Integrated field- & agent-based modelling: understanding the dynamics of the common barbel in the Meuse river

MSc Thesis

Els Kuipers
6402240

First supervisor: prof. dr. Derek Karssenberg
Second supervisor: dr. Oliver Schmitz

Universiteit Utrecht

Earth, Surface and Water MSc
Faculty of Geosciences
Utrecht University
2023 - 2024

1 Abstract

Simulation models explicitly represent the underlying processes of systems. A standardised modelling framework that incorporates natural behaviour can improve accessibility for domain experts by being semantically intuitive. This thesis explores space conceptualisation in such frameworks to represent spawning habitat availability for swimming barbels. Space is often conceptualised using field-based approaches, where attributes have values throughout the space-time domain, and agent-based approaches, where entities are spatially bound and mobile. Integrating these paradigms allows population-level properties to emerge from individual responses to the environment. The software framework Campo integrates fields and agents (de Bakker et al., 2017). In the Common Meuse, the common barbel experiences a hydropeaking flow regime, leading to fluctuating habitat suitability and connectivity. This thesis examines barbel movement and spawning success in response to environmental changes, aiming to extend Campo's functionality to better represent and understand the barbel's habitat amidst hydropeaking conditions. The objective is to determine how flow regime and barbel behaviour affect spawning success and the optimal representation of these phenomena in Campo. Various movement modes were modelled, and the framework's ability to represent agents' sensing of the field was enhanced. These several operations have increased the accessibility and representation capabilities of ecosystems in Campo. Overall, heavily fluctuating as well as low discharges were detrimental to the barbel's spawning success. Nonetheless, agent integration does require generalisation of behaviour. Different movement behaviours resulted in significantly varied outcomes, with 50% of population reaching spawning sites between 1 and 33 days. This indicates a need for a more thorough understanding of barbel behaviour and highlighting the importance of integrating the agent-based perspective when assessing habitat suitability.

Contents

List of Figures

List of Tables

2 Introduction

In our knowledge driven society, simulation models fulfil the role of mediator between theory and data (Morrison & Morgan, 1999). Simulation models differ from statistical or other conventional models as these aim at explicit representation of the underlying processes (Tang & Bennett, 2010). A good model represents reality closely in ways that are theoretically based and can be observed in nature through data. Geomorphological models such as Delft3D (Deltares, 2014) or hydrological models as PCR-GLOBWB (Sutanudjaja et al., 2018) rely on theory on mass transport of sediment or fluid mechanics and are validated using data in order to predict larger scale phenomena. In a different approach, empirical relations deduced in experiments are validated by testing their performance in different settings in a model (Troitzsch, 2004; Le Quéré et al., 2020). Nature is too complex to fully describe or grasp, and therefore we are forced to simplify. However, close analogies to nature are often achieved ineffectively, through addition of detail in models. This requires a lot of data, and problems such as equifinality arise (Favis-Mortlock, 2013). Possibly, the observed outcome can not be attributed to the modelled empirical relations because different pathways can lead to similar outcomes (Favis-Mortlock, 2013; von Elverfeldt et al., 2016). The assumption that a perfect analogy is composed of infinite detail, overlooks non-traditional cause and effect relations, such as self-organisation, which can result in equally complex patterns (von Elverfeldt et al., 2016; Favis-Mortlock, 2013). Furthermore, when the aim of modelling is to understand key mechanisms of a system, significant reduction in complexity of the model's structure, is required to make the model at least somewhat easier to understand than reality (Grimm & Railsback, 2005). Mending gaps between the model's outcome and real-world observations purely by adding empirical relations increases the complexity as well as reduces the generalisation capacity of the model (Favis-Mortlock, 2013).

Turning to these measures is understandable, as natures complex underlying processes are not easily expressed with current programming languages and frameworks (Athanasiadis & Villa, 2013). Many existing tools and languages introduce pleonasms, redundancies and boundaries, not accommodating expression of complexity through feedback loops or internal variation, mainly keeping domain experts busy with technical difficulties such as file-handling and method-calls (Athanasiadis & Villa, 2013). In addition, current languages and frameworks do not easily allow for integration of separate elements of the system in other models to promote reuse (Athanasiadis & Villa, 2013). In the scope of this, Athanasiadis and Villa (2013) coined the term Domain-Specific Language in a quest for simpler but structured software frameworks to prevent the need of excessive detail in models while still representing the characterising properties of a system.

A standard modelling framework that conceptualises the way nature behaves incorporated in the language itself is a solution to increase a model's ability to represent the complex semantics of natural systems as well as increase the accessibility of models for domain experts. Such abstract modelling environments complement the domain experts' knowledge, who are not necessarily knowledgeable in lower-level programming languages and software development. Software developers are knowledgeable in the languages needed to express the models, but the full specification of the requirements of the model depends on the model's performance as well as the domain of the research and therefore needs to be decided on during the development process (Karssenberg, 2002). Continuous adjustment of the model's requirements is best executed by the domain expert, who is best at describing reality. Therefore, modelling environments in which the domain expert is able to express their ideas through operations that meet with the domain-experts' conceptual way of thinking are a great way to mediate the gap between the domain experts' knowledge and the software developers'. This type of modelling is referred to as semantic modelling (Athanasiadis & Villa, 2013). Complexity of model assumptions is represented through operations as intrinsic features of the language and therefore should not require user attention through method calls (Athanasiadis & Villa, 2013).

In this thesis, we consider the space conceptualisation of spatio-temporal simulation models for the purpose of representing habitat dependency of and availability for organisms. In many semantic domain specific frameworks, space is conceptualised around one of two paradigms. Field-based approaches assume attributes to have a value everywhere in the space-time domain. Agent-based approaches define objects that are bound in space. These agents can be mobile, interact with other agents and change shape, as well as be addressed at differing timesteps. Though many interacting phenomena are best matched by differing space conceptualisations, the dichotomy between the two approaches is reinforced through the development of analyses that work on either of the space conceptualisations. For example, window-operations are field-based by nature, and buffer operations are agent-based (Cova & Goodchild, 2002). The decision to design a model along one of either of the paradigms influences the outcome of the model, marking the importance of dual-conceptualisations.

The modelling of population dynamics in ecosystems often requires organisms to be represented as mobile and interacting agents and their habitat as continuous fields. Agent-based modelling, or individual-based modelling, has been recognised as a promising approach to model ecosystems and let population-level traits emerge (Grimm & Railsback, 2005; McLane et al., 2011). However, current ecological agent-based models lack the representation of the agents' adaptability to a changing environment as well as the environment's response to the agent because they are constructed mostly as agent-based and do not incorporate the environment as a spatially explicit and dynamic field (McLane et al., 2011). Facilitating integration of both conceptualisations allows for accurate representation of real-world processes.

Previous attempts to integrate both fields and agents illustrate the need and potential for the method, but these modelling frameworks are often not widely applicable or have limited functionality (Hamdani et al., 2021; de Bakker et al., 2017). A common representation of objects in the field paradigm is through a density field, but this is not a precise representation, and assumes no interaction between agents, as well as of agents to be uniform. Attempts to integrate the two conceptualisations in software are manifold (Hamdani et al., 2021; Liu et al., 2008; Cova & Goodchild, 2002; Goodchild et al., 2007; Kjenstad, 2006; de Bakker et al., 2017). However, many of these frameworks are not semantically intuitive for domain experts (Hamdani et al., 2021) or not fully developed to represent a wide range of system mechanisms through queries and data manipulation capabilities (Goodchild et al., 2007). Furthermore, they are often not able to represent internal structures (Cova & Goodchild, 2002), or the spatio-temporal evolution of agents or fields (Liu et al., 2008).

The software framework Campo (de Bakker et al., 2017) integrates fields and agents through the implementation of the LUE physical data model and allows for intuitive manipulation of extensive raster and vector data through map algebra. However, the framework has a limited set of operations and its application has not been widely illustrated yet. The current incorporation of the temporal domain in the conceptual data model was hypothesised to provoke challenges when modelling processes over time (de Bakker et al., 2017). Furthermore, movement of agents has only recently been implemented in the framework and this functionality not been exemplified yet.

I test the performance of Campo by representing barbel population dynamics in the Common Meuse. The functionality of Campo as an integrated field-agent based framework still needs to be evaluated in its performance in representing larger field data that can drive agent behaviour such as movement in ecological settings. Results of other fish population agent-based simulations that integrate fields shows close similarities to observed population patterns, which illustrates the potential for an agent-based approach when simulating fish populations(Railsback et al., 2002; Beland et al., 1982). However, the fields herein are not large-scale or spatially detailed. Furthermore, since the data format of fields and agents in agent-based modelling are not uniform (de Jong & Karssenberg, 2019), the two space conceptualisations need to be addressed distinctly. Operations between fields and agents is cumbersome, and the method of the study is as a result not easily reproducible. I therefore model the evolution of fish and the fields over time using the Campo framework, to showcase the applicability of the framework as well as to incite progress to the software. The structure of this thesis is therefore twofold: while illustrating the way Campo is capable of representing habitat dependency of organisms, I also provide more insight in the dynamics of the common barbel population in the Common Meuse under conditions of a hydropeaking regime during spawning time.

Though the common barbel (*Barbus barbus*) is a species of least concern according to the IUCN Red list (IUCN, 2023), anthropogenic disturbances have caused local threats to the occurrence, such as in the Common Meuse (Britton & Pegg, 2011). The common barbel is sensitive to flow regime and therefore indicative of good habitat conditions for other rheophilic species in the Common Meuse (Liefveld & Jesse, 2006). The species' habitat preference has been used to establish minimum discharges in the Common Meuse using habitat suitability mapping (Liefveld & Jesse, 2006). However, the connectivity of suitable habitat is essential for population sustenance as it determines whether the habitat is reachable. I propose an integrated field-agent based model to study the behaviour, and survival of the common barbel in the Common Meuse under the current flow regime. Specifically, spawning possibilities are of interest for the barbel's population sustenance. The literature is inconclusive about the barbel's sensing abilities and assertiveness, particularly in group settings. Whether barbels show assertiveness by moving toward spawning grounds or randomly is unknown (Gutmann Roberts et al., 2019; Britton & Pegg, 2011). This study may shed light on the impact different types of behaviour have on spawning success.

Morphology and flow regime in the Common Meuse has been disturbed as a result of anthropogenic activity. The barbel shows sensitivity to the large differences in water levels and flow velocities in the Common Meuse as a result of upstream dams, a process called hydropeaking (van Oorschot et al., 2022; Liefveld et al., 2018). Both very high and very low flow velocities and water level occur on a diurnal scale. On smaller spatial and temporal scales, the thresholds implemented in the Common Meuse furthermore introduce spatial water depth and flow velocity fluctuations (Liefveld & Jesse, 2006). Studies have spatially addressed the mechanical effect the weirs in the Common Meuse have on some fish' species ability to migrate through the river (Deerenberg et al., 2012). However, the effects of the dams on the flow regime on the population's migration patterns is being neglected, despite evidence flow regime largely influences population dynamics (Bjørnås et al., 2021). To what extent the common barbel's sensitivity to flow regime influences population dynamics in terms of migration, death, and birth is valuable knowledge to formulate the restoration plans (McLane et al., 2011). A model to simulate and therefore validate the effects of hypothesised sensitivity to individual or group behaviour can give meaningful insight in the main drivers of the system and also predict future scenarios. Though availability of suitable habitat has been modelled and examined before (Liefveld & Jesse, 2006), the connectivity of microhabitats plays an equally important role in accommodating population sustenance (Stoffers, Buijse, Geerling, et al., 2022). Furthermore, spatial explicit representation of the ecosystem taking into account animal movement is required to gain full insight in the main shortcomings of the system in order to establish restoration plans and flow regime requirements. Specifically looking at spawning availability and the reachability of the spawning grounds, the barbel's ability to reach their preferred habitat during spawning time is assessed in this study.

The main objective of this thesis is to extend and exemplify Campo's functionality in representing and understanding the common barbel's habitat availability amidst a hydropeaking flow regime, as defined by field parameters as water depth and flow velocity, in the Common Meuse. How does flow regime affect the barbel's population dynamics when modelling an agent's response to the environment as a field, and how is this representation best achieved in Campo? Subquestions that will be addressed are:

- What does the modelling framework of Campo require in order to represent the sensitivity of the individual barbel to their environment and stimulate the framework's field-agent integration?
- How does the hydropeaking flow regime affect spawning habitat availability for the barbel?
- What effect does barbel movement behaviour have on spawning success?
- How does the model perform compared to conventional habitat suitability models?

Through literature research, I analyse characterising responses of individual barbel to their environment, specifically in the Common Meuse. Secondly, the requirements of the conceptual model to represent these characterising dynamics are formulated and implemented in Campo. I then analyse how the model is able to reproduce patterns of population responses as observed in similar models, as well as theoretic population pattern responses that would be expected. This is to provide evidence of the model's validity for predicting population-level phenomena. As a result, a model able to represent population emergence including adaptive response to the environment will evolve and showcase how Campo can be used to simulate spatially explicit habitat dependency of ecosystems. Lastly, the limiting factors for preservation of the barbel population is analysed.

I hypothesise that for Campo to accurately represent habitat dependency and availability, the framework's accommodation of field-agent interactions must be extended to include the representation of environmental sensing by agents. By incorporating spatial awareness into the barbel's movement decision-making, we may observe patterns similar to those in nature. The habitat fragmentation of the Common Meuse ecosystem could be worsened by troughs in the hydropeaking signal, suggesting that hydropeaking negatively impacts habitat availability spatially. However, as water levels and flow velocities increase over time during the peak of the signal, connectivity may be restored, facilitating movement to spawning grounds. In conditions of intense fluctuations in flow regime, barbel agents may face constantly changing environments, necessitating extensive movement to remain in habitable areas. I expect that conventional habitat suitability models can serve as a proxy for individual availability of preferred habitat, especially in a non-hydropeaking flow regime. However, the assessment of sufficient habitat for population survival depends on factors such as movement capabilities and system connectivity. This thesis will shed light on the relation between habitat suitability and individual habitat availability under these varying scenarios.

3 Background

3.1 Conceptualisation of space in models

Modelling of the scene that is real and exists on the ground requires generalisation and parameterisation of the essential qualities of the scene. The scene is characterised by the essential elements themselves and their functional dynamics (Goodchild et al., 2007). Agents are spatially bound entities which may be mobile. Fields are continuous, meaning within the definition of a system they exist over the entire two-dimensional plane. The conceptualisation of agents and fields proves valuable in modelling, aligning with our cognitive understanding of environmental structures. Recognising the significance of semantics in modelling languages that closely mirror our perception of the environment, it becomes evident that such representations are more adept at capturing complex environmental dynamics (Athanasiadis & Villa, 2013; Carré & Hamdani, 2021).

Properties of systems that are key to our understanding, may emerge from interactions between elements that are currently represented through either fields or agents. Therefore, suitable forms of representation and manipulation that extend the two conceptualisations of agents or fields are desired, because modelling such mechanisms along either of the two paradigms may yield incomplete or biased results (Hamdani et al., 2021; Goodchild et al., 2007). By enabling intuitive queries that ensure both topological accuracy and accessibility, modelling environments can enhance their semantics and applicability, ultimately fostering a deeper understanding of complex environmental systems (Goodchild et al., 2007). Furthermore, either conceptualisation mainly perceived as one can be expressed as the other (Goodchild et al., 2007). A field may be considered an agent considering larger system boundaries or an alternative definition of the characterising elements of the field, and vice versa. This furthermore stresses the incompleteness of modelling languages that only accommodate one of the two conceptualisations.

3.1.1 Agent-based modelling

Agent-based modelling allows researchers to capture individual-level behaviours and interactions as a driving factor for larger scale impacts on the environment. It simulates interactions among individual components, referred to as individuals, agents, objects or discrete particles depending on the field of research. Agents can each have their own attributes, behaviours and rules to govern their actions and interactions with other agents and the environment. The collective behaviour of agents can lead to the emergence of patterns and trends that are not explicitly programmed but arise from agents' interactions (Railsback, 2001a; Macal & North, 2005). Modelling distinct individual properties is useful in systems where heterogeneity plays a large role, such as in ecosystems (Grimm & Railsback, 2005). Furthermore, agent-based modelling is a useful tool to model systems where human decision-making is decentralised, for instance in water systems impacted by human-behaviour. Dam operators as well as water consumers are subject to a diversity of drivers that can differ on both spatial and temporal scale to release or consume water, collectively affecting water distributions greatly (Bolton & Berglund, 2023).

Besides agent representation, generic agent-based software frameworks such as NetLogo (Wilensky, 1999), GAMA (Taillandier et al., 2014) and GeoMason (Sullivan et al., 2010) also do accommodate representation of spatially unbound field entities (de Bakker et al., 2017). Domain-specific functionality is often supplied with these frameworks, such as machine learning algorithms and spatial statistics (Tang & Bennett, 2010). However, large-scale or extensively detailed spatially explicit simulations are dealt with inefficiently (de Jong & Karssenberg, 2019). The integration of external geospatial data is troublesome (Crooks & Castle, 2011). Spatial analysis in these frameworks do not accommodate interactive processes between the two paradigms, not incorporating interactive elements of fields and agents. Furthermore, field-based queries such as window operations are not easily implemented. Developing complex spatial models in NetLogo requires significant effort to customise in accordance to specific spatial environments, as the toolkit is written in its own programming language (Crooks & Castle, 2011).

3.1.2 Integrated field-agent modelling

Hamdani et al. (2021) recognised three approaches to achieve integrated field-agent modelling. Firstly, a method in which a mediator in the form of object-field associates the two space conceptualisations. Every point in the spatial domain is in addition to being mapped to a value also mapped to a discrete object, being a field or an agent (Cova & Goodchild, 2002). Because the agents and fields are in the first place conceptualised independently, dual conceptualisation is not achieved. This conceptual pathway between the two approaches is difficult to associate in the temporal domain (Hamdani et al., 2021). Dynamic modelling with the object-field requires an additional pathway, as in Hamdani et al. (2019).

Secondly, Goodchild et al. (2007) coined the terms geo-atom, from which the concepts geo-field, geo-object and field-object evolve. A geo-atom has a certain spatio-temporal domain and attributes. Geo-fields and geo-objects are aggregations of geo-atoms, and field-objects are geo-objects with internal heterogeneity. Coupling of objects and fields is achieved by geo-fields that can be associated to multiple classes. However, data manipulation is hard due to the rigid structure of the geo-atoms, which makes it hard to let fields and agents evolve. Geo-atoms can have polygonised and alternating shapes to allow for differently scaled objects, but this approach does not enable distinctive shapes for fields and agents, and furthermore objects are not modelled as evolving field-objects with internal variation (de Bakker et al., 2017; Hamdani et al., 2021).

A third category extents this approach and allows for moving objects with internal structures defined as a field (de Bakker et al., 2017; Kjenstad, 2006). Kjenstad (2006) develops fields as a set of functions from a parameter space domain to a value space, and therefore allows for agents to be fields and fields to be agents. However, it is unclear how dynamic modelling is achieved (de Bakker et al., 2017).

Many physical data models do not support easy storage of large sets of data that are able to represent both agents and fields, because they require storing agents and fields in separate data frameworks (de Jong & Karssenberg, 2019). Agents are often represented by classes, each of which represents one type of agent and its properties. Fields are represented in the format of multidimensional arrays that discretise the space. Shared attributes for agents and fields, such as shared variables or time, need to be stored several times in separate data frameworks, which is inefficient and error-prone or not easily possible to conduct operations between the separate frameworks (de Jong & Karssenberg, 2019).

3.2 Outline of Campo

In the software Campo de Bakker et al. (2017), fields are conceptualised as continuous agents. The data of both fields and agents is stored similar and therefore accommodates both internally varying agents and interactive queries between the two views without additional pathways, as in Figure 3.1. The domain of the properties of the items in a certain phenomenon is stored explicitly, accommodating tracking of the history of a state (de Bakker et al., 2017). Further operations to promote field and agents interactions are desired. Only recently, mobility of the agents is accommodated by the framework. The resulting increased complexity of an agent's behaviour requires additional operations to connect the field and agent-approach when agents move.

In Campo (de Bakker et al., 2017), field-like and agent-like entities are both represented through one conceptual framework in which fields are defined as agents with possible internal variation, see Figure 3.1.

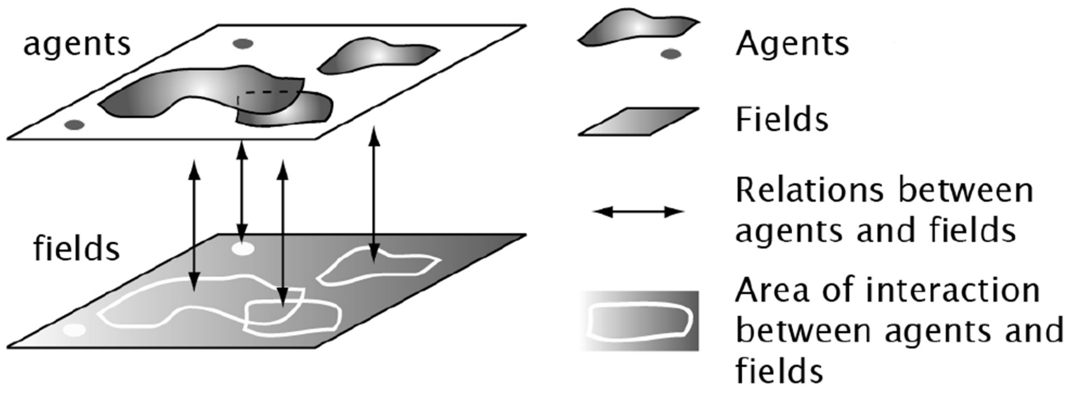
agents
fields
Agents
Fields
Relations between agents and fields
Area of interaction between agents and fields

Figure 3.1: Integrated agent-based and field-based model, consisting of agents and fields (de Bakker et al., 2017)

This conceptualisation is implemented in LUE which handles and describes the implementation of data in the model, enabling intuitive manipulation of the data within these entities. As a result, operations that work along both space conceptualisations are possible. Processes are represented in Campo through map algebra operations, as in the PCRaster framework (Karssenberg et al., 2010).

3.2.1 Conceptual data model

In Campo, entities containing data are conceptualised based on six concepts that are able to represent both fields and agents, structured as in in figure 3.2.

- The *phenomenon* is the top level concept in the data model. The *phenomenon* is a collection of the different items that can be categorised and analysed together. A *phenomenon* could be a certain species of fish, or rivers.
- An *item* is an individual contained in a phenomenon. This can be a field or an agent, for instance one fish or a family of fishes. Each fish is characterized by a similar set of properties.
- A *property* is a particular attribute of a phenomenon (e.g. the age or weight of the fish)
- A *domain* is the area and time period for which a property exists, for each item.
- The *value* represents the values for each *item* in a *phenomenon* for a particular *property*. Each property has one *value,* which may differ among the items (e.g., each fish will have one age value).

Because each item can have its own domain, this allows for there to be distinctive timesteps as well as interactions among the agents and fields. Though not yet illustrated, the software also enables movement of agents through space.

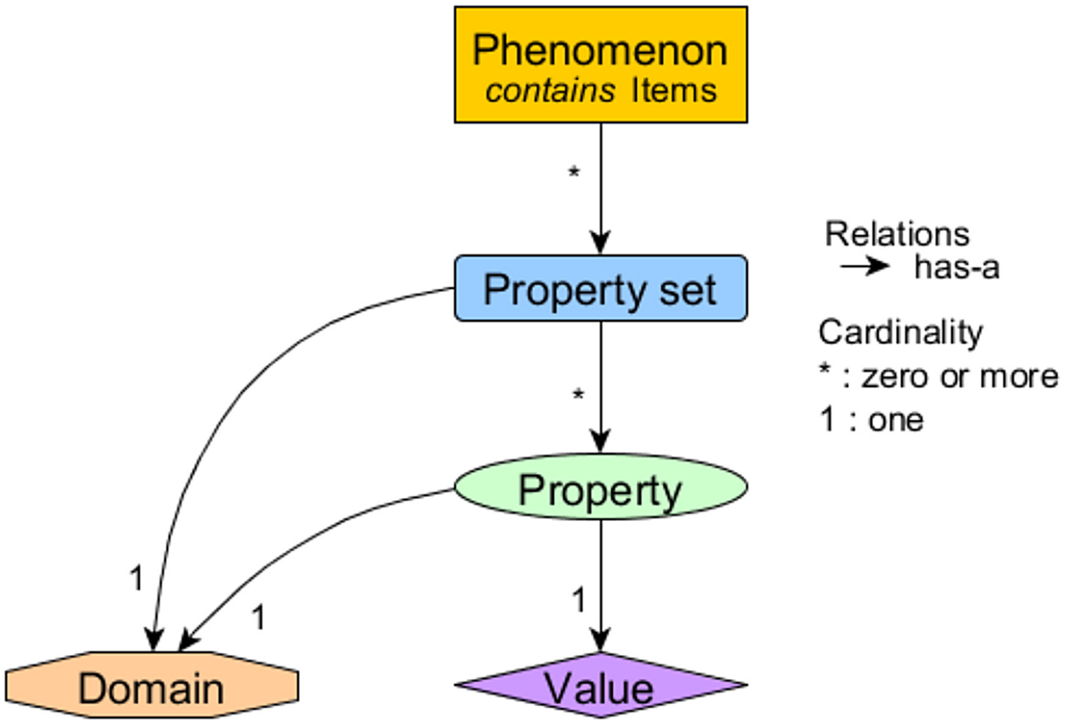

Figure 3.2: Conceptual model for spatio-temporal data (de Bakker et al., 2017)

3.2.2 Physical data model

A single physical data model that integrates data able to represent both agents and fields is required to be able to represent properties in the same conceptual model without having to be familiar with all different types of multiple data models. A physical data model determines the way data is handled and ordered. Such structuring potentially allows or limits good representation of certain real world phenomena. In case of fields and agents, these representations are often stored in separate data models due to their different nature. Fields are stored in a raster format whereas agents are stored as vectors, both formats supported by the Geospatial Data Abstraction Library (GDAL/OGR contributors, 2020). Attributes of agents that are not spatially explicit, require storage in yet another format. Adjustments to all formats need to be made to enable data storage over time. To relate the datasets to one another is error-prone, inconvenient, not intuitive and hard to reuse or maintain (de Jong & Karssenberg, 2019). In Campo, the physical data model of LUE is based on the HDF5 physical data model (The HDF Group, 1997) to store variable states for both fields and agents(de Jong & Karssenberg, 2019). This unified physical data model allows different abstractions for properties and different dimensions for different property sets. These are all stored in the same data model, therefore allowing operations among the different properties. Information about objects is stored in arrays, with one array being one type of information relevant for an agent or a field. These arrays are identified according to the shape of their data, with dynamical data and spatially differing data being distinguished. Furthermore, the data is grouped hierarchically as outlined in the conceptual model for Campo, as the HDF5 physical model enables (de Jong & Karssenberg, 2019) .

3.2.3 Modelling language

Processes in Campo are represented through map algebra (Tomlin, 1994), which is a general set of capabilities, conventions, and techniques to manipulate spatial data. Herein, single-factor map layers are treated as simple variables that can be transformed into new variables. These new variables are spatially referenced and therefore form new maps. The map algebra language is composed of primitive operations invoked through expressions familiar in algebra. An argument for the use of map algebra is its simple syntax while maintaining its ability to create a more complex model due to its ability to apply multiple algebraic expressions (Jeremy Mennis & Tomlin, 2005). We distinguish local, focal and zonal operations. Focal functions compute new values for a location as a function of the values, distances and/or directions of neighbouring locations on an existing layer. Zonal functions work similarly, as they compute each location's new value as a function of the values from one existing layer that are associated to that location's zone on another layer. The definition of zone can differ per location and therefore differs from the definition of neighbourhood (Jeremy Mennis & Tomlin, 2005).

3.3 Ecological agent-based modelling

Agent-based modelling, or individual-based modelling, has been recognised as a promising approach to model ecosystems and let population-level traits emerge (Grimm & Railsback, 2005; McLane et al., 2011). Individual agents can be addressed as a function of different timesteps. Characterising properties of an ecosystem emerge from interactions among agents, as well as through agents' interactions with the environment (Grimm & Railsback, 2005; McLane et al., 2011). In an organisms' lifespan, population dynamics through birth, migration and death, emerge from the diverse adaptive behaviour of autonomous individuals, similar to how on the timescale of evolution, new genotypes emerge as a result of genetic diversity and natural selection (Grimm & Railsback, 2005). The explicit consideration of individuals as separate agents allows for distinct adaptability and multifaceted responses to events, with decisions based on previous events or characteristics of the agent, representing the organisms' intelligence (McLane et al., 2011). Physiological states of animals are associated to the cognitive processes' perception, memory, learning, and decision-making (Tang & Bennett, 2010). The recognition of autonomous individual entities is what comprises dynamic populations. This very diversity and dynamic nature is what makes populations robust and capable of handling perturbations in environmental conditions (McLane et al., 2011). When assessing a population's resilience in restoration scenario planning, allowing for complex adaptive behaviour is therefore essential (McLane et al., 2011). Key adaptive mechanisms depend on the model's temporal and spatial scale (Railsback, 2001a). Individual behaviour-based adaptation can be modelled as decision trees based on internal states, for instance size or age of an individual.

3.3.1 Field integration

Field-based approaches assume continuous values discretised in spatial cells and uniform timesteps for each cell, which is representative of many relevant abiotic attributes of habitats, such as temperature gradients or slopes, as well as some biotic attributes such as biomass. However, current ecological agent-based models are constructed as agent-based initially and do not incorporate the environment as a dynamic spatially explicit field. Therefore, these models lack the representation of the agents' adaptability to a changing environment as well as the environment's response to the agent (McLane et al., 2011), despite knowing that abiotic and biotic attributes of the environment create constraints, threats, and opportunities for animal behaviour. External abiotic or biotic factors that compose an animal's habitat are categorised in two general classes: resources, a pulling factor, and risk, a pushing factor (Tang & Bennett, 2010).

3.3.2 Animal movement modelling

While traditional population studies focused on changes in population numbers over time, recent research has made it clear that certain population dynamics rely not just on population density, but also on individual movement patterns(Patterson et al., 2008). Movement impacts phenomena such as disease spread, invasive species, and the design of protected areas (Patterson et al., 2008). When assessing risks or resources, animal movement representation is essential because the habitat availability depends upon its reachability (Tang & Bennett, 2010). Movement is a spatial as well as temporal process, interplaying between behavioural patterns and the environment (Tang & Bennett, 2010; Nathan et al., 2022). Environments directly or indirectly drive movement, and as a consequence the environment of the animal is altered when its location is updated.

Animal movement patterns are studied and characterised at individual, population, and location level (Tang & Bennett, 2010). Hypothesized drivers for animal movement include foraging, foraging, avoiding predators, or changing habitat conditions (Swingland et al., 1983).

Different species use different movement modes, and within species, individual or population movement modes may alter based on the spatio-temporal scale as well as an individual's internal state (Tang & Bennett, 2010). Animals' ability to sense their environment influences how they move. If their senses are limited to nearby surroundings, their decisions about where to go next are based on low resolution immediate factors and their own capabilities. Conversely, if they can detect things from a distance, their movements may be driven by longer-term goals. Consider an animal navigating its environment. If it can only sense things nearby, it relies on what is immediately around it to make decisions about its next move. However, if its senses extend to distant surroundings, it might have a specific destination in mind, such as a nesting site or a feeding area. Importantly, this destination may change over time as the animal's environment changes. Literature is missing and not conclusive about the sensing abilities of the barbel and whether the fish demonstrates assertive personality traits, especially in group settings (Conradt, 2012).

When representing animal movement through modelling, the spatio-temporal scale is fundamental in considering an animal's movement path. The Nyquist-Shannon sampling theorem (Shannon, 1949) states a temporal scale time interval δt is sufficient to capture a signal that typically last $2\delta t$ or longer. The theory is explored by Nathan et al. (2022), who in this context defines behaviour and interactions between animals as a signal. The Nyquist-Shannon theorem implies behaviour should always be described at lower resolution than the timestep itself, and therefore does result in some information loss. Furthermore, this tendency to describe behaviour slower than the temporal resolution does lead to underestimation of total travel distance and the route-description is biased towards more tortuous and faster paths (Ryan et al., 2004). However, a too fine temporal grid combined with low spatial resolution leads to accumulation of errors, especially when behaviour is slow (Ryan et al., 2004). These biases indicate the importance of characterising systems at the right spatio-temporal scale (Ryan et al., 2004).

For different spatio-temporal scales, studies deduce different drivers for animal movement. At high spatio-temporal resolution of 5 second intervals, Nathan et al. (2022) found fish actively avoiding vessels. In addition, distinction between individuals becomes more relevant at a higher spatio-temporal resolution, with larger variation in behaviour among individuals of the same species (Nathan et al., 2022; Tang & Bennett, 2010). Fish behaviour characterisation on a lower resolution of 30 minutes showed fish do not avoid vessels and have little individual variation(Nathan et al., 2022).

Modelling migration requires a simulation of locations, speeds, or directions (Patterson et al., 2008). Representation of a small spatio-temporal scale of e.g. 5 second intervals as in the example by Nathan et al. (2022) implies modelling the movement trajectory of an animal as a sequence of change in direction variables. A fish actively orients itself and its behaviour is driven by its direct environment. The movement path on larger timescales such as 30 minutes (Nathan et al., 2022) may be represented by aggregation of jumps and can be modelled by means of displacement to a destination grid cell.

When sensing capabilities of the animal of interest extend to large distances, this may be a motivation to model movement on larger timescales, and simply model it as a simple displacement to a destination. This simplifies the modelling process and saves computational time. However, it is crucial to ensure that the conditions along the animal's path are suitable for it to reach its destination effectively. If an animal's sensing abilities extend to a different spatio-temporal scale than its movement abilities, decisions about direction are influenced by both the animal's internal movement capabilities and locally changing environmental factors. Conversely, when environmental factors that drive behaviour operate at a larger scale than the animal's movement timestep, its destination is determined by longer-term, goal-oriented preferences. Decision regarding letting the agents demonstrate assertiveness by moving towards spawning grounds or moving in a random pattern are highly subjective when uniformed by literature, and should be tested.

Animal movement may be modelled as random walks on the landscape, corresponding to stochastic processes that are constrained by internal states and external variables (Tang & Bennett, 2010). Movement modes may be reactive and predictive when modelling growth pressure and predation risk. For foraging, distinctive modes modelled may be low- and high-cost, random search and flocking. In many models, memory of animals is simulated as preventing the animal from moving back to a previously visited cell (Tang & Bennett, 2010). This spatial memory is represented in the form of a cognitive map that masks previously visited places from possible relocation destinations. Optimal foraging theory is based on the assumption that animals maximise their energy intake for as little energy loss. A map with the calculated net energy gain or loss for each individual is created, based on the biomass in a cell and the the distance of that cell from the individual (Tang & Bennett, 2010).

3.4 Common Meuse

As of 2013, the Common Meuse has been designated as protected area under the European project Natura2000, meaning a plan should be established to maintain or increase biodiversity with accordance to general biodiversity values established through Natura2000 (Liefveld et al., 2018). There has been a significant biodiversity reduction threat in the Common Meuse, with species not being able to sustain a population due to alternation of the morphology of the Meuse as well as through anthropogenic activities (Liefveld et al., 2018; Lieshout et al., 2003). In the Common Meuse, the common barbel is indicative of good quality and undisturbed flow regime.

3.4.1 Morphology in the Common Meuse

The Common Meuse is an armoured gravel-bed river. The river Meuse originates in France and forms the border between Belgium and the Netherlands between Maastricht and Maasbracht, and is here therefore known as the Common Meuse. The sediment in the river is generally coarse, consisting of sand, gravel, and stones. The river has a gravel bedding. The grain size of the bedding and sediment load as well as the slope of the Meuse reduces substantially downstream. Along the Common Meuse, over a distance of 43 km, grain size reduces from coarse gravel to granules and sand (Duizendstra, 2001).

The morphology of the Common Meuse is anthropologically disturbed, though not as much as other parts of the Meuse (Liefveld & Jesse, 2006). The outer floodplains are mostly used for agriculture and therefore rarely flood. The channel is uniformly shaped, and water levels are deep (Liefveld & Jesse, 2006). The width of the channel has decreased over the centuries, resulting in little variation in water depth over the channel (van Winden et al., 2001). A hydrodynamically similar river to the Common Meuse, the Allier, performs better in terms of biodiversity (Lieshout et al., 2003). Compared to the Allier, relatively steep slopes and narrow river bedding, as well as few gravel bars and little variation in bank structures are present in the Common Meuse (Lieshout et al., 2003). This has lead to little variation in flow velocity, water depth as well as in biotopes (Lieshout et al., 2003). Little variation in habitats has resulted in an overbalance of macrofauna that occur in stagnant to occasionally running water (Lieshout et al., 2003). The water quality is moderate to poor, with algae and silt settling on the river bedding and making its pores inaccessible to macrofauna (Liefveld & Jesse, 2006).

3.4.2 Flow regime in the Common Meuse

The Common Meuse is a rain-fed river and therefore shows significant natural fluctuations in discharge (Liefveld & Jesse, 2006). The upstream Ardennes have little capacity to store rainwater due to its steep slopes and rocky substrate, furthermore contributing to large natural fluctuations in discharge in the Common Meuse (Liefveld & Jesse, 2006). In the past, peat and forest areas in the surroundings had large storage capacity. In the absence of these environments, the surrounding is unable to store water when rainfall is high, leading to high discharges in winter. As a consequence, no water is supplied to the river in times of low discharges in summer (Liefveld & Jesse, 2006; Limburg, 2000).

Anthropogenic water consumption exacerbate significant discharge fluctuations, as heightened consumption often coincides with drought. Factors such as the domestic shipping through the Albertkanaal and Julianakanaal, agricultural practices, drinking water extraction and industrial demands collectively contribute to notably reduced discharge levels, sometimes dropping below 10 m^3/s during summer months (Liefveld & Jesse, 2006)

On top of natural and water consumption fluctuations, anthropogenic dams and smaller structures affect flow regime in the Common Meuse significantly. Extensive upstream control at the power generating dam at Lixhe results in hydropeaking. Dams as a means of power generation, such as the dam at Lixhe, require storing water in reservoirs during periods of low electricity demand and subsequent releasing during high electricity demand. Inaccurate managing of storage dams contributes to hydropeaking. The resulting stream flow alteration consisting of sub daily rapid and marked discharge fluctuations is called hydropeaking. Hydropeaking affects hydromorphodynamics in terms of amplitude, frequency, and rate of change of the water level. In a system experiencing hydropeaking, flow is steady during base and peak flows, and flow is unsteady during up-ramping and down-ramping (Salmaso et al., 2021). The power generating dam at Lixhe comprises four turbines, each capable of being either fully activated or deactivated. With each turbine having a capacity of 80 m^3/s, activation of one results in an increase in downstream discharge fluctuations of approximately 70 m^3/s per turbine, observable a few hours after activation. Peaks in discharges in the Common Meuse currently happen in an inconsistent manner approximately twice a day. Measurements at the South border of the Common Meuse, at Eijsden, indicate differences in the order of 100 to 200 m^3/s on an average discharge of 500 m^3/s in winter(?, ?). At low discharges, increases of 70 m^3/s add to an initial discharge of only 10 m^3/s, resulting in a large relative increase of flow velocity and water depth (Liefveld & Jesse, 2006). On a smaller scale, gravel thresholds also retain flow and alter water depth, with higher water depth but slower flow velocities upstream of a threshold, but higher flow velocities and low water depths over the thresholds.

When alternations are considerably higher compared to naturally occurring flow fluctuations, natural adaptation mechanisms of in stream organisms are often insufficient for individual survival or population sustenance (Salmaso et al., 2021). Species requiring a narrow range of hydromorphodynamics are in particular disadvantage (Salmaso et al., 2021). Unsteady flow causes drifting of species. Juveniles and little-bodied species are in particular prone to drifting because they are worse at preventing displacement as a result of their poorer swimming capacity (Salmaso et al., 2021; Lieshout et al., 2003). Base flows due to hydropeaking may result in disconnectivity of the system, resulting in trapping of individuals. During low discharges, the amount of wet surface area reduces and water temperature rises, resulting in dessication of immobile organisms, including fish eggs. Due to smaller water volumes, higher concentrations of toxic substances and lower oxygen concentrations occur, leading to algae growth and increased likelihood of botulism occurring (Liefveld & Jesse, 2006).

Restoration of species' habitat through management plans for controlling of dams is not straightforward. Expecting results of restoration as a comparison to a static previous situation neglects complex system dynamics and possible adaptive system responses through time lags or non-linear responses (van Oorschot et al., 2022). van Oorschot et al. (2022) investigated the complex system response of a.o. fish' habitat suitability to re-establishment of natural flow regime and concluded that recovery potential depends both on the magnitude of the pressure on the system by disturbance, and the timing of the restoration, and not directly on the duration of the pressure on the system. Recovery times towards pre-disturbance habitat suitability was in the order of 5 to 10 years. Habitat suitability was herein set as an indicator for recovery of the species. However, spatiality of habitat suitability in interaction with the species' location as well as interaction among individuals themselves is herein neglected. The reach of the actual species' response to the environmental attributes that make up habitat suitability in the Common Meuse needs more understanding. Habitat selection modelling has important limitations, and one important one is the movement of animals, which limits mapping their occurrence (Garshelis, 2000).

3.5 Common barbel

3.5.1 Lifecycle of the common barbel

The common barbel is a rheophilic, lithophilous and aggregative fish species. The fish prefers flowing water over stagnant water in all stages of its life, and therefore is typically encountered in the middle reaches of rivers that are referred to as the 'barbel zone'(Huet, 1949). The barbel swims in groups, close to the river bed, relying on clean gravel and sand (Liefveld & Jesse, 2006). The importance of the barbel is in its consideration as a 'flag' species. Their presence indicates fluvial habitats with little disturbance. Conservation efforts are therefore aimed at consideration of the flow regime.

Figure 3.3: Barbel swimming close to the river bed in the Common Meuse (Ravon, 2024)

The juvenile stage lasts one year before the barbel is called an adult. Small larvae (0-6 cm) have very little swimming capabilities and remain close to the spawning grounds. Bigger juveniles move to deeper and faster flowing water, eventually becoming adults and preferring midstream flow conditions. Adult barbels become up to 20 years old. (Liefveld & Jesse, 2006). The number of barbels in the population in the Common Meuse was estimated to be in the order of magnitude of 1000 (Liefveld & Jesse, 2006). Larger fish have larger swimming velocities (Stoffers, Buijse, Verreth, & Nagelkerke, 2022), with young fish still developing their swimming capacity. The spatial scale of daily activity generally varies between tens to hundreds of metres for these young fish.

It is generally unlikely for the common barbel to move long-distances unless the species is specifically looking for spawning grounds or a flood event has occurred, resulting in downstream drifting (Britton & Pegg, 2011). However, distinct behaviour between individuals has been observed, with barbels categorised either as 'resident' or 'mobile'. In a study by Hunt and Jones (1974) on the River Severn in England, the majority (86%) of 531 fish remained between 5 km of their point of release, most of them not moving at all. Other studies report range of motion of only up to 780 metres (Penaz et al., 2002). The other 'mobile' fish swam up to 34 km (Hunt & Jones, 1974) or 2 km (Penaz et al., 2002) away from their point of release in one spawning season. Movement of Barbel is generally upstream during spring and early summer, the spawning time, and downstream in autumn (Britton & Pegg, 2011).

The barbel will start looking for suitable spawning grounds when temperature of water exceeds 13,5 °C, which in the Common Meuse happens around April to May (Liefveld & Jesse, 2006). During the pre-spawning period, migratory behaviour of up to 34 km may be undertaken, with fidelity to particular spawning grounds (Britton & Pegg, 2011; Hunt & Jones, 1974). Males arrive first at the spawning grounds. Females tend to spawn once every season. Multiple spawning is inhibited by the restricted availability of spawning grounds during the low-flows, which often co-occur with the required high temperatures (Britton & Pegg, 2011). The probability for a fish to move from one locality to another during pre-spawning or spawning phases between consecutive days is over 50% (Britton & Pegg, 2011). Spawning tends to be diurnal in non-captive situations, with an average fecundity of 1650 ± to 460 eggs per spawning (Britton & Pegg, 2011).

The barbel's suitability as a flag species is furthermore marked by the fact that many other rheophilic rare fish species such as the common nase, (*Chondrostroma nasa*), schneider (*Alburnoides bipunctatus*), brook lamprey (*Lampetra planeri*), and common minnow (*Phoxinus phoxinus*), co-occur with the barbel (Liefveld & Jesse, 2006). More species benefit from undisturbed flow regimes, as a macrofauna's survival is strongly linked to flow regime. These species rely heavily on stable conditions because of their immobility (Liefveld & Jesse, 2006).

3.5.2 Sensitivity of common barbel to flow regime

The barbel is one of the most sensitive rheophilic species occurring in the Common Meuse and is therefore indicative of good habitat conditions in terms of flow regime (Liefveld & Jesse, 2006). Occurrence of the species is mostly determined by flow velocities and water depth(Liefveld & Jesse, 2006). Lower flows reduce the available spawning and incubation habitat for the barbel, and an alternation between low and high water levels traps individuals in pools, further restricting spawning (Salmaso et al., 2021; van Oorschot et al., 2022). Complete dewatering results in egg or fish desiccation, while high peak flows can destroy eggs through scouring (Casas-Mulet et al., 2015). The Common Meuse is home to the barbel and indicated as a 'barbel zone', (Huet, 1949), however current flow regime conditions are insufficient to accommodate spawning by barbels in the Common Meuse (Liefveld & Jesse, 2006). A study by Liefveld and Jesse (2006) based on flow requirements for the different life stages of the barbel as in 3.1 found that discharges below 15 m^3/s reduce areas of potentially suitable habitat for adults significantly (Liefveld & Jesse, 2006). Under such conditions, the diminished habitat suitability of the Common Meuse stems from both a decrease in wet surface area and a reduction in the proportion of wet areas exhibiting suitable flow velocity and water depth conditions (Liefveld & Jesse, 2006). To facilitate spawning, minimum discharge in the Common Meuse should be a minimum of 40 m^3/s according to Liefveld and Jesse (2006), though base flows in the Common Meuse currently reaches lows down to 10 m^3/s. The loss of connectivity of suitable area may worsen the unavailability of habitat for the common barbel. Besides the potential existence of areas unsuitable for swimming, major spawning tributaries in the Meuse are blocked due to the presences of relatively minor physical obstacles such as thresholds that the barbel could not transcend (Britton & Pegg, 2011). However, in the Common Meuse juveniles have enough habitat, especially in scenario's of low discharges (Liefveld & Jesse, 2006). With the current uniform channel of the Common Meuse, shallow, slow flowing water only occurs when discharges are very low (Liefveld & Jesse, 2006). The current uniform channel only provides shallow water surface area needed for nursing and juveniles at discharges lower than 5 m^3/s. Synthesising these results, a constant discharge at current morphological conditions of approximately 30 m^3/s was found most suitable for common barbel population sustenance (Liefveld & Jesse, 2006).

I aim to incorporate the effect of connectivity due to flow regime and water depth in the Common Meuse as a spatial element to assess the accessibility of suitable habitats for the different life stages of the barbel in the Common Meuse and how this is affected by the discharges and morphology of the Common Meuse.

Table 3.1: Habitat requirements of the Common Barbel

Lifestage or event	**Variable**	**Measure**	**Values**	**Source**
Spawning	Water velocity (m/s)	Range	0.35 - 0.50	(Liefveld & Jesse, 2006)
	Water depth (m)	Range	0.30 - 0.40	(Liefveld & Jesse, 2006)
	Period	Month	April - June	(Liefveld & Jesse, 2006)
Nursery	Mean column velocity (m/s)	Range	0.35 - 0.50	(Liefveld & Jesse, 2006)
	Water depth (m)	Range	0.50 - 1.00	(Liefveld & Jesse, 2006)
Juvenile	Water velocity (m/s)	Range	0.1 - 0.6	(Liefveld & Jesse, 2006)
	Water depth (m)	Range	0.25 - 0.7	(Liefveld & Jesse, 2006)
Adult	Water velocity (m/s)	Maximum	0.05-0.5	(Wortelboer et al., 2020)
	Water depth (m)	Range	0.1 - 1	(Wortelboer et al., 2020)

3.6 Habitat fragmentation

Habitat fragmentation describes discontinuities in an organism's habitat. This can refer to phenomena such as the reduction of the total habitable area and the relative decrease in the size of habitat patches, both of which result in a larger ratio of habitat edges compared to the interior of the habitat (Fahrig, 2003). Habitat loss and its effects on habitat fragmentation can be quantified using three measurable effects: number of patches, mean patch size, and mean isolation (Fahrig, 2003), as shown in Figure 3.4.

Habitat fragmentation directly impacts species by decreasing the habitat available to individual organisms. It also reduces genetic diversity within a population, since smaller habitat patches support smaller populations (Fahrig, 2003). However, the fragmentation threshold hypothesis predicts that the effects of habitat configuration and fragmentation may only manifest at low levels of remnant habitat area, with the exact definition of "low level" varying among landscape types (Brauer & Beheregaray, 2020).

Studies looking at stream connectivity for fish migration specifically indicate that the imperilment of fish migrating to spawning grounds is directly linked to stream fragmentation, with shorter fragments correlating with more vulnerable populations (Perkin & Gido, 2011). Previous modelling studies that relate genetic population structure to historical habitat configuration show a long-term population decline trend since stream connectivity disturbances began 160 years ago (Brauer & Beheregaray, 2020). Small spatial anthropogenic interventions, such as single in-stream barriers, lead to significant changes in population resilience over the long term (Brauer & Beheregaray, 2020). These studies support the general call to provide river connectivity over large spatial scales, encompassing hundreds of kilometres (Nilsson et al., 2005; Dudley & Platania, 2007).

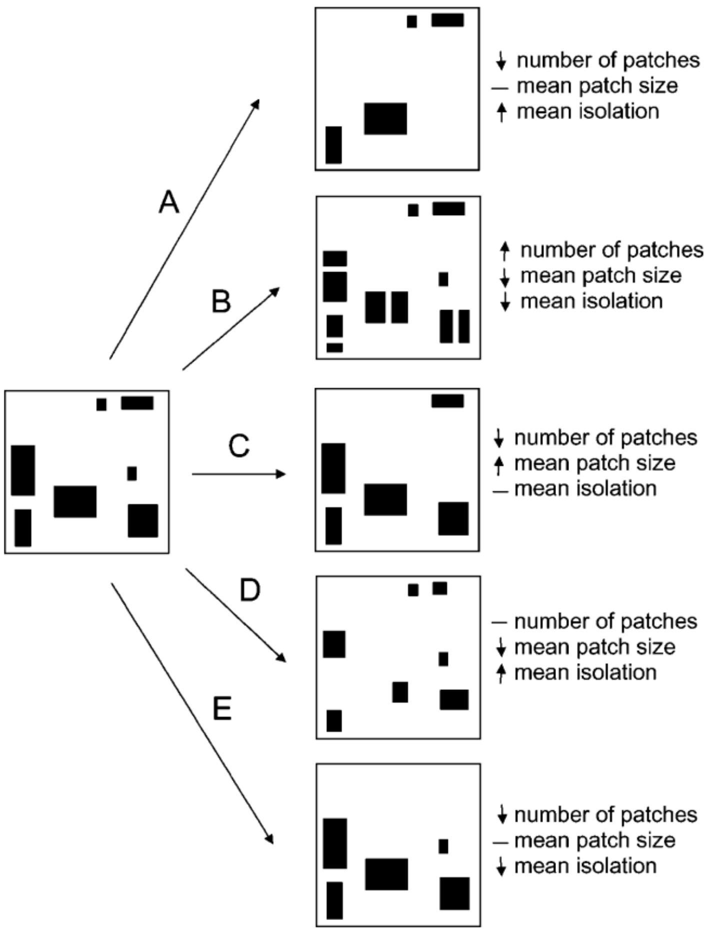

Figure 3.4: Habitat fragmentation as a result from habitat loss measured by number of patches, mean patch size and mean isolation value (Fahrig, 2003)

4 Method

I evaluated the Campo software's ability to accurately depict complexities in habitat-species relationships. Specific requirements to address any identified limitations, as mentioned in the background section, are outlined. Additionally, I introduce a set of operations designed to integrate field and agent perspectives, enhancing the spatio-temporal querying capabilities within the Campo environment. To facilitate intuitive model design, I applied these operations to the case study of the barbel to demonstrate their utility. A deductive model will be built based on the ecological requirements of the barbel in the Common Meuse, and extrapolated to the conditions in the Common Meuse.

4.1 Conceptual model and general ecological principles represented

The common barbel model simulates the swimming behaviour of the common barbel as they search for spawning grounds and try to stay in their preferred habitat amidst a hydropeaking flow regime. Their success in spawning and their movement are tracked, which is the outcome of interest. Their ability to reach spawning grounds depends on their personal daily swimming reach, as well as the areas that are available for them through the connectivity of their habitat. Each timestep, a barbel looks for spawning grounds, senses its environment, evaluates it, and moves accordingly. Barbels may respond differently to the absence of available area within their vicinity, and this varying tendency to explore is also compared.

4.1.1 Habitable area

A barbel assesses its environment for suitable spawning or swimming habitats based on the flow velocity and water depth of the Common Meuse. I assume a barbel evaluates its surroundings as either suitable or unsuitable. The entire Common Meuse is evaluated, and its flow variables change over time, spanning two months. The next step for the barbel in moving towards spawning grounds is to check whether the spawning ground is actually reachable. The area within reach depends on two factors: the connectivity of the water area the barbel is currently swimming in, and the distance the barbel can swim. The swimmable area during one timestep is visualised in Figure 4.1, with different colours representing each connected swimmable area.

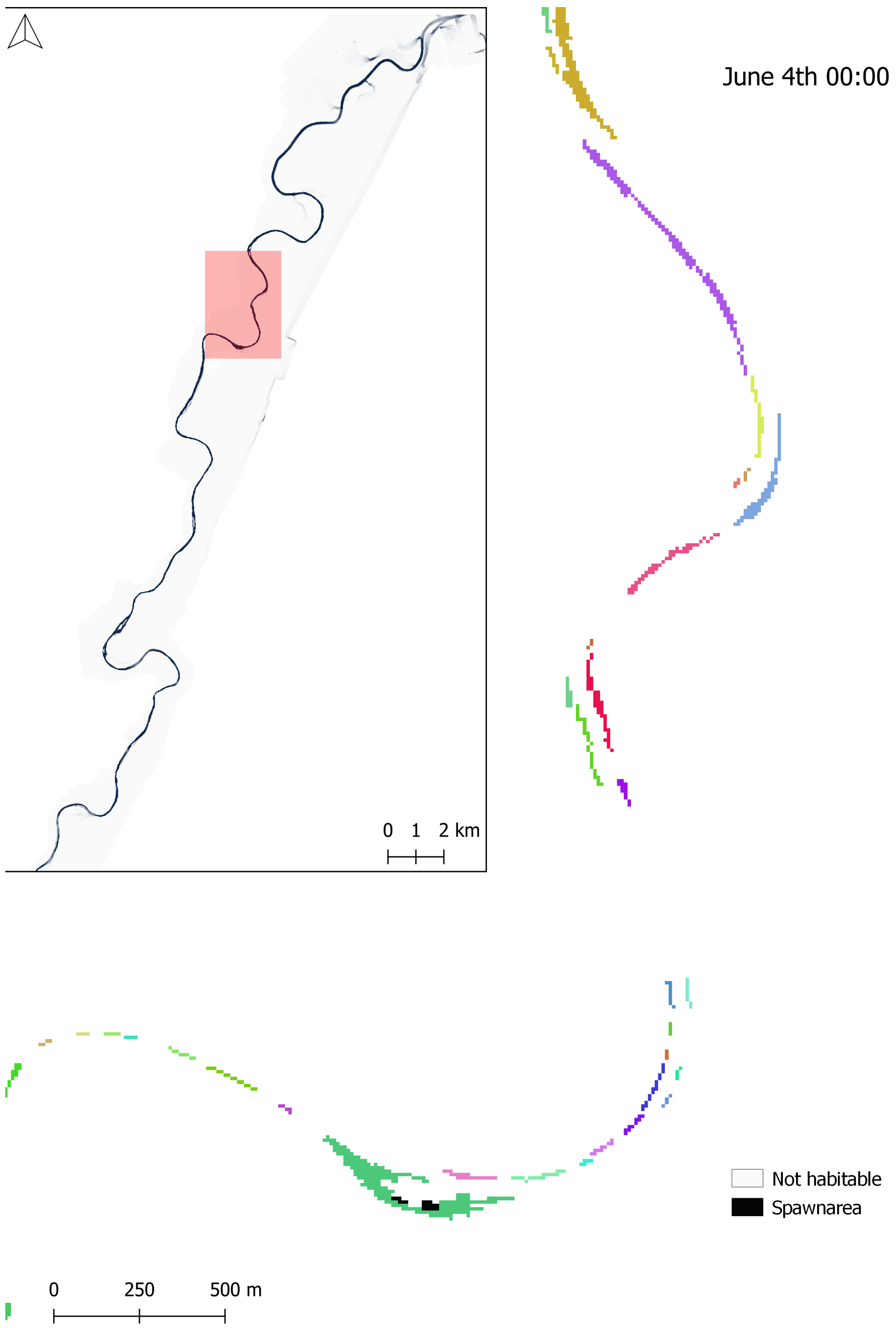

Figure 4.1: Connection of swimmable area with a different colour for each distinct connected patch

4.1.2 Movement modes

The properties attributed to the barbel determine whether they will move, as shown in Figure 4.2. If the barbel has already spawned, its movement is considered irrelevant as its aim has been achieved. If the barbel is not in a swimmable area, it moves to the nearest swimmable area to prevent drowning or desiccation. When the current area is habitable, but no spawning area is within reach, a focussed barbel population may start actively looking for spawning grounds. Barbels of the wandering type swim randomly within the area in reach. If a spawning area is available within reach in the current patch, the barbel moves to the spawning ground and spawns. Some barbels, referred to as travellers, have a greater exploration distance than others, known as homebodies. This increased exploration distance allows them to travel further per timestep.

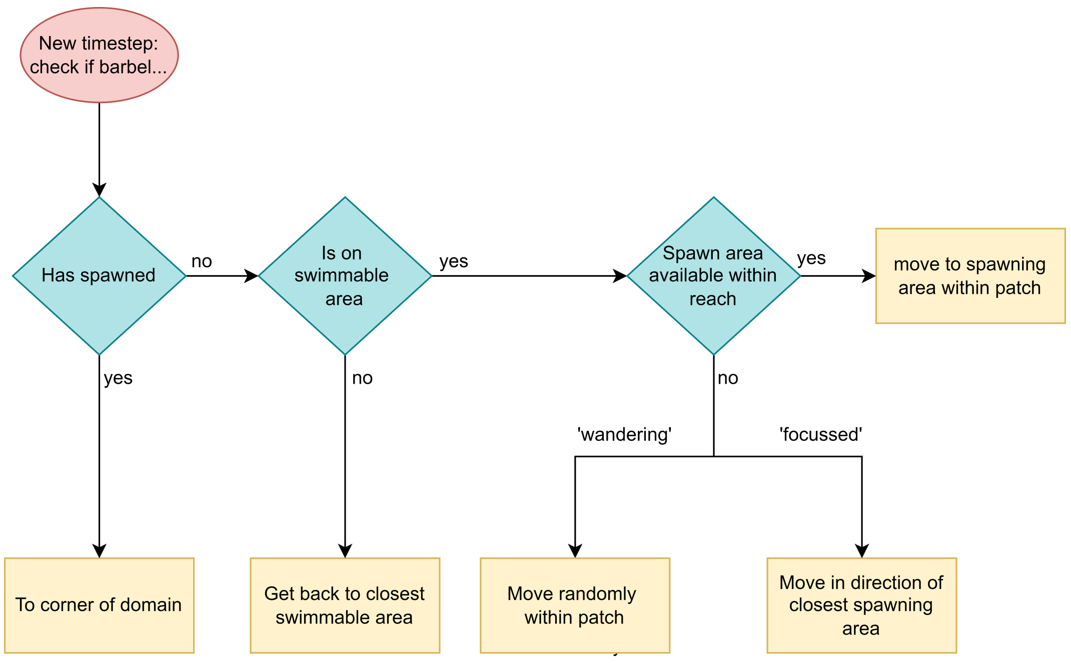
New timestep: check if barbel...
Has spawned
no
Is on swimmable area
yes
Spawn area available within reach
yes
move to spawning area within patch
yes
no
no
'wandering'
'focussed'
To corner of domain
Get back to closest swimmable area
Move randomly within patch
Move in direction of closest spawning area

Figure 4.2: Decision tree for type of movement of the barbel. A population in one model run exhibits attitudes of either type 'wandering' or of type 'focussed', meaning different decisions are made by the barbel about where to go when there is no spawn area in reach but the barbels are not in danger of drowning or desiccating.

In this model, the barbel ends up in a new location that can be reached by swimming, which happens at each timestep until the model stops running. The environmental variables describing the Common Meuse are also updated each timestep, meaning the surroundings of the barbel change both due to its own displacement and the changing environment around it. The output of the model is visualised in Figure 4.3. Barbels are positioned within the Common Meuse in places that are deemed swimmable.

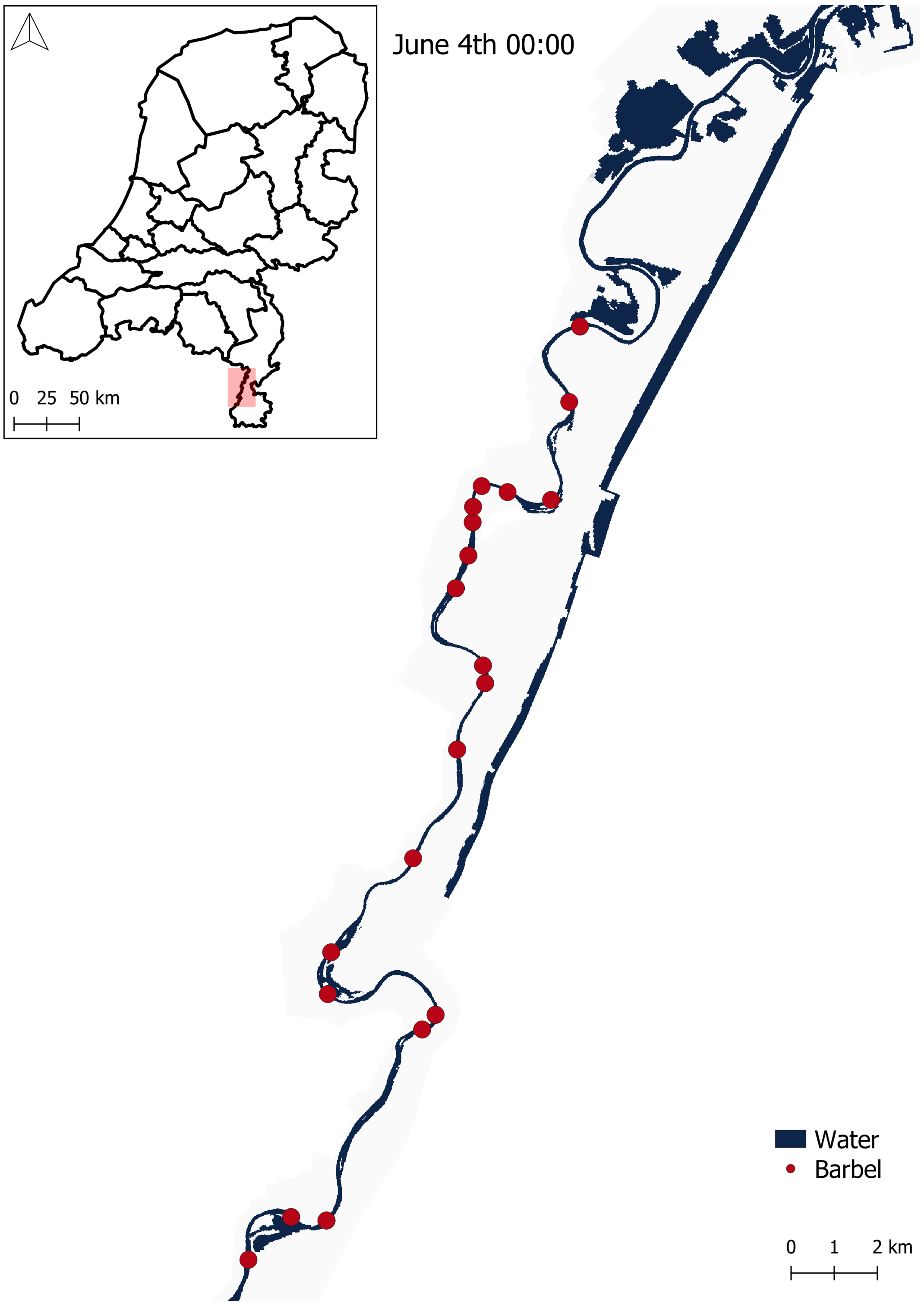

Figure 4.3: Positioning of barbels in the Common Meuse

4.2 Calculations over time

4.2.1 Concepts

In this section, I explore how the elements in the barbel model change over time. What is first computed by the model and what follows? In the initial section, the temporal and spatial domain are described, as well as the settings of parameters, as shown in Table 4.1.

1. The spatial domain of the model is defined by the coordinates of the corners of the study area. The area is divided into a grid with a given resolution for each cell, ΔL.
2. The temporal domain describes how many timesteps the model will run, and with what interval, ΔT, the model will calculate the spawning success and update the location of the barbel.

In the dynamic section of the code, the calculations executed are done so for each timestep. Each timestep, the new environment is updated and evaluated, and so is its response of the barbel.

1. The conversion timestep for the data is calculated. We convert the timestep from the model to the corresponding timestep of the data.
2. The flow velocity and water depth are updated, which are imported from the input data. Each grid cell in the Common Meuse gets a flow velocity and water depth value.
3. Following from a suitability analysis, the spawning grounds and habitable swimmable areas are updated.
4. The swimmable areas that are connected are patched together, resulting in an outcome as in Figure 4.1.
5. Each barbel group is being checked to see if they have spawned or not. If they have spawned, they are moved to the down left corner of the study area. If they have not, the following steps are taken:
6. For each agent, the classification identifier of the patch the agent is located in, is identified and attributed to the agent.
7. For each barbel, a patch array is generated. This is a 2D array with the same spatial domain as the study area, describing for each grid cell of the Common Meuse whether that grid cell is part of the same patch as the agent is currently in. All cells that are in the same patch as the agent, get value 1. All other cells get value 0.
8. For each barbel, the 2D reachable patch array is generated. Cells within the patch that cannot be reached by the barbel because they are too far away, are not taken into consideration as a potential cell for the barbels to move to. This comprises the 2D reachable patch array. What distance would be too far away for the barbel to swim to is defined in the input data.
9. The sum of total spawning area available in the reachable patch, is calculated for each individual barbel.

10. The barbels are moved within their reachable patch. If there is spawning area available in the reachable patch, they are moved to a random reachable spawn area and the attribute that describes whether they have spawned or not, is changed from 0 to 1. If there is no spawning area available, they might be moved either randomly within their reachable patch if they are of type 'wandering' barbel. If they are of type 'focussed' they move towards a nearby spawning ground, trying to get as close as possible. This is in accordance to Figure 4.2.
11. The travel distance for each barbel is calculated and summed to previously swam distances by the barbel.
12. The type of movement mode the barbel used during the timestep is also added as an attribute to the barbel.

Table 4.1: Parameter settings for the barbel model

Parameter	**Symbol**	**Initial values**
Minimal x coordinate	xmin	173 000
Maximum x coordinate	xmax	193 400
Minimal y coordinate	ymin	322 000
Maximum y coordinate	ymax	353 000
Number of groups of barbel	nrbarbel	100
Temporal resolution	ΔT	2 hours
Timesteps	t	732
Period	-	June and July
Spatial resolution or cellsize	ΔL	10 metres
Maximal swimming distance of barbel per day (euclidean)	radius	20 km
Minimal water depth for spawning	spawn_d_min	0.3 m
Maximal water depth for spawning	spawn_d_max	0.4 m
Minimal flow velocity for spawning	spawn_u_min	0.35 m/s
Maximal flow velocity for spawning	spawn_u_max	0.5 m/s
Minimal water depth for swimming	swim_d_min	0.1 m
Maximal water depth for swimming	swim_d_max	1 m
Minimal flow velocity for swimming	swim_u_min	0.05 m/s
Maximal flow velocity for swimming	swim_u_max	0.5 m/s

4.3 Running the model

4.3.1 Sensitivity analysis and barbel behaviour parameterisation

To comprehensively evaluate the performance and robustness of the barbel model, as well as to gain insight in the effect different barbel behaviour has on its spawning success, a sensitivity analysis was conducted. The model's outcome was tested under various scenarios to explore how different input parameters influence the model outputs. Three key parameters are varied: the barbel's exploration distance, the barbel's movement attitude and their range of preference for a spawning ground. For an overview of the model parameter settings, see table 4.2. A nested loop iterates through all combinations of these parameters, setting up folders for the output and running the model for each combination. A batch model run was performed for 8 model runs.

Table 4.2: Parameters describing different types of behaviour and settings for the sensitivity analysis

Parameter		**Value setting 1**	**Value setting 2**
Exploration distance		**Traveller**	**Homebody**
Distance the barbel may travel per day to reach a spawning ground or move in the direction to		20 kilometres	3 kilometres
Movement attitude		**Focussed**	**Wandering**
Type of movement a barbel may use to when there is no spawning area within their reach		Moving within their reach in the direction of the spawning ground	Moving randomly within their reach
Range of spawning conditions		**Narrow**	**Broad**
Range of conditions in which the barbel prefers to spawn	Flow velocity	min 0.35 - max 0.5 m/s	min 0.275 - max 0.575 m/s
	Water depth	min 0.3 — max 0.4 m	min 0.2 - max 0.5 m

The barbel's distance up to which it explores its environment for spawning differs within and among populations, as is elaborated on in the background section. I tested how sensitive the model is to these settings by exploring the difference between a home bound barbel, a homebody, with a maximum daily reach of 3 kilometres, and a travelling barbel, with a maximum daily reach of 20 kilometres. These reaches are represented in the model as a maximum distance the barbel may reach per timestep by multiplying the maximum daily reach with the fraction that the model's timestep is of a day.

I tested the model's sensitivity to a barbel's movement mode when a barbel is in swimmable area, but there is no spawning area within reach. Barbels may demonstrate two attitudes when encountering these conditions: focussed movement or wandering movement. Focussed barbel agents move toward a spawning area function, while barbels move randomly. In both model scenarios, the barbel agents only move within an area that is both within reach and within their connected habitable area.

The model was tested on its performance amidst conditions of varying spawning ground availability, by changing the range of spawning preference conditions. An initial setting with values from literature was tested, as well as a broader range of preferences. A broader range would comply to a more tolerant barbel.

Using the focussed traveller barbel with a narrow range of spawning preference scenario, model output was visually validated by comparing it to data from day-to-day observations by recreationalists and hobbyists (International, 2015), as well as known spawning habitat (Liefveld & Jesse, 2006).

4.3.2 Assessing the effects of hydropeaking

To assess the impact of a hydropeaking flow regime, the model was run with the hydropeaking flow regime and without. The hydropeaking flow regime is based on measurements taken during June and July 2019 (see Figure 4.4).

To simulate a non-hydropeaking signal, I filter the data by averaging it over a day. Available data is sampled throughout the day and summed for each cell, after which it is divided by the number of samples. As the data is sampled every two hours, 12 arrays are averaged for each timestep. This daily moving average effectively removes the hydropeaking signal, as dam management usually causes water levels to fluctuate approximately twice daily.

Barbels in the filtered flow regime run are of type focussed barbel, with a narrow range of preference. Broad ranges of preferences are excluded from the run, as these are not found in literature. Furthermore, focussed barbels are especially of interest, as I hypothesise their assertiveness to be specifically limited by a hydropeaking flow regime. The distinction in barbels which are more or less bound to home is also observed and therefore also compared amidst the two different regimes.

4.3.3 Data reduction and visualisation

Results from the filtered non-hydropeaking flow regime as well as the hydropeaking initial flow regime are compared. To exemplify how the Campo operations can illustrate habitat availability under conditions of habitat fragmentation, we compare them to conventional habitat suitability results. The total amount of spawning area is summed and plotted over time, as in conventional habitat suitability studies. The total percentage of barbel that spawns, for the different sensitivity analysis scenarios as well as the filtered, non-hydropeaking flow regime scenario. This elucidates whether conventional habitat suitability studies are a good proxy for the availability of habitat for the individual barbel in the Common Meuse and up to what extent the hydropeaking flow regime affects barbel's spawning success. The amount of habitat fragmentation in the Common Meuse is quantified by the amount of patches that are present in the Common Meuse and their mean size. The number of patches is plotted to the discharge rates and over time to gain further insight in habitat fragmentation and its relation to discharge. A moving average is also applied to the discharge (see Figure 4.4), which is the discharge that the outcomes of the non-hydropeaking scenario are compared to. The type of movement mode the barbel uses is plotted for the different scenarios.

The sensitivity of the model is assessed by plotting the amount of barbel that have spawned at the end of the model run for each different setting, as well as the timestep at which 50% of the population has spawned. Furthermore, average distances covered before spawning is compared among configurations. The proportion of movement modes are compared among different settings. Furthermore, as the amount of spawning area available is affected by time as well as the range of preference of the barbel, we plot the barbel that have access to the spawning grounds as a function of spawning area. This way, the the effects of the barbel's behaviour through movement attitude and reach are isolated.

4.4 Software implementation of concepts in Campo

4.4.1 Campo requirements for ecological modelling

Our overarching goal is to delineate the ecological response to habitat conditions, with a primary focus on migration patterns influenced by environmental factors. Based on the synthesis of ecological modelling literature outlined in the background section, I establish the following requirements to enable the formulation of intuitive functions that mirror ecosystem dynamics within the model's structure:

1. **Field-aware agents:** Agents should demonstrate awareness of their surroundings, perceiving and responding to changes in their environment. Their proximity to elements in the environment should influence their behaviour proportionately. It is important that the evolution of the environment over time is represented, capturing changes in habitat conditions as input to the ecosystem dynamics.

2. **Movement of agents:** Agents must possess the capability to navigate through space over time, reflecting their dynamic interactions within the ecosystem. Given the pivotal role of heterogeneity in ecological systems, it is important to model agents' movement modes as multifaceted, varying based on the agent's state or life history.

By adhering to these requirements, the Campo environment can effectively represent the complex interplay between species' migration patterns and habitat characteristics.

4.4.2 The common barbel model in Campo

The model structure follows a dynamic PCRaster framework, with each phenomenon and property set being initialised in the initial section of the model. The series of operations in the dynamic section describe the behaviour of the agents under conditions of varying fields. In Campo, all spatial and temporal domains are the same within one property set. A phenomenon may contain several property sets, each being described by properties, as described in the background section and visualised in Figure 3.2. A configuration file is composed in which alternations to the model run can be tested, providing information about the model's spatial and temporal domain, and rules to which the agents must comply. The initial section describes the zeroth timestep, after which the model is run for the specified amount of timesteps in the dynamic section as in Table 4.1.

The available data set covers the Common Meuse between the weir at Borgharen to the weir at Linne over June and July 2019 with a length of approximately 47 kilometres and an average width of approximately 60 metres. The dataset was initially constructed by modelling 2D hydrodynamic conditions (Blacow, 2023), using initial discharge data at Borgharen, see figure 4.4 (Rijkswaterstaat, 2024). The resulting data describes flow velocity and water depth, available at 30-minute intervals over a 2-month period, totalling 2930 timesteps in a flexible mesh format.

Spatial representation is delineated in two dimensions, an x and a y-dimension. The x-dimension relates to the longitude as in the projected Rijksdriehoek (Amersfoort / RD New, EPSG: 28992) coordinate reference system, and the y-dimension to the latitude.

The flexible mesh format is transformed to a 10-metre grid resolution raster. Drawing from the Courant-Friedrichs-Lewy criterion (Courant et al., 1967) and considering a timestep of 2 hours, during which a barbel may cover up to 1.66 kilometres, we determine a minimum grid size of 10 metres to accommodate the model's spatial dynamics. The Courant-Friedrichs-Lewy criterion, expressed as $C = \frac{u\Delta t}{\Delta x} \leq C_{max}$, guides

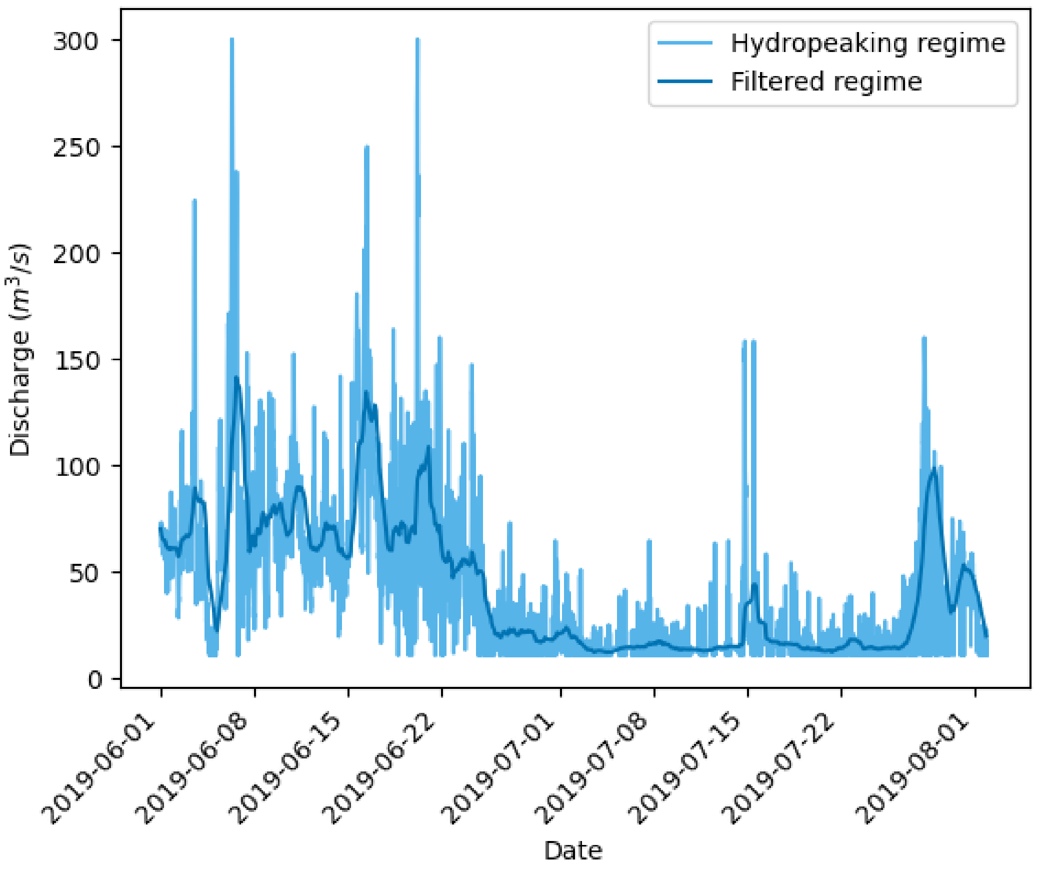

Figure 4.4: Discharge at Borgharen (Rijkswaterstaat, 2024)

our choice of grid size (Δx), where u represents the velocity magnitude of the process of interest, Δt, the timestep, and C_{max} the Courant number, set at 1. Opting for a grid size of 10 metres is significantly finer than required by the criterion but also ensures representation of the spatial variability of habitat surrounding the barbel. This spatial variance is crucial for population survival, influencing habitat suitability for spawning and accommodating different needs of both juvenile and adult fish. A grid size finer than 10 metres would demand extensive computational resources.

Considering the temporal resolution of the model I comply to the Nyquist-Shannon sampling theorem, stating δt is sufficient to capture a signal that typically last $2\delta t$ or longer, we model the barbel's migration twelve times daily in order to be able to capture the effects of the hydropeaking occurring twice a day, on spawning behaviour (Shannon, 1949). Timestep durations exceeding a day introduce biases when modelling animal-behaviour (Baras, 1998). Available spawning area is evaluated and movement is simulated twelve times daily, ensuring dynamic population simulations.

To extend the Campo framework, I address the requirements and outline the specific map algebraic implementation of the requirements as defined in section 4.4.1. First, the concepts from the common barbel model are translated to software operations. On the basis of environmental variables, the model iteratively calculates the number of barbel that have spawned, what movement mode they use and what distances they swim. The calculations and operations as they are described below are technically detailed in the following sections.

1. The current timestep as generated by the pcraster framework is multiplied with the conversion timestep. The conversion timestep is the timestep of the model divided by the timestep of the data (see A.6).

2. Flow velocity and water depth data are obtained for the new timestep using the rasterize function ('partial_reraster') to convert the flexible mesh to raster with the required resolution $DeltaL$, at the required timestep, t (see Appendix A.2).

3. The general habitat suitability assessment uses the function 'two_conditions_boolean_prop', and based on both flow velocity and water depth, results in a spawning ground and swimmable map (see Appendix A.3).

4. Using the 'campo_clump' function (see Appendix A.3), cells that are neighbouring each other are grouped together and classified.

5. The property 'has_spawned' describes whether and agent has spawned. Agents that have spawned are moved to the corner of the study area. If they have not, the following steps are taken:

6. Using the function 'raster_values_to_feature', the unique identifier for the patch the agent is located in, is attributed to the agents (see Appendix A.4, and for syntax, table 4.4).

7. The patch array describes the patch the agent is located to as a boolean map.

8. The reachable patch array describes the reachable part of the patch the agent is located in as a boolean map. To obtain this, a mask is first generated which covers all cells that are too far for the agent to reach with a 0 value (see Appendix A.3. The extent up to which a barbel swims may differ according to whether a barbel is a homebody or a traveller. This determines the radius from the barbel within which all cells are 0. The patch array is multiplied with the mask to obtain the reachable patch array.

9. The amount of available area is calculated with the 'zonal_values_to_feature' function (see AppendixA.4 and for syntax, table 4.4).

10. The barbels are moved using the 'move' functions (see A.5 and figure 4.2)

11. The distances swam by the barbel are an output of the 'move' function (see A.5), which are added to their previously swam distances in the property 'distance_swam'.

12. The type of movement mode the barbel used is added to the property 'movemode' in the barbel.

4.4.3 Data structure

The model is build up out of two phenomena, barbel and water. The fish are modelled over the Common Meuse between the weir at Borgharen and Linne over the course of the months in which they spawn. See figure 4.5 for the general outline of the data.

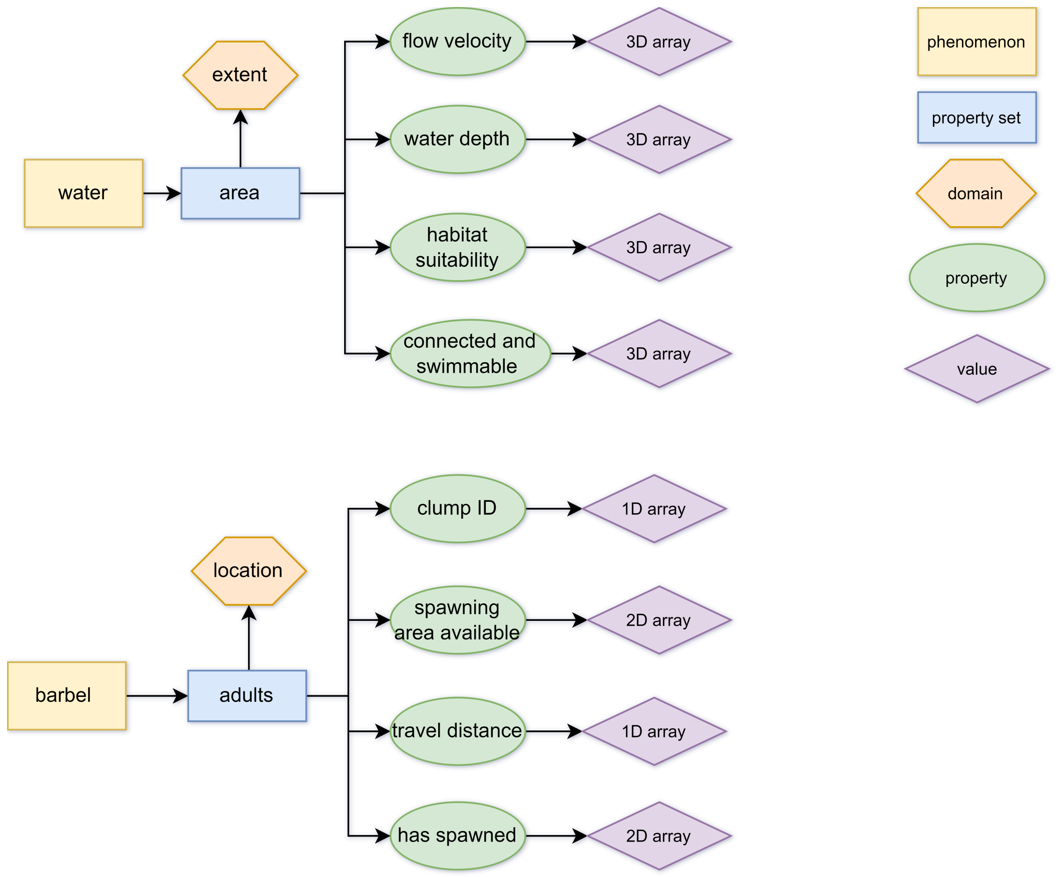

Figure 4.5: Data structure for the barbel model, with water area properties having three dimensions: x,y, and time. Barbel adult properties have two dimensions: for each barbel, over time. Barbel properties with only one dimension are properties that are the result of summation over time (travel distance) or replacement, meaning only values about the last timestep are saved.

The water depth and flow velocity, collectively termed as the flow regime, are represented in Campo as 2D numpy arrays, each forming a property. The transformation of the initial flexible mesh data format to a numpy array, is performed using the Xugrid module(Deltares, 2024), for each required timestep. These variable arrays are structured within a predefined domain, allowing for translation from simple array indexes to coordinates in a coordinate reference system. The spatial extent of the field covered by the flow regime data is encapsulated as the domain within the property set, conceptualised as a single agent with a defined spatial footprint. This spatial description encompasses six key parameters: the minimum and maximum x and y values, along with the number of columns and rows, thereby establishing a bounding box that also dictates the resolution of the data, facilitating compatibility with raster data formats. Herein, the x-axis corresponds with the columns in the array and the y-axis with the rows.

Specifically looking at spawning suitability, we create a new field-agent property which spatially describes its suitability as a spawning location. The ranges of preferences for both water depth and flow velocity, as in table 4.3), and the field-agent properties describing the water depth and flow velocity are used as an input for the function 'two_conditions_boolean_prop' (see Appendix A.3). From this function, a boolean map is generated describing the places that comply to all conditions within the range of preference of the barbel.The function operates as illustrated in figure 4.6. The information of each cell in the raster is translated into suitability maps using the parameter specific rules, α and β. The resulting suitability maps a_s and b_s are combined to create an overall suitability raster. Similarly, a boolean map is added as a property to the propertyset of waterarea describing the habitability of the Common Meuse for barbel adults.

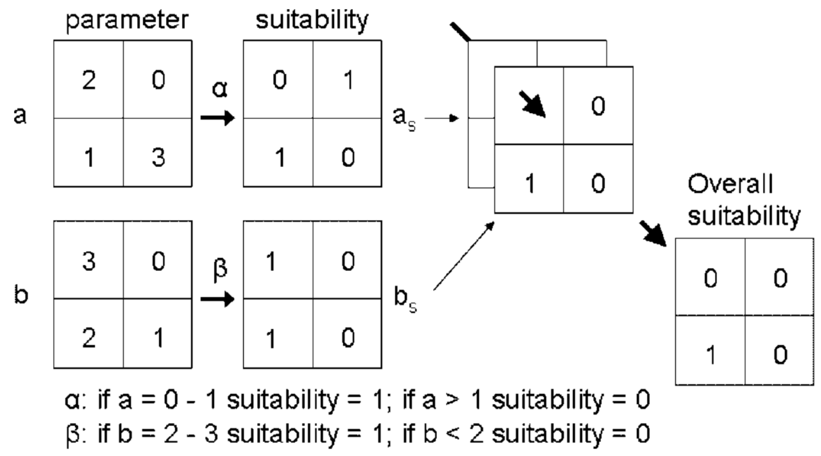
parameter
suitability
a
b
α
β
a_s
b_s
Overall suitability
α: if a = 0 - 1 suitability = 1; if a > 1 suitability = 0
β: if b = 2 - 3 suitability = 1; if b < 2 suitability = 0

Figure 4.6: Map algebra approach of calculating habitat suitability (Van de Wolfshaar et al., 2010)

Table 4.3: Habitat requirements of the Common Barbel

Lifestage or event	Variable	Measure	Values	Source
Spawning	Flow velocity (m/s)	Range	0.35 - 0.50	(Liefveld & Jesse, 2006)
	Water depth (m)	Range	0.30 - 0.40	(Liefveld & Jesse, 2006)
Adult swimming	Flow velocity (m/s)	Range	0.05-0.5	(Wortelboer et al., 2020)
	Water depth (m)	Range	0.1 - 1	(Wortelboer et al., 2020)

Grid cells deemed swimmable by the requirements of each barbel agent (see table 4.3) are grouped together if they are within the same immediate 8-cell neighbourhood, employing the function 'clump' by the PCRaster module (Karssenberg et al., 2010). This patch field-agent layer, describing areas that are both connected and swimmable, tags each connected and swimmable patch of cells with a unique ID.

100 groups of barbels are initially modelled in the Common Meuse, representing 10 barbels each. Similarly to how each flow velocity and water depth value is tagged with coordinates, each individual barbel within the model is endowed with a specific location, with the centre coordinates serving as the primary descriptor for each agent. Utilising the local 'raster_values_to_feture' function, the local patch ID values from the patch field-agent layer is assigned to each agent as a property of that agent at every timestep. This ensures that each agent interacts with the flow regime data based on its precise spatial position within the model, enabling accurate representation of environmental dynamics over time. Assigning local variables to the agents meets the requirement of our model for agents to be field-aware.

To enhance the representation of spatial awareness by the barbel, areas deemed fit for spawning that are within one agent's patch and within the exploration distance of the barbel, are summed and attributed to the agent as a property. I multiply the patches for each agent with the habitat suitability maps available as a property of the Common Meuse, resulting in a destinations map for each individual barbel. When spawning area is available, the value of the property 'has spawned' is updated from 1 to 0.

4.4.4 Field agent interactions

Although the Campo framework already enables the attribution of point agent variables to cells in field agents, it lacks the capability for point agents to be aware of and interact with variables from field agents in a reverse manner. In response, this thesis introduces three categories of operations to Campo, aimed at extending the integration between field agents. These operations align with the principles of map algebra (Takeyama & Couclelis, 1997; Tomlin, 1994), namely a local, focal and a zonal operation. Since both the field-agents and the point-agents are tagged with coordinates, their spatial relationship can be established when specifically queried. See table 4.4 for an overview of the syntax.

A local operation ('raster_values_to_feature') is developed, which attributes the value of a field-agent variable at the location of the point-agent to the point-agent itself. First, the coordinates of each point agent are retrieved and matched to the corresponding cell within the field-agent, establishing their topological relationship. The value of the variable is then queried and attributed to the point-agent as a property.

A focal operation 'window_values_to_feature' involving the aggregation of data within a defined neighbourhood or window was developed. As we integrate fields and agents, the centre of this window corresponds to the location of the point agent. Variable field values from within this window are aggregated by summation, averaging, or determining the minimum or maximum value. Subsequently, these aggregated values are assigned to the agent as a property. A similar result could be obtained using a field-based operation, by first averaging over a given window size and attributing those areas to all cells within the window. However, the size of the window or filter, can in such case not depend on the internal state of the agent.

Table 4.4: Table describing syntax of the field-agent interacting functions

Function	Input arguments	Output arguments
raster_values_to_feature Queries the raster values of one field property on the location of point agents, writes this as a new point agent property	• point_pset: property set of point-agents to attribute the local field property to (type: Propertyset) • field_pset: property set of single field-agent (type: Propertyset) • field_prop: property of field-agents of which the values are attributed to the point-agents (type: Property)	• point_prop: property of point-agents with values of local field variable (type: Property)
window_values_to_feature Queries aggregative raster values of a window of a field property based from the location of point agents, given a certain aggregative operation. Writes this as a new point agent property	• point_pset: property set of point-agents to attribute the aggregative field property to (type: Propertyset) • field_pset: property set of single field-agent (type: Propertyset) • field_prop: property of field-agents of which the values are attributed to the point-agents (type: Property) • windowsize: integer describing the length of the window over which aggregation happens, in the unit of the field-agent property set (type: Integer) • operation: aggregative operation available in numpy ('sum', 'mean', 'min', 'max', 'etc') (type: String)	• point_prop: property of point-agents with values of aggregated field variable (type: Property)
zonal_values_to_feature Queries aggregative raster values of a zone of a field property based on the location of point agents within a classification map, given a certain aggregative operation	• point_pset: property set of point-agents to attribute the field property to (type: Propertyset) • field_pset: property set of single field-agent (type: Propertyset) • field_prop_class: property of field-agent describing classes or groups of cells of which the zonal extent shall be the window of the aggregation (type: Property) • field_prop_var: property of field-agents of which the values are attributed to the point-agents (type: Property) • operation: aggregative operation available in numpy ('sum', 'mean', 'min', 'max', 'etc') (type: String)	• point_prop: property of point-agents with values of aggregated field variable (type: Property)

A zonal operation 'zonal_values_to_feature' was designed to attribute the aggregation of values in a zone in which a point agent is located, to the point agent. These zones are delineated based on the internal variation in a specific field agent variable, resulting in distinct classifications such as land use types. The zone the agent is located in is identified using the 'raster_values_to_feature' operation. Field variable values of cells within the zone are aggregated and assigned as properties of the agent.

4.4.5 Movement of agents

Migration is a pivotal event influenced by and influencing the life phase of barbel, reflecting the heterogeneity in their behaviour as well as resulting a different surrounding for the barbel. The properties attributed to the barbel determine whether they will move or not, as can be deduced from figure 4.2. A local operation ('raster_values_to_feature') is performed to check if the barbel is located in a cell that is considered habitable for swimming or not. If that is not the case, the coordinates of the nearest swimmable area are set as new coordinates, using the 'find_closest_dest' function. Directed migration ('move_directed') happens when the current area is habitable, but there is no spawning area in the current zone or patch. If there is spawning area available in the current zone, destination oriented ('randommove_to_destination', see A.5) migration happens.

Three distinct movement modes were developed, each generating a set of coordinates corresponding to the point agents in a specific order. These coordinates serve as the basis for updating the agents' locations. The design rationale behind these movement modes, namely random, destination-oriented, nearest neighbour, and directed movements, is discussed in the following paragraphs. See table 4.5 for an overview of the syntax.

The modeller can initially request the space domain of agents. The current locations of the agents are obtained through the 'get_location' function. Movement functions can be executed and used to reassign these coordinates to the desired destination location using the 'set_location' function.

The random movement function operates by determining a set of coordinates for each agent within the bounding box of the field. Initially, the bounding box of the field is acquired, outlining the maximum and minimum x and y values. Subsequently, random samples are drawn from both the x and y dimensions, with the sample size corresponding to the number of agents. The agents are moved to these sampled locations. Movement to a random 'True' cell in a boolean map is implemented via the function 'randommove_to_destination'. This function retrieves coordinate sets from a boolean map that delineates potential destinations as 'True'. Cell indices of the field-agent property are conversed to coordinates using the resolution and minimum X and Y-coordinates of the field-agent. This means that agents are always placed in the lower left corner of a cell that corresponds to their destination. From the sets of 'True'-coordinates, a random sample is extracted, with the sample size corresponding to the number of agents. For this purpose, the random library is employed, which does not replace elements after they have been sampled. This means that when sampling for the entire population at once, agents are dispersed and do not end up at the same cell. However, when sampling one by one for a personalised destination boolean map, several agents may be placed in one cell, potentially leading to crowding of the cell.

The 'find_closest_destination' function determines the nearest destination for an individual agent. Initially, it retrieves the location of the agent and translates this value to the corresponding cell location on the boolean field-agent, where a value of 'True' signifies a potential destination. The Nearest Neighbors function from Pedregosa et al. (2011) is applied, starting from the cell corresponding to the point agent's location, finding the closest destination cell.

The 'move_directed' function represents an agents' ability to sense its surroundings, and its inability to get there in that same timestep. The directed move operates by first finding the closest destination, utilising the 'find_closest_destination' function. From these coordinates, the closest cell complying to the agent's maximum reach is found, once more with the 'find_closest_destination'. The argument for the destination map in this function call describes all cells within an agent's reach as 'True'.

Table 4.5: Table describing syntax of the movement functions for point-agents

Function	Input arguments	Output arguments
randommove_to_boolean Generates a list of sets of coordinates randomly picked from a boolean field.	• field_pset: property set of single field-agent (type: Propertyset) • boolean_fieldprop: field-agent property describing the destination with value 'True' (type: Property) • nr_agents: amount of agents to be generated new coordinates for (type: Integer)	• xcoords: list describing the x-coordinates (type: Array) • ycoords: list describing the y-coordinates (type: Array)
find_closest_dest Finds the set of coordinates with a true value that is closest to the location of the agent.	• field_pset: property set of single field-agent (type: Propertyset) • boolean_fieldprop: field-agent property describing the destination with value 'True' (type: Property) • point_pset_orX: property set of point-agents to attribute the field property to (type: Propertyset), or x-coordinate describing the x-coordinate of the agent, only when the y-coordinate is given next (type: Integer). • pidx_orY: the index of the point agent to be attributed a new coordinate set to (type: Integer) or the y-coordinate describing the y-coordinate of the agent (type:Integer)	• xcoord: x-coordinate (type: Integer) • ycoord: y-coordinate (type: Integer) • travel_distance: the Euclidean distance travelled by the agent (type: Integer)
move_directed Finds the set of coordinates that is most in the direction of the closest destination to the location of the agent, but within the reach of the agent.	• field_pset: property set of single field-agent (type: Propertyset) • dest_boolean_fieldprop: field-agent property describing the destination with 'True' (type: Property) • boolean_clump_fieldprop: field-agent property describing the cells that are within the reach of the agent with 'True' (type: Property) • point_pset: property set of point-agents to attribute the field property to (type: Propertyset). • pidx: the index of the point agent to be attributed a new coordinate set to (type: Integer).	• xcoord: x-coordinate (type: Integer) • ycoord: y-coordinate (type: Integer) • travel_distance: the Euclidean distance travelled by the agent (type: Integer)

5 Results

5.1 Discharge and habitat dynamics

5.1.1 Habitat availability

The response of habitat availability to discharge is complex, as illustrated in Figure 5.1. Over time, there are two distinct discharge regimes in the data: in June, discharges exhibit high peaks and significant fluctuations around an average of about 60 m^3/s. In contrast, the Common Meuse in July experiences generally dry conditions, with discharges averaging around 10 m^3/s and showing smaller variations.

First considering the unfiltered hydropeaking flow regime with the initial range of preference for the spawning grounds by the barbel. Periods of frequent, large fluctuating discharges, such as in June, correlate with highly fluctuating availability of spawning habitat for barbels, as shown in the second plot of Figure 5.1. Both discharge and spawning area can be described as signals in terms of amplitude, frequency, and magnitude. As discharges and their absolute changes decrease in July, spawning area generally increases, showing smaller amplitude and lower frequency fluctuations. Under these dry conditions, troughs in the spawning area signal often coincide with peaks in discharge. However, this relationship is not evident during the high and fluctuating discharges in June. This is further indicated by the lack of a significant trend in Figure 5.2. From this figure, it can be derived that in general, higher discharge rates result in a reduction of the available spawning habitat. However, the significance of this trend is low (R^2: -0.47) and the slope of the trend is not high compared to the order of magnitude of habitat availability. This suggests that a wide range of spawning habitat availability exists across various discharge scenarios.

In scenarios where the barbel has a broad range of preference, highly fluctuating discharge before July leads to equally fluctuating availability of spawning habitat. Low discharge scenarios show a slight general increase in spawning area with much smaller amplitude changes. During the drought regime, large increases in discharge cause an initial increase followed by a significant decrease in spawning area. This indicates that not only the magnitude of discharge but also the rate and relative change matter. Figure 5.2 shows a significant trend; however, the low slope suggests that a wide range of spawning area may be present under various discharge scenarios.

A filtered flow regime, results in a similarly filtered spawning area availability compared to the unfiltered flow regime, but with a lower frequency of fluctuating spawning areas. However, the habitable area for swimming is higher for similar rates of discharge compared to the unfiltered flow regime (see Figure 5.3). The linear regression analysis shows a much steeper relationship between discharge and swim area for the filtered flow regime compared to the unfiltered flow regime.

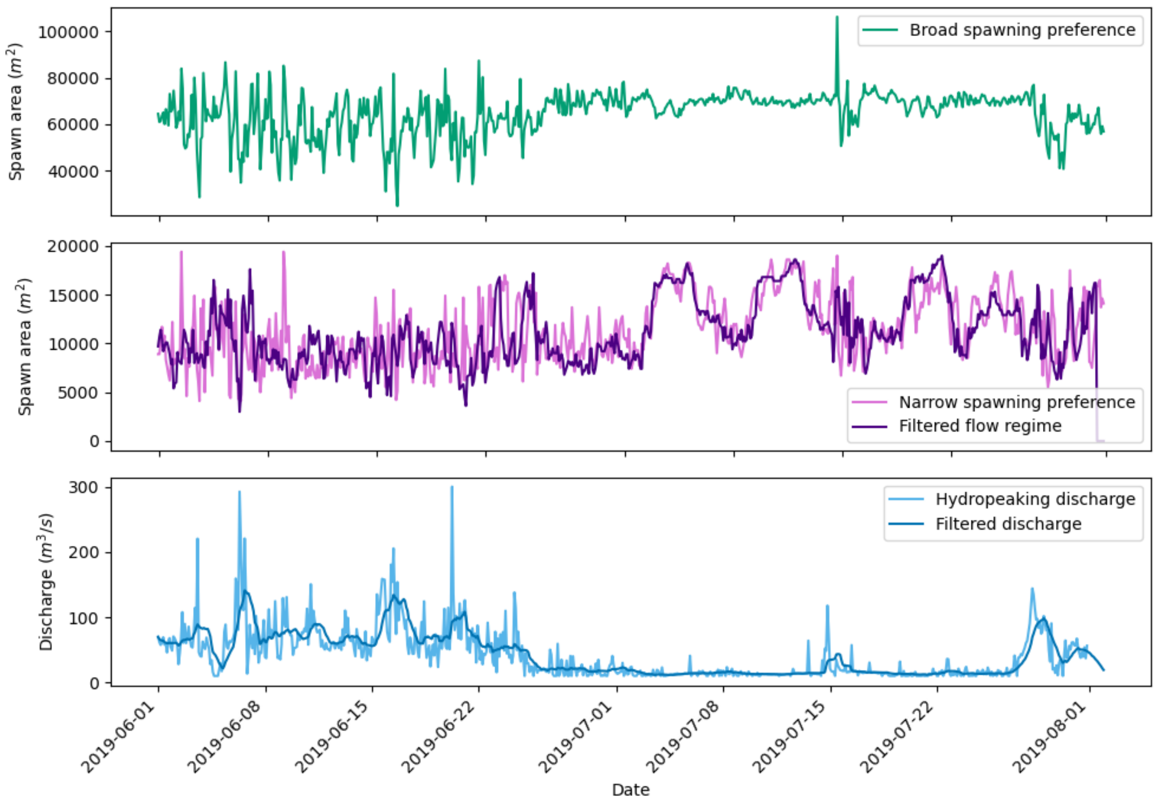

Figure 5.1: The spawn area compared to the discharge at Borgharen. The upper plot shows spawn area over time for barbels with a broad range of spawning preference. The centre plot shows spawn area over time for barbels with narrow range of preference for spawning ground, both for a hydropeaking and filtered flow regime. Note how upper and centre plot have different scales. The lower plot shows discharge over time.

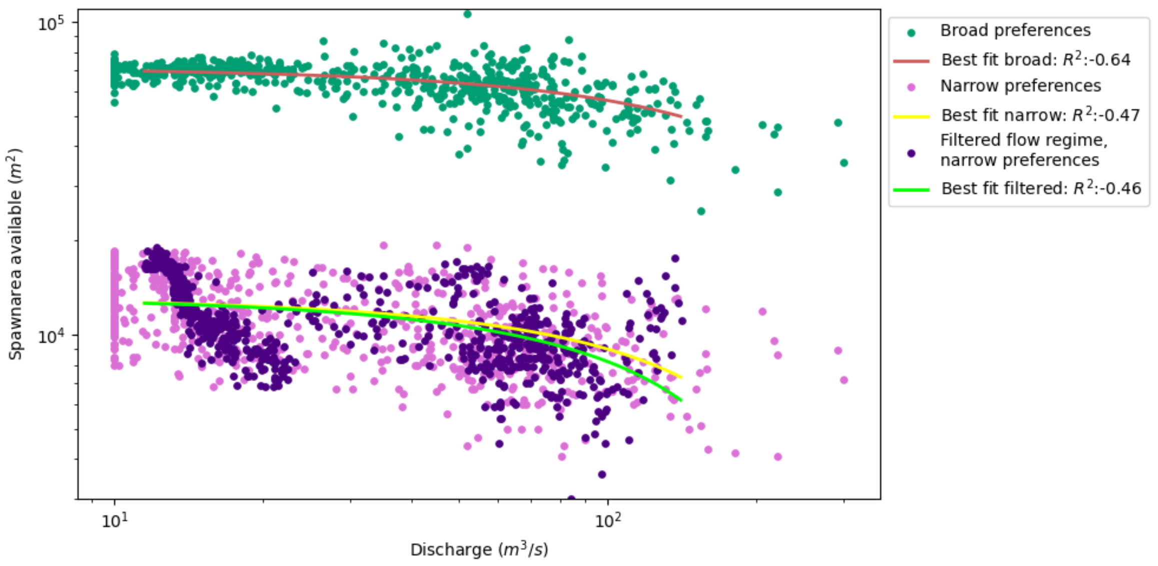

Figure 5.2: Relation between the discharge at Borgharen and the downstream available spawnarea

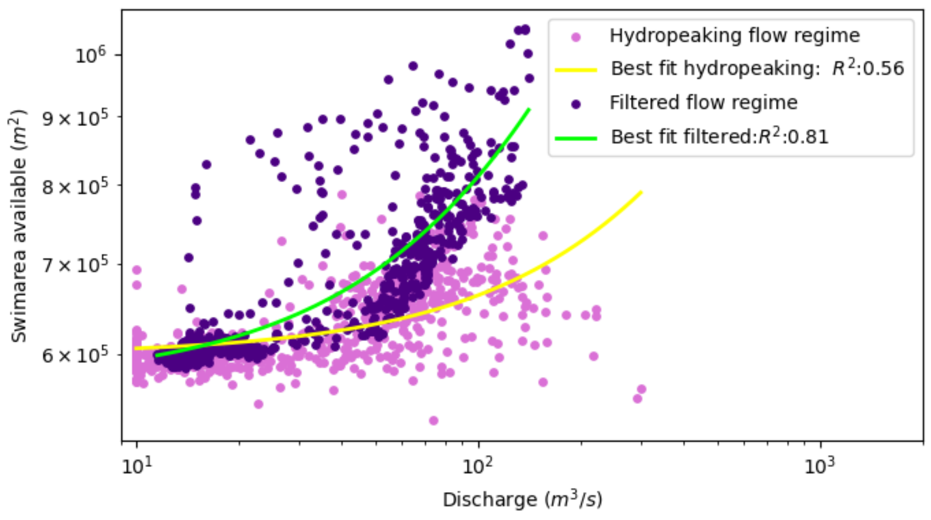

Figure 5.3: Relation between the discharge at Borgharen and the downstream spawn area available. Outcome of analysis with filtered flow regime assumes a narrow preference of spawning habitat by the barbel.

5.1.2 Habitat fragmentation

Similarly to how frequency and magnitude of discharge relate to the amount of habitat available, the fragmentation is also affected by it (see Figure 5.4). Higher discharges lead to a higher number of habitat patches, and low discharges to a low number of patches. This is true for filtered as well as unfiltered flow regimes. From figure 5.5, the general significant relation between the number of patches and discharge can be established, with an increasing number of patches for an increasing discharge.

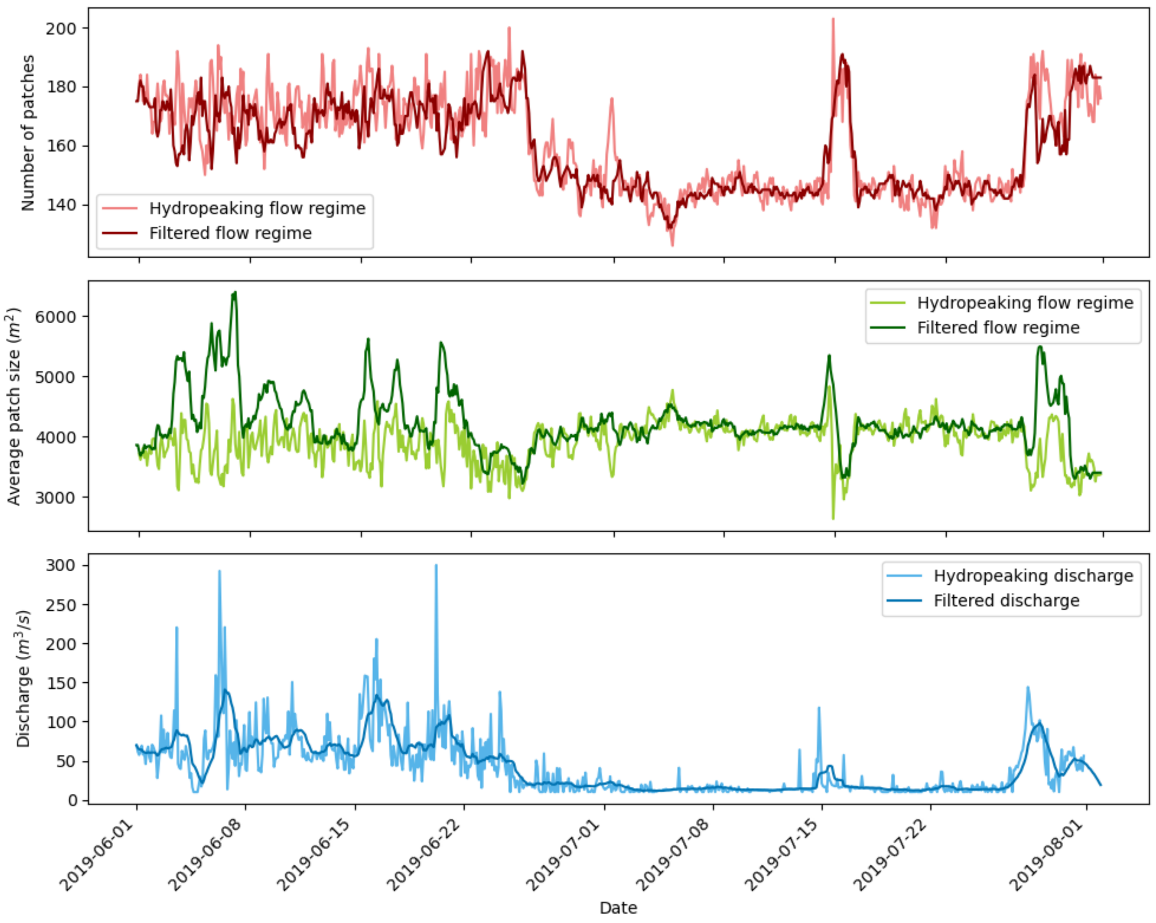

Figure 5.4: The habitat fragmentation over time compared to the discharge at Borgharen. The upper plot shows number of patches present in the entire study area over time, centre plot the average size of each patch over time and lower plot the discharge over time.

The average patch size amidst an unfiltered hydropeaking flow regime, as shown in the second plot of Figure 5.4, is only slightly affected by discharge. It fluctuates around 3700 in June and slightly increases to around 4000 in July. In contrast, the filtered flow regime responds differently to the overall decrease in discharge in July. The average patch size signal shows a skewed pattern with occasional high patch sizes, averaging approximately 4500, featuring high but short peaks and longer but less deep troughs. Although the average magnitude of patch size does not increase in July, it slightly decreases with a smaller frequency and relative magnitude of change. This difference in patch size response to discharge between hydropeaking and non-hydropeaking flow regimes can also be seen in Figure 5.6.

Relative small habitat fragmentation measures, meaning a small number of patches and a high average patch size (Fig. 5.4), for the filtered flow regime, were present during July.

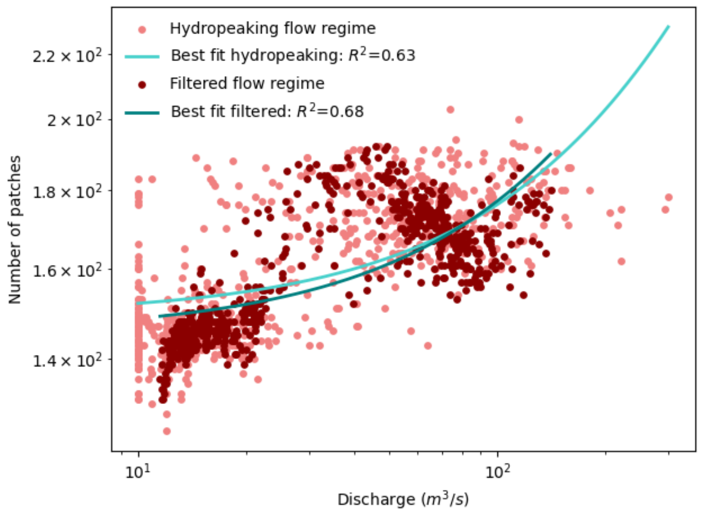

Figure 5.5: Number of patches or clumps compared to the discharge at Borgharen.

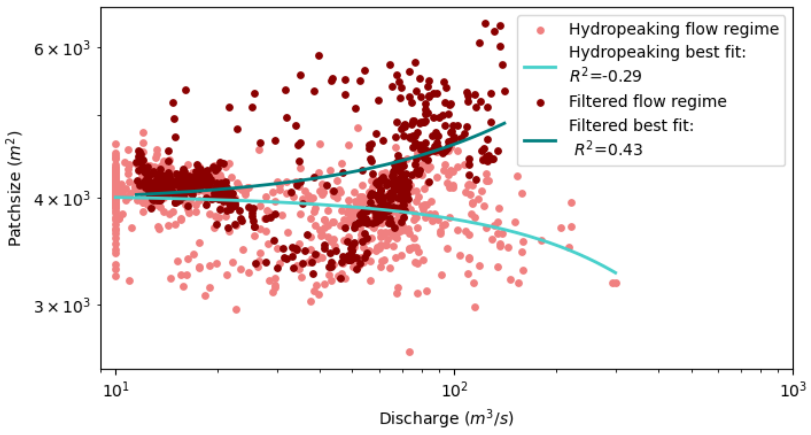

Figure 5.6: Relation between the discharge at Borgharen and the downstream patch sizes.

Habitat suitability maps over time indicate that in the initial spawning preference scenario, some spawning locations remain consistently suitable, while most spawning areas disappear after one or a few timesteps. New spawning locations frequently emerge, highlighting the dynamic nature of the environment. Table5.1 shows the dynamics of spawning area habitat availability across three scenarios: the hydropeaking flow regime, both broad and initial range of spawning preference, and the initial range for the filtered flow regime. Comparing for the initial range of preferences, the hydropeaking scenario with the filtered flow regime, the biggest difference is the duration of a spawn cell's existence, with filtered spawn cells generally persisting longer in a filtered flow regime. For barbels with broad preferences, some spawning habitat cells are present of up to 58% of the time. However, on average, a spawn cell is only present 11% of the time, with an average duration of 12 hours per cell. In the filtered flow regime, the maximum presence of spawn cells is much lower, but the average duration of each spawn cell is similar.

Table 5.1: Spawning area duration considering different scenarios. Presence of a spawncell in percentages describes, for a cell that is defined as a spawning cell at some point over time, for what percentage of the run this spawncell is actually a spawning cell. Duration of the spawncell before transition describes how long this cell exists continuously in time, before transitioning to a non-spawning cell.

	Hydropeaking		Filtered
	Broad range	Initial range	Initial range
Average presence spawncell over time	11%	4.5%	6.0%
Median presence spawncell over time	4.2%	1.5%	3.1%
Maximum presence spawncell over time	58%	42%	40%
Average duration before transition	12 hr	7.3 hr	11 hr
Median duration before transition	5.6 hr	3.7 hr	5.8 hr

5.2 Fish movement in response to habitat dynamics

In the Common Meuse, the barbel zone extends north to coordinates 186000, 345000, before the bend around Herenlaak near Maaseik (see Figure 5.7). Both swimmable areas and suitable spawning habitats are present up to this point. Some areas, such as the bend around coordinates 179000, 330000, may be unnecessary to cross as there is enough spawning habitat nearby. Many barbels spawn here without crossing the area. The area covered by wandering versus focused travellers differs, with wandering barbels generally covering more areas. However, certain areas, like the bend around coordinates 179000, 325000, are consistently crossed by focused barbels but not by wandering ones. The locations that the model defines as destinations for the barbel to spawn, are also known as such in the field as in Figure 5.9.

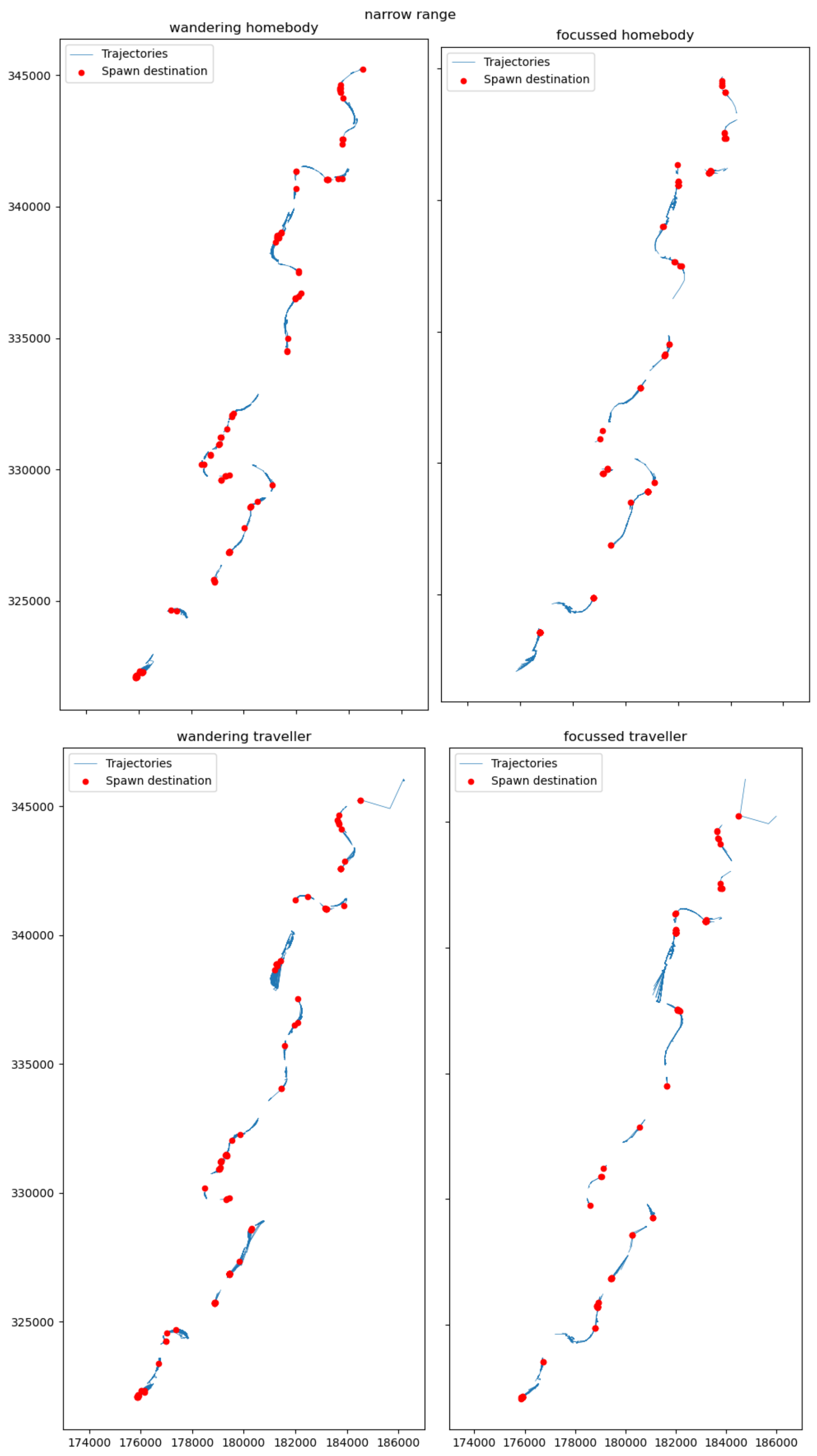

Figure 5.7: Trajectories and destination for spawning by the barbel with the initial range of spawning preference. Tracks of all barbels are plotted, with thicker lines representing more often used paths. Consult Table 4.2 for specific parameterisation of the fish behaviour definitions.

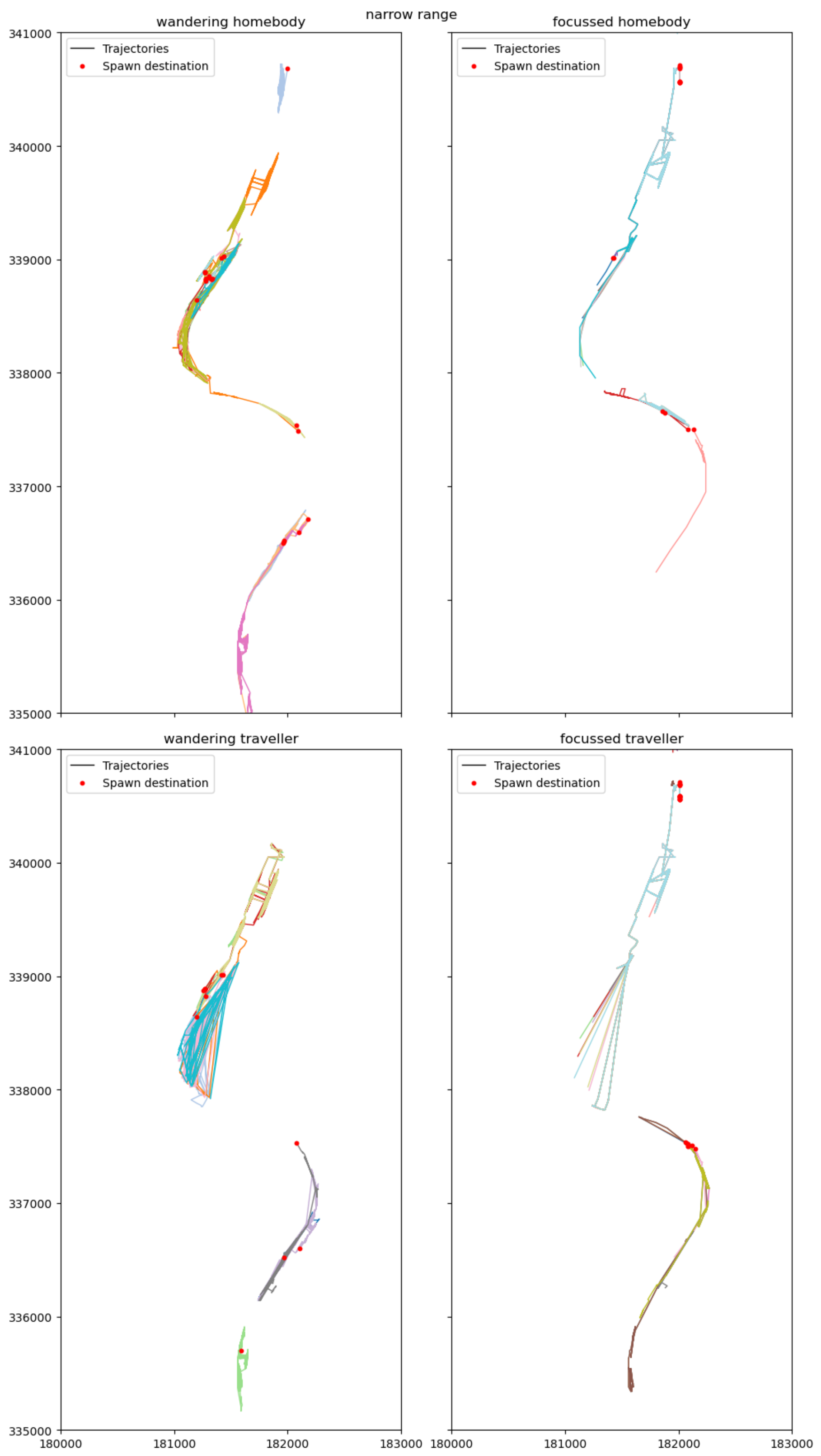

Figure 5.8: A zoom in of trajectories and destination for spawning by the barbel with the narrow range of spawning preference. Each distinct coloured line represents the track of a single group of barbels. Consult Table 4.2 for specific parameterisation of the fish behaviour definitions.

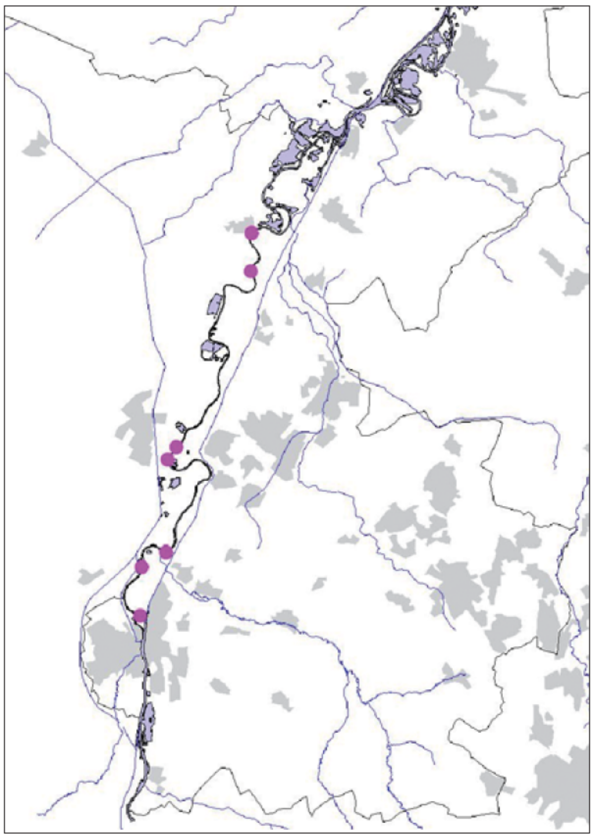

Figure 5.9: Locations of spawning grounds known in the field in the Common Meuse (Liefveld & Jesse, 2006).

Barbels with broad ranges of spawning preference displace themselves less due to the high availability of spawning areas. Their spawn destinations in Figure 5.10 show a distribution of spawning areas mostly before and after bends in the channel. Barbels in Figure 5.11 find spawning area faster compared to barbels with a narrow range of preference.

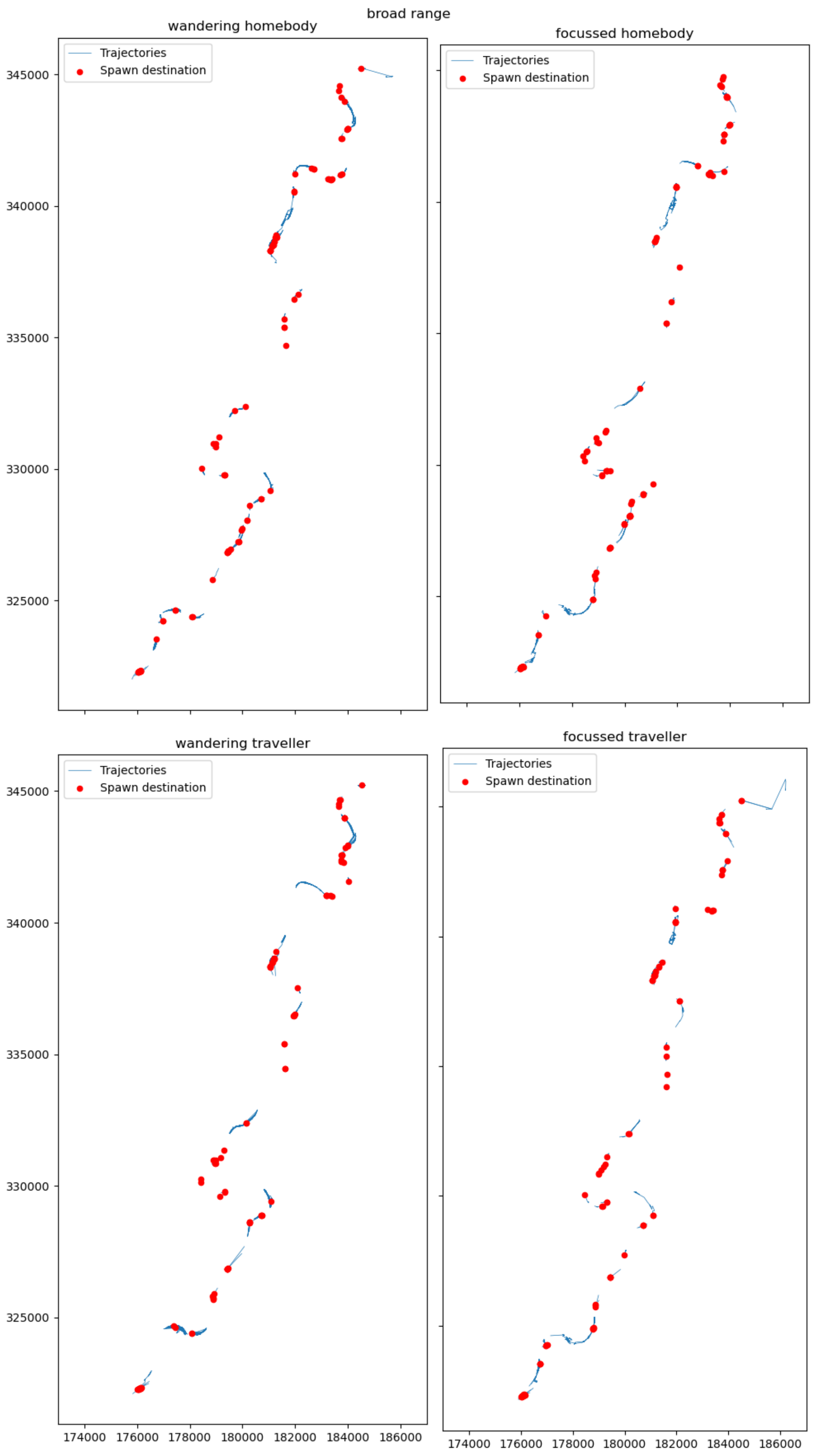

Figure 5.10: Trajectories and destination for spawning by the barbel, with the broad range of spawning preference. Tracks of all barbels are plotted, with thicker lines representing more often used paths.Consult Table 4.2 for specific parameterisation of the fish behaviour definitions.

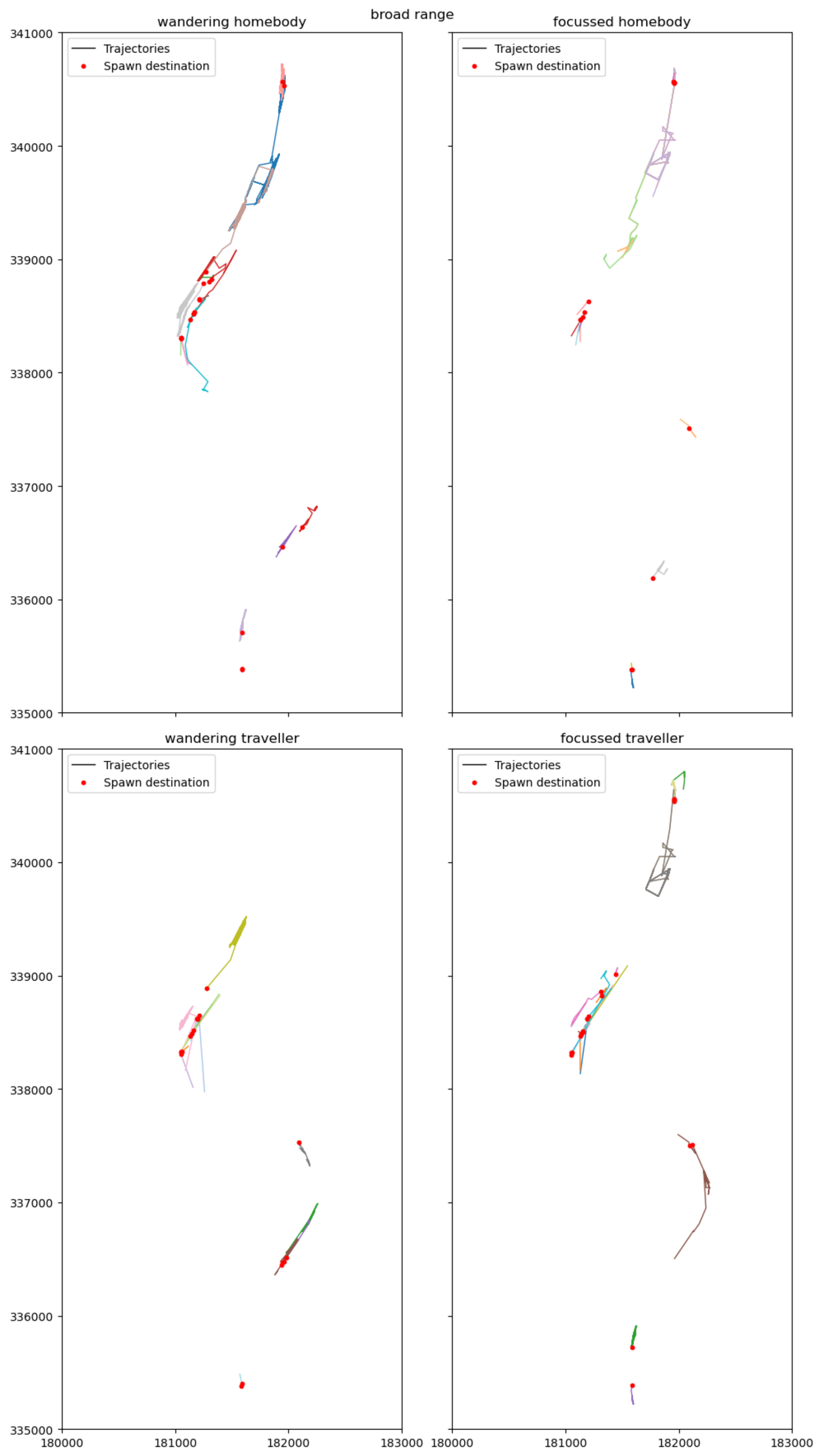

Figure 5.11: A zoom in of trajectories and destination for spawning by the barbel, with the broad range of spawning preference. Each distinct coloured line represents the track of a single group of barbels.Consult Table 4.2 for specific parameterisation of the fish behaviour definitions.

5.3 Spawning success as a function of fish parameterisation

The model shows sensitivity to barbel movement attitude, exploration distances and their range of preference (Figure 5.12). The model is most sensitive to movement attitude and range of preferences, with differences of up to 40 % in spawning success. This difference is most prevalent during the first month of the model run. Spawning success rates increase rapidly in the initial days after the model starts. The movement behaviour of the barbel significantly influences spawning success, with wandering barbels achieving higher success rates than focussed barbels. In July, spawning success stagnates for all populations. Very successful populations, such as the wandering traveller with a broad range of preference, experience stagnation already before. Populations reaching a spawning success of 80% only increase in spawning success slightly afterwards, up until a maximum of approximately 90 to 100 %.

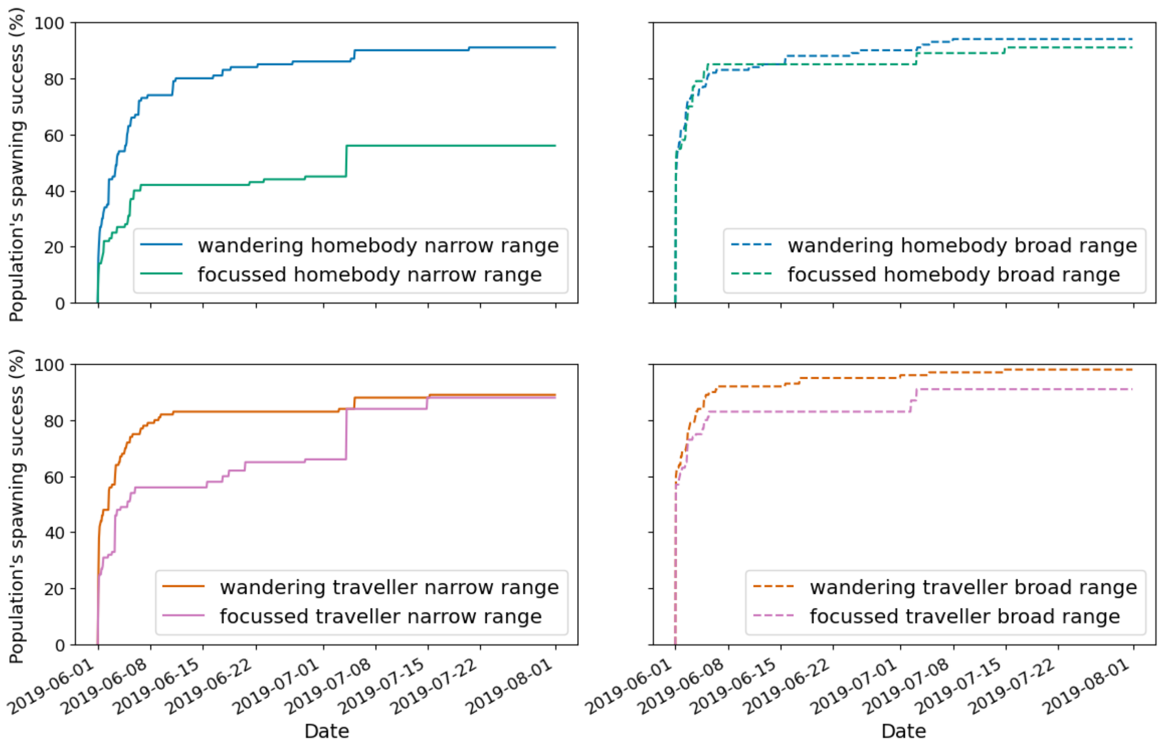

Figure 5.12: Spawning success over time, considering different types of barbel movement attitude. Consult Table 4.2 for specific parameterisation of the fish behaviour definitions.

The maximum daily exploration distance significantly impacts barbel spawning success, particularly when other behavioural factors are not ideal (Figure 5.13. For focussed barbels with a limited preference range, a larger exploration distance greatly enhances spawning success. Conversely, for wandering barbels, whether they are homebodies or travellers has minimal effect. Both focussed traveller and focussed homebody populations increase their spawning success substantially around the 4th of July.

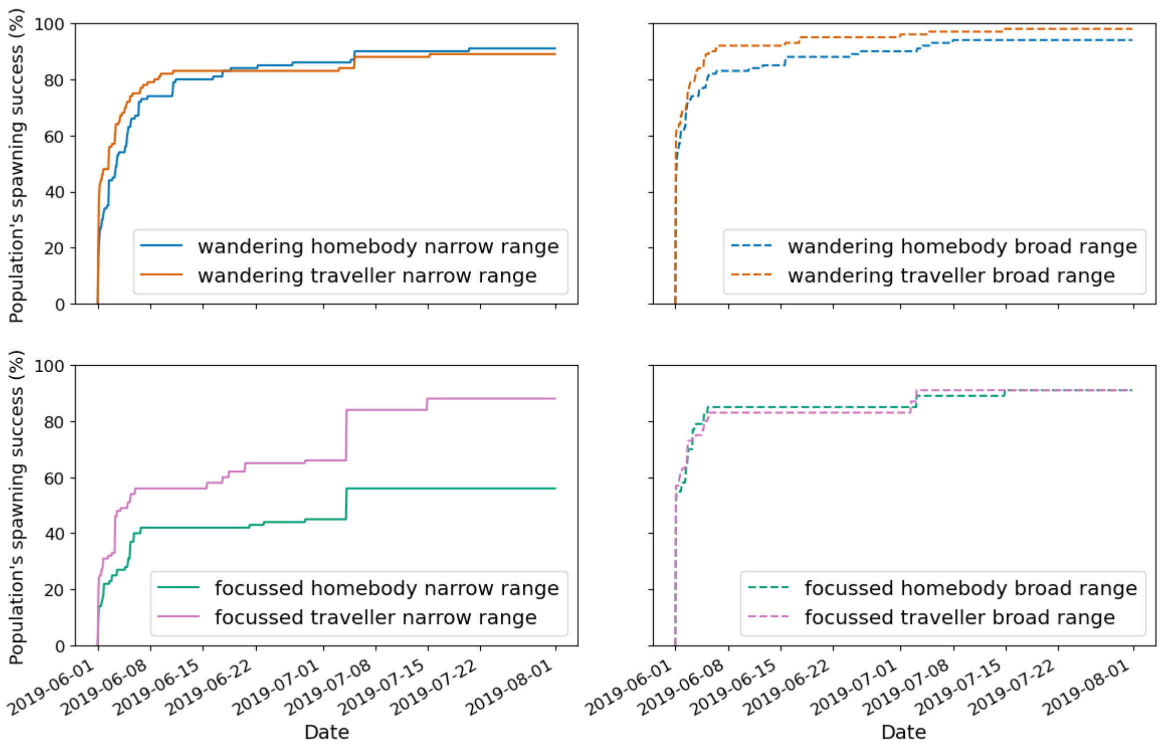

Figure 5.13: Spawning success over time, considering different barbel exploration distances. Consult Table 4.2 for specific parameterisation of the fish behaviour definitions.

Figure 5.14 shows that wandering barbels are relatively insensitive to varying ranges of spawning preferences, unlike focussed barbels. Focussed homebody barbels experience a significant difference in spawning success, with up to a 40% variation. An increase in spawning success around the 4th of July can be observed among barbels with a narrow range of preference. This co-occurs with a decrease in the number of patches and a slight increase in average patch size 5.4, as well as co-occurs with a peak in spawn area 5.1. There is no distinct peak in the spawning area for barbels with a broader range of preference during this time 5.1, who show no increase in spawning success.

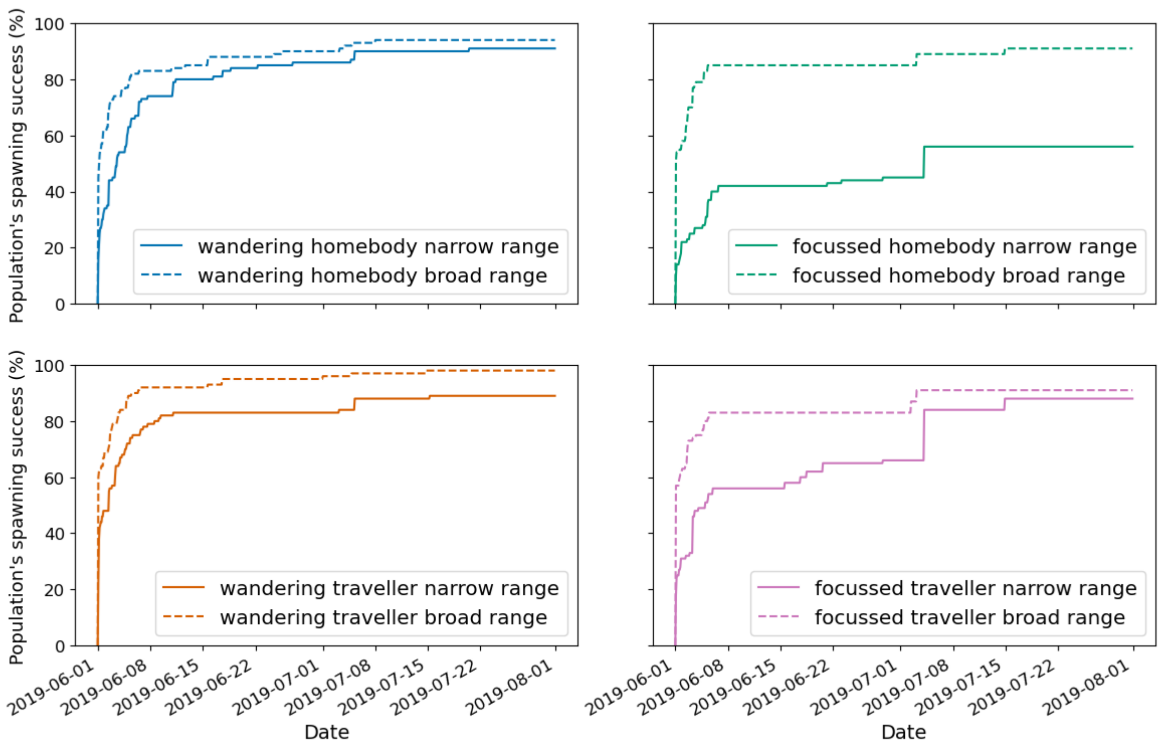

Figure 5.14: Spawning success over time, considering broader or initial ranges of spawning preferences by the common barbel. Consult Table 4.2 for specific parameterisation of fish behaviour definitions.

Barbels which have a preference for water depths as in Wortelboer et al. (2020), find spawning grounds fast, as can be derived from Figure 5.15.

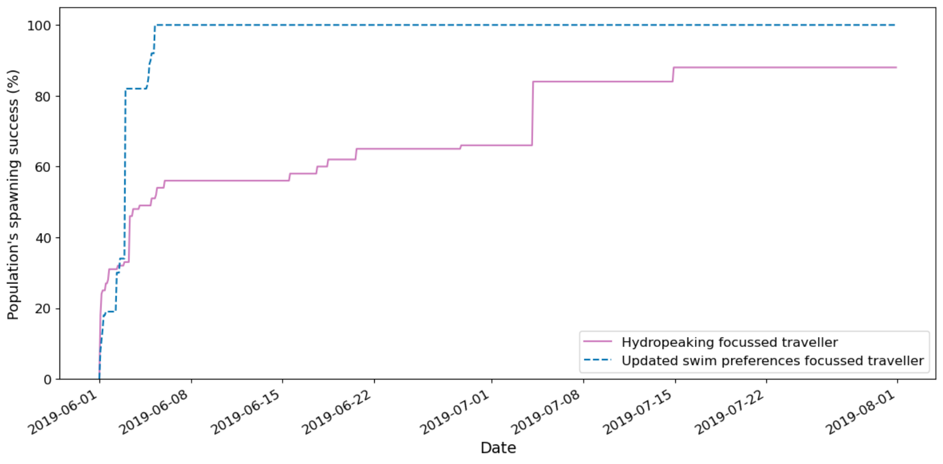

Figure 5.15: Spawning success over time, considering a new swimming preference by the common barbel.

5.4 The effect of hydropeaking on fish movement and spawning success

5.4.1 Spawning success

The filtered flow regime, representing a scenario without hydropeaking, results in higher spawning success rates, particularly in June (see Figure 5.16). This regime may contribute to up to a 30% increase in spawning success. However, the differences between non-hydropeaking and hydropeaking flow regimes diminishes in July.

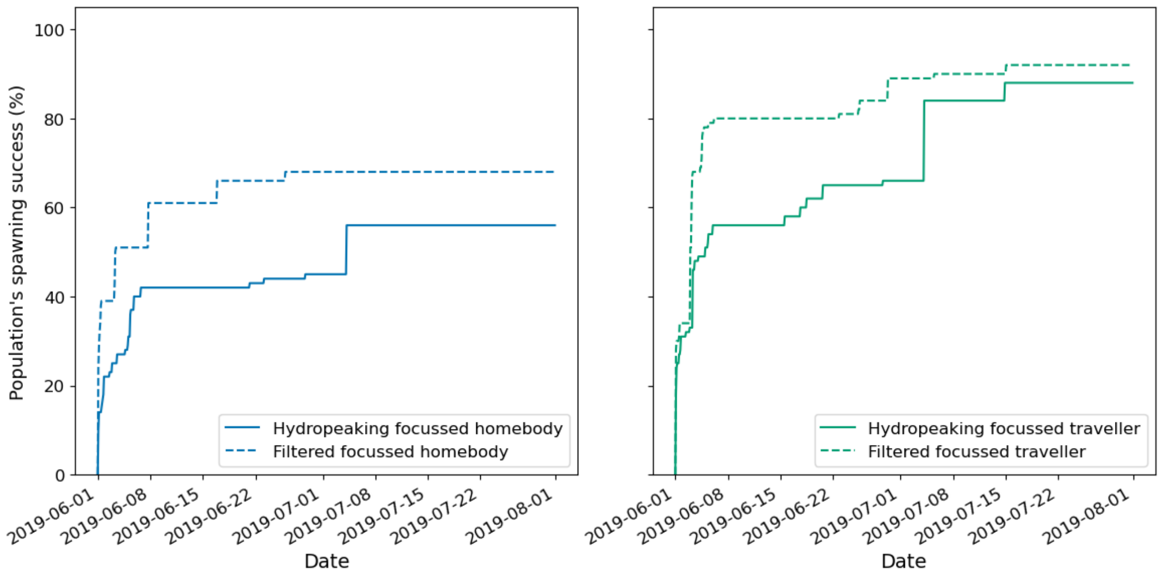

Figure 5.16: Spawning success for barbels under conditions of hydropeaking and a filtered flow regime. Left graph shows a focussed homebody population spawning success, right graph shows the spawning success for a population consisting of focussed travellers.

5.4.2 Movemodes

Barbels in a non-hydropeaking flow regime spend significantly less time in movement modes transitioning from uninhabitable to habitable areas. They use fewer movements overall before spawning (see Figure 5.17). Barbels in filtered flow regimes do not end up on unswimmable areas as often as barbels amidst a hydropeaking flow regime, as indicated by the relatively smaller fraction of 'nearest' movement used during a filtered flow regime. This movement mode was employed when barbels were in not swimmable area. Their more effective movement use can also be derived from Figure B.2.

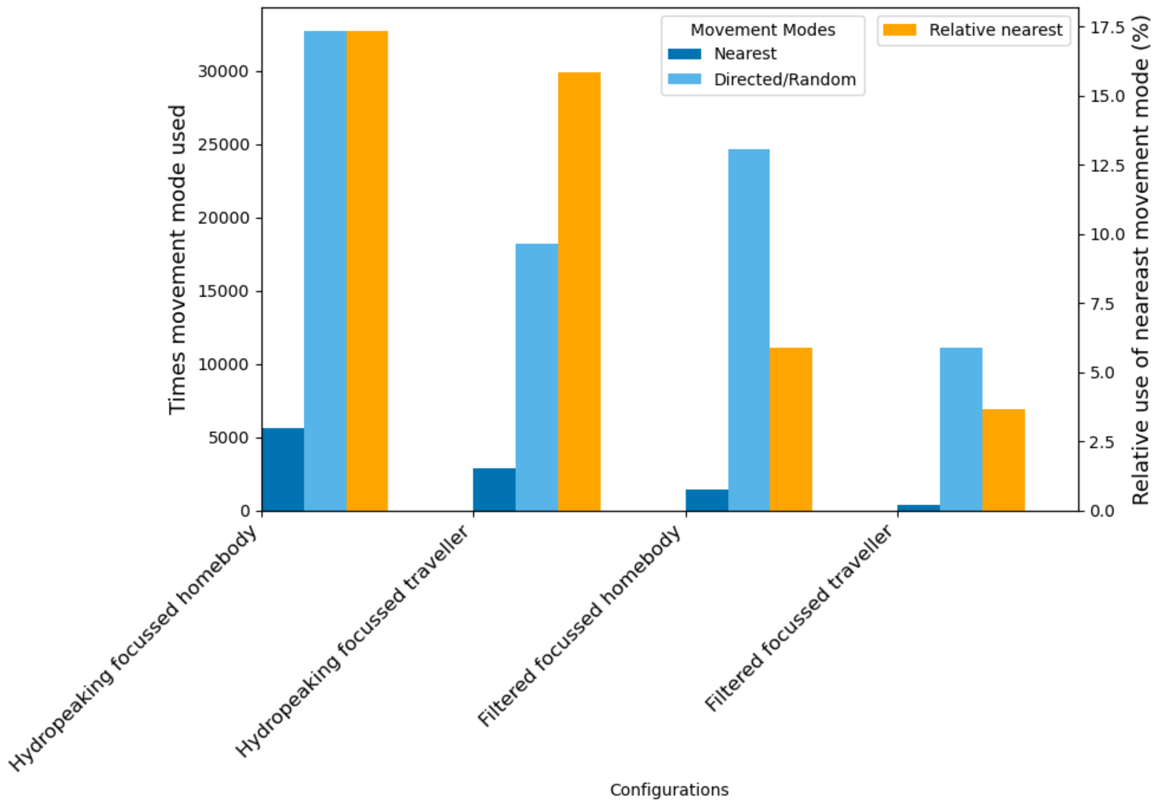

Figure 5.17: Move modes used by barbels, either experiencing a hydropeaking or a filtered flow regime. The sum of blue bars represents the movements used by the population which has not spawned yet. The orange bar represents the relative use of the 'nearest' function.

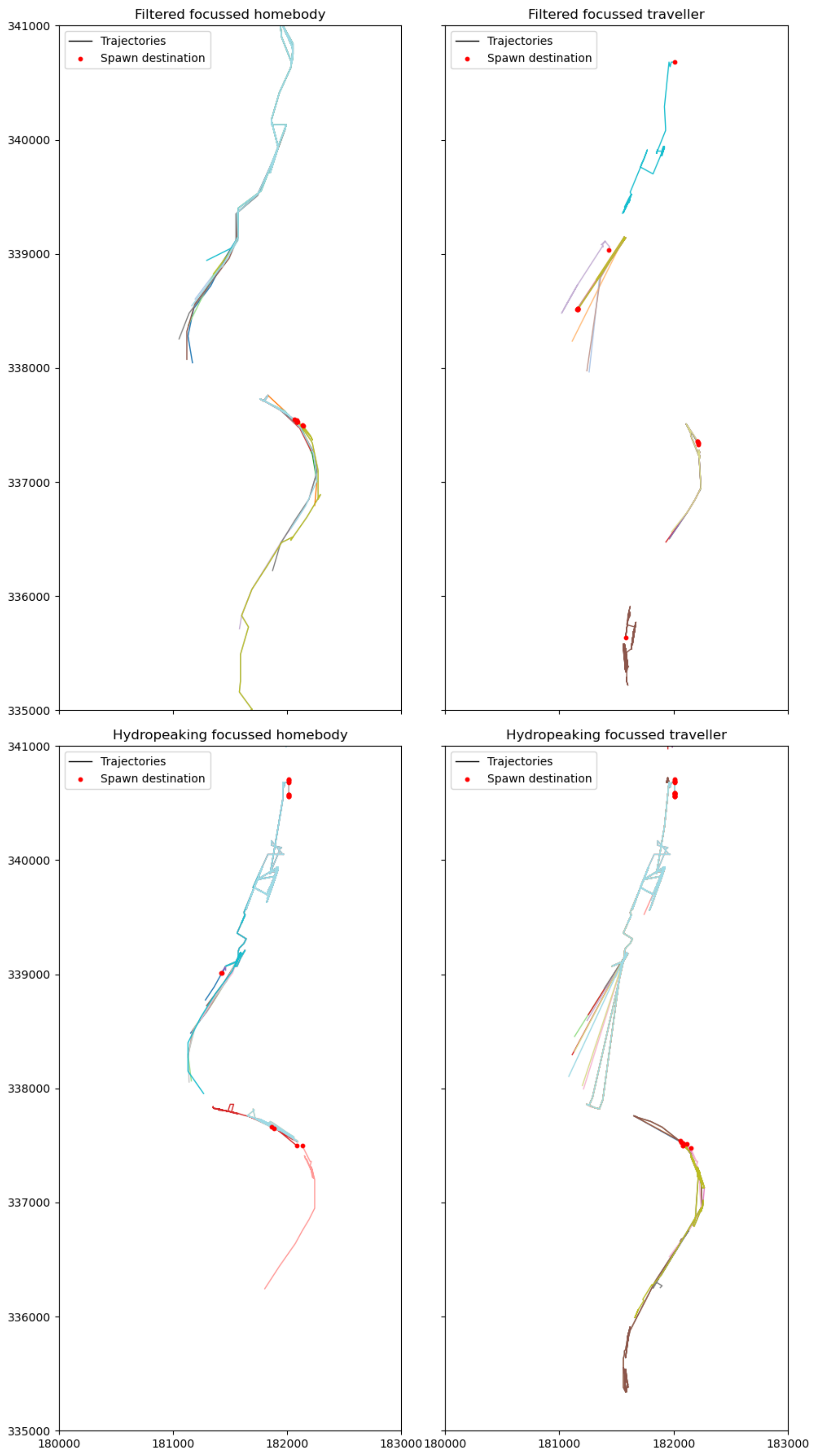

Figure 5.18: A zoom in of trajectories and destination for spawning by the barbel with different flow regimes. Each distinct coloured line represents the track of a single group of barbels.

6 Discussion

The primary objective of this thesis was to extend and exemplify Campo's functionality in representing and understanding the habitat availability for the common barbel under conditions of habitat fragmentation, specifically in the Common Meuse. This was achieved by examining how field parameters such as water depth and flow velocity influence the barbel's population dynamics when modelling an agent's response to the environment as a field. This section will discuss the implementation of the model, the barbel's population dynamics in various flow regimes, and a comparison to conventional habitat suitability models.

6.1 Barbel population dynamics amidst various flow regimes

The analysis of the hydropeaking flow regime and its impact on movement of the barbel during spawning time provides several key insights.

Habitat for swimming by adults and spawning are defined by a set range of preferences for flow velocities and water depths (see Table 4.3). These parameters are related to the discharge and the bed profile, as $Q = u * h$, in which Q is water depth, u is flow velocity and h is water depth. This makes these habitats directly related to the flow regime. Higher discharges generally result in an increase in swimmable area (Fig. 5.3, as barbels are rheophilic and thrive in faster flowing, deeper waters. Spawning happens in more calm environments, which may be present under a variety of discharge scenarios (Fig. 5.2). While low discharges can provide for a more calm environment in general, otherwise dry areas may drown under higher discharge regimes, providing for low flow velocities and water depths as well.

The area suitable for spawning remains more or less constant for different discharges amidst different flow regimes (Fig. 5.2), indicating that the physical presence of spawning sites is not directly affected by the fluctuating flow conditions. However, spawning success is significantly higher under non-hydropeaking conditions (Fig. 5.16). This suggests that other factors, beyond mere spawning habitat availability, plays a crucial role in enhancing reproductive outcomes under variable flow regimes. An increase in habitat does not directly result in increased connectivity. While the number of habitat patches increases with higher discharge (Fig. 5.5) and habitat availability (Fig. 5.3), the relationship between patch size and discharge is insignificant (Fig. 5.6). Simply increasing habitable area does not enhance connectivity between these patches. From Table 5.1 we can derive that though spawning area may be equally present during similar discharge scenarios, spawn area presence fluctuates a lot more over time in a hydropeaking scenario.

The properties of habitable area, as outlined in the paragraphs above, are not linearly related to spawning success. Though a broad range of preferences for spawning habitat resulted in approximately 6 times more spawning area than under narrow ranges of preferences (Fig. 5.2), this increase in spawning habitat only lead up to an increase in spawning success of maximum 40%, considering the hydropeaking flow regime (Fig.5.14). Comparing the filtered flow regime to the hydropeaking flow regime with a narrow range of preference, we see how amidst a filtered flow regime, the swimmable area is generally more abundant, but not as significant as the increase in spawning habitat by a broader range of preference. However, magnitudes of 40% increase in spawning success can be found considering a filtered flow regime compared to the hydropeaking flow regime with a narrow range of preference. The higher spawning success is specifically significant during June (Fig. 5.16), when habitat fragmentation measures are also smaller compared to the hydropeaking flow regime (Fig. 5.4).

The increased spawning success under filtered flow regime conditions can furthermore be attributed to the barbel's behavioural orientation in a stable versus changing environment. Barbels in a filtered flow regime spend less time travelling between uninhabitable and habitable areas (Fig. 5.17). Additionally, a stable spawning area reduces the frequency of back-and-forth movement by focussed barbels toward disappearing and reappearing spawning sites, enhancing efficiency.

The sensitivity analysis elucidates how assumptions about barbel behaviour affect spawning success in a similar magnitude as a different discharge regime or a broader range of spawning preference, either showing a maximum difference in spawning success of about 40%.

The results in this study once more indicate the importance of providing different microhabitats at a small scale (Stoffers, 2022; Liefveld & Jesse, 2006), especially amidst a hydropeaking flow regime (Boavida et al., 2015). The abundance of diverse morphological structures accommodates the barbel's needs during different life stages, and, when existing connected and in proximity to each other, this allows for easy transitioning between different habitats (Boavida et al., 2015). Boavida et al. (2015) showed ramping of the hydropeaking signal is site specific, with a more diverse river-bed accommodating alternative habitat for the Iberian barbel to move to when peak or base flows occurred. Heterogeneity drives survival of species in general (Stoffers, 2022), but also the connectivity of these habitats is relevant.

It is important to note that upstream discharge filtering is not equivalent to filtering each downstream water depth and flow velocity cell. Filtering of the hydropeaking signal occurs across all cells on a raster that describes flow velocity and water depth responses to discharge. These local parameters can vary spatially and temporally, often responding non-linearly to filtered discharge signals. Therefore, filtering a field differs from filtering an upstream discharge measure. This explains why for the same discharge levels, habitable swimmable area is larger (Fig. 5.3), indicating filtering the downstream grid cells of the study area is more sensitive to filtering, resulting in a more moderate and therefore more accommodating habitat, than would be present when filtering discharge.

Furthermore, assumptions were made about the range of preferences the barbel has about its environment. The range of preference of water depth for the adult barbel as in Table 4.3 did not comply with the literature, which is assumed to be between 0.3 and 100 metres by Wortelboer et al. (2020), instead of 0.1 and 1 metres, as in this model. Assumptions about preferences of the barbel affect outcomes significantly, as is illustrated by the barbels with a broad range of preference for spawning area. Though a barbel is likely to swim under these conditions, as other parameters are assumed in the first place as well, it is likely that in a simulation with this wider range of swimming preference of the barbel, it will be easier for the barbels to find their spawning grounds.

The observed stagnation in spawning success after a certain number of individuals have spawned, which occurs across all model run scenarios, suggests that some barbels are inevitably trapped in their habitat, unable to reach spawning areas. This indicates that extending the model run would not yield additional insights, especially considering the spawning season only lasts 2 to 3 months. Since barbels are placed randomly at the beginning of each run, varying numbers of barbels may find themselves in these 'trapped' habitats. However, it is possible that this trapping is specifically associated with the dry flow regime present during the latter part of the model run, in July. Altering the sequence of flow regimes might change the rate at which each barbel locates spawning grounds, but it is likely to result in a similar overall spawning success.

In conclusion, the hydropeaking flow regime impacts barbel spawning success more through behavioural adaptations and connectivity rather than just habitat availability. Understanding these dynamics can lead to better management and conservation strategies for barbel populations in fluctuating river systems.

6.2 Comparison to conventional habitat suitability models

Ecological niche theory is founded by the idea that individual species only thrive within definite ranges of environmental conditions (Hirzel & Le Lay, 2008). When the fitness of individuals can be described on the basis of these quantitative measures of the environment, this can be modelled and mapped, providing insight in the ability of an environment to sustain a population. Habitat suitability models are useful when assessing similarities or differences between species, or to potentially explain the absence of species. However, traditional habitat suitability models fail to address niche issues such as interactions, community, and evolution or movement of species (Hirzel & Le Lay, 2008). Agent-integration aims to address these niche issues. This section outlines the shortcomings of conventional habitat suitability and how the barbel model paves the way for a more thorough understanding of species-environment interaction.

The study by Liefveld and Jesse (2006), emphasises the necessity of a minimum discharge for the survival of barbels across all life stages. It was indeed found that higher discharges lead to higher availability of swimmable area. Though in general, spawning area slightly decreases with higher discharges, the relationship was neither steep nor very significant, indicating that a wide variety of spawning area may be present considering different discharge regimes. Furthermore, none of the populations show a significant increase in spawning success during the dry flow regime in July. A minimum discharge may thus indeed be beneficial for the survival and success of the adult barbel searching for spawning habitat.

However, our findings mostly highlight the complex interplay between movement, spawn area availability, swim area availability, and connectivity. In this study, not all barbels are able to find spawning grounds, despite the availability of sufficient spawning area in all flow regimes. This finding challenges the conventional approach and suggests a more nuanced understanding of habitat requirements. Further proof of this argument is provided by the significant different outcomes in spawning success for barbels portraying different types of behaviour, which shows the importance of incorporating animal movement behaviour under various discharge regimes. As barbel behaviour is neglected in conventional suitability models, this shows that agent-based integration in field-based modelling is an essential step in assessing suitability. This is especially true when different life stages require different habitats, necessitating movement for survival or reproduction. This also questions the barbel, or any species, in its consideration as a flagship species to evaluate habitat requirements, as its species' specific behaviour affect its access to habitat. A general field-based analysis does not incorporate any of these behavioural differences.

The spatial distribution of the barbel showed similar patterns across different behaviours. Though spatial habitat suitability modelling is a good and simple measure to describe the presence or absence of barbels, it cannot predict movement or reproduction, especially in a temporally changing system. Agents may also demonstrate exhaustion or inability to find spawning grounds.

6.3 Practical application of the barbel model

This thesis demonstrated the accessibility of agent-based modelling while maintaining high geospatial data resolution and manipulation capabilities. Before we are able to implement barbels as agents navigating their environment, more research about their behaviour to represent them is needed. In an abductive approach, through fish tracking and subsequent comparison of their tracks with movement paths as in this study, we can generalise drivers for different types of movement behaviour and apply it to the organism of interest (Railsback, 2001b). A similar approach can be found in Nathan et al. (2022), showing the potential of the method. The approach in this thesis could provide an additional validation of the generalisations found by comparing them with the observed movement patterns.

Addressing an agents' movement response to its environment, as in this thesis, is specifically valuable when considering species with different habitat requirements over their lifetime, a system with a significantly changing environment over time, or a fragmented habitat. Incorporating the self-organizing aspects of populations that may be affected by an intervention, integrating the agent-based view is useful to inform decision-making in complex systems (Hammond, 2015; Le Page et al., 2017).

6.4 Challenges and future improvements

The Campo framework was extended by several operations tailored to the needs of representing the barbel population in the Common Meuse. These operations resulted in patterns that could be observed in nature. To accurately represent the sensitivity of the individual barbel to their environment and enhance the framework's field-agent integration, the sensing of the environment by the agent was added as a functionality.

6.4.1 Field-agent interaction in Campo

Field-agent interaction was implemented by aggregating window variables around each agent. The proximity effect was initially modelled using a focal operation with a window size determined by an internal property of the agent. However, this approach did not differentiate variables based on their exact proximity within the window. To address this, multiple queries of field variables at varying window sizes can be used to approximate proximity effects.

A more precise but computationally intensive method involves creating a distance raster centred on each agent. This raster assigns values based on the agent's distance from each point, requiring a unique raster for every agent. While this provides detailed proximity data, it can be computationally expensive, especially in large domains with many agents. This requires the formulation of separate field-agents for each agent in the simulation. The extent of the field-agent would be similar to the domain of the study area, one for each agent present in the study area.

A hybrid approach, where a distance raster is calculated for each agent, but only up to the extent of a selective window, could balance accuracy and computational feasibility. This approach assumes that not all environmental elements are equally relevant for decision-making by individual agents, but proximity to certain key elements must be considered (Schmitz et al., 2013). This could be achieved by the creation of a moving window field-agent. By focusing on key elements within a limited, moving window around each agent, the system can dynamically adjust its focus area. This selective approach reduces computational load while ensuring agents have the necessary information to make informed decisions. Currently, the framework of Campo doesn't support a domain that changes over time. When something is defined as a window field agent specifically, the framework can override the default domain settings to accommodate the changing domain of the window field-agent. The moving window field-agent allows the agent to consider distance in its decision-making process without overwhelming computational resources. The framework remains efficient because it processes only relevant information within the moving window.

A moving field-agent would also be a useful representation when a moving organism is internally differently affected by the spatial variability of its environment. A field-agent may be a more accurate representation of such a moving organism because its response to the environment may differ spatially. The body temperature distribution of a whale may be altered differently on a spatial scale when large spatial gradients are present in water. Herein, it is relevant to consider both the relevant spatial resolution of the moving organism and of its environment. When the relationship is reverse, a field-agent representation of a moving organism may also be useful, with the fin of the whale affecting e.g. flow velocity more than its head would.

In our barbel model, the surrounding environment's window was defined around the grid cell the agent was located in, with window sizes rounded down to avoid partial cell inclusion. Future improvements could incorporate fractional cell coverage to refine this representation (Pullar, 2001).

In a scenario where moving agents are larger than a cell on a raster, the window size of the sensing of the environment has to take into account the magnitude of the moving agent. If needed, the extent of the agent on both dimensions could simply be added to the window in correspondence to the dimensions of the extent of the agent.

6.4.2 Movement in Campo

Movement was a key element when simulating a barbel's spawning success. Peaks in spawn area availability do co-occur with increases in spawning success, but also with high connectivity rates among habitats. Results for different fish parameterisation revealed that spawning success varies among barbel agents with different movement modes, underscoring the importance of accommodating these different movement modes within the modelling framework. Despite the variations in spawning success, the resulting spatial distribution patterns in all scenarios aligned well with natural observations.

Literature emphasises the significance of different movement modes in representation of fish behaviour, specifically when considering a hydropeaking flow regime. For instance Gutmann Roberts (2018) showed exploited situations and poor habitat conditions influence swimming activity in barbel. Alternative behaviour under conditions of hydropeaking has specifically been addressed by Costa et al. (2019), who observed higher swimming velocities during higher flow velocities, as well as drifting of juveniles. A decreased swimmable area causes fish to move rapidly from one place to another (Costa et al., 2019). Within fishes' population, considerable individual variation in movement was present (Costa et al., 2019). Incorporating these behaviours into the model can enhance its accuracy. I propose developing movement modes based on environmental values. For instance, agents could be displaced to neighbouring cells based on the direction of flow vectors. The timing of this displacement depends on the flow velocity, deciding at which timing the barbel will arrive and thus at what time the environment should be updated.

Theories formulating required timesteps to distinguish signals such as movement and environmental change (Shannon, 1949) may result in insufficient temporal resolution to capture the change that happens to an agents' environment when the agent moves. Longer timesteps may overlook rapid habitat changes, leading to unrealistic movement patterns (Ryan et al., 2004). Conversely, shorter timesteps may result in a too assertive representation of the barbel than is likely, leading to the accumulation of errors by constantly estimating its destination (Ryan et al., 2004).

6.4.3 The barbel model

Several other factors could potentially influence barbel population dynamics and warrant further investigation. The accumulation of fine sediments in the Common Meuse is hypothesized to degrade habitat for species relying on gravel (Boon et al., 2024). Specifically for the barbel, sediment colmation could reduce the suitability of habitat for reproduction (Nagel et al., 2020). Mapping grainsizes and the extent of sediment colmation and incorporating it as a spatial field for habitat suitability mapping could improve representation. Pollution, such as PCB contamination, poses significant risks to the health and survival of barbel populations, specifically having a negative effect on reproduction (Hugula et al., 1995).

Future research could focus on the sensitivity of individual barbels within a population that includes replacement, investigating the direct effects of connectivity and spawn area availability under various settings. This could provide deeper insights into the population dynamics and reproductive success under different environmental conditions. Buffering the field by defining a risk zone could further refine the model's accuracy in predicting barbel behaviour.

7 Conclusion

To accurately represent habitat dependency and availability, Campo's framework was extended to include field-agent interactions and environmental sensing by agents. By incorporating spatial awareness into the barbel's movement decision-making, the results underscore the critical importance of understanding barbel behaviour when assessing their spawning capabilities.

Although habitat fragmentation or spawnarea availability in the Common Meuse ecosystem was not observably exacerbated by hydropeaking signals, spawning success was higher during non-hydropeaking scenarios. Under a hydropeaking flow regime, barbel agents encounter continuously changing environments, necessitating extensive movement to stay within habitable areas. Providing different microhabitats through heterogenous morphology is imperative for the survival of species with different habitat requirements throughout their lifetime, especially amidst a hydropeaking flow regime. As discharge levels increase, the availability of spawning habitats generally remains constant, while the overall habitable area expands, facilitating movement to spawning grounds. This indicates the detrimental aspect of low discharges, but as this thesis mostly points at the complex interplay between habitat availability and barbel behaviour, it challenges the need for a minimum discharge as was stressed by Liefveld and Jesse (2006).

Conventional habitat suitability models are a useful tool to understand how habitat availability can limit or promote species occurrence but fall short in accounting for adaptive responses, especially when considering temporally changing environmental factors and their impact on species migrating to find suitable habitats, such as a barbel amidst a hydropeaking flow regime. The significant difference in spawning success among barbels with varying behaviours highlights the crucial role of movement behaviour in spawning success. Ultimately, the success and survival of the barbel population hinge on their movement modes and capabilities, as well as fluctuations in habitat availability over time.

Several operations were developed which have increased the accessibility and representation capabilities of ecosystems in Campo. With these operations only requiring properties or propertysets as defined as concepts in Campo as an input, anyone familiar with the framework could use them. However, agent integration does require generalisation of behaviour of the organism of interest. While maintaining high resolution geospatial data manipulation capabilities, the accessible and semantically intuitive nature of the model paves the way for the assessment of anthropogenic interventions in complex systems. Field-agent integration and movement functions could further be extended in the Campo framework to accommodate more complex responses of the agent to the environment while maintaining computational efficiency. This will provide the domain expert with more tools to test or illustrate the adaptive response of species to their environment. Environmental issues, such as colmation or pollution, and its effect on barbel population dynamics, could be further assessed. However, theory on barbel movement is needed first, in which this model may also play a validating role.

References

Athanasiadis, I. N., & Villa, F. (2013). A roadmap to domain specific programming languages for environmental modeling: key requirements and concepts. In *Proceedings of the 2013 acm workshop on domain-specific modeling* (pp. 27–32). doi: https://doi.org/10.1145/2541928.2541934

Baras, E. (1998). Selection of optimal positioning intervals in fish tracking: an experimental study on barbus barbus. In *Advances in invertebrates and fish telemetry: Proceedings of the second conference on fish telemetry in europe, held in la rochelle, france, 5–9 april 1997* (pp. 19–28).

Beland, K. F., Jordan, R. M., & Meister, A. L. (1982). Water depth and velocity preferences of spawning atlantic salmon in maine rivers. *North American Journal of Fisheries Management*, *2*(1), 11–13.

Bjørnås, K. L., Railsback, S. F., Calles, O., & Piccolo, J. J. (2021). Modeling atlantic salmon (salmo salar) and brown trout (s. trutta) population responses and interactions under increased minimum flow in a regulated river. *Ecological Engineering*, *162*, 106182. doi: https://doi.org/10.1016/j.ecoleng.2021.106182

Blacow, C. (2023, September). *Hydrodynamic characterisation for habitat mapping* [Guided Research].

Boavida, I., Santos, J. M., Ferreira, T., & Pinheiro, A. (2015). Barbel habitat alterations due to hydropeaking. *Journal of hydro-environment research*, *9*(2), 237–247. doi: https://doi.org/10.1016/j.jher.2014.07.009

Bolton, E. R., & Berglund, E. Z. (2023). Agent-based modeling to assess decentralized water systems: Micro-trading rainwater for aquifer recharge. *Journal of Hydrology*, *618*, 129151. doi: https://doi.org/10.1016/j.jhydrol.2023.129151

Boon, A., Blacow, C., Kleinhans, M. G., & Straatsma, M. W. (2024). Hydrodynamic modelling of habitat availability in the common meuse. *TOMORROW'S RIVERS*, 21.

Brauer, C. J., & Beheregaray, L. B. (2020). Recent and rapid anthropogenic habitat fragmentation increases extinction risk for freshwater biodiversity. *Evolutionary applications*, *13*(10), 2857–2869. doi: https://doi.org/10.1111/eva.13128

Britton, J. R., & Pegg, J. (2011). Ecology of european barbel barbus barbus: Implications for river, fishery, and conservation management. *Reviews in Fisheries Science*, *19*(4), 321-330. doi: 10.1080/10641262.2011.599886

Carré, C., & Hamdani, Y. (2021). Pyramidal framework: Guidance for the next generation of gis spatial-temporal models. *ISPRS International Journal of Geo-Information*, *10*(3). doi: https://doi.org/10.3390/ijgi10030188

Casas-Mulet, R., Saltveit, S. J., & Alfredsen, K. (2015). The survival of atlantic salmon (salmo salar) eggs during dewatering in a river subjected to hydropeaking. *River Research and Applications*, *31*(4), 433–446. doi: https://doi.org/10.1002/rra.2827

Conradt, L. (2012). Models in animal collective decision-making: information uncertainty and conflicting preferences. *Interface focus*, *2*(2), 226–240. doi: https://doi.org/10.1098/rsfs.2011.0090

Costa, M., Fuentes-Pérez, J., Boavida, I., Tuhtan, J., & Pinheiro, A. (2019). Fish under pressure: Examining behavioural responses of iberian barbel under simulated hydropeaking with instream structures. *PLoS One*, *14*(1). doi: https://doi.org/

10.1371/journal.pone.0211115

Courant, R., Friedrichs, K., & Lewy, H. (1967). On the partial difference equations of mathematical physics. *IBM Journal of Research and Development*, *11*(2), 215-234. doi: 10.1147/rd.112.0215

Cova, T. J., & Goodchild, M. F. (2002). Extending geographical representation to include fields of spatial objects. *International Journal of geographical information science*, *16*(6), 509–532. doi: https://doi.org/10.1080/13658810210137040

Crooks, A. T., & Castle, C. J. (2011). The integration of agent-based modelling and geographical information for geospatial simulation. In *Agent-based models of geographical systems* (pp. 219–251). Springer. doi: https://doi.org/10.1007/978-90-481-8927-4_12

de Bakker, M. P., de Jong, K., Schmitz, O., & Karssenberg, D. (2017). Design and demonstration of a data model to integrate agent-based and field-based modelling. *Environmental Modelling I& Software*, *89*, 172-189. doi: https://doi.org/10.1016/j.envsoft.2016.11.016

de Jong, K., & Karssenberg, D. (2019). A physical data model for spatio-temporal objects. *Environmental Modelling I& Software*, *122*, 104553. doi: https://doi.org/10.1016/j.envsoft.2019.104553

Deerenberg, C., Machiels, M., Van Kooten, T., Van der Sluis, M., & Paijmans, A. (2012). *Beoordelingssystematiek beschermde vissoorten van de grensmaas* (Tech. Rep.). IMARES.

Deltares. (2014). Users manual.(2014). *Simulation of multi-dimensional hydrodynamic flows and transport phe-nomena, including sediments.*

Deltares. (2024). *Xugrid.* (https://github.com/Deltares/xugrid/)

Dudley, R. K., & Platania, S. P. (2007). Flow regulation and fragmentation imperil pelagic-spawning riverine fishes. *Ecological Applications*, *17*(7), 2074–2086. doi: https://doi.org/10.1890/06-1252.1

Duizendstra, H. (2001). Determination of the sediment transport in an armoured gravel-bed river. *Earth Surface Processes and Landforms*, *26*(13), 1381–1393. doi: https://doi.org/10.1002/esp.302

Fahrig, L. (2003). Effects of habitat fragmentation on biodiversity. *Annual review of ecology, evolution, and systematics*, *34*(1), 487–515. doi: https://doi.org/10.1146/annurev.ecolsys.34.011802.132419

Favis-Mortlock, D. (2013). 1.14 systems and complexity in geomorphology. *Treatise on Geomorphology, edited by: Shroder, JF*, 257–270. doi: https://doi.org/10.1016/B978-0-12-374739-6.00014-2

Garshelis, D. L. (2000). Delusions in habitat evaluation: measuring use, selection, and importance. *Research techniques in animal ecology: controversies and consequences*, *2*, 111–164.

GDAL/OGR contributors. (2020). GDAL/OGR geospatial data abstraction software library [Computer software manual]. Retrieved from `https://gdal.org`

Goodchild, M. F., Yuan, M., & Cova, T. J. (2007). Towards a general theory of geographic representation in gis. *International journal of geographical information science*, *21*(3), 239–260. doi: https://doi.org/10.1080/13658810600965271

Grimm, V., & Railsback, S. F. (2005). *Individual-based modeling and ecology.* Princeton university press. doi: https://doi.org/10.1515/9781400850624

Gutmann Roberts, C. (2018). *Population ecology and behaviour of european barbel* *barbus barbus, a recreationally important, translocated fish.* (Unpublished doctoral dissertation). Bournemouth University.

Gutmann Roberts, C., Hindes, A. M., & Britton, J. R. (2019). Factors influencing individual movements and behaviours of invasive european barbel barbus barbus in a regulated river. *Hydrobiologia*, *830*, 213–228. doi: https://doi.org/10.1007/s10750-018-3864-9

Hamdani, Y., Thibaud, R., & Claramunt, C. (2019). A hybrid temporal gis representation for coastal dynamics. In *Proceedings of the 27th acm sigspatial international conference on advances in geographic information systems* (p. 584–587). New York, NY, USA: Association for Computing Machinery. doi: https://doi.org/10.1145/3347146.3359357

Hamdani, Y., Thibaud, R., & Claramunt, C. (2021). A hybrid data model for dynamic gis: application to marine geomorphological dynamics. *International Journal of Geographical Information Science*, *35*(8), 1475–1499. doi: https://doi.org/10.1080/13658816.2020.1829628

Hammond, R. A. (2015). Considerations and best practices in agent-based modeling to inform policy. In *Assessing the use of agent-based models for tobacco regulation.* National Academies Press (US).

Hirzel, A. H., & Le Lay, G. (2008). Habitat suitability modelling and niche theory. *Journal of applied ecology*, *45*(5), 1372–1381. doi: https://doi.org/10.1111/j.1365-2664.2008.01524.x

Huet, M. (1949). Aperçu des relations entre la pente et les populations piscicoles des eaux courantes. *Schweizerische Zeitschrift für Hydrologie*, *11*, 332–351.

Hugula, J., Philippart, J.-C., Kremers, P., Goffinet, G., & Thomé, J. (1995). Pcb contamination of the common barbel, barbus barbus (pisces, cyprinidae), in the river meuse in relation to hepatic monooxygenase activity and ultrastructural liver change. *Netherland Journal of Aquatic Ecology*, *29*, 135–145. doi: https://doi.org/10.1007/BF02061796

Hunt, P. C., & Jones, J. W. (1974). A population study of barbus barbus (l.) in the river severn, england. *Journal of Fish Biology*, *6*(3), 269-278. doi: https://doi.org/10.1111/j.1095-8649.1974.tb04544.x

International, S. O. (2015). *Observation international.* Retrieved from `https://observation-international.org/en/`

IUCN. (2023). *The iucn red list of threatened species.* Retrieved 2024-01-18, from `https://www.iucnredlist.org`

Jeremy Mennis, R. V., & Tomlin, C. D. (2005). Cubic map algebra functions for spatio-temporal analysis. *Cartography and Geographic Information Science*, *32*(1), 17-32. doi: 10.1559/1523040053270765

Karssenberg, D. (2002). The value of environmental modelling languages for building distributed hydrological models. *Hydrological Processes*, *16*(14), 2751–2766. doi: https://doi.org/10.1002/hyp.1068

Karssenberg, D., Schmitz, O., Salamon, P., de Jong, K., & Bierkens, M. F. (2010). A software framework for construction of process-based stochastic spatio-temporal models and data assimilation. *Environmental Modelling I& Software*, *25*(4), 489-502. doi: https://doi.org/10.1016/j.envsoft.2009.10.004

Kjenstad, K. (2006). On the integration of object-based models and field-based models in gis. *International Journal of Geographical Information Science*, *20*(5), 491-509.

Retrieved from https://doi.org/10.1080/13658810600607329 doi: https://doi.org/10.1080/13658810600607329

Le Page, C., Bazile, D., Becu, N., Bommel, P., Bousquet, F., Etienne, M., … Weber, J. (2017). Agent-based modelling and simulation applied to environmental management. *Simulating Social Complexity: A Handbook*, 569–613. doi: https://doi.org/10.1007/978-3-319-66948-9_22

Le Quéré, P. A., Nistor, I., & Mohammadian, A. (2020). Numerical modeling of tsunami-induced scouring around a square column: Performance assessment of flow-3d and delft3d. *Journal of Coastal Research*, *36*(6), 1278–1291. doi: https://doi.org/10.2112/jcoastres-d-19-00181.1

Liefveld, W., & Jesse, P. (2006). *Minimale afvoer van de grensmaas: inschatting van ecologische effecten met rhasim.* Rijkswaterstaat, RIZA.

Liefveld, W., van Vliet, F., Winden, A. V., van Gogh, I., & Lensink, R. (2018). *Effectenanalyse huidige activiteiten Grensmaas* (Tech. Rep.). Culemborg: Bureau Waardenburg and Bureau Stroming.

Lieshout, F. v., Peeters, E., & Franken, R. (2003). The allier, ecological reference for the grensmaas? the macrofauna biotic community related to the ecological restoration signalling new bottle necks. *Natuurhistorisch Maandblad (Netherlands)*.

Limburg, R. (2000). Iwaco: Internationale ecologische verkenning maas (evim). *Historisch ecologische oriëntatie op het stroomgebied (fase 2a), Rijkswaterstaat Directie Limburg, Afdeling Integraal Waterbeleid, Maastricht, The Netherlands*.

Liu, Y., Goodchild, M. F., Guo, Q., Tian, Y., & Wu, L. (2008). Towards a general field model and its order in gis. *International Journal of Geographical Information Science*, *22*(6), 623-643. doi: https://doi.org/10.1080/13658810701587727

Macal, C., & North, M. (2005). Tutorial on agent-based modeling and simulation. In *Proceedings of the winter simulation conference, 2005.* (p. 14 pp.-). doi: 10.1109/WSC.2005.1574234

McLane, A. J., Semeniuk, C., McDermid, G. J., & Marceau, D. J. (2011). The role of agent-based models in wildlife ecology and management. *Ecological Modelling*, *222*(8), 1544-1556. doi: https://doi.org/10.1016/j.ecolmodel.2011.01.020

Morrison, M., & Morgan, M. (1999). Models as mediators. *Perspectives on Natural and Social Science. Cambridge University Press, Cambridge*. doi: https://doi.org/10.1017/cbo9780511660108.004

Nagel, C., Mueller, M., Pander, J., & Geist, J. (2020). Making up the bed: Gravel cleaning as a contribution to nase (chondrostoma nasus l.) spawning and recruitment success. *Aquatic Conservation: Marine and Freshwater Ecosystems*, *30*(12), 2269–2283. doi: https://doi.org/10.1002/aqc.3458

Nathan, R., Monk, C. T., Arlinghaus, R., Adam, T., Alós, J., Assaf, M., … Jarić, I. (2022). Big-data approaches lead to an increased understanding of the ecology of animal movement. *Science*, *375*(6582), eabg1780. doi: 10.1126/science.abg1780

Nilsson, C., Reidy, C. A., Dynesius, M., & Revenga, C. (2005). Fragmentation and flow regulation of the world's large river systems. *Science*, *308*(5720), 405–408. doi: https://doi.org/10.1126/science.1107887

Patterson, T. A., Thomas, L., Wilcox, C., Ovaskainen, O., & Matthiopoulos, J. (2008). State–space models of individual animal movement. *Trends in Ecology Evolution*, *23*(2), 87-94. doi: https://doi.org/10.1016/j.tree.2007.10.009

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., …

Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Penaz, M., Barus, V., Prokes, M., & Homolka, M. (2002). Movements of barbel, barbus barbus (pisces: Cyprinidae). *Folia Zoologica (Czech Republic)*, *51*(1).

Perkin, J. S., & Gido, K. B. (2011). Stream fragmentation thresholds for a reproductive guild of great plains fishes. *Fisheries*, *36*(8), 371–383. doi: https://doi.org/10.1080/03632415.2011.597666

Pullar, D. (2001). Mapscript: A map algebra programming language incorporating neighborhood analysis. *GeoInformatica*, *5*, 145–163. doi: https://doi.org/10.1023/A:1011438215225

Railsback, S. F. (2001a). Concepts from complex adaptive systems as a framework for individual-based modelling. *Ecological modelling*, *139*(1), 47–62. doi: https://doi.org/10.1016/S0304-3800(01)00228-9

Railsback, S. F. (2001b). Getting "results": The pattern-oriented approach to analyzing complex systems with agent-based models. In *Proceedings of the 4th annual swarm user group meeting: Swarmfest 2000: March 11-13, 2000, utah state university, logan, utah* (Vol. 8, p. 41). doi: https://doi.org/10.1111/j.1939-7445.2001.tb00069.x

Railsback, S. F., Harvey, B. C., Lamberson, R. H., Lee, D. E., Claasen, N. J., & Yoshihara, S. (2002). Population-level analysis and validation of an individual-based cutthroat trout model. *Natural Resource Modeling*, *15*(1), 83–110. doi: https://doi.org/10.1216/nrm/1030539098

Ravon. (2024). Barbeel. *Stichting Natuur Onderzoek Nederland*.

Rijkswaterstaat. (2024, Jan). *Waterdata.* Retrieved from `https://www.rijkswaterstaat.nl/water/waterdata-en-waterberichtgeving/waterdata`

Ryan, P., Petersen, S., Peters, G., & Grémillet, D. (2004). Gps tracking a marine predator: the effects of precision, resolution and sampling rate on foraging tracks of african penguins. *Marine biology*, *145*, 215–223. doi: https://doi.org/10.1007/s00227-004-1328-4

Salmaso, F., Servanzi, L., Crosa, G., Quadroni, S., & Espa, P. (2021). Assessing the impacts of hydropeaking on river benthic macroinvertebrates: A state-of-the-art methodological overview. *Environments*, *8*(7), 67. doi: https://doi.org/10.3390/environments8070067

Schmitz, O., Karssenberg, D., De Jong, K., De Kok, J.-L., & De Jong, S. M. (2013). Map algebra and model algebra for integrated model building. *Environmental modelling & software*, *48*, 113–128. doi: https://doi.org/10.1016/j.envsoft.2013.06.009

Shannon, C. (1949). Communication in the presence of noise. *Proceedings of the IRE*, *37*(1), 10-21. doi: https://doi.org/10.1109/JRPROC.1949.232969

Stoffers, T. (2022). *Surviving and growing in dynamic floodplains: Habitat heterogeneity drives communities of young fish in the lower river rhine* (Unpublished doctoral dissertation). Wageningen University and Research.

Stoffers, T., Buijse, A., Geerling, G., Jans, L., Schoor, M., Poos, J., ... Nagelkerke, L. (2022). Freshwater fish biodiversity restoration in floodplain rivers requires connectivity and habitat heterogeneity at multiple spatial scales. *Science of the Total Environment*, *838*, 156509. doi: https://doi.org/10.1016/j.scitotenv.2022.156509

Stoffers, T., Buijse, A., Verreth, J., & Nagelkerke, L. (2022). Environmental requirements and heterogeneity of rheophilic fish nursery habitats in european lowland rivers: Current insights and future challenges. *Fish and Fisheries*, *23*(1), 162–182. doi: https://doi.org/10.1111/faf.12606

Sullivan, K., Coletti, M., & Luke, S. (2010). *Geomason: Geospatial support for mason* (Tech. Rep.). Department of Computer Science, George Mason University.

Sutanudjaja, E. H., van Beek, R., Wanders, N., Wada, Y., Bosmans, J. H. C., Drost, N., ... Bierkens, M. F. P. (2018). Pcr-globwb 2: a 5 arcmin global hydrological and water resources model. *Geoscientific Model Development*, *11*(6), 2429–2453. doi: 10.5194/gmd-11-2429-2018

Swingland, I. R., Greenwood, P. J., et al. (1983). *The ecology of animal movement.* Clarendon Press Oxford. doi: https://doi.org/10.2307/4647

Taillandier, P., Grignard, A., Gaudou, B., & Drogoul, A. (2014). Des données géographiques à la simulation à base d'agents: application de la plate-forme gama. *Cybergeo: European Journal of Geography*. doi: https://doi.org/10.4000/cybergeo.26263

Takeyama, M., & Couclelis, H. (1997). Map dynamics: integrating cellular automata and gis through geo-algebra. *International Journal of Geographical Information Science*, *11*(1), 73–91. doi: https://doi.org/10.1080/136588197242509

Tang, W., & Bennett, D. A. (2010). Agent-based modeling of animal movement: a review. *Geography Compass*, *4*(7), 682–700. doi: https://doi.org/10.1111/j.1749-8198.2010.00337.x

The HDF Group. (1997). *Hierarchical Data Format, version 5.* (https://www.hdfgroup.org/HDF5/)

Tomlin, C. (1994). Map algebra: one perspective. *Landscape and Urban Planning*, *30*(1), 3-12. (Special Issue Landscape Planning: Expanding the Tool Kit) doi: https://doi.org/10.1016/0169-2046(94)90063-9

Troitzsch, K. G. (2004). Validating simulation models. In *Proceedings of the 18th european simulation multiconference* (pp. 98–106).

van Oorschot, M., Kleinhans, M., Geerling, G., Buijse, T., & Middelkoop, H. (2022). Modelling restoration of natural flow regimes in dam impaired systems: Biomorphodynamic effects and recovery times. *Geomorphology*, *413*, 108327. doi: https://doi.org/10.1016/j.geomorph.2022.108327

Van de Wolfshaar, K., Ruizeveld de Winter, A., Straatsma, M., Van Den Brink, N., & De Leeuw, J. (2010). Estimating spawning habitat availability in flooded areas of the river waal, the netherlands. *River Research and Applications*, *26*(4), 487–498. doi: https://doi.org/10.1002/rra.1306

van Winden, A., Reker, J., & Overmars, W. (2001). Dynamische processen in de grensmaas: Hoe de morfologische dynamiek in de 19e eeuw tot stilstand kwam en de mogelijkheden die er zijn voor herstel. *Natuurhistorisch Maandblad*, *90*(10), 221–226.

von Elverfeldt, K., Embleton-Hamann, C., & Slaymaker, O. (2016). Self-organizing change? on drivers, causes and global environmental change. *Geomorphology*, *253*, 48-58. doi: https://doi.org/10.1016/j.geomorph.2015.09.026

Wilensky, U. (1999). *Netlogo* (http://ccl.northwestern.edu/netlogo/). Northwestern University, Evanston, IL: Center for Connected Learning and Computer-Based Modeling. Retrieved from `http://ccl.northwestern.edu/netlogo/`

Wortelboer, R., Buijse, T., van Geest, G., Harezlak, V., & van den Roovaart, J. (2020). *Krw-verkenner module ecologie rijkswateren* (Tech. Rep.). Deltares.

8 Appendix

A Code Listings

Code associated with this thesis can also be found in this repository on GitHub: https://github.com/ERKuipers/CoupledFieldFish/

A.1 Model

```python
import datetime
import os
import sys
from pathlib import Path
cur = Path.cwd()
up_dir = cur.parent
post_processing = up_dir / 'post_processing'
working = up_dir / 'working'
sys.path.append(f'{post_processing}')
sys.path.append(f'{working}')
from pathlib import Path
import pcraster as pcr
import pcraster.framework as pcrfw
import campo
import numpy as np
from matplotlib import pyplot as plt
from lifecycle_pref import two_conditions_boolean_prop, campo_clump
from moving_to_coordinates import move
from xugrid_func import partial_reraster

#########
# model #
#########

class FishEnvironment(pcrfw.DynamicModel, ):
     def __init__(self, input_dir, output, map_nc, spatial_resolution,
    temporal_resolution, conversion_T, xmin, ymin, xmax, ymax,
    nrbarbels, spawning_conditions, adult_conditions, radius, attitude)
    :
        pcrfw.DynamicModel.__init__(self)
        # Framework requires a clone
        # set a dummy clone
        pcr.setclone(10, 20, 10, 0, 0)
        self.input_dir = input_dir
        self.output = output
        self.map_nc = map_nc
        self.resolution = spatial_resolution
        self.delta_t = temporal_resolution # delta timesteps
        self.xmin = xmin
        self.ymin = ymin
        self.xmax = xmax
        self.ymax = ymax
        self.nrbarbels = nrbarbels
        self.spawning_conditions = spawning_conditions
        self.adult_conditions = adult_conditions
        self.radius = radius
        self.attitude = attitude
        self.conversion_T = conversion_T # t to multiply the timesteps with to make it fit the timestep of the model
    def initial(self):
        init_start = datetime.datetime.now()
        self.fishenv = campo.Campo(seed = 1)

        # create real time settings for lue:
        date = datetime.date(2019, 6, 1)
        time = datetime.time(00,00)
        start = datetime.datetime.combine(date, time)
        unit = campo.TimeUnit.hour
        stepsize = self.delta_t

        # create the output lue data set
        self.fishenv.create_dataset(f'{self.output}/fish_environment.lue')
        self.fishenv.set_time(start, unit, stepsize, self.nrTimeSteps())
        self.data_t = int(self.currentTimeStep()*self.conversion_T)

        ###################
        # Phenomenon barbel #
        ###################
        self.barbel = self.fishenv.add_phenomenon ('barbel') # could we possibly reduce the first step of this?
        self.barbel.set_epsg(28992)
        # Property Set barbel
        self.barbel.add_property_set ('adults', self.input_dir / 'Fish.csv' ) # the water area always has the same spatial as well as temporal extent (it always exists)
        self.barbel.adults.is_mobile = True

        # Properties for barbel
        self.barbel.adults.movemode = 0
        self.barbel.adults.movemode.is_dynamic = True
        self.barbel.adults.spawning_area = 0
        self.barbel.adults.spawning_area.is_dynamic = True
        self.barbel.adults.has_spawned = 0
        self.barbel.adults.has_spawned.is_dynamic = True
        self.barbel.adults.swimdistance = 0
        self.barbel.adults.swimdistance.is_dynamic = True
        self.barbel.adults.surrounding = 0
        self.barbel.adults.surrounding.is_dynamic = True

        ####################
        # Phenomenon Water #
        ####################
        self.water = self.fishenv.add_phenomenon ('water') # could we possibly reduce the first step of this?
        self.water.set_epsg(28992)
        # Property Set Area #
        self.water.add_property_set ('area', self.input_dir / 'CommonMeuse.csv') # the water area always has the same spatial as well as temporal extent (it always exists)

        # May come in handy
        self.water.area.zero = 0
        self.water.area.one = 1
        #  Property Flow velocity #
        u_array = partial_reraster (self.map_nc, self.resolution, self.data_t, 'mesh2d_ucmag', self.xmin, self.xmax, self.ymin, self.ymax)
        self.water.area.flow_velocity = u_array [np.newaxis, :, :]
        self.water.area.flow_velocity.is_dynamic = True
        # Property Water Depth #
        d_array = partial_reraster (self.map_nc, self.resolution, self.data_t, 'mesh2d_waterdepth', self.xmin, self.xmax, self.ymin, self.ymax)
        self.water.area.water_depth = d_array [np.newaxis, :, :]
        self.water.area.water_depth.is_dynamic = True

        self.water.area.spawning_grounds = two_conditions_boolean_prop (self, self.water.area.water_depth, self.water.area.flow_velocity, self.spawning_conditions)
        self.water.area.spawning_grounds.is_dynamic = True
        self.water.area.swimmable = two_conditions_boolean_prop (self, self.water.area.water_depth, self.water.area.flow_velocity, self.adult_conditions)
        self.water.area.swimmable.is_dynamic = True
        self.water.area.connected_swimmable = campo_clump (self, self.water.area.swimmable)
        self.water.area.connected_swimmable.is_dynamic = True
        self.fishenv.write() # write the lue dataset
        end = datetime.datetime.now() - init_start # print the run duration
        print(f'init: {end}, timestep: {self.currentTimeStep()}')

    def dynamic(self):
        start = datetime.datetime.now()
        self.data_t = int(self.currentTimeStep()*self.conversion_T) # update the  given data timestep with updated current timestep as by the pcraster framework
        # first setting environmental variables, then positioning the barbels as a response to the alternation in habitat
        u_array = partial_reraster (self.map_nc, self.resolution, self.data_t, 'mesh2d_ucmag', self.xmin, self.xmax, self.ymin, self.ymax)
        self.water.area.flow_velocity = u_array [np.newaxis, :, :]

        # Property Water Depth #
        d_array = partial_reraster (self.map_nc, self.resolution, self.data_t, 'mesh2d_waterdepth', self.xmin, self.xmax, self.ymin, self.ymax)
        self.water.area.water_depth = d_array [np.newaxis, :, :]
        # Creating boolean and clump fields describing swimmable and spawning grounds
        self.water.area.spawning_grounds = two_conditions_boolean_prop (self, self.water.area.water_depth, self.water.area.flow_velocity, self.spawning_conditions)
        self.water.area.swimmable = two_conditions_boolean_prop(self, self.water.area.water_depth, self.water.area.flow_velocity, self.adult_conditions)
        self.water.area.connected_swimmable = campo_clump (self, self.water.area.swimmable)

        # Moving barbel and getting information about barbel movement:
        movingX, movingY, spawning_area, travel_distance, has_spawned, movemode = move (self.water.area.connected_swimmable, self.water.area.swimmable, self.water.area.spawning_grounds, self.barbel.adults, self.water.area, self.currentTimeStep(), self.barbel.adults.has_spawned, self.radius, self.attitude)
        # Move agents over field:
        barbel_coords = self.barbel.adults.get_space_domain(self.currentTimeStep())
        barbel_coords.xcoord = movingX
        barbel_coords.ycoord = movingY
        self.barbel.adults.set_space_domain(barbel_coords, (self.currentTimeStep ()))
        self.barbel.adults.spawning_area = spawning_area
        self.barbel.adults.has_spawned = has_spawned
        self.barbel.adults.justswam = travel_distance
        self.barbel.adults.movemode = movemode
        self.barbel.adults.swimdistance = self.barbel.adults.swimdistance + self.barbel.adults.justswam # keep on adding the swimming distance

        # write to lue
        self.fishenv.write(self.currentTimeStep())
        end = datetime.datetime.now() - start
        print(f'ts:  {end}  write, timestep: {self.currentTimeStep()}')
```

Listing 1: The common barbel model

A.2 Obtaining data

```python
import xarray as xr
import xugrid as xu
import pandas as pd
import numpy as np
def partial_reraster (ugrid_filelocation, resolution, timestep, var, xmin,xmax,ymin,ymax):
    '''
    Parameters
    ----------
    ugrid_filelocation : location of .nc file (Type: String, Path)
    resolution : resolution to rasterize variable in (Type: Integer)
    Timestep : timestep of the dataset to rasterize (Type: Integer)
    Var: Variable to get, 'mesh2d_ucmag' for u and 'mesh2d_waterdepth' for d (Type: String)
    Xmin: minimal X-coordinate
    Xmax: maximum X-coordinate
    Ymin: minimal Y-coordinate
    Ymax: maximum Y-coordinate

    Returns
    -------
    xr_raster : xarray data array within the spatial extent given (Type: np.Array)

    '''
    ds = xr.open_dataset(ugrid_filelocation)
    uds = (xu.UgridDataset(ds))
    # when not sure what variable to get, run following line to print variable name:
    # print(uds.data_vars)

    x_coords = np.arange(xmin,xmax, resolution)  # resolution becomes the cell length of the raster.
    y_coords = np.arange(ymin,ymax,resolution)

    da_clone = xr.DataArray(data=np.ones((len(x_coords), len(y_coords))),
                            coords={'x':x_coords,
                                    'y':y_coords},
                            dims=['x', 'y'])

    xr_raster = uds[str(var)].isel(time=timestep).ugrid.rasterize_like(da_clone)
    xr_ds = xr_raster.rio.write_crs ("epsg:28992")   # xr Data array
    xr_df = xr_ds.to_dataframe()
    pd_xy = xr_df.reset_index()[['x','y', str(var)]]
    # make from long raster format with columns x,y and variable the indx x, columns y and the variable the value
    reshaped = pd_xy.pivot(index = ['y'], columns = ['x'], values=str(var))
    raster_array_rev = reshaped.to_numpy()
    raster_array = np.flip (raster_array_rev, axis = 0) # needs to be flipped in order to show up correctly
    return raster_array
```

Listing 2: Rasterize the flexible mesh

A.3 Habitat suitability and fragmentation

```python
import numpy as np
def two_conditions_boolean_prop (water_depth, flow_velocity, conditions):
    ''' returns a boolean fieldproperty on the basis of  input conditions two (equally sized) field properties
    and conditions determined by the modeller
    Parameters:
         water_depth: boolean fieldproperty (Type: Property)
         flow_v: boolean fieldproperty (Type: Property)
         conditions: a list with length 4, contains the conditions on the basis of which the boolean map will be created, with idx
              0: water_depth_min
              1: water_depth_max
              2: flow_velocity_min
              3: flow_velocity_max
              (Type: List, np.Array)
    Returns: a boolean fieldprop for which both conditions in relation to flow velocity and water depth are true (type: Property)
```

```
        '''
    water_d = water_depth.values()[0]
    flow_v = flow_velocity.values()[0]
    water_depth_min = conditions[0]
    water_depth_max = conditions[1]
    flow_v_min = conditions[2]
    flow_v_max = conditions [3]
    condition = (water_d >= water_depth_min) & (water_d <=
   water_depth_max) & (flow_v<=flow_v_max) & (flow_v >= flow_v_min)
    boolean_ar = np.where(condition, 1,0)
    boolean_prop = boolean_ar [np.newaxis,:,:]
    return boolean_prop
```

Listing 3: Generating habitat suitability map

```
import campo
import pcraster as pcr
def campo_clump (boolean_fieldprop, field_pset):
    connected_boolean_prop = Property("
   _new_property_from_property_name", boolean_fieldprop._pset_uuid,
   boolean_fieldprop._pset_domain, boolean_fieldprop._shape)
    for fidx, area in enumerate (field_pset.space_domain):
        nrCols = int(area[5])
        nrRows = int(area[4])
        west = area [0]
        north = area [3]
        cellSize = math.fabs (area[2] - west)/nrCols
    boolean_ar = (boolean_fieldprop.values()[0]).astype(int)
    plt.imshow(boolean_ar)
    plt.colorbar()
    plt.show()
    pcr.setclone (nrRows, nrCols, cellSize, west, north)
    arg_raster = pcr.numpy2pcr(pcr.Boolean, boolean_ar, -1000)
    pcr.plot(arg_raster)
    result_raster = pcr.clump(arg_raster)
    result_ar = pcr.pcr2numpy(result_raster, -1000)
    # overruling value 0 for areas that are not connected to the big '
   non-swimmable land' but are still non-swimmable !
    # value 0 for any clump which is non-swimmable
    #boolean_fieldprop.values()[0], shows correct but im not sure if
   its boolean, seems like it

    connected_boolean_ar = np.where (boolean_fieldprop.values()[0] ==
   0, 0, result_ar)
    connected_boolean_prop.values()[0] = connected_boolean_ar
    return connected_boolean_prop
```

Listing 4: Generating patches based on boolean map

```
import numpy as np
import math
import matplotlib.pyplot as plt
def generate_mask (point_pset, pidx, field_pset, radius):
    '''radius = in unit of model, so probably metres '''

  # Loop over space attributes of the different points in the point
   agents propertyset
    point_x = point_pset.space_domain.xcoord[pidx]
    point_y = point_pset.space_domain.ycoord[pidx]

    for fidx,area in enumerate(field_pset.space_domain):
    # Get bounding box of field
        nr_rows = int(area[4])
        nr_cols = int(area[5]) #
        minX = area [0]
        minY = area [1]

        # Translate point coordinate to index on the field array
        cellsize = math.fabs(area[2] - minX) / nr_cols # in unit of length
        ix = math.floor((point_x - minX) / cellsize)
        iy = math.floor((point_y - minY) / cellsize)


    mask_unflipped = np.zeros((nr_rows, nr_cols))  # Initialize mask with NaN

    # Generate grid of coordinates
    x, y = np.meshgrid(np.arange(nr_cols), np.arange(nr_rows))

    # Calculate distance from each point to the center
    distance = np.sqrt((x - ix)**2 + (y - iy)**2)
    # Convert model unit to number of cells
    cell_radius = math.floor(radius / cellsize)
    # Set values inside the radius to 1
    mask_unflipped[distance <= cell_radius] = 1
    mask = np.flip (mask_unflipped, axis=0)
    return mask
```

Listing 5: Generate mask to represent maximum reach of barbel

A.4 Field-agent interactions

```
from osgeo import gdal
from osgeo import osr
from osgeo import ogr
gdal.UseExceptions()
import math
import numpy as np
from campo.property import Property
def raster_values_to_feature(point_pset, field_pset, field_prop):
  ''' Queries the raster values of one field property on the location of point agents,
  writes this as a new point agent property set
  Parameters:
    point_pset: property set of the point agents to be attributed the location
    field_pset: property set of a single field
    field_prop: property of which the values are attributed to the newly generated point property
  Returns:
    point_prop: property of point agents with values of local field values'''

```

```python
  # generate empty point property given point property space definitions
  point_prop = Property('emptycreatename', point_pset.uuid, point_pset.space_domain, point_pset.shapes)

  # loop over space attributes of the different points in the point agents propertyset
  for pidx,coordinate in enumerate(point_pset.space_domain):
    point_x = point_pset.space_domain.xcoord[pidx]
    point_y = point_pset.space_domain.ycoord[pidx]

    point_value = np.zeros(1)
    for fidx,area in enumerate(field_pset.space_domain):

      # get bounding box of field
      nr_cols = int(area[5]) #
      minX = area [0]
      minY = area [1]

      # translate point coordinate to index on the field array
      cellsize = math.fabs(area[2] - minX) / nr_cols
      ix = math.floor((point_x - minX) / cellsize)
      iy = math.floor((point_y - minY) / cellsize)

      # reshape property to a mirrored numpy field array to accommodate right use of point and agent indexes
      reshaped = np.flip (field_prop.values()[fidx], axis=0)
      # query the field attribute given point location
      point_value[fidx] = reshaped[iy,ix]

    # write the value to the point property for each point agent
    point_prop.values()[pidx] = point_value.item()

  return point_prop
```

Listing 6: Local function

```python
from osgeo import gdal
from osgeo import osr
from osgeo import ogr
gdal.UseExceptions()
import math
import numpy as np
from campo.property import Property
def window_values_to_feature(point_pset, field_pset, field_prop, windowsize, operation):
  ''' Queries aggregative raster values of a window of a field property based from the location of point agents,
  Given a certain aggregative operation. Writes this as a new point agent property.
  Parameters:
    point_pset: property set of the point agents to be attributed the location
    field_pset: property set of a single field
    field_prop: property of which the values are attributed to the newly generated point property
    windowsize: we make square windows: windowsize describes the length of the window in the unit of the field_property
    operation: aggregative numpy operation as a string ('sum', 'mean', 'min', 'max', 'etc')
  Returns:
    point_prop: property of point agents with values of aggregated field values'''

  # Generate operator, first checking if operation is available in numpy
  if not hasattr (np, operation):
    raise ValueError (f'Unsupported numpy operation, {operation}')
  operator = getattr(np, operation)

  # Generate empty point property given point property space definitions
  point_prop = Property('emptycreatename', point_pset.uuid, point_pset.space_domain, point_pset.shapes)
  # Loop over space attributes of the different points in the point agents propertyset
  for pidx,coordinate in enumerate(point_pset.space_domain):
    point_x = point_pset.space_domain.xcoord[pidx]
    point_y = point_pset.space_domain.ycoord[pidx]

    window_value = np.zeros(1)
    for fidx,area in enumerate(field_pset.space_domain):
      # Get bounding box of field
      nr_rows = int(area[4])
      nr_cols = int(area[5]) #
      minX = area [0]
      minY = area [1]

      # Translate point coordinate to index on the field array
      cellsize = math.fabs(area[2] - minX) / nr_cols # in unit of length
      ix = math.floor((point_x - minX) / cellsize)
      iy = math.floor((point_y - minY) / cellsize)

      nr_windowcells = math.floor(windowsize/cellsize)

      ws_iy = math.floor(iy - 0.5*nr_windowcells)
      we_iy = math.floor(iy + 0.5*nr_windowcells)
      ws_ix = math.floor (ix - 0.5*nr_windowcells)
      we_ix = math.floor (ix + 0.5*nr_windowcells)
      # Reshape field property to a mirrored numpy field array to accommodate right use of point and agent indexes
      reshaped = np.flip (field_prop.values()[fidx], axis=0)

      # Query the field attribute given point location
      window_value[fidx] = operator (reshaped [ws_iy:we_iy, ws_ix:we_ix])

    # Write the value to the point property for each point agent
    point_prop.values()[pidx] = window_value.item()

  return point_prop
```

Listing 7: Window operation

```python
from osgeo import gdal
from osgeo import osr
from osgeo import ogr
gdal.UseExceptions()
import math
import numpy as np
from campo.property import Property
def zonal_values_to_feature(point_pset, field_pset, field_prop_class, field_prop_var, operation):
    ''' Queries aggregative raster values of a zone of a field property based on the location
      of point agents within a classification map, given a certain aggregative operation.
      Writes this as a new point agent property. Only works for one fieldagent existing at the location.
      Fieldagents with overlapping domains cannot generate an output
      E.g.: According to both the agent's location and a soil map (field_prop_class), the agent is positioned in
           soil '2' , which is clay. With the operation 'mean', the mean rainfall (from field_prop_var)
           is calculated and attributed to the agent
    Parameters:
      point_pset: property set of the point agents to be attributed the location
      field_pset: property set of a single field
      field_prop_class: property describing classes or groups of cells of which the zonal extent shall be the windowsize of the aggregation
      field_prop_var: property describing the variable which needs to be aggregated,
        in case of a boolean map, 'True' values are 1
      operation: operation: aggregative numpy operation as a string ('sum', 'mean', 'min', 'max', 'etc')
    Returns:
      point_prop: property of point agents with values of aggregated field values'''

    # Generate operator, first checking if operation is available in numpy
    if not hasattr (np, operation):
      raise ValueError (f'Unsupported numpy operation, {operation}')
    operator = getattr(np, operation)

    # Identifying the zone the agent is in
    agents_zoneIDs = raster_values_to_feature(point_pset,field_pset, field_prop_class)
    # Generate empty point property given point property space definitions
    point_prop = Property('emptycreatename', point_pset.uuid, point_pset.space_domain, point_pset.shapes)
    # make as many field properties as there are agents:
    # Loop over space attributes of the different points in the point agents propertyset
    for pidx, ID in enumerate(agents_zoneIDs.values()):
      # Making a boolean map concerning the extent of the zone for each agent
      aggr_zone_var = np.zeros(1)
      for fidx,area in enumerate(field_pset.space_domain):
```

```python
        zone_extent = np.where (field_prop_class.values()[fidx] == ID, 1, 0)
        variable_zone_array = np.multiply (zone_extent, field_prop_var.values()[fidx])
        # we don't need to flip this time, since the raster_values_to_feature already gave
        # the right topological relationship between the field and the agent:
        # the zone_extent array describes the zone in which the agent is positioned.
        # This array might be flipped, but this won't lead to any different outcomes of aggregative operations
        aggr_zone_var[fidx] = operator (variable_zone_array)
    #field_prop_array [pidx] = field_prop# array as long as the number of agents filled with a field prop for each agent
    # Write the value to the point property for each point agent
    point_prop.values()[pidx] = aggr_zone_var.item()

  return point_prop
```

Listing 8: Zonal operation

```python
import numpy as np
import math
import matplotlib.pyplot as plt
def generate_mask (point_pset, pidx, field_pset, radius):
    '''radius = in unit of model, so probably metres '''

  # Loop over space attributes of the different points in the point agents propertyset
    point_x = point_pset.space_domain.xcoord[pidx]
    point_y = point_pset.space_domain.ycoord[pidx]

    for fidx,area in enumerate(field_pset.space_domain):
    # Get bounding box of field
        nr_rows = int(area[4])
        nr_cols = int(area[5]) #
        minX = area [0]
        minY = area [1]

        # Translate point coordinate to index on the field array
        cellsize = math.fabs(area[2] - minX) / nr_cols # in unit of length
        ix = math.floor((point_x - minX) / cellsize)
        iy = math.floor((point_y - minY) / cellsize)


    mask_unflipped = np.zeros((nr_rows, nr_cols))  # Initialize mask with NaN

    # Generate grid of coordinates
    x, y = np.meshgrid(np.arange(nr_cols), np.arange(nr_rows))

    # Calculate distance from each point to the center
    distance = np.sqrt((x - ix)**2 + (y - iy)**2)
    # Convert model unit to number of cells
    cell_radius = math.floor(radius / cellsize)
    # Set values inside the radius to 1
    mask_unflipped[distance <= cell_radius] = 1
    mask = np.flip (mask_unflipped, axis=0)
    return mask
```

Listing 9: Generating the maximum reach mask

A.5 Move functions

```python
def randommove_to_boolean (boolean_fieldprop, field_pset, point_prop):
    '''
    - boolean_fieldprop = a property that is a field and contains true values for places where an agent may move to
    - field_pset = a property set describing the domain of the field of the study area (Type: propertyset)
    - point_propset = the property set that has the move (= point agents), can be of any property of the point agents
    return lists of coordinates with the same length as the number of agents in a point propertyset (sueful wehn you want to make them move there)
    '''
    # need to flip again because when calculating with this map points have different orientation than field (see rerasterize)
    if isinstance(boolean_fieldprop, np.ndarray):
        map_flipped = np.flip (boolean_fieldprop, axis=0)
    elif isinstance (boolean_fieldprop, campo.property.Property):
        map_flipped = np.flip (boolean_fieldprop.values()[0], axis=0)
    else:
        raise TypeError ('boolean_fieldprop needs to be of type campo.property or of a numpy array with same dimensions as field property values')

    nragents = len (pointprop.values().values.values())
    for fidx, area in enumerate (field_pset.space_domain):
        nr_cols = int(area[5])
        xmin = area [0]
        ymin = area [1]
        resolution = math.fabs (area[2] - xmin)/nr_cols
    # finding the indices of the places where the fieldcondition is true
    coords_idx = np.argwhere (map_flipped) #coordinates of the spawning grounds in [y,x]
    # collecting the coordinate combination in a tuple so as to prevent them from being 'disconnected' from eachother
    coords_list = [tuple(row) for row in coords_idx]
    random_newindex = random.sample (coords_list, nragents) # nr of agents is the subsetsize
    # seperating the tuples in the list in two seperate list
    yindex, xindex = zip(*random_newindex) #assuming that tuple list is reversed, first gives y then x as in column = x and row = y
    xindex = np.array (xindex)
    yindex = np.array (yindex)
    xcoords = np.zeros(nragents)
    ycoords = np.zeros(nragents)

    # make from x index a x coordinate by using resolution and bounding box information
    for i, xvalue in enumerate(xindex):
        xcoords [i] = (xvalue*resolution + xmin)
    for j, yvalue in enumerate(yindex):
        ycoords[j] = (yvalue *resolution + ymin)
    return xcoords, ycoords
```

Listing 10: Random move to a destination

```python
def find_closest_dest (field_pset, boolean_fieldprop, point_pset_orX, pidx_orY):
    ''' find closest point complying to the boolean fieldprop, from a point
    if point_pset and pidx is inavailable, point_pset may be a xcoordinate  and pidx may be the ycoordinate
    of the point of which a new destination needs to be found
    parameters:
        boolean_fieldprop: a boolean map, either as a field-agent property or as a numpy array, describing with 1s the destination
        poi'''
    if isinstance(point_pset_orX, campo.propertyset.PropertySet):
        point_x = point_pset_orX.space_domain.xcoord [pidx_orY]
        point_y = point_pset_orX.space_domain.ycoord [pidx_orY]
    elif isinstance (point_pset_orX, (np.int64, int, np.float64, float)):
        point_x = point_pset_orX
        point_y = pidx_orY
    else:
        raise TypeError('make sure third and fourth argument give enough information to substract coordinates, by being a propertyset or integers describing coordinates')

    for fidx,area in enumerate(field_pset.space_domain):
      # Get bounding box of field
      nr_cols = int(area[5]) #
      minX = area [0]
      minY = area [1]
      resolution =  math.fabs(area[2] - minX) / nr_cols

    ix = math.floor((point_x - minX) / resolution) # needs to be rouned down since we define it by the minimum and therefore lower border
    iy = math.floor((point_y - minY) / resolution)
    point = np.array([iy, ix]) # in indexes as in the field , with first row = y, column = x

    # field proerty may be of type property or the values of such a property
    if isinstance (boolean_fieldprop, campo.property.Property):
        field_array = np.flip(boolean_fieldprop.values()[0])

    elif isinstance (boolean_fieldprop, np.ndarray):
        field_array = np.flip (boolean_fieldprop, axis=0)
    else:
        raise TypeError ('boolean_fieldprop needs to be of type campo.property or of a numpy array with same dimensions as field property values')
    boolean_array = np.where (field_array == 0, 0, 1) # in case it was not boolean yet
    # Generate a list with all potential destinations, also accommodates for a clump field in which all possible destinations are not 0
    potential_dest_idxs = np.argwhere (boolean_array)

    # Convert indices to a 2D array of points
    # Use NearestNeighbors to find the closest '1'
    nbrs = NearestNeighbors (n_neighbors= 1, algorithm='ball_tree').fit(potential_dest_idxs)
    distances, indices = nbrs.kneighbors([point])
    # these indices are based on the flipped point, so the terugvertaling naar punt gaat dan niet meer, omdat het nu dus een geflipt punt is
    # the flip operation however has to be performed, otherwise the topological relation is not correctly established
    new_yidx, new_xidx= tuple(potential_dest_idxs[indices[0][0]]) # but now what do these indices mean on the non flipped array
    xcoord = new_xidx*resolution + minX
    ycoord = new_yidx*resolution + minY
    travel_distance = float(distances [0][0])*resolution
    return xcoord, ycoord, travel_distance
```

Listing 11: Function move to closest destination

```
def move_directed (field_pset, dest_boolean_fieldprop, boolean_clump_fieldprop, point_pset, pidx):
    '''boolean_clump_fieldprop = the current clump as a boolean map ( is all available area for the current location of the )'''
    # Find closest spawning area pixel destination
    closest_destX, closest_destY, dist1 = find_closest_dest(field_pset, dest_boolean_fieldprop, point_pset, pidx)
    # from this pixel, find closest clump pixel , which will be in the right direction
    xcoord, ycoord, dist2 = find_closest_dest(field_pset, boolean_clump_fieldprop, closest_destX, closest_destY)

    initialX = point_pset.space_domain.xcoord[pidx]
    initialY = point_pset.space_domain.ycoord[pidx]
    travel_distance = np.sqrt((xcoord - initialX)**2 +(ycoord - initialY)**2)
    return xcoord, ycoord, travel_distance
```

Listing 12: Move in the direction of

```
def move (clump_fieldprop, boolean_fieldprop, dest_fieldprop, point_pset, field_pset, timestep, has_spawned_pointprop, radius, fishbehaviour):
    '''Moving to a place which is connected to the initial location of the agent by allowing potential destinations

    self: the object class of the model
    clump_fieldprop: a field agent property describing certain clumps as defined by the pcraster function 'clump'; each clump having a unique ID
    boolean_fieldprop: relates to the location in which the connection is defined. A boolean map which allows for proper connection: this can be a place
    that can be bridged, like the swimmable or walkable area
```

```python
    dest_fieldprop: a boolean map relating to possible destinations, for instance: spawning grounds. May be filled in with 'True' / '1' or can be
    removed if all destinations are accepted
    point_pset : the  property set of the points to be moved
    '''
    for fidx, area in enumerate (field_pset.space_domain):
        nr_cols = int(area[5])
        xmin = area [0]
        ymin = area [1]
        resolution = math.fabs (area[2] - xmin)/nr_cols # adjust if resolution is different for x and y, then this is the x-resolution

    agent_clumpID = raster_values_to_feature (point_pset, field_pset, clump_fieldprop) # property describing the clump ID where the agent is
    nragents = len (has_spawned_pointprop.values().values.values())

    xcoords = np.zeros ((nragents))
    ycoords = np.zeros ((nragents))
    available_area = np.zeros((nragents))
    travel_distances = np.zeros ((nragents))
    movemode = np.zeros ((nragents))
    spawns = np.hstack(list(has_spawned_pointprop.values().values.values())) # creating an numpy array while using the property values
     # really need to do this for all given that

    # this needs to be implemented from the fieldprop so that it does not get overwritten by a 0 value in a next timestep
    for pidx, ID in enumerate (agent_clumpID.values()):
        # print (ID)
        mask = generate_mask (point_pset, pidx, field_pset, radius) # check, is now flipped
        fieldprop_boolean_value = np.where (clump_fieldprop.values()[0] == ID, 1, 0) # the boolean map describing the clump for each individual fish
        reachable_array = np.multiply (fieldprop_boolean_value, mask) # reachable within clump, works
        array_dest = dest_fieldprop.values()[0] # the eventual destination, boolean map of it
        prob_destination = np.multiply (reachable_array, array_dest)
        available_area [pidx] = np.sum(prob_destination)*(resolution**2)
        if timestep == 1: # first moving the
            xcoord_array, ycoord_array = randommove_to_boolean (clump_fieldprop, resolution, xmin, ymin, 1)
            xcoords [pidx] = xcoord_array.item()
            ycoords [pidx] = ycoord_array.item()

        elif has_spawned_pointprop.values()[pidx]==1:
            xcoords [pidx] = xmin # moving spawners to the corner !!
            ycoords [pidx] = ymin
            # writing the available area so that also when spawning, the available area can be printed
            movemode [pidx]= 1
            # print (f'been there, done that (the spawning), {pidx}
    out')

        elif ID == 0:# dryswimming
            # find the closest non-dry land to go to
            movemode [pidx] = 2 # nearest destination
            # print (f'I, {pidx}, am dryswimming! help me get back')
            swim_array = boolean_fieldprop.values()[0]
            #reachable and swimmable within buffer, not taking into account own clump:
            masked_swimmable = np.multiply(swim_array, mask)

            if np.sum(masked_swimmable) > 0: # make next operation faster by feeding find_closest_dest a smaller subset of potential indices to go to
                xcoords [pidx], ycoords [pidx], travel_distances [pidx] = find_closest_dest (field_pset, masked_swimmable, point_pset, pidx)
            else: # if there is no swimmable area in the direct vicinity
                xcoords [pidx], ycoords [pidx], travel_distances [pidx] = find_closest_dest (field_pset, swim_array, point_pset, pidx)
        else:
            # if theres no spawning in proximate area, move in the direction of the closest
            if available_area[pidx] == 0: # has not spawned yet but no available area within clump and radius
                movemode [pidx] = 3 # directed move
                # print (f'I {pidx} rate the spawning availability over here 0/5 stars')
                if fishbehaviour == 'focussed': # distuingishing the different
                    xcoords [pidx], ycoords [pidx], travel_distances [pidx] = move_directed (field_pset, dest_fieldprop, reachable_array, point_pset, pidx)
                elif fishbehaviour == 'wandering':
                    xcoord_array, ycoord_array = randommove_to_boolean (reachable_array, resolution, xmin, ymin, 1)
                    xcoords [pidx] = xcoord_array.item()
                    ycoords [pidx] = ycoord_array.item()
                    travel_distances [pidx] = np.sqrt ((point_pset.space_domain.xcoord[pidx]-xcoords[pidx])**2 + (point_pset.space_domain.ycoord[pidx]-ycoords[pidx])**2)
                else:
                    raise ValueError ('Fishbehaviour can be either focussed or wandering')
            else: # spawning:
                # generating destination maps, moving there
                xcoord_array, ycoord_array = randommove_to_boolean (prob_destination, resolution, xmin, ymin, 1)
                xcoords [pidx] = xcoord_array.item()
                ycoords [pidx] = ycoord_array.item()
                travel_distances [pidx] = np.sqrt ((point_pset.space_domain.xcoord[pidx]-xcoords[pidx])**2 + (point_pset.space_domain.ycoord[pidx]-ycoords[pidx])**2)
                spawns [pidx] = 1
                movemode [pidx]= 4 # destination oriented
                # print (f'dope ! #sex {pidx}')
    return xcoords, ycoords, available_area, travel_distances, spawns,
    movemode
```
Listing 13: Moving for for different scenarios

Listing 14: Rasterize the flexible mesh

A.6 Configurations and running the model

```
# commonBarbel = Fish(cfg.nr_barbel, cfg.xmin, cfg.ymin, cfg.xmax, cfg
    .ymax, cfg.input_d)
# commonMeuse = CommonMeuse (cfg.xmin, cfg.ymin, cfg.xmax, cfg.ymax,
    cfg.spatial_resolution, cfg.map_nc, cfg.timesteps, cfg.
    temporal_resolution, cfg.data_T_res, cfg.input_d)
# commonMeuse.extent() # generates a csv file describing the extent of
    the Meuse
# commonBarbel.extent() # generetes a csv file describing
    coordinatesets for each barbel
# commonMeuse.time_domain()
for fishadventuring in cfg.fish_exploring.keys():
    for fishattitude in cfg.attitude.keys():
        for spawningrange in cfg.spawning_conditions.keys():
            # Create a folder for this parameter combination
            folder_name = f"{fishattitude}_{fishadventuring}_{
    spawningrange}"
            folder_path = os.path.join(cfg.sens_output_dir,
    folder_name)
            os.makedirs(folder_path, exist_ok=True)
            print (f'running the config: {fishattitude}_{
    fishadventuring}_{spawningrange}')
            myModel = FishEnvironment(cfg.input_d, folder_path, cfg.
    map_nc, cfg.spatial_resolution, cfg.temporal_resolution, cfg.
    conversion_T, cfg.xmin, cfg.ymin, cfg.xmax, cfg.ymax, cfg.nr_barbel
    , cfg.spawning_conditions[f'{spawningrange}'], cfg.adult_conditions
    , cfg.fish_exploring[f'{fishadventuring}'], cfg.attitude[f'{
    fishattitude}'])
            dynFrw = pcrfw.DynamicFramework(myModel, cfg.timesteps)
            dynFrw.run()
            # exporting the results to csvs, gpgks and tifs
            print (f'exporting the config:{fishattitude}_{
    fishadventuring}_{spawningrange}')
            export = Export(folder_path, cfg.timesteps, cfg.
    spatial_resolution)
            export.Barbel()
            print ('exporting barbel csvs...:')
            export.Barbel_csv()
            print ('exporting clump csvs...:')
            export.CommonMeuse_clumpcsv()
            print ('exporting spawn csvs...:')
            export.CommonMeuse_csv('spawn')
            export.Barbel_gpkg()
            export.CommonMeuse_tif('spawn')
            export.CommonMeuse_tif('connected_swim')
            print (f'done with the config: {fishattitude}_{
    fishadventuring}_{spawningrange}')
```
Listing 15: Batch run of the model

A.7 Exporting of data

```python
#%%
from pathlib import Path

import os
import sys
cur_dir = Path.cwd()
up_dir = cur_dir.parent
working = up_dir / 'working'
post_processing = up_dir / 'post_processing'
sys.path.append(f'{working}')
sys.path.append(str(post_processing))
import model_config as cfg
import lue.data_model as ldm
import campo
import numpy as np
import csv


class Export():
    def __init__ (self, output_d, timesteps, spatial_resolution):
        self.output_dir = output_d
        self.timesteps = timesteps
        self.dyn_timevector = np.arange (0,(int(timesteps)),1)
        self.coords_timevector = np.arange (1,(int(timesteps)+1),1)
        self.spatial_resolution = spatial_resolution
        self.fish_env = f'{self.output_dir}/fish_environment.lue'
        self.dataset  = ldm.open_dataset(f"{self.fish_env}")

    def Barbel (self):
        self.movemode_df = campo.dataframe.select (self.dataset.barbel, property_names = [f'movemode'])
        self.has_spawned_df = campo.dataframe.select(self.dataset.barbel, property_names =[f'has_spawned'])
        self.barbelarea_available_df = campo.dataframe.select(self.dataset.barbel, property_names =[f'spawning_area'])
        self.distance_df = campo.dataframe.select(self.dataset.barbel, property_names =[f'swimdistance'])

    def Barbel_gpkg (self):
        for t_coords in self.coords_timevector:
            coords = campo.dataframe.coordinates(self.dataset, "barbel", "adults", t_coords)
            tmp_df = campo.to_df(self.movemode_df, t_coords)  # is only for dataframe before starting at t =1
            campo.mobile_points_to_gpkg(coords, tmp_df,(f"{self.output_dir}/barbel_{t_coords}.gpkg"), 'EPSG:28992')

    def Barbel_csv (self):
        os.chdir (f'{self.output_dir}')
        campo.to_csv(self.barbelarea_available_df, f'available_area')
        campo.to_csv(self.distance_df, f'distance_swam')
        campo.to_csv (self.movemode_df, f'movemode')
        campo.to_csv (self.has_spawned_df, f'has_spawned')

    def CommonMeuse_sumcsv (self, property):
        #change it to make sure outputs are stored
        # self.flow_velocity_df = campo.dataframe.select(self.dataset.water, property_names=[f'flow_velocity']) # space type = static_diff_field but should be dynamic field
        # no space type distinction, however proper shape
        df = campo.dataframe.select(self.dataset.water, property_names=[f'{property}'])
        #self.depth_df = campo.dataframe.select(self.dataset.water, property_names=[f'water_depth'])
        # self.swim_df = campo.dataframe.select(self.dataset.water, property_names=[f'swimmable'])
        total_area = np.zeros ((self.timesteps+1))
        #for t in agent_timevector:
        for t in self.dyn_timevector:
            # let op : neemt alleen laatste key mee!!!!! als df
            raster = df["water"]["area"][f'{property}'][0][t]
            total_area [t+1] = self.spatial_resolution**2*np.sum(raster)

        total_csv = f'{self.output_dir}/{property}.csv'
        with open(f'{total_csv}', 'w', newline='') as f:
            # Create a CSV writer object
            csv_writer = csv.writer(f, delimiter=',',quoting=csv.QUOTE_MINIMAL)
            csv_writer.writerow(total_area)

    def CommonMeuse_clumpcsv (self):
        connected_swim_df = campo.dataframe.select (self.dataset.water, property_names=[f'connected_swimmable'])
        nr_clumps = np.zeros ((self.timesteps+1))
        for t in self.dyn_timevector:
            connected_swimraster = connected_swim_df['water']["area"]['connected_swimmable'][0][t]
            nr_clumps [t+1] = np.max (connected_swimraster)

        clump_csv = f'{self.output_dir}/clump.csv'
        with open(f'{clump_csv}', 'w', newline='') as f:
            # Create a CSV writer object
            csv_writer = csv.writer(f, delimiter=',',quoting=csv.QUOTE_MINIMAL)
            csv_writer.writerow(nr_clumps)

    def CommonMeuse_tif(self, property):
        '''property: may be 'connected_swim','flow_velocity','water_depth', 'swimmable', 'spawn'. Type =String'''
        df = campo.dataframe.select(self.dataset.water, property_names=[f'{property}'])
        for t in self.dyn_timevector:
            # let op : neemt alleen laatste key mee!!!!! als df
            raster = df["water"]["area"][f'{property}'][0][t]
            campo.to_geotiff(raster, (f"{self.output_dir}/{property}_{t+1}.tif"), 'EPSG:28992')
```
Listing 16: Exporting data from LUE data file

B Figures

B.1 Fish movement

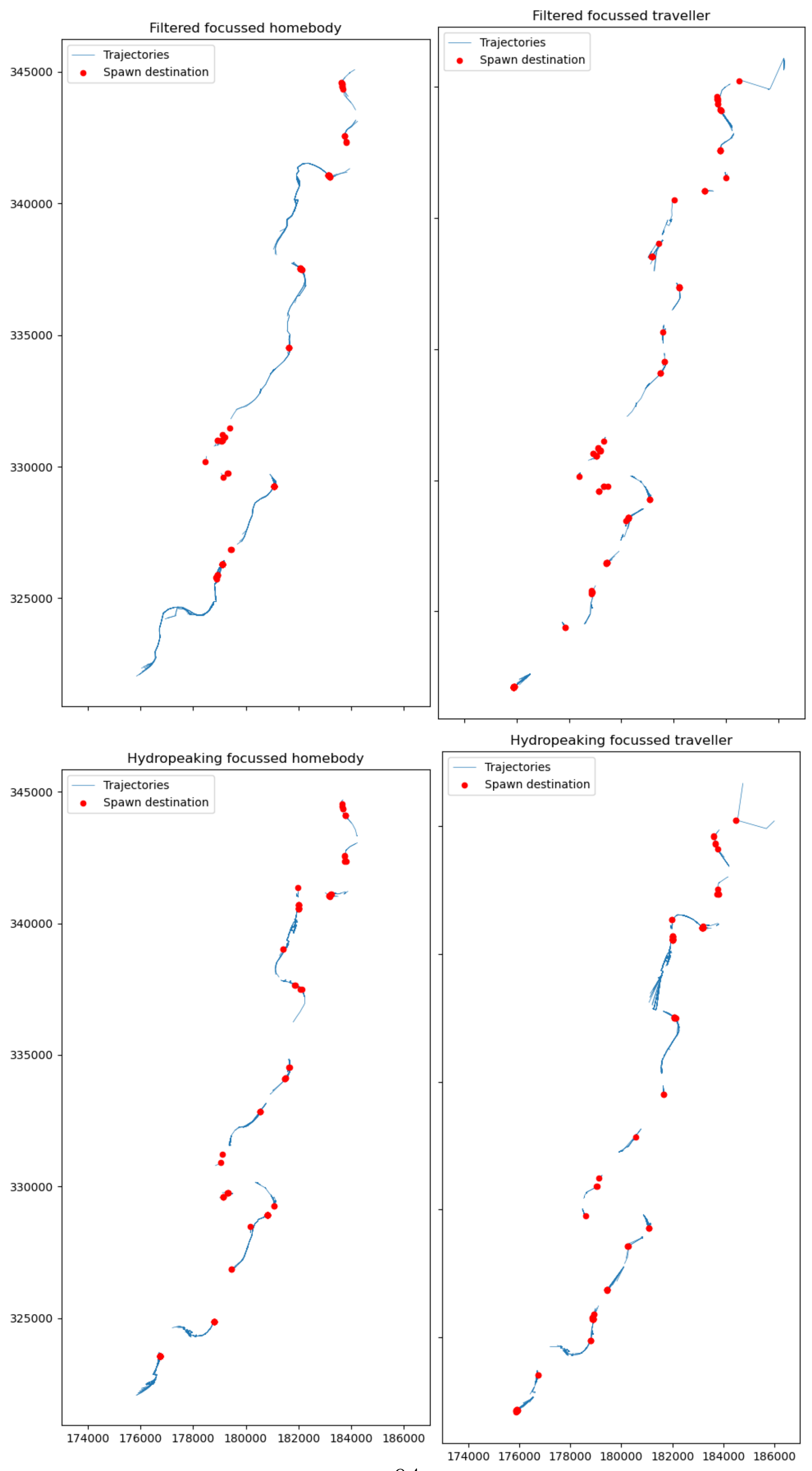

Figure B.1: Trajectories and destination for spawning by the Barbel considering different flow regimes.

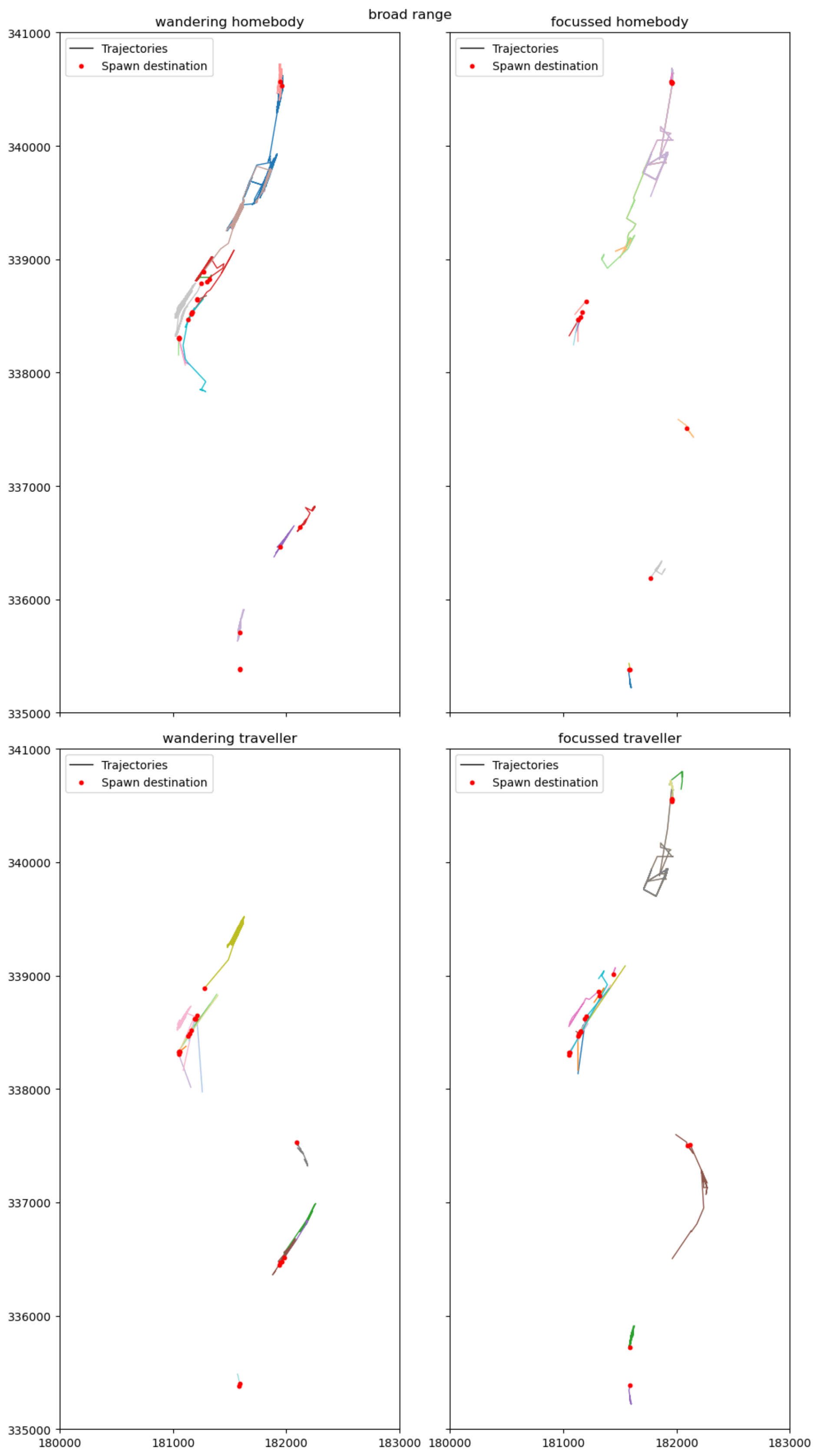

Figure B.2: A zoom in of trajectories and destination for spawning by the barbel with different behaviours, each having a broad preference for flow regime. Each distinct coloured line represents the track of a single group of barbels.

B.2 Movemodes

The most successful spawners are wandering travellers with a broad range of spawning preferences, moving about 5000 times in total, averaging 50 moves or 200 hours before spawning. These barbels find spawning grounds the fastest. In contrast, focused homebodies with the initial range of preferences perform approximately 38,000 movements, while other spawners move between 7,000 and 21,000 times. All focussed barbels spend a relatively bigger number of movement transitioning from uninhabitable to habitable areas compared to wandering barbels.

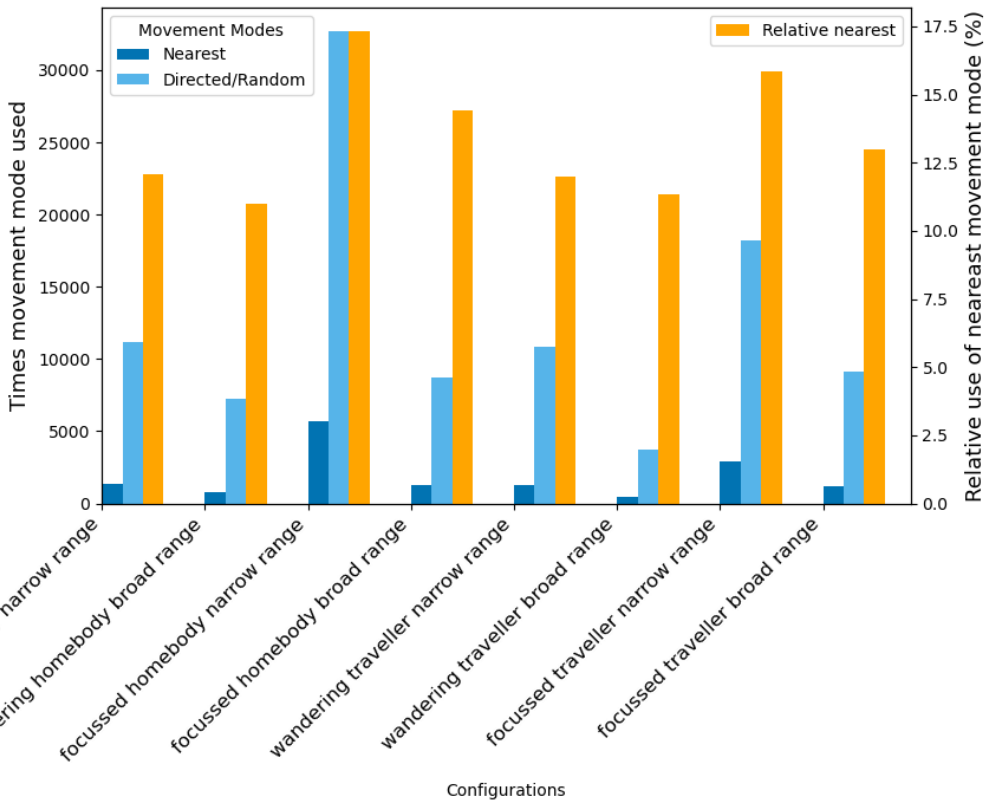

Figure B.3: The movement modes accessed by the barbel agents in the model for the various scenarios.