**Supplementary Material**

**Materials and methods: model description**

The models were coded in the C programming language. The model description follows the ODD protocol (Grimm et al. 2006; Grimm et al. 2010).

*Purpose*

 The purpose of the model was to examine how adaptive movements emerged, which was dependent on resource distribution. I allowed the agent to scan the local food distribution and estimate an abundant area (or sparse area). The agent then modified its directional rule. I named the proposed model the direction-modified model (or the DM model). The Brownian walker model was used as a control model.

*Entities, state variables, and scales*

I developed two different models, and these models included two types of entities: an agent and cells. The agent has the state variable *Navigational state*, which has two values: *Navigational state* = {Brownian, biased}. A cell has the state variable *Resource*; this value represents the number of food resources at each cell.

The field size was defined as 1000 × 1000 cells and periodic boundaries were assumed for the agent movements. Each trial was run for 1000 time-steps. Cells have food resources, which are randomly distributed or patchily distributed. In the latter condition, a landscape containing patchily distributed resources was used. For simplicity, it was assumed that each patch had the same size (5 × 5 cells) and contained 10 food items; the center of each patch was randomly distributed in the field and the 10 food items were randomly distributed in each patch (Figure S1), two patches could overlap, but patches were not permitted to roll over the periodic edges of the field.

*Process overview and scheduling*

In both models, the agent at coordinate (*x*, *y*) always updates its position with one of the four coordinates (*x*−1, *y*), (*x*+1, *y*), (*x*, *y*−1), or (*x*, *y*+1) at each time-step. The Brownian walker model, in which the agent always chooses one direction from four directions (–*x*, +*x*, -*y*, +*y*), was used to evaluate the resource search ability of the DM model. I used a von Neumann neighborhood for all agent behaviors (movement, search, and consumption).

In the DM model, at each time-step, the agent was allowed to scan the local resource distribution using the nearest four neighbors (see sub-model entitled “Local pattern calculation”). Based on the local pattern, the agent subjectively estimated the directions of abundant and sparse resource areas. Afterwards, the agent sometimes coordinated its directional rule (see sub-model entitled “Directional rule coordination”) and consumed food items if they were located within the agent’s consumption field (see sub-model entitled “Consuming food items”). The agent then updated its position (see sub-model entitled “Position updating”). Finally, some food items, which had previously been consumed by the agent, reappeared on the field (see sub-model entitled “Reappearance of food items”).

In the Brownian model, at each time-step, the agent always updated its position using a fixed directional rule. Therefore, only three sub-models were used for this model: “Consuming food items,” “Position updating,” and “Reappearance of food items.”

*Design concepts*

The movement patterns, diffusive properties, and food exploitation capabilities are the emergent properties of the model. Sensing is important since the agent scans the local food distribution. Stochasticity was used to determine in which direction the agent moved when it obeyed a Brownian-like movement.

*Initialization*

I initially set the agent at the coordinates (0, 0). The default *Navigational state* was set to Brownian at the beginning of each trial. The resource density was set to 0.30. Therefore, in both conditions (randomly distributed or patchily distributed), there were 300,000 food items in the field.

*Sub-models of the DM model*

On each time-step *t*,

STEP 1: *Local pattern calculation*

STEP 2: *Directional rule reset (as required)*

STEP 3: *Directional rule coordination (as required: if local food pattern at time t−1 is biased)*

STEP 4: *Consuming food items*

STEP 5: *Position updating*

STEP 6: *Reappearance of food items*

STEP 7: *t* → *t*+1

*Sub-model* “*Local pattern calculation”*

It was assumed that the agent could not visually calculate global patterns of food distribution. Therefore, at each time-step, the agent strictly calculates local patterns of food distribution, which indicate the spatial distribution of food. Food resources are sometimes uniformly distributed and sometimes distributed in an unbalanced manner on the field. The agent scans the nearest four cells to make a pattern calculation. When one or three cells are occupied by food, the agent estimates that the local food distribution is biased in a certain direction. For the agent, biased local food distributions represent a boundary between an abundant area and a sparse area. For example, the agent located at coordinate (*x*, *y*) estimates that the abundant or sparse area is located in the right or left direction if the three cells (*x* + 1, *y*), (*x*, *y*−1), and (*x*, *y* + 1) are occupied by food (also see Fig. 1).

Thus, for an agent located at coordinate (*x*, *y*),

*Direction*t = *right*,

if three cells, (*x* + 1, *y*), (*x*, *y*−1), and (*x*, *y* + 1), or one cell, (*x* + 1, *y*), are occupied by food.

*Direction*t = *left*,

else if three cells, (*x* - 1, *y*), (*x*, *y*−1), and (*x*, *y* + 1), or one cell, (*x* - 1, *y*), are occupied by food.

*Direction*t = *upper*,

else if three cells, (*x*, *y*+1), (*x*+1 *y*), and (*x*−1, *y*), or one cell, (*x*, *y*+1), are occupied by food.

*Direction*t = *lower*,

else if three cells, (*x*, *y*−1), (*x*+1 *y*), and (*x*−1, *y*), or one cell, (*x*, *y*−1), are occupied by food.

Here, *Direction*t represents the estimated direction of the food resources (an abundant area) at time *t*.

*Sub-model “Directional rule coordination (if local food pattern at time t−1 is biased)”*

If the agent estimates that the local food distribution is biased at time *t*−1, it coordinates its directional rules at time *t* as follows only when its *Navigational state* = Brownian:

if one cell or no cells of the nearest four cells are occupied by food at time *t*,

*Rule*for the opposite direction of *Direction*t−1 = *Prob*high,

*Rule*for *Others* = (1.00 − *Prob*high)/3.

else if two or more cells of the nearest four cells are occupied by food at time *t*,

*Rule*for the opposite direction of *Direction*t−1= *Prob*low,

*Rule*for *Others* = (1.00 − *Prob*low)/3.

Here, *Rule*for *A* represents the directional rule by which the agent moves in the direction *A*, which satisfies *A* = {*left, right, lower, upper*},

*Rule*for *A* = [0.0, 1.0],

*Rule*for *left* + *Rule*for *right* + *Rule*for *lower* + *Rule*for *upper* = 1.00.

Thus, the agent moves in the direction *A* with a probability “*Rule* for *A*” at each time-step. Figure 2 presents examples of directional rule coordination. After directional rule coordination occurs, the agent obeys the replaced directional rules, i.e., it produces biased movements until the directional rule resets (below). In addition, at the end of this sub-model, the value of *Navigational state* becomes biased where previously it was Brownian.

*Sub-model “Directional rule reset (as needed)”*

The agent resets its directional rule to the default rule as follows only when *Navigational state* = biased and three or four cells of the nearest cells are all occupied by food:

*Rule*for *left* = 0.25,

*Rule*for *right* = 0.25,

*Rule*for *lower* = 0.25,

*Rule*for *upper* = 0.25.

*Sub-model “Consuming food items”*

The agent consumes food items when they are located within its consumption field (≤*C*onsumption radii for the agent = 1.00) like follows;

 if sqrt((*x*\*-*x*)2 + (*y*\*-*y*)2) ≤ *C*onsumption and *Reappear*\*= 0

then the agent consumes the food item and set *Reappear*\*= *threshold*

Here, (*x*\*, *y*\*) and (*x*, *y*) present the position of the food item \* and the agent respectively. *Reappear*\* = 0 indicates that the food item \* appears on the field. When *Reappear*\* is bigger than 0, food item \* does not appear on the field.

*Sub-model “Position updating”*

The agent updates its position as follows using the directional rules:

*x* = *x*−1, *y* = *y*, with a probability *Rule*for *left*

*x* = *x*+1, *y* = *y*, with a probability *Rule*for *right*

*x* = *x*, *y* = *y*−1, with a probability *Rule*for *lower*

*x* = *x*, *y* = *y*+1, with a probability *Rule*for *upper*

*Sub-model “Reappearance of food items”*

After the agent consumes food items, those items disappear for a period (*threshold\_vanish* = 20); therefore, the agent is unable to detect and consume food items that they have already consumed within this period, i.e., a given number of time-steps.

After those time-steps have passed, the disappeared food items reappear on the field as follows;

For the food item \*,

if *Reappear*\*> 0, then *Reappear*\*= *Reappear*\*- 1

*Brownian model’s sub-models*

On each time-step *t*,

STEP 1: *Consuming food items*

STEP 2: *Position updating*

STEP 3: *Reappearance of food items*

STEP 4: *t* → *t*+1

Each sub-model for the Brownian model is the same as the sub-models for the DM model.

**Results-Parameter Effects.**

The performance of the DM model was compared against that of a model in which the agent always moved towards local resources. In the latter model, *Prob*high was changed from 0.90 to 0.10, which forced the agent to move in the direction of a (local) abundant area after it determined the local food pattern to be biased. The DM model once again outperformed the comparative model in terms of food exploitation ability (DM model = 0.59 ± 0.17 vs. comparative (*Prob*high = 0.10) model = 0.40 ± 0.027, *χ*2= 35.92, d*f* = 1, *p* < 0.001).

Similarly, I assessed the influence of replications (number of trials) using the default resource situation. I found that similar results were obtained in respect with the food exploitation even after number of trials was replaced with 200 trials (DM model = 0.59 ± 0.17 vs. comparative (*N* of trials = 200) model = 0.60 ± 0.16, *χ*2< 0.001, d*f* = 1, *p* = 0.98). This suggests that 100 trials are sufficient.

References

1. Grimm V et al (2006) A standard protocol for describing individual-based and agent-based models. Ecol Model 198: 115–126
2. Grimm V, Berger U, DeAngelis DL, Polhill JG, Giske J, Railsback SF (2010) The ODD protocol: a review and first update. Ecol Model 221(23): 2760–2768

Table S1. Food exploitation results and H for a randomly distributed resource landscape.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Density** | **0.3** | **0.4** | **0.5** | **0.6** |
| **N of consumed food/step (DM)** | 0.59 | 0.83 | 0.97 | 1.1 |
| **N of consumed food/step (Brownian)** | 0.37 | 0.49 | 0.60 | 0.73 |
| **H (DM)** | 0.72 | 0.55 | 0.55 | 0.54 |

Table S2. Food exploitation results and H values for a patchily distributed resource landscape.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Density** | **0.3** | **0.4** | **0.5** | **0.6** |
| **N of consumed food/step (DM)** | 0.59 | 0.78 | 0.92 | 1.1 |
| **N of consumed food/step (Brownian)** | 0.37 | 0.49 | 0.61 | 0.73 |
| **H (DM)** | 0.60 | 0.61 | 0.57 | 0.53 |



**Fig. S1.** Examples of the food resource distribution (resource density = 0.3) of the patchily distributed landscape. An enlarged view (100 × 100 size) is shown in both cases.



**Fig. S2.** Step length and its cumulative frequency obtained from one trial. Step length along (A) the x-axis and (B) the y-axis. Diagonal lines correspond to power-law fits.