DaNCES: a framework for data-inspired agent-based models of collective escape [⋆] [⋆⋆]

Marina Papadopoulou[1,2], Hanno Hildenbrandt[2], and Charlotte K. Hemelrijk[2]

[1] Biosciences, School of Biosciences, Geography and Physics, Swansea University, Swansea, UK

[2] Groningen Institute for Evolutionary Life Sciences, University of Groningen, Groningen, The Netherlands

Abstract. The collective escape of prey by prey is a classic example of adaptive behavior in animal groups. Across species, prey has evolved a large repertoire of individual evasive maneuvers they can use to evade predators. With recent technological advances, more empirical data of collective escape is becoming available, and a large variation in the collective dynamics of different species is apparent. However, given the complexity of patterns of collective escape, we are still lacking the tools to understand their emergence. Computational models that can link rules of individual behavior to patterns of collective escape are needed, but species-specific motion and escape characteristics that will allow the link between behavior and eco-evolutionary dynamics of a given species are rarely included in agent-based models of collective behavior. Here, to tackle this challenge, we introduce a framework that uses individual-based state machines to model spatio-temporal dynamics of collective escape. A synthetic agent in our framework can switch its behavior between 'flocking' with different coordination specifics (e.g., quicker interactions when vigilant) and 'escape' with various maneuvers through a dynamic Markov-chain, depending on its local information (e.g., its relative position to the predator). A user can compose a new agent-based model adjusted to empirical data by choosing a set of states (which includes rules of motion, interaction, and escape), their temporal order, and a detailed parameterization. The flexibility and structure of our software allows substantial changes in a model with very minimal code alterations, showing great potential for future use to identify the underlying mechanisms of collective escape across species and ecological contexts.

Keywords: Collective escape · Agent-based modeling · Predator-prey interactions.

[⋆] Supported by the Netherlands Organization for Scientific Research (NWO), project 14723 awarded to CKH: "Preventing bird strikes: Developing RoboFalcons to deter bird flocks."

[⋆⋆] **Publication acknowledgment:** This version of the manuscript has been accepted for publication, after peer review but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: http://dx.doi.org/10.1007/978-3-031-71533-4_15. Use of this Accepted Version is subject to the publisher's Accepted Manuscript terms of use https://www.springernature.com/gp/open-research/policies/accepted-manuscript-terms.

1 Introduction

Since the first model that reproduced patterns of collective escape in fish schools was published more than 20 years ago [19], few studies have attempted to model patterns of collective escape across taxa (e.g.,[5,38,14,15]), leaving many gaps in our understanding of collective escape. Even though species differ a lot in the specifics of their escape reactions, that can also be predator-specific [7,6], models of collective escape have been usually generic (e.g.,[38,1,2,15]). With technological advantages (e.g., unmanned aerial vehicles and underwater cameras) enabling the collection of empirical data of collective escape across many systems [32,12], a better link between data and theory is the next step forwards. However, species-specific adjustments in a model [18,27,11] can be time-consuming and challenging in an existing model, given the increasing model complexity as species-specific characteristics are added. Striving for a balance between mechanistic simplicity and data complexity is also of high importance, since overfitting a model to data may compromise the identification of self-organized mechanisms and make a model 'single-use', delaying advances in the field and limiting model comparisons [10].

To adjust a model of collective escape to an empirical system, the specifics of reaction of individual prey to the predator should be carefully considered. Most models include a single rule based on which prey avoids a predator. This rule is either 'continuous', balanced with interaction rules that make group members coordinate (for instance a tendency to turn away from the predator's position [19,38]), or 'discrete' (also referred to as 'fixed'), instantaneous turning that 'interrupts' the regular coordinated motion of the group (usually applied to study wave propagation, e.g., [15,14]). The escape reactions we see in nature may be represented by both continuous and discrete cognitive rules; which type of escape reaction individuals perform when their group is under attack is still hard to distinguish in empirical data. Additionally, more complex escape maneuvers (e.g. protean motion [17,21,22]) are rarely modeled in the context of collective behavior. Thus, in order to improve our modeling of collective escape, both continuous and discrete escape reactions should be tested at the individual level. Nevertheless, we first need to model the collective motion of a given species [27].

Most agent-based models of collective animal behavior are based on the rules of attraction, alignment and avoidance [5,30], even though species differ in their dynamics of collective motion (their specifics of order, shape and density) [23]. Mere changes in a model's parameters are often not enough to realistically simulate a group of a specific species, adjustments in the interaction rules are needed. For instance, in a model aiming to simulate groups of pigeons, an extra rule of speed adjustment was needed [27], with individuals deviating from their preferred speed to stay with the flock, in accordance to empirical findings in homing pigeons [31,29]. How exactly a single coordination rule of flocking is modeled may also vary; for instance, an individual may avoid collisions by turning away from the position of its neighbor, by aligning with the heading of its neighbor, or by decelerating. How these modeling decisions affect the emerging collective motion has been rarely studied. Additionally, most computational models of collective behavior assume that individuals are identical, even though individual variation in specific traits within a group has been identified in many species [20,31].

Here, we recognize the necessity for a new modeling structure that can counteract the aforementioned issues and model the diversity of collective escape we see in nature (Fig. 1). We want to create a model that is flexible and easily adjustable to different species and ecological contexts. Therefore, it should accommodate several (and perhaps diverse) types of individual motion (especially escape), from continuous coordinated motion to sudden reactions to the predator. To achieve these aims, we develop a modeling framework (named DANCES, DAta-iNspired Collective EScape) that uses individual-based state-machines to model collective escape. By combining internal states at the individual level with a composable software structure, we further highlight the potential of such conceptualization [36] to study collective escape. While developing our framework, we created three agent-based models of bird flocks [26,25,28], and present some of their technical details here as examples for demonstrating the full structure and functionality that our framework offers.

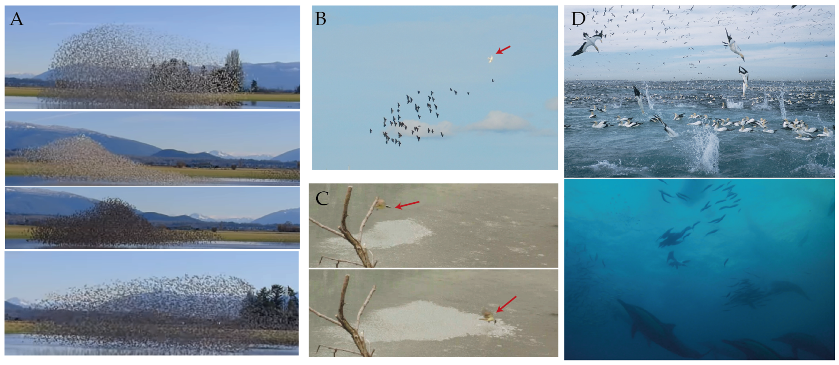


Fig. 1. Collective escape in animal groups **A.** Aerial displays of flock of dunlins (*Calidris alpina*) reacting to the attack of a peregrine falcon (*Falco peregrinus*). Screenshots taken from the video 'Dance of the Dunlins' (2013) by Ray Hamlyn. **B.** A flock of corvids turning away from a robotic falcon [33]. **C.** Propagation of a 'diving' escape wave in schools of sulphur mollies (*Poecilia sulphuraria*) as a response to the attack of a great kiskadee (*Pitangus sulphuratus*), screenshots taken from video by [8]. **D.** A group of Cape gannets (up, Mathieu Simonet/Galatée Films 2009) attacking a fish school surrounding by more predator (bottom, Galatée Films 2009), taken from [35].

2 Modeling framework

Our framework is implemented in C++, using OpenGL for the visualization and DearImGui [4] for a user interface, and can be accessed in our GitHub repo [24].

2.1 Internal states & social interactions

An agent in our framework is represented by its 'internal state'. In the context of collective escape, this internal state consists primarily of the agent's position and velocity in a global coordinate system. Since agents in computational models of collective behavior move in space based on their interactions with one another, we conceptualize an individual-level 'rule' in our model that can alter the internal state of an agent (and thus control its motion), as a stand-alone 'internal-state control unit' (referred to as an *ISC-unit*). Examples of such units are an alignment interaction (that makes the individual align its heading with its neighbors), a roosting behavior (that makes it turn towards its roost), or a noisy motion (that adds a random error to its heading). Each ISC-unit owns a set of parameters (e.g. the number of interacting neighbors or the position of the roost). We treat each internal-state control unit as a building block; the modeler combines the pieces of interest according to their study system. Thus, all available ISC-units can be simply used to build a new model or examine alternative interaction rules in existing models, while modeling new units is facilitated (by copying the main structure of an existing unit) and does not risk interfering with other functionalities of the software.

All social ISC-units need information about the social environment of the agent at each timepoint. To provide that, a set of neighbors' information (`neighbor_info`) is defined and can be accessed by an agent (currently the id, state and stress of another agent, as well as the distance between them). An ISC-unit can access this information and indicate a set of neighbors for the agent to 'sense' (`while_topo.hpp`). For instance, at its current state, the framework supports metric and topological interactions along with the definition of a simple visual field: 'neighbors' are defined by a given number of agents (parameter `topo`), positioned within a given radius (parameter `maxView`), and within a field of view (an angle, parameter `foV`, e.g. prohibiting the sensing of individuals in the back). By parameterizing `topo` to be equal to the total population size and restricting the `maxView` one can have a 'traditional' metric model, and by setting `maxView` to a large value and restricting `topo`, a topological. These settings are ISC-unit specific, meaning that an agent can use different social information per behavioral rule (e.g., avoid collision with only a single neighbor but copy an escape maneuver from any surrounding neighbor).

2.2 Individual-based state machines

A set of ISC-units defines a 'state', that controls an agent's behavior (similar to a Markov-Chain [37]). For instance, a state that represents a 'flocking' behavior may consist of four ISC-units that affect the agent's behavior: alignment, centroid-attraction, avoidance and noise. Thus, a state is designed around the nature of its effect on individual motion. At its current state, our framework implements social interactions and individual motion through a 'social' force that is applied to the internal state of each individual. Thus, each ISC-unit adds a steering vector (glm::vec3 type) to the main social force that ultimately controls the motion of each agent (see the OODD Protocol of the HoPE model in [26]). The use of such steering vector is however not enforced by our framework; one can choose a different variable to be changed and control the agent's motion.

Apart from the parameters of each ISC-unit of a state, a state also has its own parameterization. This can include, for instance, the frequency with which the steering vector is updated (recalculating the effect of each ISC-unit, also known as 'reaction frequency'[2,18]). A description of example states in existing models based on our framework is given in *Section 3*. Overall, a model with a single state in our framework is the equivalent of a classic model of collective behavior, in which a constant set of rules at the individual level affects the motion of all agents identically.

To model discrete reactions and switches between different behaviors (related to the predator or the ecological context), our framework enables several states to be included in a model. An agent is thus defined by not only its internal state, but also a finite-state machine that controls its behavior. Each state can comprise a different combination of ISC-units or the same ISC-units with different parameterization. For instance, a state of 'flocking' can be followed by an 'escape' state, during which an agent flocks while also avoiding a predator. An example of the definition of a state machine of a prey agent is given in Listing 1.1.

Listing 1.1. Example of a definition of a state machine of a 'prey' agent with several states. A description of all ISC-units currently available in the framework is given in our repository [24]. The ones listed here represent alignment, cohesion, repulsion (by turning away from the position of its neighbors), roosting (attraction to a global point in space), random noise in the direction of motion (`wiggle`), and escape by a level turn or diving. To change the behaviour of an agent, the user needs to change the functions included in the defined 'package'(e.g., remove **`roost_attraction`** or replace **`avoid_n_position`** with **`avoid_n_direction`**: avoid collision points by turning parallel to the neighbour), and the parameters relating to each ISC-unit in the *.json* parameter file.

```
// States package of a PreyType agent
using AP = states::package<
     states::transient<iscus::package< // 1.regular flocking
         PreyType,
         iscus::align_n<PreyType>, // alignment
         iscus::cohere_centroid_distance<PreyType>, // cohesion
         iscus::avoid_n_position<PreyType>, // separation
         iscus::roost_attraction<PreyType>,
         iscus::wiggle<PreyType> >>,
      flee_state, // 2. escape multi-state
      states::persistent<iscus::package< // 3. refraction (persistent flocking)
         PreyType,
         iscus::align_n<PreyType>,
         iscus::cohere_centroid_distance<PreyType>,
         iscus::avoid_n_position<PreyType>,
         iscus::roost_attraction<PreyType>,
         iscus::wiggle<PreyType> >> >;

using flee_state = states::multi_state<PreyType, // multistate definition
      states::persistent<iscus::package<PreyType,
        iscus::random_t_turn_gamma_pred<PreyType>,
        iscus::wiggle<PreyType> >>,  // 1.escape turn
      states::persistent<iscus::package<PreyType,
        iscus::dive<PreyType> >> >; // 2. dive
```

Transitions between states are controlled by a transition machine, owned by each agent. Each state has a duration that ranges from a single reaction step to any parameterized duration. For conceptual differentiation, a state is defined as 'transient' (effective duration of a single update cycle, an individual can switch state at any point) or 'persistent', with a user-define duration (only after a given period is passed the agent may switch state). At the exit of a state, the transition machine assigns the next state as a function of the current state and a set of probabilities, i.e. a transition matrix. This matrix can be dynamic or constant, and is predefined by parameterization. For instance, deterministic transitions are used in our existing models for the predator so its behavior follows a closed loop [26,27,25]: the predator has a probability of 1 to transition between its 'pursuit', 'attack' and 'retreat' states that all have a pre-defined duration.

Dynamic transition matrices are modeled for transitions that may depend on specific conditions during a simulation. For instance, an individual may switch to an escape state only if the predator is approaching. To account for this, each transition machine also owns a variable that can alter the probability of switching to a specific state. In the existing state of the framework, such transition-altering mechanism is modeled through a 'stress' variable that is affected by a set of 'stress source' functions, namely neighbors (*neighbors_stress*) or predator (*predator_distance*): the closer the predator or the higher the stress of the neighbors of an agent, the more stress is accumulated in each agent, and the higher its probability to perform an escape maneuver is. A decay rate parameter is used to stabilize this accumulating value, so that in the absence of a predator, the probability of escaping returns to 0. Thus, the exact transition matrix at a given update time step is calculated by the transition machine through a piecewise linear interpolator that uses the stress value, and a number of user-defined transition matrices and interpolation edges.

Overall, depending on the characteristics of the empirical systems, the transition probabilities between states for each agent can vary through time or be constant. Based on the probabilities of each current state, an agent selects its next state. Algorithm 1 presents a simplified version of the above mentioned processes.

In a real system, an individual may choose to react with an escape maneuver to a predator, but the exact maneuver may depend on the relative position of the prey to the predator [6]. Translated in our framework, to account for cases where the state selection should be performed by a specific ISC-unit and not the transition machine, we further included 'multi-states' in our framework. A multi-state is a mere collection of states. The states of a multi-state (sub-states) have a conceptual connection that affects their selection during a simulation. The transition machine may decide that an individual should perform an escape maneuver, but the probability of selecting each one can be altered by each ISC-unit during the simulation (for instance if the predator attacks from above, a diving maneuver may be evaluated as more effective than a turning one), or from the parameterization of the multi-state (for instance, empirical data may show that the frequency of turning is higher than the frequency of diving). An

Algorithm 1 Pseudo-code for the main structure of a simulation, showing the timing of processes and the handling of state transitions.

```
1: for $t_i : 1, \ldots, T_{max}$ do                          //With $T_{max}$ the total simulation time
2:     $integrate\_motion(\psi)$                          //Calculate new position and velocity of agent
3:     $s_i \leftarrow s_{i-1} - r_{decay}$                                            //Stress decay
4:     if $t_i = t_{update}$ then
5:         $\psi \leftarrow 0$                                      //Start new steering vector ($\psi$)
6:         $\psi \leftarrow chain\_ISC\_units(S)$           //Accumulate steering vector from the state
7:         $t_{update} \leftarrow t_i + t_{react}$                                  //Next update step
8:     end if
9:     if $t_i = t_{exit\_state}$ then
10:        $s_i \leftarrow accumulate\_stress()$
11:        $TM \leftarrow get\_transition\_matrix(s_i)$
12:        $S \leftarrow sample\_new\_state(TM)$                                    //Get new state
13:        $t_{exit\_state} \leftarrow t_i + \Delta t_S$    //Next switch from the duration of the state ($\Delta t_S$)
14:    end if
15: end for
```

example of the definition of a multi-state is given in Listing 1.1 (‘flee_state’). This allows a model to be improved as new empirical data on individual escape reactions are becoming available, facilitating the link between observations and theory.

2.3 Multi-level parameterization

Parameters in a model on our framework can be at the level of a state (for instance a state-specific speed), at the level of an ISC-unit within a specific state (such as the number of topological neighbors for alignment), and at the level of the agent (such as its mass). To control all these, a configuration file with all the necessary parameters should be given as input in the model (at run time). We implemented this as a JSON file given its nested nature that allows for parameterization of very complex state-machines. We can thus control every detail of a model, making conscious assumptions and allowing simulations of long sequences of collective escape seen in nature. This is particularly helpful when running a large set of simulations with different parameterization: ISC-units can be activated or deactivated and thus completely change the model between runs.

Our implementation of the parameterization of the transition machine is inspired by behavioral chains found in bird flocks [34], so that transition probabilities can be directly informed by empirical data. We define a Markov chain through a transition matrix in our configuration file. Given that these probabilities may need to be adjusted during a simulation, depending on external conditions such as the proximity of an individual to the predator, we implemented an interpolator between several transition matrices. In detail, by giving the model several matrices (3 by default, for low, medium and high threat) for some extreme conditions, the exact transition matrix is created at every state-switching step based on the parameter of the transition machine (*stress*). For instance, the interpolator could be used when the probability of an individual to perform an extreme escape maneuver should be 0 when the predator is not present and only increase when the predator gets very close.

It is important to note that ISC-units are not specific to the type of the agent in a model. They can be used by any agent as long as all its necessary variables (parameters) are there. For instance, predator and prey can use the same ISC units to move around, but to use the ISC-unit that allows an agent to select a target from a group (conceptually a predatory behavior), the agent needs to own a 'target' variable. This flexibility allows for all of the ISC-units in our framework to be reusable, even when introducing a completely new type of agent.

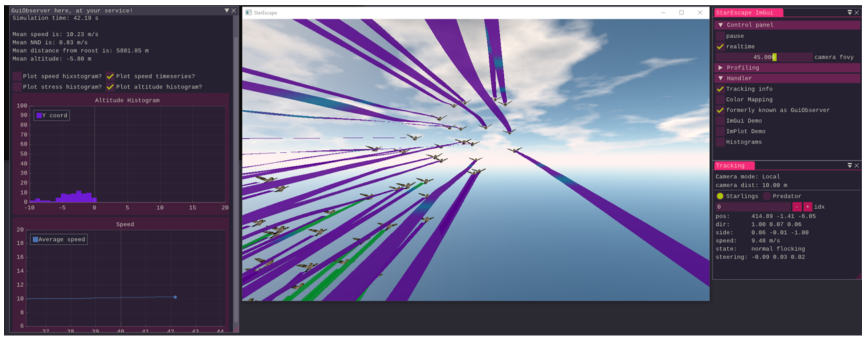
2.4 Customizable locomotion type

The type of motion of grouping individuals (e.g. flying, swimming, or walking) is crucial to the emerging collective behavior [13,11]. Currently, our framework has been used to model bird flocks, and thus the locomotion module resembles flying. For our framework to be extendable to non-flying species, the way that a steering vector is affecting the motion of an agent is controlled by a stand-alone motion integrator. This integrator functions independently of the specifics of an agent (its variables and states). This allows the adjustment of the core of a model to other study systems, without changes being necessary on the structure of the agents, the ISC-units or the states.

2.5 Real-time visualization & data analysis

To incorporate analysis and visualization in our simulations we use the concept of 'observers' (a software design pattern mainly used in event-driven computing [9]). Observers access variables of a simulation in real time without interfering with the model. Based on a chain of specialized observers, we can perform data analysis on the simulated trajectories in real time (for instance calculating the average nearest neighbor distance or the neighbor stability). With a specific observer per analysis type, our simulations can output analyzed data along with raw trajectories in '.csv' format. All exported files are stored within a unique folder, along with a copy of the configuration file of the simulation. The folder is created at the beginning of each simulation, within a user-defined folder (parameter `data_folder`) in the "*bin/sim_data*" directory. This is extremely valuable for pairwise measurements, e.g., neighbor stability, that are computationally very demanding [25] .

The observers enable the real-time visualization of a simulation (for visual debugging and calibration). By changing the representation of an agent, effects of body coloration that affect the observed patterns of collective escape can also be studied (e.g. Fig.1A). A 'head-less' version (without visualization) can also be turned-on in order to speed up the simulation, for instance when running a large number of simulations during data collection. Finally, a graphic user interface


```
GuiObserver here, at your service!
Simulation time: 42.19 s

Mean speed is: 10.23 m/s
Mean NND is: 0.83 m/s
Mean distance from roost is: 5881.85 m
Mean altitude: -5.80 m

[ ] Plot speed hixstogram?   [✓] Plot speed timeseries?
[ ] Plot stress histogram?   [✓] Plot altitude histogram?
```

```
StarEscape ImGui
▼ Control panel
[ ] pause
[✓] realtime
45.000  camera fovy
▶ Profiling
▼ Handler
[✓] Tracking info
[ ] Color Mapping
[✓] formerly known as GuiObserver
[ ] ImGui Demo
[ ] ImPlot Demo
[ ] Histograms
```

```
Tracking
Camera mode: Local
camera dist: 10.00 m
● Starlings  ○ Predator
0  - + idx
pos:      414.89 -1.41 -6.05
dir:      1.00 0.07 0.06
side:     0.06 -0.01 -1.00
speed:    9.48 m/s
state:    normal flocking
steering: -0.09 0.03 0.02
```

Fig. 2. Screenshot of the GUI (created in *Dear ImGui*) and visualization (in OpenGL) of a 3D model of bird flocks based on *DaNCES*. Information from the simulation is collected through *Observers* and thus the visualization and data analysis do not interfere with the model itself, ensuring that the model can be run without them (headless version) and software complexity does not increase. The color of the trails of each agent shows its current state.

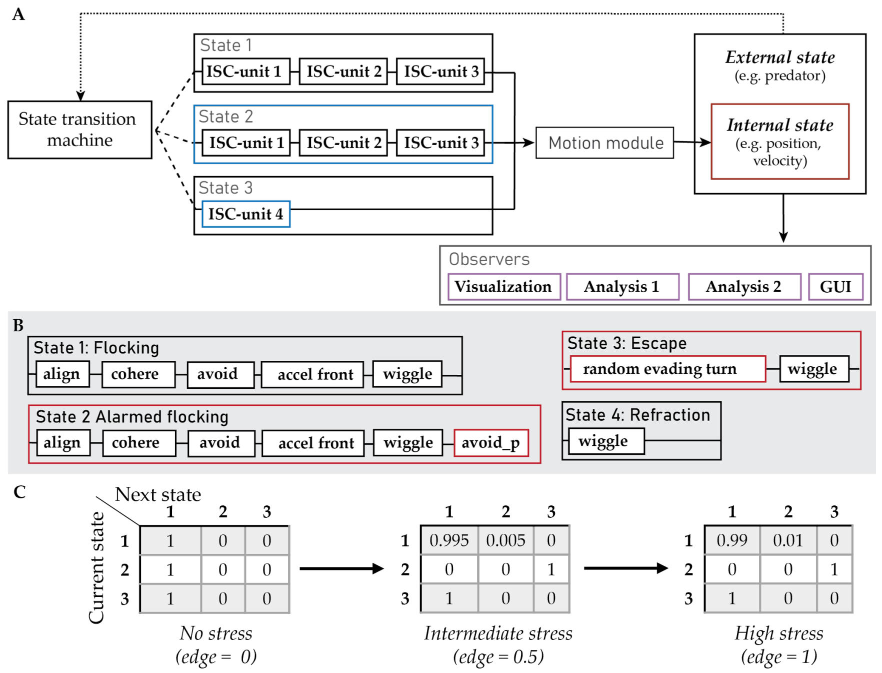
(GUI) allows the real-time plotting and visualization of several variables in the model, such as the current speed or state of each agent (Fig. 2).

An overview of a state machine and its connection with other elements of our framework is given in Fig. 3.

3 Use cases

Our framework has so far been used for four models of bird flocks [27,26,25,28]. *HoPE*, a model adjusted to the collective escape of pigeons, first consisted of a single state with 5 ISC-units: alignment, centroid-attraction, speeding attraction, noise, and predator avoidance [27]. Since predator avoidance is modeled as a continuous tendency to turn away from the predator while coordinating, similar to many models of collective escape [19,38], an extra state to model escape was not needed. Given that this model couldn't capture all patterns of collective escape seen in pigeons, an extension of the model was created to study the propagation of escape maneuvers [26]. Patterns of collective escape were conceptualized to being initialized by one individual that stops coordinating with its neighbors and maneuvers to escape the predator for a specific time period. Additionally, a refractory time period is added after an escape maneuver, during which an individual does not coordinate with its neighbors to ensure that the emergence of splitting and collective turning is the effect of the group's 'decision'. Thus, the extended *HoPE* model consists of three states: a flocking state (the one from its previous version [27]), an escaping state (with two ISC-units, an escape maneuver and noise), and a refraction state (with a single ISC-unit that adds noise to the straight heading of the initiator).

A model of higher complexity based on our framework is the 3-dimensional model of starlings (*Sturnus vulgaris*), *StarEscape* [28]. Aiming to reproduce their



	1	2	3
1	1	0	0
2	1	0	0
3	1	0	0

	1	2	3
1	0.995	0.005	0
2	0	0	1
3	1	0	0

	1	2	3
1	0.99	0.01	0
2	0	0	1
3	1	0	0

Fig. 3. **A.** Schematic representation of a state machine and its connection with the other elements of our framework. The transition machine selects a state depending on the agent's internal (e.g., its position, the previously selected state), and external state (e.g., its distance to the predator). States can be composed of a different combination of ICS-units and have their own parameterization. The different border colors represent different parameter values. State 1 and 2 are identical in their ISC-units but differ in their state-level parameter values (for instance in their duration or the agent's reaction frequency). The output of the ISC-units changes the internal state of an agent depending on the specifics of the motion module. Observers have access to an agent's external and internal state. Several observers with different functions (e.g., to export, analyze or visualize the simulated data) can be added in a model. **B.** An example of the states and ISC-units similar to the ones used for the extended HoPE model [26]. Red borders relate to escape-related blocks. **C.** An example of 3 user-defined transition matrices for a model with 3 states (1:flocking, 2:escape, 3:refraction). As stress increases, the probability to escape (transition from state 1 to 2) should also increase. The transition machine will interpolate across the given values depending on the value of stress and the input edges of the interpolation.

aerial displays [3], the model includes 4 states: flocking, alarmed flocking, escape (which as mentioned earlier is a multi-state), and refraction. Flocking in *StarEscape* includes the following 6 ISC-units: alignment, centroid-attraction (distance biased, ensuring sharp edges of the flock similar to real starlings [18]), avoidance, noise, roost attraction, and altitude attraction. The last two ISC-units constrain the movement of the flock within the 3-dimensional 'infinite' space: they ensure that individuals will stay within a given radius and flying at a preferred altitude. The alarmed flocking state is structurally identical with the flocking state, apart from being parameterized with a higher reaction frequency (imitating more vigilant individuals when the predator is pursuing the flock [2,16]) and an extra ISC-unit that affects the transition machine: a ‘copy-escape’ behavior. This ISC-unit checks whether any neighbor is in the escape state. If so, it sets the future state in the transition machine to be the same as its neighbors. The escape multi-state has two states that include the coordination-related ISC-units of the flocking states, and an additional maneuvering ISC-unit, one for a diving maneuver and one for a turning maneuver. Finally, the refraction state, has the coordination-related ISC-units of the flocking states but a predefined duration. Separating these states ensures that individuals will not keep reacting to the predator in an unrealistic manner.

4 Summary and potential

In our new framework, a model is composable, flexible and reusable. By the user changing a few lines of code in the definition of an agent’s state machine and the configuration file, a new model can be created. A large set of already available ISC-units can be found in our repo [24]. Furthermore, given the stand-alone nature of ISC-units, adding a new one requires minimal coding and does not interfere with the chain of the model itself. Apart from adding new rules, the user can also quickly remove elements of a model for comparing simulations of decreasing complexity. This is extremely valuable when searching for a simple self-organized process to explain the emergence of a collective pattern.

Following the example of the ‘Medawar’ zone of pattern-oriented models [10], we should emphasize that the increasing complexity of models that our framework supports should be used with caution: a very large parameter space can limit the explanatory power of the model and constrain our understanding of the underlying mechanisms of the complex system. Finding the balance between minimal models and models derived from complex empirical data is challenging, and should be reached while keeping the study’s research question in mind. We hope that DaNCES supports modelers while seeking this balance, trains students in the modeling of animal behavior, and promotes clear (and conscious) assumptions and modeling decisions.

Apart from the flexibility of our models, our framework’s structure enables the modeling of different behaviors that are crucial to collective escape, for instance the interplay between continuous and instantaneous reactions. Because of its ‘building-blocks’, we can develop complex models that reproduce long sequences of collective escape similar to the ones seen in nature without increasing the complexity of the code itself. This reduced code complexity also includes the avoidance of long series of nested conditional blocks (if-statements) which, as a model grows, become error prone, difficult to work with, and introduce optimization issues (code that is rarely used increases the computation time). Every block in our framework (being an ISC-unit, a state, a transition machine or a motion interpolator) is a pure function that owns a set of parameters and effectively alters the state of an agent. Because of their simplicity, they are reusable, can be more easily optimized, while providing code clarity and debugging efficiency. Thus, the extension and growth of a model based on our framework is less error prone and less time consuming.

To conclude, our simulation framework provides a skeleton to build a new agent-based model rather than being just a function library. Its building-block structure and detailed parameterization allows for a model to be informed with as many empirical data are available, increasing its biological relevance. Programming-wise, it does not impose our implementation, allowing a more experienced user to work in their own modeling style. Overall, if one can conceptualize a set of rules and actions that agents should follow, along with the interconnection (time sequence) of these actions, it can be modeled in our framework. We hope that it will enable the modeling of many systems for which our theoretical understanding is currently lacking, such as collective escape from multiple predators [12] or with highly complex displays [34] (Fig.1A,1D). Despite our focus on collective escape, our framework can be used for the spatially explicit modeling of collectives across contexts.

References

1. Angelani, L.: Collective predation and escape strategies. Physical Review Letters **109**(11), 1–5 (2012). https://doi.org/10.1103/PhysRevLett.109.118104
2. Bode, N.W., Faria, J.J., Franks, D.W., Krause, J., Wood, A.J.: How perceived threat increases synchronization in collectively moving animal groups. Proceedings of the Royal Society B: Biological Sciences **277**(1697), 3065–3070 (2010)
3. Carere, C., Montanino, S., Moreschini, F., Zoratto, F., Chiarotti, F., Santucci, D., Alleva, E.: Aerial flocking patterns of wintering starlings, Sturnus vulgaris, under different predation risk. Animal Behaviour **77**(1), 101–107 (2009)
4. Cornut, O.: Dear imgui: Bloat-free graphical user interface for c++ with minimal dependencies (2024), `https://github.com/ocornut/imgui`
5. Couzin, I.D., Krause, J.: Self-Organization and Collective Behavior in Vertebrates. Advances in the Study of Behavior **32**, 1–75 (2003)
6. Domenici, P.: Context-dependent variability in the components of fish escape response: Integrating locomotor performance and behavior. Journal of Experimental Zoology Part A: Ecological Genetics and Physiology **313 A**(2), 59–79 (2010)
7. Domenici, P., Blagburn, J.M., Bacon, J.P.: Animal escapology I: theoretical issues and emerging trends in escape trajectories. Journal of Experimental Biology **214**(15), 2463–2473 (2011). https://doi.org/10.1242/jeb.029652
8. Doran, C., Bierbach, D., Lukas, J., Klamser, P., Landgraf, T., Klenz, H., Habedank, M., Arias-Rodriguez, L., Krause, S., Romanczuk, P., Krause, J.: Fish waves as emergent collective antipredator behavior. Current Biology **32**(3), 708–714.e4 (Feb 2022). https://doi.org/10.1016/j.cub.2021.11.068
9. Gamma, E., Helm, R., Johnson, R., Johnson, R., Vlissides, J.: Design patterns : elements of reusable object-oriented software. Pearson Deutschland GmbH (1995)
10. Grimm, V., Revilla, E., Berger, U., Jeltsch, F., Mooij, W.M., Railsback, S.F., Thulke, H.H., Weiner, J., Wiegand, T., DeAngelis, D.L.: Pattern-oriented modeling

of agent-based complex systems: Lessons from ecology. Science **310**(5750), 987–991 (2005). https://doi.org/10.1126/science.1116681
11. Gyllingberg, L., Szorkovszky, A., Sumpter, D.J.: Using neuronal models to capture burst-and-glide motion and leadership in fish. Journal of the Royal Society Interface **20**(204), 20230212 (2023)
12. Hansen, M.J., Domenici, P., Bartashevich, P., Burns, A., Krause, J.: Mechanisms of group-hunting in vertebrates. Biological Reviews **98**(5), 1687–1711 (2023)
13. Hemelrijk, C.K., Hildenbrandt, H.: Schools of fish and flocks of birds: their shape and internal structure by self-organization. Interface Focus **2**(6), 726–737 (2012)
14. Hemelrijk, C.K., van Zuidam, L., Hildenbrandt, H.: What underlies waves of agitation in starling flocks. Behavioral Ecology and Sociobiology **69**(5), 755–764 (2015)
15. Herbert-Read, J.E., Buhl, J., Hu, F., Ward, A.J.W., Sumpter, D.J.: Initiation and spread of escape waves within animal groups. Royal Society Open Science **2**(4), 140355–140355 (2015). https://doi.org/10.1098/rsos.140355
16. Herbert-Read, J.E., Rosén, E., Szorkovszky, A., Ioannou, C.C., Rogell, B., Perna, A., Ramnarine, I.W., Kotrschal, A., Kolm, N., Krause, J., Sumpter, D.J.: How predation shapes the social interaction rules of shoaling fish. Proceedings of the Royal Society B: Biological Sciences **284**(1861) (2017)
17. Herbert-Read, J.E., Ward, A.J.W., Sumpter, D.J., Mann, R.P.: Escape path complexity and its context dependency in Pacific blue-eyes (Pseudomugil signifer). The Journal of Experimental Biology **220**(11), 2076–2081 (2017)
18. Hildenbrandt, H., Carere, C., Hemelrijk, C.K.: Self-organized aerial displays of thousands of starlings: A model. Behavioral Ecology **21**(6), 1349–1359 (2010)
19. Inada, Y., Kawachi, K.: Order and Flexibility in the Motion of Fish Schools. Journal of Theoretical Biology **214**(3), 371–387 (2002)
20. Jolles, J.W., King, A.J., Killen, S.S.: The Role of Individual Heterogeneity in Collective Animal Behaviour. Trends in Ecology & Evolution **35**(3), 278–291 (Mar 2020). https://doi.org/10.1016/j.tree.2019.11.001
21. Jones, K.A., Jackson, A.L., Ruxton, G.D.: Prey jitters; protean behaviour in grouped prey. Behavioral Ecology **22**, 831–836 (2011)
22. Mills, R., Hildenbrandt, H., Taylor, G.K., Hemelrijk, C.K.: Physics-based simulations of aerial attacks by peregrine falcons reveal that stooping at high speed maximizes catch success against agile prey. PLoS Computational Biology **14**(4), 1–38 (2018). https://doi.org/https://doi.org/10.1371/journal.pcbi.1006044
23. Papadopoulou, M., Fürtbauer, I., O'Bryan, L.R., Garnier, S., Georgopoulou, D.G., Bracken, A.M., Christensen, C., King, A.J.: Dynamics of collective motion across time and species. Philosophical Transactions of the Royal Society B **378**(1874), 20220068 (2023)
24. Papadopoulou, M., Hildenbrandt, H.: DaNCES framework (2024), `https://github.com/marinapapa/DaNCES_framework`
25. Papadopoulou, M., Hildenbrandt, H., Hemelrijk, C.K.: Diffusion during collective turns in bird flocks under predation. Frontiers in Ecology and Evolution **11**, 1198248 (2023)
26. Papadopoulou, M., Hildenbrandt, H., Sankey, D.W.E., Portugal, S.J., Hemelrijk, C.K.: Emergence of splits and collective turns in pigeon flocks under predation. Royal Society Open Science **9**, 211898 (2022)
27. Papadopoulou, M., Hildenbrandt, H., Sankey, D.W.E., Portugal, S.J., Hemelrijk, C.K.: Self-organization of collective escape in pigeon flocks. PLOS Computational Biology **18**(1), e1009772 (Jan 2022). https://doi.org/10.1371/journal.pcbi.1009772
28. Papadopoulou, M., Hildenbrandt, H., Storms, R., Hemelrijk, C.K.: Starling murmurations under predation. In prep. (2024)

29. Pettit, B., Perna, A., Biro, D., Sumpter, D.J.: Interaction rules underlying group decisions in homing pigeons. Journal of the Royal Society Interface **10**(89) (2013)
30. Reynolds, C.W.: Flocks, herds and schools: A distributed behavioral model. ACM Computer Graphics **21**(4), 25–34 (1987). https://doi.org/10.1145/37402.37406
31. Sankey, D.W., Shepard, E.L., Biro, D., Portugal, S.J.: Speed consensus and the 'Goldilocks principle' in flocking birds (Columba livia). Animal Behaviour **157**, 105–119 (2019). https://doi.org/10.1016/j.anbehav.2019.09.001
32. Sankey, D.W., Storms, R.F., Musters, R.J., Russell, T.W., Hemelrijk, C.K., Portugal, S.J.: Absence of "selfish herd" dynamics in bird flocks under threat. Current Biology pp. 1–7 (2021). https://doi.org/10.1016/j.cub.2021.05.009
33. Storms, R.F., Carere, C., Musters, R., Van Gasteren, H., Verhulst, S., Hemelrijk, C.K.: Deterrence of birds with an artificial predator, the robotfalcon. Journal of the Royal Society Interface **19**(195), 20220497 (2022)
34. Storms, R.F., Carere, C., Zoratto, F., Hemelrijk, C.K.: Complex collective motion: collective escape patterns of starling flocks under predation. Behavioral Ecology and Sociobiology **73**(10) (2019). https://doi.org/10.1007/s00265-018-2609-0
35. Thiebault, A., Semeria, M., Lett, C., Tremblay, Y.: How to capture fish in a school? Effect of successive predator attacks on seabird feeding success. Journal of Animal Ecology **85**(1), 157–167 (2016). https://doi.org/10.1111/1365-2656.12455
36. Tunstrøm, K., Katz, Y., Ioannou, C.C., Huepe, C., Lutz, M.J., Couzin, I.D.: Collective States, Multistability and Transitional Behavior in Schooling Fish. PLoS Computational Biology **9**(2) (2013). https://doi.org/10.1371/journal.pcbi.1002915
37. Wilson, A.D., Krause, S., James, R., Croft, D.P., Ramnarine, I.W., Borner, K.K., Clement, R.J., Krause, J.: Dynamic social networks in guppies (poecilia reticulata). Behavioral Ecology and Sociobiology **68**(6), 915–925 (2014)
38. Zheng, M., Kashimori, Y., Hoshino, O., Fujita, K., Kambara, T.: Behavior pattern (innate action) of individuals in fish schools generating efficient collective evasion from predation. Journal of Theoretical Biology **235**(2), 153–167 (2005)