FLARE: Future LApwing Responses to Expansions

An Agent-Based Simulation Approach to Analyse Current and Future Movement of *Vanellus vanellus Linn.* in a Changing Landscape

Submitted By

Jannatul Ferdous
Master's Student
Chair of Tropical Forestry
Technische Universität Dresden

Supervisor:

Dr. Elina Takola
Postdoctoral researcher
Department of Computational Landscape Ecology,
Helmholtz Centrefor Environmental Research - UFZ.

Leipzig, May 2025

HELMHOLTZ
Zentrum für Umweltforschung

Contents

List of Figures

List of Tables

1. Background

In recent years, the Netherlands has been working to adapt nature-based approaches to achieve the Sustainable Development Goals (SDGs). But over the years, nature-based solutions have only been implemented through unplanned expansion of green areas without considering the necessity of different habitat types and neglecting biodiversity and social equity. (Bulkeley et al., 2023). However, the recent policy mandate Vision 2021 focuses on developing a high-quality nature-based solution plan, avoiding the tendency of unplanned green expansion, instead, it talks more about circular agriculture, energy-efficient horticulture using greenhouses, climate-smart management of peatlands, forests and different habitat types (Baptist et al. 2019) . Despite their focus towards developing a high-quality management plan, there's still a growing need for urbanisation in the Netherlands. Although most of the urbanisations are being carried out in the primary built-up areas, the rate of rural urbanisation is also not negligible. Dellar et al., 2024 has created hypothetical future land use maps which present the possible land use scenarios of the Netherlands in 2050. Although this study is focused more on analysing shared socio-economic pathways and health risks in those scenarios, the projection of land use changes can act as a base for biodiversity studies.

The species Northern Lapwings (*Vanellus vanellus Linn.*) are long migratory birds distributed across the northern Palearctic region (Musters et al. 2010). They have been facing a declining population for the past few decades and have been listed as Vulnerable in the European Red List of species (HELCOM Red List Bird Expert Group 2013). Remarkably, the Netherlands comes under the zone of their breeding habitat, which makes them susceptible to any change of land uses that will be carried out in the Netherlands (international 2016).

Therefore, this project aims to predict the shift in future habitat preference of *V. vanellus* in different hypothetical scenarios. To attain this aim, an agent-based model simulation approach is being developed, which will first analyse the sensitivity of *V. vanellus* to different land use classes and, based on their preferences, it will predict the future habitat shift or preference with different land use change scenarios.

Data source: The data in this study has been obtained from Movebank's global open-source database (Kranstauber et al. 2011). The map used for the studies has been downloaded from the open-source map database of Wageningen Environmental Research (Hazeu et al. 2023) and then cropped into the targeted region of Friesland for computational efficiency. For model precision, a 5m resolution map has been used.

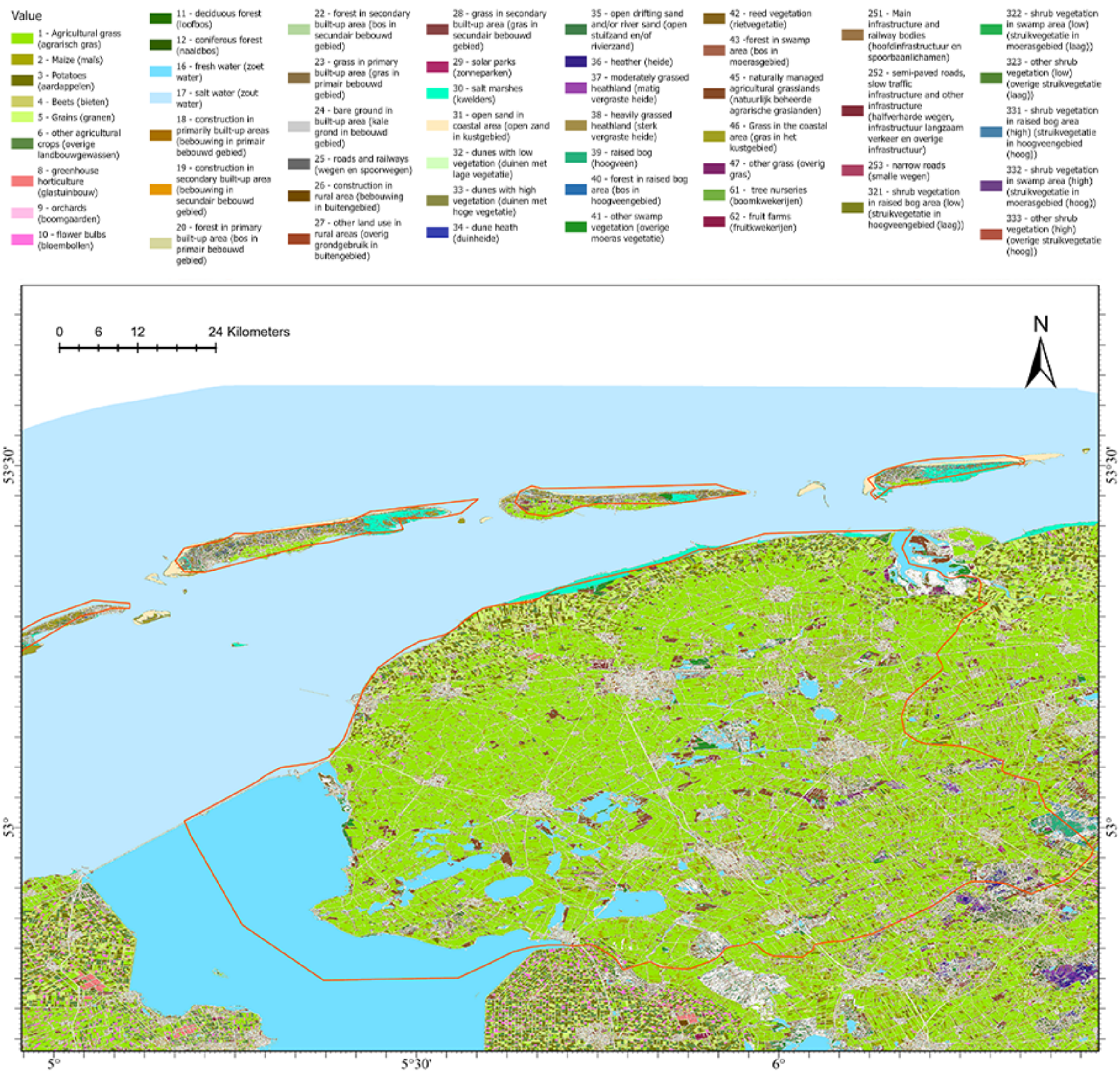

Figure 1. 5m resolution land use and land cover map of Friesland

2. Working Framework

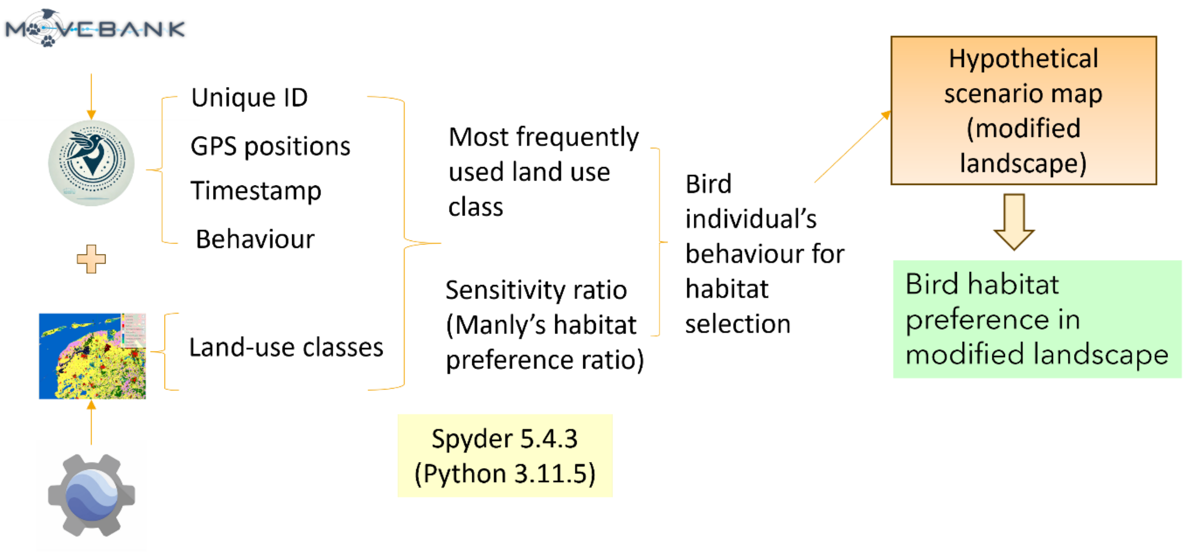

Figure 2. Working framework

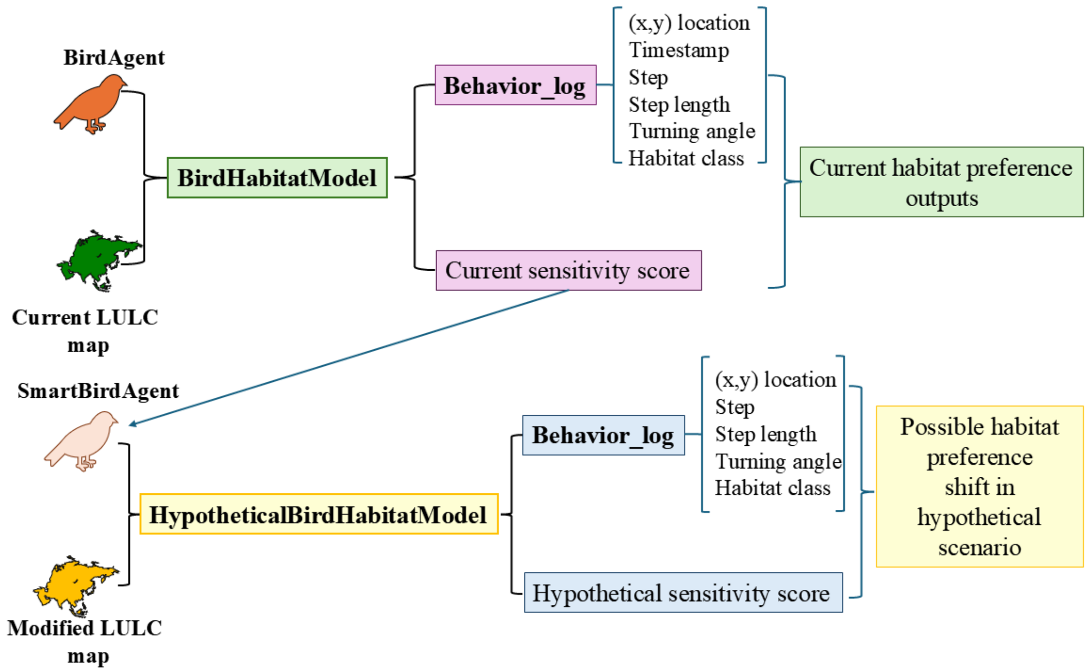

Figure 3. Model workflow with corresponding agents and environments

3. Agent-based Model Components

Agents: Northern Lapwing individuals

1. BirdAgent
2. SmartBirdAgent

Environment: Landcover classes

1. Current scenario land cover map
2. Hypothetical scenario landcover map

Variables:

1. Behaviour log
    a. Step
    b. Step length
    c. Habitat class
    d. Turning angle
    e. Latitude, longitude
    f. Timestamp (for current model only)
2. Sensitivity score (Manly's habitat selection ratio)

4. Model Description[1]

BirdAgent: The BirdAgents resemble the actual birds who follow the real time GPS tracking paths.

```python
# BirdAgent class
class BirdAgent:
    def __init__(self, unique_id, movement_data):
        self.unique_id = unique_id
        self.movement_data = movement_data  # Bird's recorded movement data
        self.current_step = 0  # Track movement data index
        self.current_position = None
        self.previous_position = None
        self.habitat_preference = {}
        self.behavior_log = []  # To record each step's habitat and behavior

    def step(self, model, sensitivity=None):
        """Move the bird based on movement data and record habitat preferences."""
        if self.current_step < len(self.movement_data):
            # Read the bird's location
            data = self.movement_data.iloc[self.current_step]
            timestamp = data['timestamp']
            current_position = Point(data['location.long'], data['location.lat'])

            # Get the habitat type at the current position
            habitat_value = model.get_habitat_at_position(current_position)

            # Adjust step length based on sensitivity
            step_length_multiplier = 1.0  # Default multiplier
            if habitat_value is not None and sensitivity is not None:
                habitat_sensitivity = sensitivity.get(habitat_value, 1.0)
                step_length_multiplier = habitat_sensitivity
```

Definition of BirdAgent, so it will create the exact number of agents according to the unique IDs, record the bird movement data and also record down the habitat classes in the behavior_log[]

Step initiation function which takes data from the step positions and preserves it in the behaviour_log[]

Figure 4. BirdAgent definition and initiation functions

Instead of calculating step length in a straight line, this function calculates the step length considering the turning angle, which resembles the actual distance that the bird has covered from one point to another

```python
            # Calculate step length and turning angle
            step_length = 0
            turning_angle = None

            if self.previous_position:
                dx = (current_position.x - self.previous_position.x) * step_length_multiplier
                dy = (current_position.y - self.previous_position.y) * step_length_multiplier
                step_length = np.sqrt(dx**2 + dy**2)

                if self.current_position:
                    prev_dx = self.current_position.x - self.previous_position.x
                    prev_dy = self.current_position.y - self.previous_position.y

                    dot_product = prev_dx * dx + prev_dy * dy
                    magnitude_a = np.sqrt(prev_dx**2 + prev_dy**2)
                    magnitude_b = np.sqrt(dx**2 + dy**2)

                    if magnitude_a > 0 and magnitude_b > 0:
                        cosine_angle = dot_product / (magnitude_a * magnitude_b)
                        cosine_angle = np.clip(cosine_angle, -1.0, 1.0)  # <--- safe clipping
                        turning_angle = np.arccos(cosine_angle)
                        turning_angle = np.degrees(turning_angle)


            # Update positions
            self.previous_position = self.current_position
            self.current_position = current_position
```

Figure 5. BirdAgent turning angle and step length calculation

[1] The model will be checked for further bugs and in case of any bugs, further versions will be available with the bug fixes

```python
        # Update positions
        self.previous_position = self.current_position
        self.current_position = current_position

        # Record habitat preference and behavior
        if habitat_value is not None:
            self.habitat_preference[habitat_value] = self.habitat_preference.get(habitat_value, 0) + 1

        self.behavior_log.append({
            "Step": self.current_step + 1,
            "Timestamp": timestamp,
            "Habitat Class": habitat_value,
            "Longitude": current_position.x,
            "Latitude": current_position.y,
            "Step Length": step_length,
            "Turning Angle": turning_angle
        })

        self.current_step += 1
```

This function here directs which variables will be preserved in the behaviour_log[]. This includes number of step, timestamp, habitat class, lat-long data of the position, step length and turning angle

Figure 6. BirdAgent behavior_log[] elements

SmartBirdAgent: The SmartBirdAgent is another agent class of this model, which is also created based on the unique IDs of individual birds from our original data. But instead of following the exact GPS points, they integrate the recorded behaviour of our BirdAgents and their Sensitivity to a habitat. Hence, when the model starts, it starts looking for a preferred habitat around it. It analyses every pixel around it and then selects one of the most preferred habitats randomly from the sensitivity scores that are attained from the BirdHabitatModel. Here, introducing the randomness avoids the bias of choosing the best and keeps the choices more natural.

```python
#SmartBirdAgent class for hypothetical model
class SmartBirdAgent(BirdAgent):
    """BirdAgent that moves based on habitat sensitivity."""

    def step(self, model, sensitivity=None):
        """Move the bird intelligently based on habitat preferences."""
        if self.current_step == 0:
            # Initialize at the real first position
            data = self.movement_data.iloc[self.current_step]
            current_position = Point(data['location.long'], data['location.lat'])
            self.current_position = current_position
            self.previous_position = None
            self.current_step += 1
            return

        if self.current_position is None:
            return

        directions = [
            (0.0005, 0), (0.0005, 0.0005), (0, 0.0005), (-0.0005, 0.0005),
            (-0.0005, 0), (-0.0005, -0.0005), (0, -0.0005), (0.0005, -0.0005)
        ]
```

This is the most important array of this agent class. These eight pairs of values define the eight directions and the searching range

Figure 7. SmartBirdAgent class direction array

The direction array follows the Cartesian coordinate system. It works like a vector grid, for example: (0.0005,0) represents 50m towards East.

```python
candidates = []
for dx, dy in directions:
    candidate_pos = Point(self.current_position.x + dx, self.current_position.y + dy)
    habitat_value = model.get_habitat_at_position(candidate_pos)

    if habitat_value is not None:
        # if normal valid move record location
        score = sensitivity.get(habitat_value, 1.0)
        candidates.append((score, candidate_pos, habitat_value))
    else:
        # Outside raster: Bounce back within the map boundary
        bounce_pos = Point(self.current_position.x - dx, self.current_position.y - dy)
        bounce_habitat_value = model.get_habitat_at_position(bounce_pos)

        if bounce_habitat_value is not None:
            bounce_score = sensitivity.get(bounce_habitat_value, 1.0)
            candidates.append((bounce_score, bounce_pos, bounce_habitat_value))

# Now choose among candidates
if not candidates:
    # No valid moves even after bouncing
    self.behavior_log.append({
        "Step": self.current_step + 1,
        "Timestamp": None,
        "Habitat Class": None,
        "Longitude": self.current_position.x,
        "Latitude": self.current_position.y,
        "Step Length": 0,
        "Turning Angle": None
    })
    self.current step += 1
```

This loop of SmartBirdAgent ensures that the bird individuals don't go outside the map boundary.

Figure 8. SmartBirdAgent: loop for maintaining the movement into Raster boundary

As the birds in SmartBirdAgent class move independently, they might show a tendency of going outside the raster boundary. Especially, if the study area has segmented islands like Friesland and some GPS points have been recorded in the ocean or near the edges. This loop ensures that even if the birds cross the raster boundary, they will bounce back into the boundary and follow their normal behaviour from a different direction.

```python
# Correct sorting - sort by score descending
candidates.sort(key=lambda x: x[0], reverse=True)

# pick randomly among top 26
top_n = min(6, len(candidates))
best_score, best_position, best_habitat = random.choice(candidates[:top_n])


self.previous_position = self.current_position
self.current_position = best_position

step_length = np.sqrt((self.current_position.x - self.previous_position.x)**2 +
                      (self.current_position.y - self.previous_position.y)**2)

turning_angle = None

self.habitat_preference[best_habitat] = self.habitat_preference.get(best_habitat, 0) + 1

self.behavior_log.append({
    "Step": self.current_step + 1,
    "Timestamp": None, #self.movement_data.iloc[self.current_step]['timestamp'],
    "Habitat Class": best_habitat,
    "Longitude": self.current_position.x,
    "Latitude": self.current_position.y,
    "Step Length": step_length,
    "Turning Angle": turning_angle
})

self.current_step += 1
```

This is the command for randomness which selects one preferred habitat randomly, sorting from most preferred habitats

Figure 9. Randomization command of SmartBirdAgent to resemble more like natural movement

In this random loop, we have used 6, as only 6 habitats showed a sensitivity score close to or more than 1. According to Manly's selection ratio, which we have used as our sensitivity score, a value <1 means avoidance of the habitat, where =1 resembles neutrality and >1 represents preference.

BirdHabitatModel class: BirdHabitatModel represents the model which mimics the actual data and analyses the habitat preference score or sensitivity score of birds to each habitat type.

```python
# BirdHabitatModel class
class BirdHabitatModel:
    def __init__(self, landcover_file, bird_movement_file):
        self.landcover = rasterio.open(landcover_file)
        self.landcover_data = self.landcover.read(1)
        # 🔄 Merge Ocean (0) into Salt Water (17) (Case specific decision)
        self.landcover_data[self.landcover_data == 0] = 17

        # Initialize agents
        self.bird_data = pd.read_csv(bird_movement_file)
        self.bird_agents = {}
        self.bird_colors = {}

        # Assign colors to birds
        color_list = ['red', 'pink', 'orange', 'purple']
        for idx, bird_id in enumerate(self.bird_data['tag.local.identifier'].unique()):
            movement_data = self.bird_data[self.bird_data['tag.local.identifier'] == bird_id]
            agent = BirdAgent(bird_id, movement_data)
            self.bird_agents[bird_id] = agent
            self.bird_colors[bird_id] = color_list[idx % len(color_list)]

    def get_habitat_at_position(self, position):
        """Get the habitat class from the raster at the given position."""
        lon, lat = position.x, position.y
        try:
            row, col = rowcol(self.landcover.transform, lon, lat)
            habitat_value = self.landcover_data[row, col]
            return habitat_value
        except IndexError:
            print(f"Position {position} is outside the raster bounds.")
            return None
```

Figure 10. BirdHabitatModel class definition and initiation

```python
    def get_habitat_at_position(self, position):
        """Get the habitat class from the raster at the given position."""
        lon, lat = position.x, position.y
        try:
            row, col = rowcol(self.landcover.transform, lon, lat)
            habitat_value = self.landcover_data[row, col]
            return habitat_value
        except IndexError:
            print(f"Position {position} is outside the raster bounds.")
            return None

    def step(self):
        """Advance the model by one time step."""
        for agent in self.bird_agents.values():
            agent.step(self, sensitivity=self.sensitivity if hasattr(self, 'sensitivity') else None)


    def get_results(self):
        """Retrieve habitat preference results and behavior logs."""
        results = []
        for bird_id, agent in self.bird_agents.items():
            for log in agent.behavior_log:
                results.append({
                    "Bird ID": bird_id,
                    **log  # Include all log fields
                })
        return results


    def calculate_sensitivity(self):
        """Calculate sensitivity for each landcover type."""
```

This command directs the model to record the (x,y) position data, retaking step after bouncing back if crossed raster boundary and the retrieving habitat preference results and behaviour_log[]

Figure 11.BirdHabitatModel: Model Elements

```python
                    **log  # Include all log fields
                })
        return results


    def calculate_sensitivity(self):
        """Calculate sensitivity for each landcover type."""
        # Count usage for each habitat type
        habitat_usage = {}
        for agent in self.bird_agents.values():
            for habitat, count in agent.habitat_preference.items():
                habitat_usage[habitat] = habitat_usage.get(habitat, 0) + count

        # Count availability of each habitat type in the landcover map
        unique, counts = np.unique(self.landcover_data, return_counts=True)
        habitat_availability = dict(zip(unique, counts))

        # Calculate sensitivity scores
        sensitivity = {}
        total_usage = sum(list(habitat_usage.values()))
        total_availability = self.landcover_data.size
        for habitat, usage_count in habitat_usage.items():
            usage_proportion = usage_count / total_usage if total_usage > 0 else 0
            availability_proportion = habitat_availability.get(habitat, 0) / total_availability if total_availability > 0 else 0
            sensitivity[habitat] = usage_proportion / availability_proportion if availability_proportion > 0 else 0

        print("Sensitivity Scores:", sensitivity)
        return sensitivity
```

Figure 12. BirdHabitatModel: Sensitivity score calculation

The simplified equation of Manly's habitat selection ratio can be presented as :

Manly's selection ration = usage proportion / availability proportion (Palminteri and Peres 2012).

This has been used to avoid the bias that can arise from the difference in availability of a habitat.

HypotheticalBirdHabitatModel class is where we modify the habitats resembling a hypothetical scenario. In our project, we have altered the most preferred habitat class 41-Other swamp vegetation. According to (Dellar et al. 2024) SSP5, by 2050, the Netherlands will have a 39% increase in construction in rural areas, where rural areas are referred to as grassland, croplands, pastures and swamps. Moreover, according to SSP1, there will be 33% increase of forested land. We used these percentages as references and tried to create three similar hypothetical scenarios where (1) Northern Lapwing's most preferred habitat will be replaced 33% by deciduous forest (Class11), (2) again in 2nd scenario, it will be replaced by 33% coniferous forest (Class12) and (3) in 3rd scenario, 39% of swamps will be replaced by construction in rural areas (Class26).

```python
# HypotheticalBirdHabitatModel
class HypotheticalBirdHabitatModel(BirdHabitatModel):
    def __init__(self, landcover_file, bird_movement_file, sensitivity):
        super().__init__(landcover_file, bird_movement_file)
        self.sensitivity = sensitivity

        # Modify the landcover map
        most_used_habitat = max(sensitivity, key=sensitivity.get)
        self.hypothetical_landcover_data = self.landcover_data.copy()

        class_indices = np.argwhere(self.hypothetical_landcover_data == most_used_habitat)
        num_to_convert = int(len(class_indices) * 0.33)  # Replace %

        selected_indices = random.sample(list(class_indices), num_to_convert)
        for row, col in selected_indices:
            self.hypothetical_landcover_data[row, col] = 12  # New habitat class

        print(f"Replaced 39% of class {most_used_habitat} with class 26.")

        # Replace bird agents with SmartBirdAgents
        self.bird_agents = {}
        self.bird_colors = {}

        color_list = ['red', 'pink', 'orange', 'purple']

        for idx, bird_id in enumerate(self.bird_data['tag.local.identifier'].unique()):
            movement_data = self.bird_data[self.bird_data['tag.local.identifier'] == bird_id]
            agent = SmartBirdAgent(bird_id, movement_data)  # <--- Use SmartBirdAgent here!
            self.bird_agents[bird_id] = agent
            self.bird_colors[bird_id] = color_list[idx % len(color_list)]
```

Habitat modification command

Figure 13. HypotheticalBirdHabitatModel: Modifying the landscape

5. Results

5.1 Changes in the frequency of habitat usage over different scenarios

For determining the significance of changes in habitat usage by the individuals in different scenarios, a chi-square test has been conducted. As some of the habitats had a very low frequency, the significance is presented with a simulated p-value instead of the actual p-value, which is calculated with a Monte Carlo randomisation value of 10,000.

The results show X^2 (N = 120,000) = 166227, p = 0.00009999 (simulated, based on 10,000 replicates), which depicts a large deviation from the expected under the null hypothesis. This represents that the usage of habitats in the hypothetical scenario is significantly different from the actual or current scenario.

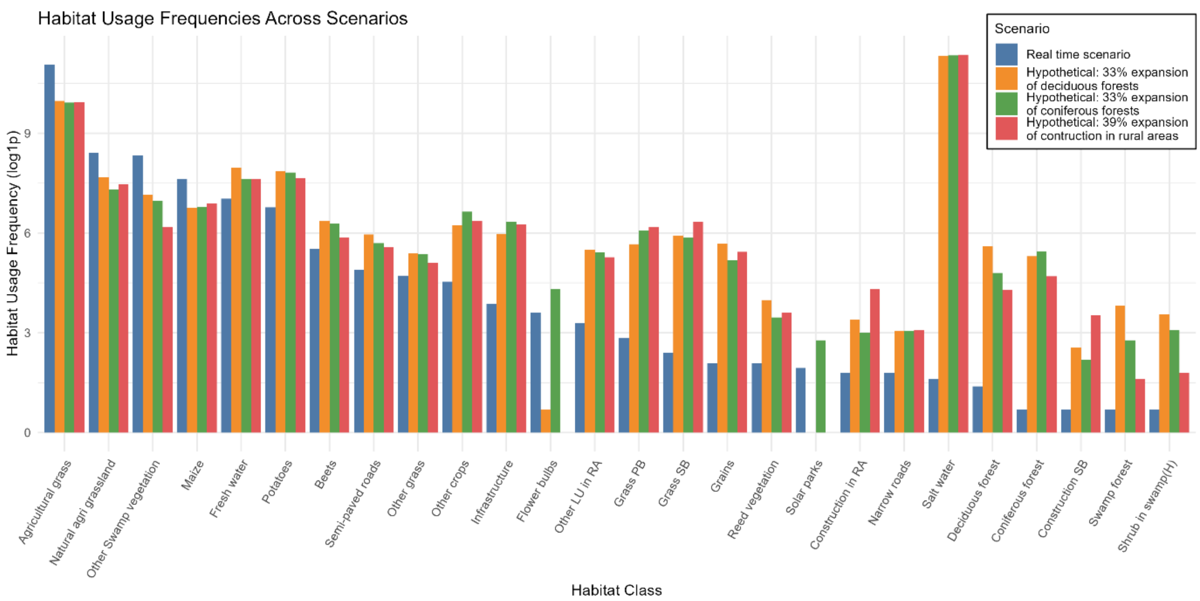

Figure 14. Habitat usage frequencies across different scenarios; The x-axis shows different habitats while the y-axis represents the log (frequency+1) values (The classes are ordered according to their sensitivity score in current model in a descending order from left to right)

The bar plot shows that the birds have been consistently emerging more in the habitats that they used to avoid in the current scenario. As the availability of their preferred habitat is being compromised, they have been appearing more in other habitats close to their preferred one, like different crop fields (eg. Potatoes, Maize), Shrubs in swamps, Swamp forests, Fresh water, Reed vegetations. Their emergence in the non-preferred habitats in the hypothetical scenario is also remarkably noticeable but it can happen due to availability rather than preference.

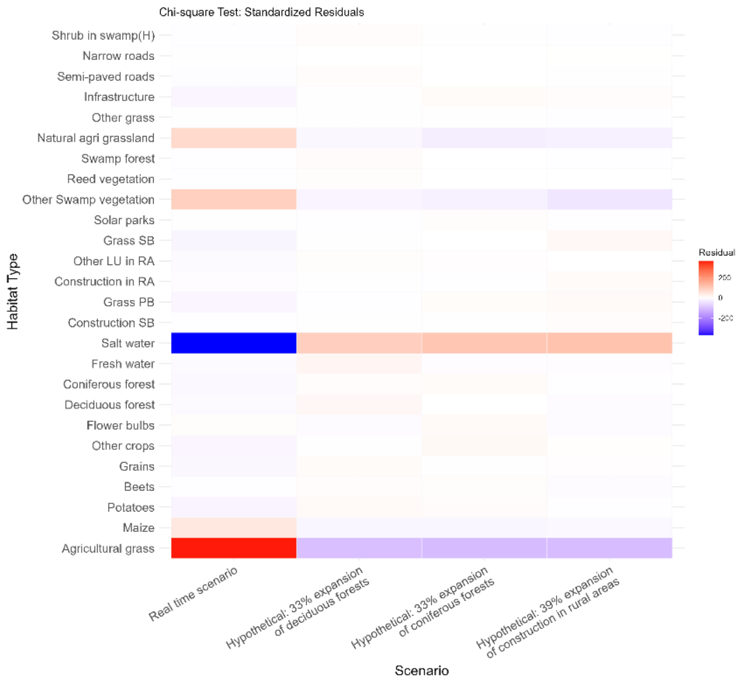

Figure 15. Chi-square residuals showing which habitats had affected the results most

The heatmap shows that the changes in appearance in habitat types like Agricultural grass, Salt water (Ocean), Natural agricultural grasslands and Other swamp vegetation probably affect the result more. Although Northern Lapwings prefer fresh water but the heatmap shows an increased emergence in the Ocean area. This raises a question on whether more appearances in a habitat means preference or it is due to the large availability of the habitat.

5.2 Does higher frequency mean high preference?

The sensitivity score from the hypothetical models has been compared with the sensitivity score of the current scenario model using the Wilcoxon rank sum test.

Table 1. Wilcoxon rank sum test result outputs comparing different hypothetical scenario scenarios to current scenario

Comparisons	W	P
Other swamp vegetation → Deciduous forest	316	0.004802
Other swamp vegetation → Coniferous forest	292	0.001113
Other swamp vegetation → Contruction in rural area	327	0.01548

The Wilcoxon rank sum test shows a significant difference in sensitivity scores between the current scenario and all three hypothetical scenarios.

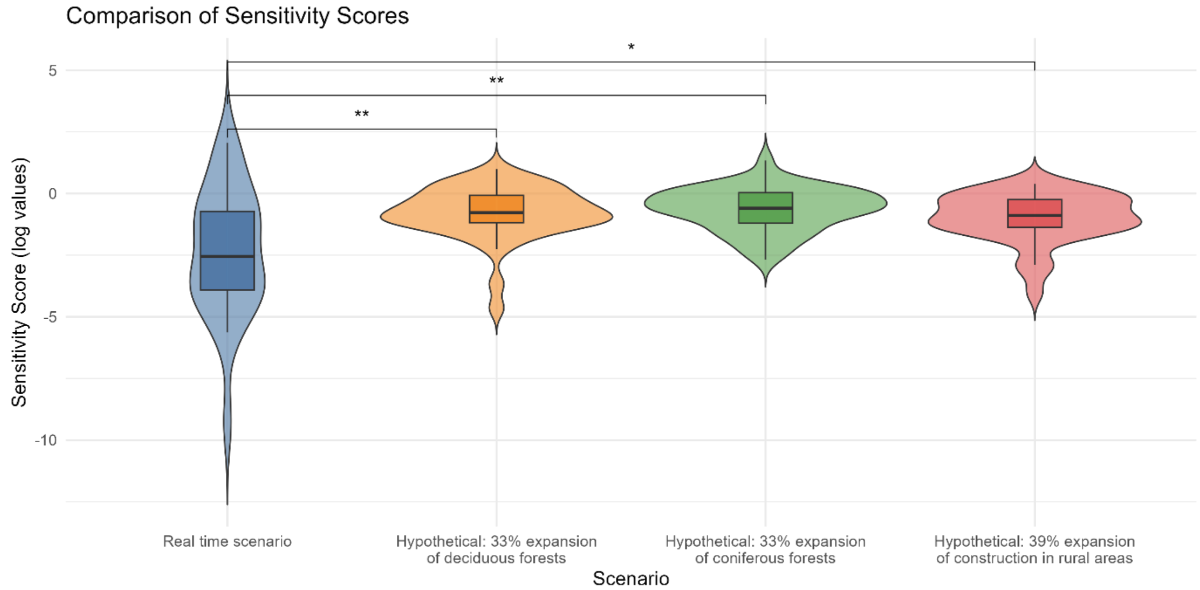

Figure 16. Comparison of sensitivity scores over different scenarios; The x-axis represents the different scenarios and the y-axis represents the natural logarithm of sensitivity scores

Figure 17. Heatmap showing the changes in sensitivity scores over different scenarios

From the heatmap, it's clear that although some unusual habitats had higher appearances, but their sensitivity scores are still <1, which proves that the increased number of frequencies have been recorded due to their availability rather than preference. On the other hand, the sensitivity scores of some similar habitats to the other swamp habitat seem to have an increased sensitivity score, for instance, shrubs in swamps and dune heath. Hence, more frequent visits does not represent a higher preference for habitat for Northern Lapwings.

Comparing the step movement of the birds, it seems that in current scenario, the bird movements are somewhat clustered in a similar region, also they seem to have less longitudinal deviation. But in all three hypothetical scenarios, the birds seem to move continuously in lower longitude, which is probably due to the reduction of preferred habitat. As their most preferred habitat has been altered, they have been moving towards the edges in search of suitable habitats.

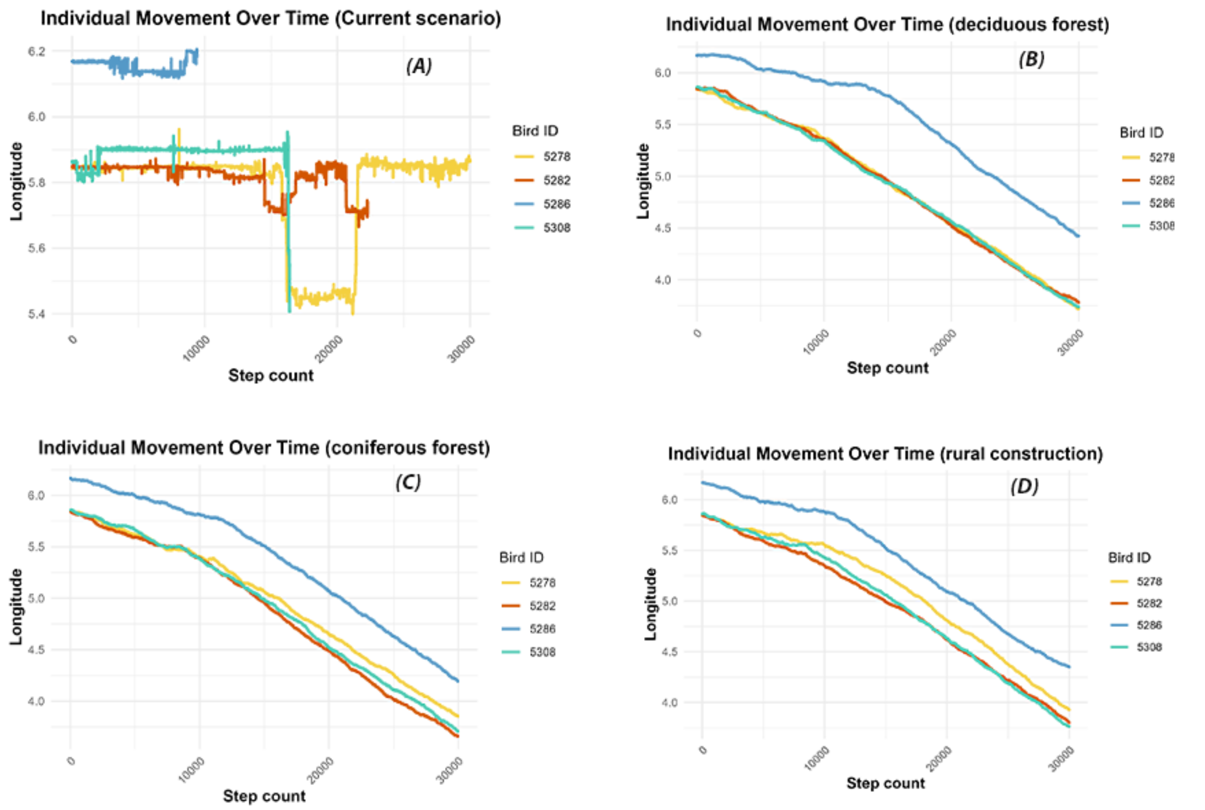

Figure 18. Movement steps of individual birds in different scenarios; (A) Current scenario, (B) 33% replacement with deciduous forest, (C)33% replacement with coniferous forest, (D) 39% replacement with rural construction

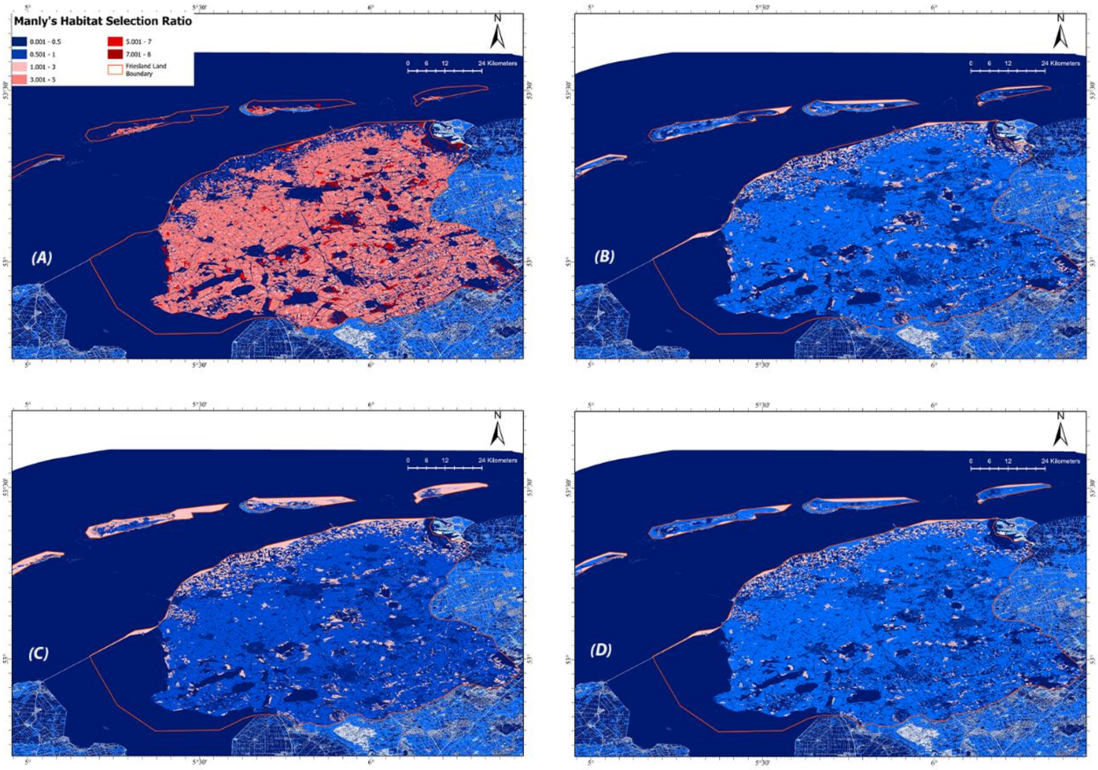

Figure 19. Maps showing sensitivity scores across the area; (A) Preferable habitat in current situation, (B)Preferable habitat locations in case of 33% deciduous forest expansion, (C) Preferable habitat locations in case of 33% coniferous forest expansion, (D) 39% expansion of rural construction

The map shows a clear shift in habitat regions towards the north-east coasts of the area, which explains their increased sightings in the saltwater area near the sea. Full animated movements are available with this link.

6. Discussion and Conclusion

In this study, we explored how Northern Lapwings (*V. vanellus*) respond to changes in their landscape using an agent-based model. By comparing the current habitat in Friesland with three hypothetical future scenarios, we were able to simulate bird behaviour and assess how their habitat preference shifts with certain land use and land cover changes.

Under the current landscape, Northern Lapwings showed a clear preference for swamp vegetation (habitat class 41), as indicated by the highest sensitivity score (Manly's preference ratio). This score helped us go beyond just counting how often habitats were used. In fact, it showed us which habitats were actually selected more than would be expected based on their availability (Palminteri and Peres 2012)

When parts of this preferred habitat were replaced with deciduous forest (class 11), coniferous forest (class 12), or rural construction (class 26), we observed noticeable shifts in behaviour. Although the birds appeared more often in open or water-related areas like saltwater (Ocean) (class 17), these habitats had very low sensitivity scores. This suggests that Northern Lapwings weren't choosing these areas rather, they were likely just passing through or had no suitable options nearby.

Our statistical tests supported this. The chi-square analysis confirmed significant differences in habitat usage across scenarios ($p < 0.0001$), and Wilcoxon rank-sum tests showed a significant drop in habitat preference in all three modified landscapes compared to the current one.

We also noticed changes in how the birds moved across the landscape. In the hypothetical scenarios, especially the one with expanded rural construction, birds were more scattered, often moving toward less suitable zones like coastal water. This likely reflects stress or uncertainty due to a lack of suitable habitats, which has also been seen as a reason of the population reduction of many other waders all over Europe (Eglington et al. 2010).

Importantly, this study highlights that just because a habitat appears frequently in tracking data does not mean it is preferred. Sensitivity scores helped to avoid this availability bias.

In conclusion, there's a high possibility that even small changes to preferred habitats can have a noticeable impact on Northern Lapwing behaviour. While they might adapt to some extent, their movement patterns and habitat use suggest that they are being affected by these shifts. If we want to support species like the Northern Lapwing, especially in breeding regions like Friesland, it's essential to integrate species behaviour into land-use planning. Future work could build on this model by incorporating complex land use change combinations, more behavioural variables, breeding or nesting suitability index, or even long-term environmental and movement data for a fuller picture of habitat suitability.

References

Baptist M, van Hattum T, Reinhard S, et al (2019) A nature-based future for the Netherlands in 2120. Wageningen University & Research 20

Dellar M, Geerling G, Kasper K, et al (2024) Future land use maps for the Netherlands for the Dutch One Health SSPs. Sci Data 1–16. https://doi.org/10.1038/s41597-024-04059-5

Eglington SM, Bolton M, Smart MA, et al (2010) Managing water levels on wet grasslands to improve foraging conditions for breeding northern lapwing Vanellus vanellus. Journal of Applied Ecology 47:451–458. https://doi.org/10.1111/j.1365-2664.2010.01783.x

Hazeu G, Schuiling R, Thomas D, et al (2023) Landelijk Grondgebruiksbestand Nederland 2021 (LGN2021) : achtergronden, methodiek en validatie

HELCOM Red List Bird Expert Group (2013) Vanellus vanellus Vanellus vanellus Distribution and status in the Baltic Sea region

international B (2016) Vanellus vanellus: BirdLife International. IUCN Red List of Threatened Species 8235

Kranstauber B, Cameron A, Weinzerl R, et al (2011) The Movebank data model for animal tracking. Environmental Modelling and Software 26:834–835. https://doi.org/10.1016/j.envsoft.2010.12.005

Musters CJM, Ter Keurs WJ, De Snoo GR (2010) Timing of the breeding season of Black-tailed Godwit Limosa limosa and Northern Lapwing Vanellus vanellus in the Netherlands. Ardea 98:195–202. https://doi.org/10.5253/078.098.0209

Palminteri S, Peres CA (2012) Habitat Selection and Use of Space by Bald-Faced Sakis (Pithecia irrorata) in Southwestern Amazonia: Lessons from a Multiyear, Multigroup Study. Int J Primatol 33:401–417. https://doi.org/10.1007/s10764-011-9573-0