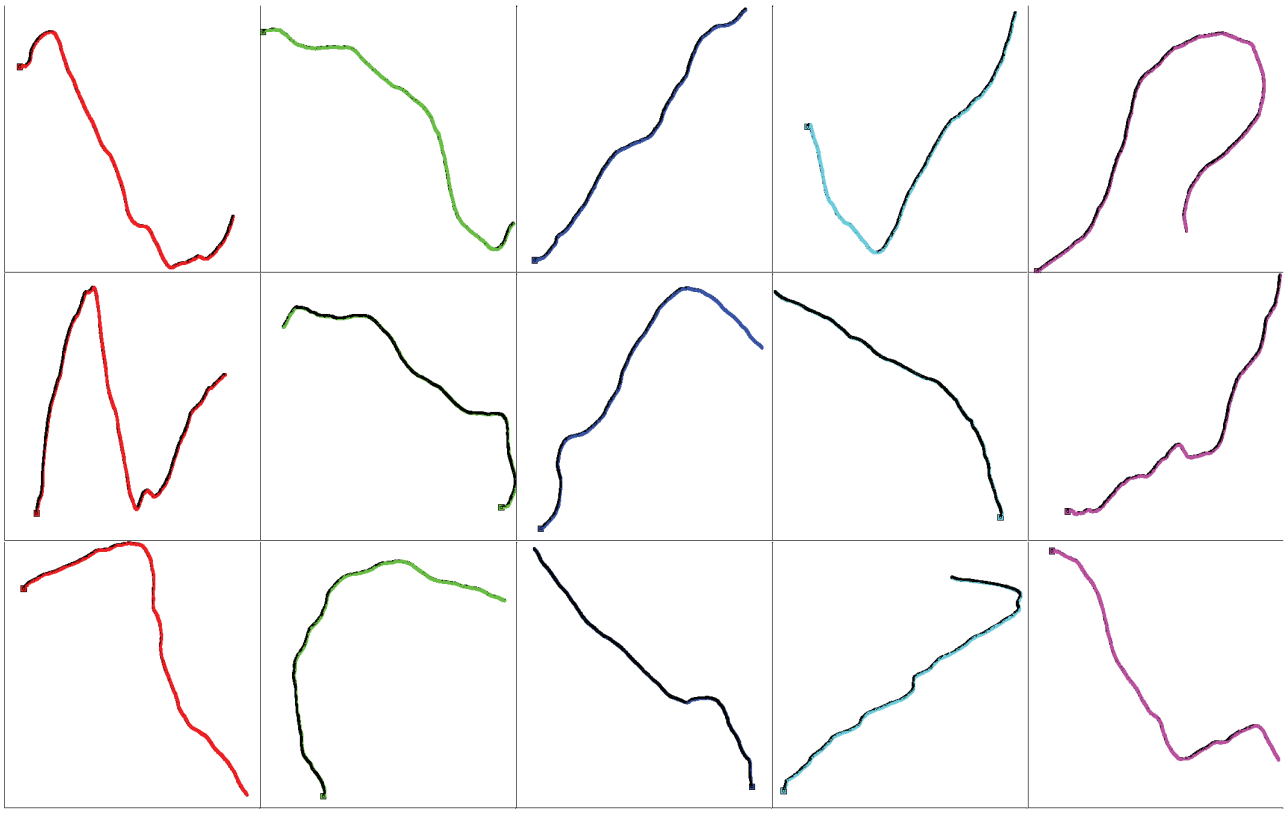

Fig. 10. A collection of 15 random-like trajectories of 4500 points generated by (12) — (18) with $a = h_x = h_y = 0.1$, $b = 0.025$, where the pseudorandom number generator seed is the same as for the collection in Fig. 9

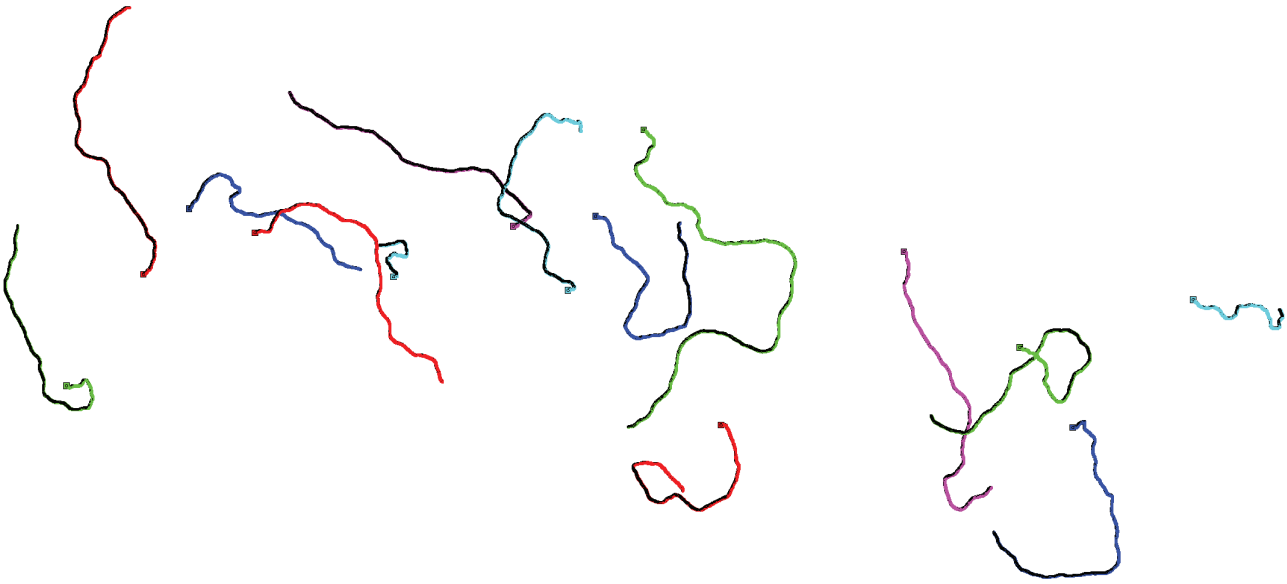

Fig. 11. A collection of 14 random-like trajectories of 2000 points in the same domain, where the trajectories are generated by (12) — (18) with $a = h_x = h_y = 0.1$, $b = 0.075$ (a different pseudorandom number generator seed is used)

However, a more realistic scenario may require sudden changes of the trajectory "style". It is easy to implement by imposing a condition by which one or more parameters (18) are set to different values. An example of such a condition is:

$$\rho > \rho_0 \text{ and } i > i_0$$
$$\text{for } \rho_0 \in (0;1) \text{ and } 2 < i_0 < N-1 \qquad (19)$$

for a value ρ of a UDRV and some threshold values ρ_0, i_0 given beforehand. So, if (19) is true, during a trajectory is generated, one or more parameters (18) are changed (updated) and further generation is executed at the updated parameters. Condition (19) can be invoked multiple times. Usually it is sufficient to update the angle-scattering parameter solely. An example of invoking condition

$$\rho > 0.9975 \text{ and } i > \frac{N}{3} \qquad (20)$$

once is visualized in Fig. 12 for $N = 2000$, where a collection of six trajectories is presented. As soon as (20) turns true, the angle-scattering parameter is set to a different value (namely, it is 10 times increased), and the trajectory is generated onward with the new, much severer, angle-scatterer. Practically, condition (20) along with subsequently increasing parameter b from 0.015 to 0.15 mean that the object must drastically strengthen its manoeuvring upon passing the one third of its path, but not exactly at the point. Therefore, each of the trajectories in Fig. 12 starts smoothly and predictably, and subsequently it develops much more randomly, being far less predictable in its general heading or even unpredictable.

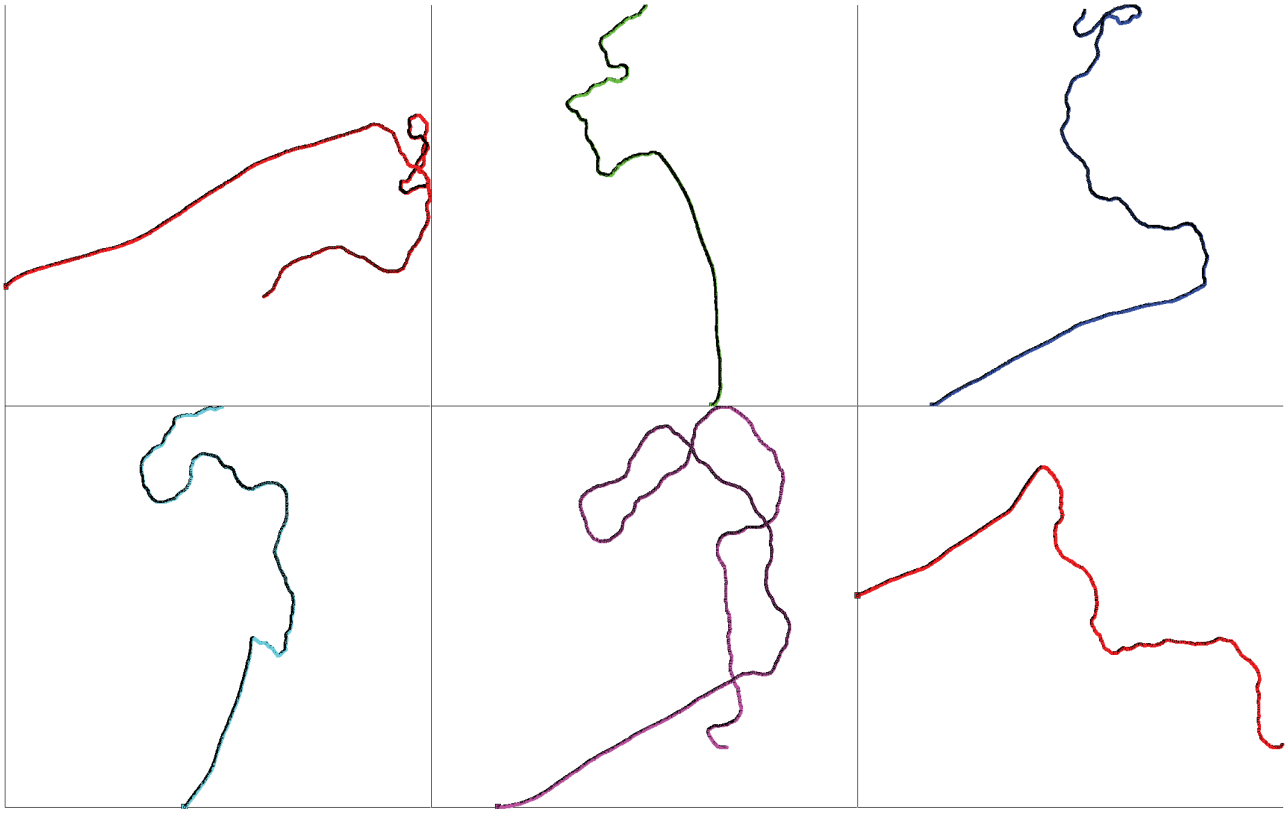

Fig. 12. A collection of six random-like trajectories of 2000 points generated by (12) — (18) with $a = h_x = h_y = 0.1$, $b = 0.015$, and by condition (20) with $N = 2000$ to set $b = 0.15$

5. Discussion

As experience has shown, the random-like trajectory generator has the same time complexity as its predecessors, including the direction randomization generator by (1) — (10) and its modifications. Overall, trajectories by the direction randomization approach remind of pipeline screensaving (see Fig. 5, although Fig. 2 and 3 fit as well), so they hardly can be used in most practical tasks. Self-intersections are important to manoeuvre and confuse the opponent side. The respective exemplary trajectories in Fig. 9 appear very realistic, as well as some trajectories in Fig. 8, 11, 12.

A trajectory of N points requires generating

$$4 + 2 \cdot (N-2) + 2 \cdot (N-2) = 4N - 4$$

values of SNDRVs and four values of UDRVs. An additional UDRV is required to add supplementary conditions like (19) in order to change trajectory properties during its generation. Speaking less strictly, all those pseudorandom values can be drawn from only one SNDRV and one UDRV — this will not significantly decline non-correlatedness and other stochastic properties of such values.

A few additional constraints can be embedded. They relate to domain limits. Say, if an object trajectory must be within a rectangular domain, four simple conditions

$$x_i < x_{\min},\ x_i > x_{\max},\ y_i < y_{\min},\ y_i > y_{\max} \qquad (21)$$

are additionally checked while the trajectory is generated. If one of conditions (21) turns true, the respective coordinate is re-generated until it turns false. Conditions (21) would be similar to fit within circular or elliptic domain limits, where only right terms in (21) should be substituted for respective curve functions.

6. Conclusion

The suggested model of generating random-like planar trajectories consists in preliminarily generating two starting points (4), (5), calculating distance (12) and angle (13) between them, which then are successively updated as (14) and (15) to calculate new trajectory points by (16) using the polar coordinate system. The suggested approach uses six independent SNDRVs and two independent UDRVs. The trajectory has four parameters (18) to adjust its heading, scattering of points, and intensity of turns and twists. These parameters serve as magnitudes to amplify the respective properties. The highest influence has the angle-scattering parameter b. Unlike the direction randomization approach, which generates rather too chaotic trajectories or too rectangularish trajectories, the polar coordinate system approach generates sufficiently smooth trajectories. Moreover, the trajectory "style" can be varied during its generation by adding supplementary conditions like (19) with some threshold values, upon exceeding which one or more parameters (18) are set to different values. So, apart from rectangular domain conditions (21), the polar coordinate system approach generates controllable trajectories also.

The polar coordinate system approach allows easily balancing smoothness and randomness of the trajectory. The intensity and size of self-intersections, as well as intersections with other trajectories generated in the same domain, cannot be straightforwardly regulated, though. Nevertheless, unpredictability of the trajectory is increased as the angle-scattering parameter is set to a higher value. It efficiently masks the genuine destination or purposes of the manoeuvring object. The suggested model of generating random-like planar trajectories with intersections should serve either for generating trajectory datasets to train manoeuvring-object detectors on them or for masking reconnaissance.

The research must be furthered by studying methods of identifying multiple trajectories generated in the same domain by the polar coordinate system approach. For this, in particular, density-based and shape-free clustering methods like DBSCAN [21], [22] may be used. However, handling trajectory intersections is likely to be an additional problem to prevent merging different trajectories and severing one trajectory with self-intersections (with simple or sophisticated knots).

References

1. J. M. Kelner, W. Burzynski, and W. Stecz, "Modeling UAV swarm flight trajectories using Rapidly-exploring Random Tree algorithm," *Journal of King Saud University — Computer and Information Sciences*, vol. 36, iss. 1, 101909, 2024.
https://doi.org/10.1016/j.jksuci.2023.101909
2. K. Gromada and W. Stecz, "Determining UAV flight trajectory for target recognition using EO/IR and SAR," *Sensors*, vol. 20, iss. 19, 5712, 2020.
https://doi.org/10.3390/s20195712
3. M. A. Jasim, H. Shakhatreh, N. Siasi, A. H. Sawalmeh, A. Aldalbahi, and A. Al-Fuqaha, "A survey on spectrum management for unmanned aerial vehicles (UAVs)," *IEEE Access*, vol. 10, pp. 11443 — 11499, 2022.
https://doi.org/10.1109/ACCESS.2021.3138048
4. M. Chodnicki, B. Siemiatkowska, W. Stecz, and S. Stepien, "Energy efficient UAV flight control method in an environment with obstacles and gusts of wind," *Energies*, vol. 15, iss. 10, 3730, 2022.
https://doi.org/10.3390/en15103730
5. A. A. Laghari, A. K. Jumani, R. A. Laghari, and H. Nawaz, "Unmanned aerial vehicles: A review," *Cognitive Robotics*, vol. 3, pp. 8 — 22, 2023.
https://doi.org/10.1016/j.cogr.2022.12.004
6. R. Gui, Z. Zheng, and W.-Q. Wang, "Cognitive FDA radar transmit power allocation for target tracking in spectrally dense scenario," *Signal Processing*, vol. 183, 108006, 2021.
https://doi.org/10.1016/j.sigpro.2021.108006
7. V. V. Romanuke, "Accurate detection of multiple targets by uniform rectangular array radar with threshold soft update and area rescanning," *Information and Telecommunication Sciences*, vol. 13, no. 2, pp. 62 — 71, 2022.
https://doi.org/10.20535/2411-2976.22022.62-71
8. J. L. Zhao and H. Y. Li, "Maneuvering identification method for non-cooperative aircraft based on sparse orbital information," *Systems Engineering and Electronics*, vol. 44, iss. 6, pp. 1950 — 1956, 2022.
9. R. Kneusel, *Random Numbers and Computers*. Springer

International Publishing, 2018.
https://doi.org/10.1007/978-3-319-77697-2
10. V. V. Romanuke, A. Y. Romanov, and M. O. Malaksiano, “Pseudorandom number generator influence on the genetic algorithm performance to minimize maritime cargo delivery route length,” *Pomorstvo. Scientific Journal of Maritime Research*, vol. 36, pp. 249 — 262, 2022.
https://doi.org/10.31217/p.36.2.9
11. B. Yang, G. Zhang, H. Rao, S. Wang, B. Yang, and Z. Sun, “Numerical simulation of the maneuvering performance of ships in broken ice area,” *Ocean Engineering*, vol. 294, 116783, 2024.
https://doi.org/10.1016/j.oceaneng.2024.116783
12. Z.-L. Ouyang, S.-Y. Liu, and Z.-J. Zou, “Nonparametric modeling of ship maneuvering motion in waves based on Gaussian process regression,” *Ocean Engineering*, vol. 264, 112100, 2022.
https://doi.org/10.1016/j.oceaneng.2022.112100
13. E. Cortina, D. Otero, and C. E. D’Attellis, “Maneuvering target tracking using extended Kalman filter,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 27, iss. 1, pp. 155 — 158, 1991.
14. R. Vera-Amaro, M. E. Rivero-Ángeles, and A. Luviano-Juárez, “Phase-type distributions of animal trajectories with random walks,” *Mathematics*, vol. 11, 3671, 2023.
https://doi.org/10.3390/math11173671
15. N. L. Sund, G. M. Porta, and D. Bolster, “Upscaling of dilution and mixing using a trajectory based Spatial Markov random walk model in a periodic flow domain,” *Advances in Water Resources*, vol. 103, pp. 76 — 85, 2017.
https://doi.org/10.1016/j.advwatres.2017.02.018
16. J. Goldberger and D. Burshtein, “Scaled random trajectory segment models,” *Computer Speech and Language*, vol. 12, pp. 51 — 73, 1998.
17. Z. Xi, Y. Kou, Z. Li, Y. Lv, and Y. Li, “An air combat maneuver pattern extraction based on time series segmentation and clustering analysis,” *Defence Technology*, 2023.
https://doi.org/10.1016/j.dt.2023.11.010
18. A. R. Pitombeira-Neto, H. P. Santos, T. L. Coelho da Silva, and J. A. F. de Macedo, “Trajectory modeling via random utility inverse reinforcement learning,” *Information Sciences*, vol. 660, 120128, 2024.
https://doi.org/10.1016/j.ins.2024.120128
19. G. Technitis and R. Weibel, “An Algorithm for Random Trajectory Generation Between Two Endpoints, Honoring Time and Speed Constraints,” in *Proceedings GIScience 2014*, 24 — 26 September 2014, Vienna.
20. Y. A. Ahmed, “Mathematical model of the manoeuvring motion of a ship,” in *Engineering Applications for New Materials and Technologies*, Springer, 2018, pp. 551 — 566.
https://doi.org/10.1007/978-3-319-72697-7_45
21. M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, AAAI Press, 1996, pp. 226 — 231.
22. X. Bai, Z. Xie, X. Xu, and Y. Xiao, “An adaptive threshold fast DBSCAN algorithm with preserved trajectory feature points for vessel trajectory clustering,” *Ocean Engineering*, vol. 280, 114930, 2023.
https://doi.org/10.1016/j.oceaneng.2023.114930

Романюк В.В., Пабіх М.П.

Модель випадкових на вигляд траєкторій з перетинаннями на площині

Проблематика. Останнім часом задача виявлення та ідентифікації траєкторій об’єктів, чиї справжні наміри або невизначені, або загрожують ударами, стала екстремально важливою. Відомі підходи продукують недостатньо гладкі траєкторії.

Мета дослідження. Побудувати модель генерування випадкових на вигляд траєкторій на площині, які мали би достатньо гладкі криві частини. Траєкторія може мати перетинання самої себе та може перетинати інші траєкторії.

Методика реалізації. Першопочатково на площині генеруються дві точки. Обчислюються відстань та кут між цими точками. Ці відстань та кут далі послідовно оновлюються для обчислення нових точок траєкторії з використанням системи полярних координат. Траєкторія з N точок генерується з використанням $4N-4$ значень нормально розподілених випадкових змінних з нульовим середнім й одиничною дисперсією та чотирьох значень рівномірно розподілених на інтервалі $(0;1)$ випадкових змінних.

Результати дослідження. Генератор випадкових на вигляд траєкторій має ту саму часову складність, як його попередники, включно з генератором на основі рандомізації напрямку та його модифікацій. Прикладові траєкторії виглядають вельми реалістично. Перетинання траєкторії самої себе важливі для маневрування та заплутування сторони противника. Траєкторія має чотири параметри для підлаштування її напрямку, розсіювання точок та інтенсивності поворотів і вигинів. Ці параметри виступають у якості амплітуд для підсилення відповідних властивостей. Найсильніший вплив має параметр кутового розсіювання. Чотири простих умови можуть бути додані для того, щоб згенерувати траєкторію у межах деякої прямокутної області.

Висновки. Запропонована модель повинна слугувати або для генерування наборів траєкторій з метою навчання систем виявлення маневруючих об’єктів, або для маскування розвідувальних акцій. Ця модель дозволяє збалансувати гладкість та випадковість траєкторії.

Ключові слова: *спостереження за об’єктом; випадкова траєкторія; випадковий шлях; напрямок; система полярних координат; маневреність.*