CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

A Generic Model of Motivation in Artificial Animals Based on Reinforcement Learning

Master's thesis in Computer science and engineering

BIRGER KLEVE AND PIETRO FERRARI

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2021

Master's thesis 2021

A Generic Model of Motivation in Artificial Animals Based on Reinforcement Learning

BIRGER KLEVE AND PIETRO FERRARI

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2021

A Generic Model of Motivation in Artificial Animals Based on Reinforcement Learning
BIRGER KLEVE AND PIETRO FERRARI

Supervisor: Claes Strannegård, Department of Computer Science and Engineering
Examiner: Devdatt Dubhashi, Department of Computer Science and Engineering

Master's Thesis 2021
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Gothenburg, Sweden 2021

A Generic Model of Motivation in Artificial Animals Based on Reinforcement Learning

BIRGER KLEVE AND PIETRO FERRARI
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

This thesis is a part of a broader research project at Chalmers University of Technology focused on ecosystems' simulations using reinforcement learning artificial animals, called *animats*. The scope of this project is to provide animats with a reward signal which should ultimately drive animats' learning towards adaptation of their environment.

We introduce a framework based on basic biological mechanisms of homeostatic regulation, i.e. the regulation of physiological conditions, to reward animats for maintaining their optimal homeostatic state, i.e. for maintaining *homeostasis*. As such, homeostasis is each animat's objective.

Previous, theoretical work adopting homeostatic regulation as a mechanism of reward generation lack the ability of regulating needs' importance and needs' interaction, and as shown by our results fail in environments where animats eventually die.

We extend on previous theoretical efforts of modeling homeostatic regulation by defining the animat's happiness as a function of its needs through several simple univariate utility functions. Modeling the utility of each need singularly enables high flexibility in design and easily configurable interactions between different needs. Moreover, in this framework vital needs have priority over non-vital or sensory needs.

We show that this framework can be used to elicit six important animat behaviors, emulations of real animal behaviours, and in particular can be used to recreate typical behaviours observed in free-living planktonic copepods such as quick escape reactions from fast-approaching predators and diel vertical migration. We compare 2 models for reward generation utilizing different happiness functions to previous theoretical work and to a generalization of said previous work in a diverse array of environments, showing that one model of motivation is superior in all tested environments and allows animats to learn the six objective behaviours. The models are also compared against a baseline reward, rewarding staying alive. We show that the proposed models produce a better performance compared to the baseline model, implicating that motivational models based on homeostatic regulation are a good choice for reward generation for animats. Finally, we test the models in a more general marine environment, showing that using this framework animats can learn copepod behaviour.

Keywords: reinforcement, learning, reward, shaping, animat, homeostasis, motivation, ecosystem.

Acknowledgements

We would like to thank our supervisor Claes Strannegård for giving us the possibility to work at this project, part of the bigger project Ecotwin, currently in development at Dynamic Topologies. We had the fortune to work in close contact with Hans Glimmerfors, Victor Skoglund and Tobias Karlsson, who have also been developing their Master's thesis for Ecotwin and have been a constant resource of ideas, support and productive discussion. Finally, we would like to thank Marcus Hilding Södergren for creating the original Ecotwin framework which was used as the basis for our work.

Pietro Ferrari & Birger Kleve, Gothenburg, June 2021

Contents

List of Figures

List of Tables

1

Introduction

Ecosystem simulations are useful in a variety of applications, from the estimation of fish tonnes in different marine areas during different seasons, to avoiding animals' extinction through the prediction of their future population trends.

One of the branches of population dynamics models is that of computer simulated animal interaction. Such models are known as *computer models* and can be used to study population dynamics by simulating large scale population interactions. When organisms in the population are modeled on an individual level, the model belongs to individual-based models. If such individuals are reinforcement learning agents, the model is referred to as an agent-based model and finally, when agent-based models aim at simulating artificial animals, as in this project, the agents are called *animats*.

Animats were first introduced by S. Wilson in 1986 [1]. They constitute a large field of interest and exploration in artificial intelligence [2].

The basic hypothesis that drives research in general artificial intelligence through animats is that human common sense can be reached by simulating and understanding simpler animal-like systems [2]. Wilson compare the animat approach to artificial intelligence to the approach of Turing's *child-machine* [3], reasoning that both are to be equipped with a learning-through-experience apparatus and to be situated in a sensory environment, hence the learning. This holistic approach to artificial intelligence is on the other side of the spectrum to the standard approach, which aims for excellent performance in very specific tasks, such as chess playing[2].

For this reason, we believe that advancement in animat research, by the ability to construct general artificial animal agents that can learn how to succeed in an environment, might also benefit the progress in general artificial intelligence.

With agent-based modeling of ecosystem comes the question of defining a proper reward for the reinforcement learning agents. Rewards can be yielded as a consequence of certain agent actions or certain environmental events, but some problems might arise. The reward signal in reinforcement learning tends to be sparse, delayed and uninformative. Standard approaches of Reinforcement Learning can achieve asymptotically optimal solutions, if allowed sufficient training, in stationary environments[4]. However, in several contexts, such as biological simulations with reinforcement learning artificial animals, the environment is not stationary and agents' lifespan is usually limited. Agents might thus die before benefiting from asymptotic results.

An existing alternative approach for providing a reward signal to animats uses the concept of homeostatic regulation[5].

1.1 Homeostasis for Animats

Homeostasis is the biological mechanism observed in nature, which regulates the internal, physiological conditions within animals[6]. For this reason, homeostasis can also be called homeostatic regulation.

The approach of reinforcement learning for animats with a representation of physiology leads to the concept of homeostatic regulation. If such physiological representation is sufficiently descriptive of the simulated animal's physiology, it also becomes possible to define homeostasis as the animats' aim, i.e. their motivation.

In this vein, it has been theorized by Keramati et al. that a way of producing a reward signal for animats is through homeostatic regulation[5]. They introduced a basic functional form of drive towards a desired physiological state, defined on the space of the agent's internal homeostatic variables. Through the difference of the animat's drive in time, they derive the reward used by the reinforcement learning algorithm.

However, Keramati et al.'s model suffers from lack of generality. Infact, all homeostatic variables possess the same weight on the reward and there is limited interaction between variables, not allowing the reward function to be adaptable in scenarios where certain variables are more critical than others. These flaws showed repercussions in our experiments, and in particular when animats regulate both vital and non-vital needs.

1.2 Animats' Happiness for Reward Generation

We thus introduce two new generic models of motivation for animats, based on a definition of animat's *happiness*. The models are inspired by homeostatic regulation and produce a reward signal for maintaining homeostasis. Moreover, these models associate agents' sensory perception to the homeostatic variables in order to anticipate future rewards by sensory cues. Additionally, we extend the work of Keramati et al.[5] by generalizing their proposed drive function with the use of sensory variables and the introduction of variable-specific weights, alleviating its critical aspects mentioned above.

1.3 Models Evaluation and Copepod Behaviours

We test said motivation models in a number of ecosystem simulations built with the Unity game engine[7]. We show the utility of homeostatic regulation for animats and, in particular, we test the models' ability to simulate a few characteristic behaviours of planktonic copepods, which are quite sophisticated with respect to their limited cognition[8, 9, 10]. Finally, we compare the different tested motivational models and conclude by attesting the superiority of one model, which succeeds in all environments.

This project is a key part of a broader research project on ecosystem simulations at Chalmers University of Technology and Dynamic Topologies AB, using

reiforcement learning artificial animals, i.e. animats, with the purpose of building multi-agent simulations of ecosystems. Our scope is to provide animats with a reward signal based on a set of accurately defined physiological variables. Ultimately, the reward signal should drive animats' learning towards adaptation to the environment and survival.

The code developed during this project will be released as open-souce.

1.4 Aim

The aim of this project is to create a generic model of motivation for animats, i.e. a model that is efficient in several, diverse reinforcement learning scenarios. Each scenario that we consider is a reinforcement learning environment focusing on particular aspects of real ecosystems. In particular, a few of the environments in which our model is tested serve to replicate some behaviours typical of planktonic copepods, such as their daily vertical migrations and the detection and avoidance of predators.

The main research questions in the project are:

- Is homeostatic regulation a feasible generic model of motivation in artificial animals based on reinforcement learning?
- Can such a model be used for replicating basic behaviors observed in some copepod species?

1.5 Problem Description

In Biology, *taxis* and *kinesis* refer to the movement of an animal in response to stimuli. taxis is specific and directed motion while kinesis is undirected and random motion. Various forms of taxis and kinesis exist, for example, *chemotaxis* refers to a specific motion in response to a chemical stimulus, such as following the smell of a scent trail, and *barotaxis* refers to the response to a pressure stimulus, such as pressure changes caused by animals moving in water. This project aims at providing a general model of basic cognition based on *taxis* and *kinesis*.

Computationally, the model performs reward shaping by introducing information-dense rewards, e.g. in the form of smells or visual stimuli. This approach overcomes problems of sparse reward signals, greedy optimization and aims at achieving additional common sense such as feeling threatened by predators. As such this generic model of motivation might be a step towards relatively flexible AI agents with separate motivational modules that are easy to replace and modify, and a key part in simulating animats.

In particular, a set of six behaviours of animats will be replicated:

- B1: Regulation of hunger when food is scarce, only eat when not satiated.
- B2: Chemotaxis, by selective eating by differentiating different types of nutritious food and harmful food.
- B3: Chemotaxis, by following scent trails.

- B4: Phototaxis, by moving towards and away from light. This is observed in free-living planktonic copepods' and known as Diel Vertical Migrations (DVM). The migration happens daily, between the bottom of the *daylight zone* - top of the *twilight zone* and the surface of the body of water. In order to avoid visual predators, copepods hide in the deep water, where little sunlight can reach, during the day, and raise to the water surface regions, where more food can be found, during the night [11].
- B5: Barokinesis to escape close and fast-approaching predators.
- B6: Chemotaxis to escape predators sensed by scent.

For free-living planktonic copepods, behaviours B4 and B5 have been observed in the wild [12, 11, 9], while behaviour B6 has been observed in laboratory conditions [13].

The evaluation setting will consist of six separate environments where, respectively, each of these six behaviours will be necessary for the animats to survive. The survival time of animats using homeostatic regulation will be compared to a baseline agent simply using survival time as reward function, i.e. at each timestep alive, it will receive a constant reward. Moreover, in each environment, different reward signals based on homeostatic regulation, which we define as happiness functions, are compared.

1.6 Scope

This project will not incorporate evolution, which is a key part of ecosystems. This limits the ability of agents to learn from several episodes, in a realistic manner. Instead we use pretraining of the models, which can be interpreted as an emulation of evolution, leading to a current state of animals' behaviour, adapted to the environment.

In addition, in environments where predators are present, since they are not the focus of this particular project, their implementation might not be sophisticated enough to simulate real world predators as their purpose solely consists on evaluating their prey animats.

Finally, as the focus of this thesis is on motivation and reinforcement learning, the realism of the simulation is limited to what is feasible within the time limit of this thesis.

This project, being part of a larger project on ecosystems simulations, takes large inspiration from real observed natural phenomena. Each designed environment is meant to focus on one particular aspect of ecosystem dynamics, and thus on one particular corresponding animal behaviour. The behaviours of reference have been listed in Section1.5.

A necessary component in this project's goal is to construct a framework for dynamically complex digital ecosystems. Thus, while simulations have been designed striving for realistic representations of particular aspects of ecosystems and animal behaviour listed in Section1.5, since real ecosystem dynamics are highly complex and the result of co-evolution between an high number of living beings, simplifications of real dynamics had to be made, see Section6.1.

1.7 Contribution

In this thesis we propose a way of building motivational modules for animats. These modules combine internal homeostatic signals like energy level with external signals like smells to create a reward signal.

During this project, significant work has gone into building the experiments and implementing a general framework for various, diverse ecosystem simulations. In particular, a great portion of our work dealt with the code structure for animats and environments. We implemented animat's senses and dynamics modularly, allowing different types of animats, and environments, to be easily instantiated by selecting modules and configurations they use.

A module for homeostatic regulation, which is the core of this project, was implemented in an accessible yet flexible library, allowing further research to build on our work. Particular care was taken to allow homeostatic regulation to work with preconfigured values, when such values are not specified.

Finally, we compare a selection of new models, which allow flexible interaction between different homeostatic and sensory variables, with previous work related to homeostatic regulation.

2
Background

2.1 Overview on Ecosystem Simulations

One the most renowned mathematical models of an ecosystem is the population-prey dynamical system proposed by Lotka A.J.[14] and Volterra V.[15]. This model was originally intended, in Volterra's work, to explain the fluctuations in marine populations of fish and sharks in the Adriatic Sea, and introduces key features of species interactions such as predation and breeding in the form of a population's growth rate and mortality.

Population dynamics models, such as the predator-prey model, are called mathematical models or analytical models, as they are defined by a set of conditions specifying the relations between observed variables and the parameters.

Analytical models can describe phenomena on very large scales, when said relations between the observables and the parameters are realistic. Nonetheless, real biological dynamics are highly complex, accounting for spatial interactions and additional environmental conditions can drastically raise the model's complexity, requiring other approaches to biological models.

Experimental models constitute an additional type of population dynamics study, for example by isolating different bacteria cultures in a Petri dish in laboratory conditions[16]. Nonetheless, this kind of models are not always applicable when scale is augmented and can raise ethical concerns.

Lastly, the branch of population dynamics models which we treat in this work is that of computer simulated animal interaction. Such models are known as *computer models* and can be used to study population dynamics by simulating large scale population interactions.

A more recent example of an computer model is the Ecopath simulator for marine ecosystems[17], which considers a marine environment as a grid where each block has a set of attributes such as a specified population number for each specie considered. Each specie's diet along with other characteristics are defined, allowing the simulation to be run automatically.

2.1.1 Individual-based Models

One branch of computer models, simulates biological populations by modeling organisms at an individual level[18]. These are called *individual-based* models.

Examples of individual-based models are the JABOWA model[19], which simulates the succession of tree communities in forests where a gap has been created, multiple instances of models simulating the recruitment in fish populations, reviewed

by Sibly et al.[20], or models for assessing the risk of extinction of other animal species, such as the brown bear in the Cordillera Cantabrica, northern Spain[21].

Often when modeling biological populations on an individual scale, environmental factors are important in the dynamics. Marine or terrestrial environments might be detailed with features, striving to resemble real environmental features, such as light conditions[19], terrain and other topographic conditions[21], chemical concentrations and so fourth.

Some individual-based models describe individual attributes as a fixed set, or as a set of attributes evolving in a fixed manner. In this cases, animals can be considered as reflex machines, since their response to the environmental conditions is predictable by an external observer. This can constitute a limitation when it is intended to model individuals which can learn from the environment, striving to adapt to changing environmental conditions.

2.1.2 Agent-based Models

An alternative for ecosystem simulations with learning animals exists through *agent-based* models, which are characterized by the use of reinforcement learning algorithms to determine the behaviour of the entities, called agents, within the environment.

The use of simulations and the utilization of reinforcement learning provide a distinct point of view on ecosystems compared to the population-dynamics models based on differential equations, such as the Lotka-Volterra model[22, 15], possibly allowing for new insights into the complexity of ecosystems. Moreover, if highly-adaptive and efficient behaviours are obtained, this approach might give greater insights into cognition, general intelligence and evolution.

2.2 Animats

Animat behaviour is built as a simulation of animals' necessity for need-satisfaction, thus setting the *inductive bias* in the reinforcement learning framework [23]. These needs can often be represented by a *homeostatic variable* such as *energy, water* or a *mating hormone*, which reflect different physiological conditions and characteristics of an animal at any given moment in time [24]. Since animals experience varying degrees of need-satisfaction, an intuitive way of modeling homeostatic variables is as real-valued variables in the closed interval $[0, 1]$.

2.2.1 Animat Cognition

A *sense* is an agent variable dependent on the environment state and the agent's action. Borrowing the terminology used in an analogy between the animat and finite-state machines by S. Wilson [2], the sensory stimulus E at the next timestep can be described by:

$$E(t+1) = G(Q(t), A(t))$$

where Q is the environment's state, A the agent's action and G a general function, serving as a filter between the environment and the agent's observations.

At any given timestep, the agent performs an action in the environment. Examples of possible actions are *move right, move left, idle, attack prey*. The effect of any given animat action on the environment, of which the animat is part, are handled by the environment itself, including the eventual non-feasibility of an action such as *attack prey* when no prey is present.

The decision-making is handled by the *agent's brain*, a reinforcement learning algorithm which, given a set of sensory stimuli and homeostatic variables of the agent, decides on one of the possible actions. Moreover, as in any reinforcement learning framework, the agent receives a reward signal from the environment.

2.3 Reinforcement Learning

Reinforcement Learning, RL, is used in this project to facilitate the cognition of the animats, i.e. their decision making and learning. The purpose of this section is to introduce the key concepts and build up to the algorithms and techniques used in this project. The choice of algorithm and its parameters play an important role for the behaviour of the animats. However, the most important concept for this project is the *reward signal*, as this is the basis of motivation for animats which is the subject of this thesis.

Learning is performed by an *agent* A which manipulates its *environment* E using a set of actions. The environment updates itself after an action is taken and emits a new *state* s and reward signal r. The reward signal's purpose is to guide the agent into desired behaviour, by promoting actions that lead to good outcomes with a larger reward and discouraging actions that lead to bad outcomes with a lower reward[25]. Reinforcement learning can be seen as an interaction loop, illustrated in Figure 2.1. Note that the states might not be fully observable, i.e. the agent might not be able to observe the full state of the environment, which is especially true in complex environments such as ecosystems.

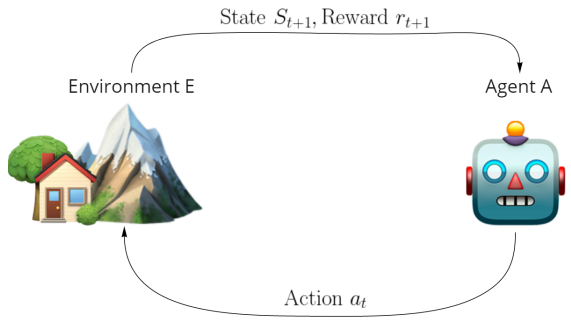


Figure 2.1: Agent-Environment interaction loop in Reinforcement Learning.

The agent's goal is to learn the best actions to perform, depending on the state, in order to maximize its expected cumulative reward, i.e. the sum of all reward it manages to collect. An episode is the time between the beginning of the

agent's interaction with the environment and the terminal state, be it death, a preset maximum number of time steps or another termination condition.

At each time step t the agent selects an action a_t using its *policy* π, depending on the agents state s_t [25]. More specifically, the policy is a distribution over actions which the agent can sample:

$$a_t \sim \pi(\cdot|s_t)$$

Performing action a_t on the environment will update the state s_t to s_{t+1} and emit a reward r_t. Each episode consists of a set of interactions, also called *trajectory*, which contain a sequence of states and actions:

$$\tau = (s_0, a_0, s_1, a_1...., s_T)$$

where T is the episode length. The sum of all rewards in a trajectory is called the return. Future rewards can be discounted using a *discount factor* $\gamma \in [0, 1]$, thus the return takes the form:

$$R(\tau) = \sum_{t=0}^{T} \gamma^t r_t$$

The discount factor γ is important as it controls how short sighted the agent should be. That is, how much weight it should give to future, less certain, rewards compared to rewards in the immediate future. A low γ values immediate rewards significantly more, while a large γ distributes value of future rewards more evenly, albeit still valuing more rewards that are closer in time. Only if $\gamma = 1$ each reward within the time horizon T is given an equal weight.

The agent's objective is to maximize its expected cumulative reward, which can be expressed as optimizing its expected *undiscounted* return, i.e. with $\gamma = 1$. The expected return from a policy π over a trajectory τ is called $J(\pi)$:

$$J(\pi) = E_\pi[R_{\gamma=1}(\tau)]$$

It is important to note that the trajectory of following a policy is a random variable, as both the policy and environment are stochastic processes. While the expected return depends on both the policy and the environment, we choose to only subscript the policy, as that captures the animat's behaviour, which is what the animat learns.

The optimal policy can be defined as:

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, J(\pi) \tag{2.1}$$

From this follows the *value function* $V^\pi(s)$, which gives the value of being in a state s and following the policy π, as the expected return:

$$V^\pi(s) = E_\pi[R(\tau)\,|s_0 = s]$$

where τ is the trajectory starting in s. The superscript policy π emphasizes that the agent will continue following, or sampling actions, from the policy π.

Similarly, the *action-value function* $Q^\pi(s, a)$ gives the expected return starting in state s, performing action a, and then following the policy π:

$$Q^\pi(s, a) = E_\pi[R(\tau)\,|s_0 = s, a_0 = a]$$

The *advantage function* $A^\pi(s, a)$ captures how much better action a is in state s, relative to the average behaviour of policy π:

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$$

2.3.1 Reinforcement Learning Methods

Reinforcement Learning methods can be categorized by what they learn, and whether or not they are given a model of their environment. In this section the distinctions of these groups are outlined, and the choice of specific method for our simulations is motivated.

Methods that are provided models of their environment are called *model-based* while methods that needs to explore their environment are called *model-free* [25]. Model-based methods allows finding the optimal policy through planning, without interacting with the environment. These methods require an accurate model of the environment. Since in this project the environment simulates an ecosystems with possibly multiple agents and stochastic elements, creating a sufficiently accurate model is not feasible. Moreover, even when feasible, a model-based method would be out of the scope when simulating animal's perception in an environment. The other category, model-free methods needs to learn both their environment, by exploring, and how to exploit it. Model-free methods can be further categorized by what is being learned.

Value-based model-free methods learn a function approximation of the optimal policy's action-value function:

$$Q^{\pi*}(s, a; \theta) \approx Q^{\pi*}(s, a)$$

with θ as the function approximation's parameters. The action-value function for some policy π can be learned using trajectories from some other policy π'. As such, these methods can make use of a second exploration policy to explore its environment, making them more resilient to local optima. Such methods are thus also called *off-policy* methods [26].

Policy-based model-free methods instead aim to learn the policy directly. As the same policy that is being learned is also used to explore the environment, these methods are called *on-policy* methods.

One such method is the *Policy Gradient*, which parameterizes the policy $\pi(a|s; \theta)$ and updates the parameters θ using gradient ascent to maximize the expected return $J(\theta) = E_\theta[R(\tau)]$. In other words, these methods try to directly find the optimal policy in Equation 2.1.

The gradient ascent updates for the parameter θ_k are thus:

$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta J(\theta_k) = \theta_k + \alpha \nabla_\theta E_{\theta_k}[R(\tau)]$$

where α is the learning rate, i.e. the size of the parameter updates [25][26]. Using the unbiased estimator $\nabla_\theta \log \pi(a_t|s_t; \theta) R_t$ of the gradient $\nabla_\theta E_{\theta_k}[R(\tau)]$, the gradient ascent rule can be written as [26]:

$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta \log \pi(a_t|s_t; \theta) R_t$$

Policy gradient methods of this form has unstable learning due to their high variance estimates, and are poorly suited for neural networks policy function approximators. Additionally, because they use the expected return of the *current* policy as loss function, when the policy's parameters are changed, the expected return

changes distribution thus making the previous loss function irrelevant. The policy gradient methods are thus prone to converge to local optima [27].

A third category of methods, known as *Actor-Critic* methods, tries to combine the policy- and value-based methods. By using Actor-Critic methods, the variance of the policy gradient methods can be reduced. This is done by subtracting the return with some baseline $b_t(s_t)$ which reduces the variance and keeps the estimator unbiased [26]:

$$\theta_{k+1} = \theta_k + \nabla_\theta log\pi(a_t|s_t;\theta)[R_t - b_t(s_t)]$$

Note that by selecting the baseline to an approximation of the value function $b_t(s_t) = V(s_t;\theta')$ the term $R_t - b_t(s_t)$ can be written as an approximation of the advantage function:

$$R_t - b_s(s_t) = R_t - V(s_t;\theta') = Q(a_t, s_t) - V(s_t;\theta') \approx A(a_t, s_t)$$

since R_t is an estimator of $Q(a_t, s_t)$ [26]. Policy gradients using this method to reduce their variance is called actor-critic as they learn both the actor π and the critic $A(a_t, s_t)$ [25][26].

An important requirement of the reinforcement learning methods used in this project is that they are on-policy methods. As on-policy methods is required for lifelong learning, which simulate animals' learning during their lifetime. One state-of-the-art on-policy method are the Proximal Policy Optimization method, *PPO*, which outperform other online policy gradient methods [27]. On-policy methods are not as sample efficient as off-policy methods as they can not reuse experience, and are less stable, which previously has made them infeasible to train neural networks. However, PPO mitigates the stability issues with a bias-variance trade-off to estimate its critic, the advantage function, and controls the size of its policy updates. Additionally, PPO uses asynchronous samples from several agents to update one policy, which facilitates several minibatch epoch updates of neural networks as it breaks the correlation between samples. The underlying function approximator trained by the PPO uses a two layer 256-neuron LSTM, for both the policy and value estimation.

2.3.2 Proximal Policy Optimization

The key idea of the Proximal Policy Optimization algorithm is to directly optimize the policy in such a way that the new policy is not drastically changed. That is, striving for reasonable policy updates [27].

PPO extends on-policy gradient methods and tries to mitigate their high variance by controlling their main flaw, i.e. that they might collapse into local optima as they optimize a loss function that is not directly connected to the actual expected return [27].

PPO uses a "surrogate" loss function based on the probability ratio $r_t(\theta_k) = \frac{\pi_{\theta_k}(a_t|s_t)}{\pi_{\theta_{k-1}}(a_t|s_t)}$ and the advantage function $A(s_t, a_t)$ [27]:

$$L(\theta_k) = E\left[\frac{\pi_{\theta_k}(a_t|s_t)}{\pi_{\theta_{k-1}}(a_t|s_t)}A(s_t, a_t)\right] = E[r_t(\theta_k)A(s_t, a_t)]$$

The advantage function's sign is positive if the action a_t yields higher expected return than the policy's default behaviour in state s_t while the probability ratio indicates whether the policy update is more likely to select action a_t. If the action is better than the default behaviour, the gradient ascent should increase the probability ratio $r_t(\theta_k)$ for the updated policy to select a_t compared to the previous policy.

PPO controls the magnitude of policy updates by clipping its loss function:

$$L^{clip}(\theta_k) = E_t\Big[\min\Big(r_t(\theta_k)A(s_t,a_t), \text{clip}(r_t(\theta_k), 1-\epsilon, 1+\epsilon)A(s_t,a_t)\Big)\Big]$$

The clip parameter ϵ controls how far the new policy update is allowed to update from the old, creating a pessimistic bound for the performance of the policy [27].

PPO estimate the advantage function using Generalized Advantage Estimation, GAE, which enables efficient trade-off between bias and variance using an exponential weighted-average of advantage estimators:

$$\hat{A}^{GAE(\gamma,\lambda)}(a_t, s_t) = \sum_{l=0}^{\infty}(\gamma\lambda)^l[r_{t+l} + \gamma V(s_{t+1+l}) - V(s_{t+l})] \tag{2.2}$$

where γ is the discount factor, discussed above, down-weighting distant rewards and thus reducing the variance at the cost of bias, and the exponential-weighted average parameter $0 < \lambda < 1$ controlling bias variance trade-off between the high variance, unbiased advantage estimator, with $\lambda = 1$:

$$\hat{A}^{GAE(\gamma,\lambda=1)}(a_t, s_t) = \sum_{l=0}^{\infty}\gamma^l r_{t+l} - V(s_t)$$

and the biased low variance estimator with $\lambda = 0$:

$$\hat{A}^{GAE(\gamma,\lambda=0)}(a_t, s_t) = r_t + \gamma V(s_{t+1}) - V(s_t)$$

Note that both parameters γ, λ control the bias-variance trade-off, where λ is far more efficient in controlling variance [28].

PPO shares weights between the value function estimator used in $\hat{A}^{GAE(\gamma,\lambda=0)}(a_t, s_t)$ and the policy. It is common to only let the last layer differ between the policy, which uses a softmax layer, and the value function, which uses a linear layer. Since they share weights the parameters must be updated on both objectives by adding the squared-error value loss:

$$L^{VF}(\theta) = (V(s_t;\theta) - V_t^{target})^2 \tag{2.3}$$

to the surrogate loss [26] [27].

Finally, an entropy term is added to discourage premature convergence and thus improve exploration [26][27]. The final loss function becomes:

$$L^{clip+VF+S}(\theta_k) = E_t\Big[L^{clip}(\theta_k) - c_1 L^{VF}(\theta_k) - c_2 S[\pi_{\theta_k}](s_t)\Big] \tag{2.4}$$

where c_1, c_2 are coefficients that can be tuned and S is the entropy bonus term for the policy [27].

PPO enables training neural networks by breaking sample correlation using asynchronously sampled trajectories from multiple agents. This architecture is called

Asynchronous Advantage Actor-Critic, A3C, which has shown great performance in training neural networks over several tasks [26] [27].

For each training iteration t one or more agents collect experience for T time steps, where T is the *time-horizon*, using the current policy π_{θ_k}. After collecting the experience, the parameters are updated using minibatch gradient ascent for K epochs on the surrogate loss function $L^{clip+VF+S}(\theta_k)$. Note that the number of epochs K controls how well the model will optimize to the surrogate loss, that is the variance of the model [27]. Training for a large number of epochs K will lower the loss, and thus bias, for the specific experience collected. This increases the variance of the model as experience collected with future policies might be different. PPO is able to run several epochs with tolerable levels of variance and low risk of collapsing into local optima, using the A3C architecture, GAE with appropriate parameters, asynchronous samples and the surrogate loss function with clipping [27] [26].

2.3.2.1 Hyperparameters

The PPO has a set of important hyperparameters [27]:

- Clip ratio ϵ - Controls how large the updates for new policies should be with respect to the previous policy.
- Time horizon T - Controls variance/bias trade-off. Using a longer time horizon gives a less biased but higher varianced estimate. T should be large enough to capture important behaviour within a sequence of actions, but should be much less than an episode's length.
- Value function coefficient c_1 - Controls the weight of the value function loss in the surrogate.
- Entropy coefficient c_2 - Controls the strength of the entropy used to facilitate exploration. Note that this decreases with learning.
- epochs K - Number of epochs to run the mini-batch updates. More epochs means higher variance as the model fits better to the current set of exploration; lower value thus means more stable learning, albeit slower.
- Discount factor γ - Controls how far into the future rewards are relevant for the agent.
- Learning rate α - Controls the size of updates in the gradient ascent. It is common to linearly decrease the learning rate as training progresses.
- GAE smoothing λ - Controls the smoothing of the GAE estimator.

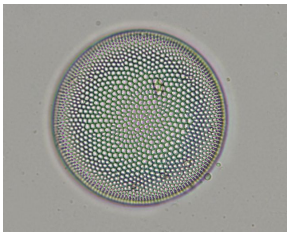
2.4 Copepods

The purpose of the experiments in this project is to elicit some general animal behaviours. Behaviour B4 and B5 in particular are specific references to behaviours observed in *planktonic copepods*.

Copepods are a subclass of crustaceans, comprising of 10 orders and more than 25.000 currently known species [29]. Copepods can be free-living or parasitic, copepods can also be planktonic or benthic, where the former are drifting in sea waters, while the latter live on the ocean floor.

Since we are interested in copepod behaviours exhibited by some free-living planktonic copepods, our referenced copepod is of the order *Calanoida*, which are free-living and planktonic, see Figure 2.2a. Calanoid copepods are teardrop shaped with long antennae which they use as means of mechanoreception and chemoreception [30]. They typically range between 0.5mm and 2mm in length. Calanoid copepods are an essential specie in marine food webs, in fact they constitute the majority of oceans' plankton [31].

(a) A calanoid copepod.



(b) A coscinodiscus.

(c) A krill.

Figure 2.2: Images of reference animat, food and predator used in the simulations.

2.4.1 Diel Vertical Migration

Behaviour B4 references to copepods' Diel Vertical Migration, *DVM*, which is a typical behaviour of some copepod species, including calanoid copepods. At night these organisms feed near the water surface, where phytoplankton grows, while during the day they sink, by turning their oils into lipids [10], into the deeper levels of the body of water, reaching the bottom of the daylight zone or even the top of the twilight zone, accounting for a drop of around 200m [32]. This behaviour has been linked to the presence of visual predators, which can threaten copepods' survival when and where light is more intense, i.e. especially near the surface of the water during daytime [33].

2.4.2 Quick Escape Reactions

The modality of locomotion can differ substantially between different species of free-living copepods: an example is *Clausocalanus furcatus*, which was shown to move at a speed of around $10mm\,s^{-1}$, which corresponds to approximately 10 body lenghts per second, along convoluted small loops, with occasional immotile behaviour which can lead to sinking, and some high-speed somersaultings [34]. One motile behaviour of interest, typical of most planktonic copepods including calanoid copepods, is an extremely fast jump produced by specialized propulsion muscles.

Such reactions generally happen when a predator is sensed, leading to very high speed jumps over few centimetres [9], which are in fact a matter of life or death for copepods. It is thus logical that selective pressure has led to many species having neurons surrounded by myelin for increased neuronal conduction speed and reactivity, which enables quicker escape reactions [8].

Since most copepods have very poor vision and some live in rather muddy waters, these copepods mostly rely on chemical and hydromechanical cues to detect, avoid and escape predators [35]. Underlining the causality between sensed turbulence or fluid deformation and the fast escape response, it has been shown that copepods are successfully hunted by slow-swimming seahorses, which, although being much larger than copepods, manage to approach them gradually and slowly enough to not produce any perceivable turbulence, until they have the preyed copepods within reach. Then, finally, they quickly suck them into their snout [36].

In environment E5 we enable the copepod animats to learn this fast escape jump behaviour as a reaction to sensing the fluid deformation produced by an approaching predator. We label this behaviour B5.

2.4.3 Predator

Meganyctiphanes norvegica, known as krill, are the most common predators for copepods, see Figure 2.2c. Most krills are between 1cm and 2cm long. The reference krill predator used in this project is 5 times larger than our reference copepod.

2.4.4 Copepod Diet

Copepods mostly feed on *phytoplankton* [37], which are microplankton, i.e. plankton with size of the order of micrometers.

Being self-feeding through photosynthesis, phytoplankton live in the euphotic zone, i.e. close to the water surface, in order to receive sunlight. This characteristic is used in environments E4, E5 and E6, where phytoplankton can only spawn near the water surface. Among phytoplankton, the most common food source for copepods are diatoms, an algae group present in waters. Diatom can range between $2\mu m$ to $200\mu m$ in diameter. but can also form larger aggregates.

Some of the food sources present in our simulation are diatoms, in particular we choose *Coscinodiscus granii*, see Figure 2.2b, as the reference food source.

2.4.5 Sensing Fluid Deformation

Haury et al. (1980) set up an experiment to measure the rate of avoidance reaction of *Calanus finmarchius* from an approaching larger obstacle, representing a possible predator [38]. Their experimental apparatus had an obstacle moving with respect to the water in which the copepod was placed. They also measured the distance from the obstacle when the reaction was observed, along with the flight angle between the direction of the observed flight and the direction of the approaching obstacle. Based on their experimental results, Haury et al. argue that the fluid forces on the copepod in the region where they observed avoidance flight reactions are dominated by the quasi-steady deformation of the fluid in the neighbourhood of the copepod. The fact that fluid deformation is the predominant perceived property is also argued by Kiørboe et al. for a small copepod (of around 100μm in length), while for larger copepods of 1cm in length the slip velocity could be the predominant signal [39].
Haury et al.'s results show no difference in the distance of avoidance initiation, i.e. the distance from the obstacle when initiating the escape response, when varying

the size of the obstacle or its speed. Nonetheless, the average distance of avoidance flights (the total distance range of the escape) changed depending on the obstacle's size. Moreover, their experimental results indicate that copepods, when performing an escape reaction, are more likely to escape away from the obstacle's stagnation line (the direction of approach). These experimental results helped us design the copepod animats' perception of fluid deformation, see Section 4.2.3, and design their escape reaction in Environment E5 and E6, see Section 4.4.5 and 4.4.6.

3

Theory

3.1 Happiness Functions

3.1.1 Homeostatic and sensory variables

Homeostatic variables, or homeostatic conditions, are defined in the fields of Biology as the physico-chemical conditions of a living being's physiology, which have a steady state [6]. Homeostatic regulation strives to maintain *homeostasis*, balancing the homeostatic variables around their respective steady-states.

Keramati et al. developed a framework for producing a reward signal which rewards homeostatic regulation for a reinforcement learning agent with simulated homeostatic variables [5]. While homeostatic regulation constitutes the basis of our approach to reward shaping, sensorial information can itself be considered an important part of the reward. We thus define, along with the animat's set of homeostatic variables H, the set of sensory variables S.

It is important to mention that in our model all homeostatic variables and sensory variables of one animat also constitute observation inputs for the reinforcement learning algorithm, thus the set of variables used to compute the reward is always a subset of the input variables of the policy network.

3.1.2 Critical Variables

In our reinforcement learning framework, the technical difference between homeostatic variables and sensory variables is the fact that only the former can be critical, i.e. can lead to death. Each of the animat's critical homeostatic variables h_i must be within their surival bounds: $h_i \in [S_{L,i}, S_{U,i}]$. If any critical homeostatic variable is outside the survival bounds the animat is terminated. In the case of copepods, for example, the homeostatic variable *energy* is critical with a lower boundary for survival at 0. However, copepods do not gain any more energy than 1, and thus do not have any upper survival boundary for their energy.

3.1.3 Homeostatic-sensory Regulation

Each environment E1-E6 is designed to elicit behaviour B1-B6, which leads to the optimal survival rate. The reward functions based on homeostatic-sensory regulation are designed to help the animats learn to maintain homeostasis in each environment. By striving to maintain homeostasis and sensory balance in each environment, with

an appropriate set of homeostatic and sensory variables, each behaviour can be elicited.

3.1.4 Happiness

The underlying mechanism of homeostatic regulation and the reward signal is here defined as the animat's *happiness*. In addition the following definitions are used throughout this thesis:

- The set of homeostatic variables $H = (h_1, \ldots, h_{N_h})$ with $N_h = \#H$;
- The set of sensory variables $S = (s_1, \ldots, s_{N_s})$ with $N_s = \#S$;
- we denote *happiness variable* any homeostatic or sensory variable, since it takes part in computing the happiness value below;
- The set of happiness variables $V = H \cup S$;
- The total number of happiness variables as N_v, where $N_v = N_h + N_s$, since $H \cap S = \emptyset$

Finally, we define the value of an homeostatic-sensory state in the animat's physiology as their *happiness*.

$$happiness = F(V) \tag{3.1}$$

for some function $F(.)$ and some set of happiness variables $V = (v_1, \ldots, v_{N_v})$.

At any time-step t, after performing an action, the animat's happiness is computed with the happiness variables' current values as $happiness_t = F(V_t)$. Finally, the animat receives a reward r_t as the step-increase in happiness:

$$r_t = happiness_t - happiness_{t-1} \tag{3.2}$$

The update rule for r_t is defined for $t \geq 1, t \in \mathbb{N}^+$ by initializing the animat's happiness as the happiness given by their initial state.

By selecting the happiness function carefully the reward will be higher if the agent improves from a low happiness state, and lower if the agent improves from a high happiness state. Figure 3.1 shows an example of happiness function where $happiness_t = F(V_t) = F(h)$ is a function of one variable h.

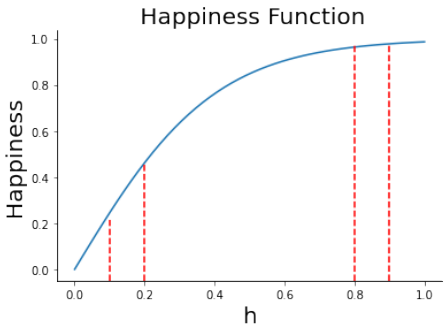


Figure 3.1: Example of decreasing marginal utility with increasing homeostatic variable h.

3.1.5 Simple Happiness Functions

Happiness comes in many shapes and forms, some of which might be captured by simple functions. We propose different happiness functions reflecting different methods of aggregating terms for different homeostatic and sensory variables. One of such methods is by simple summation of the different variables' weighted terms, as in the following:

$$happiness_t(V_t) = \sum_{i=1}^{N_v} w_{v_i} u_{v_i}(v_{i,t}) \tag{3.3}$$

where $V_t = (H_t, S_t) = (h_{1,t}, \ldots, h_{N_h,t}, s_{1,t}, \ldots, s_{N_s,t}) = (v_{1,t}, \ldots, v_{N_v,t})$

for some set of utility functions $\{u_{v_i}\}_{i=1}^{N_v}$, where V_t are the animat's happiness variables at time step t and w_{v_i} is the weight relative to variable v_i's term. The happiness function in Equation 3.3 is denoted f_1 for brevity.

Yet another approach to compute the happiness value is by multiplication of the happiness variables' terms. A general happiness function of this form is proposed in Equation 3.4:

$$happiness_t(V_t) = \prod_{i=1}^{N_v} (a_{v_i} + w_{v_i} u_{v_i}(v_{i,t})) \tag{3.4}$$

for some set of utility functions $\{u_{v_i}\}_{i=1}^{N_v}$, parameters $\{a_{v_i}\}_{i=1}^{N_v}$ and weights $\{w_{v_i}\}_{i=1}^{N_v}$.

Equation 3.4 is a general form for computing happiness through the multiplication of terms relative to each happiness variable. When utility functions u_{v_i} have their image between 0 and 1, the choice of parameters a_{v_i} and weights w_{v_i} allows for many choices of modeling the happiness function. For example, choosing a_{v_i} and w_{v_i} so that $0 \leq a_{v_i} < a_{v_i} + w_{v_i} = 1$ each term multiplied is between a_{v_i} and 1, yielding a maximum happiness value of 1 and the possibility to model non-critical variables' terms with an higher a_{v_i}, thus not inhibiting the critical variables.

During this project, when using the multiplication function above to calculate happiness, some parameter choices have been made: for all homeostatic variables h_i, $a_{h_i} = 0$ and $w_{h_i} = 1$, while for all sensory variables s_i, $a_{s_i} = 1$ and $w_{s_i} < 1$. Moreover, utility functions $\{u_{v_i}\}_{i=1}^{N_v}$ have their image in the interval $[0, 1]$. This choice implies that each homeostatic variable, which is critical, i.e. can lead to the animat's death, can lower the total happiness value to 0. Moreover, all homeostatic variables are equally important, having their term's image ranging from 0 to 1. Finally, sensory variables, which are not critical by design, can only increase the product computed with the homeostatic terms, not inhibiting the critical terms by excessively lowering the happiness value. The following equation is a particular case of Equation 3.4 and constitutes the multiplicatory happiness function adopted in all experiments in this projects, denoted f_2 for brevity:

$$f_2(V_t) = happiness_t(V_t) = \prod_{i=1}^{N_h} u_{h_i}(h_{i,t}) * \prod_{i=1}^{N_s} (1 + w_{s_i} u_{s_i}(s_{i,t})) \tag{3.5}$$

for some set of utility functions $\{u_{v_i}\}_{i=1}^{N_v}$ and weights $\{w_{s_i}\}_{i=1}^{N_s}$.

Both methods enable separately modeling the utility of each variable through univariate functions $\{u_{v_i}\}_{i=1}^{N_v}$.

In Equations 3.3, 3.5, different physiological needs are considered through different utility functions. We try to capture the utility of particular needs by utilizing functions with particular features, such as in Figure 3.2. Monotonically increasing or decreasing functions such as linear and logarithmic functions are adopted for variables representing physiological needs that need to be maximized or minimized. Functions with a single maximum are adopted for variables representing a physiological need that has to be balanced in the inner of a domain, such as a body temperature value.

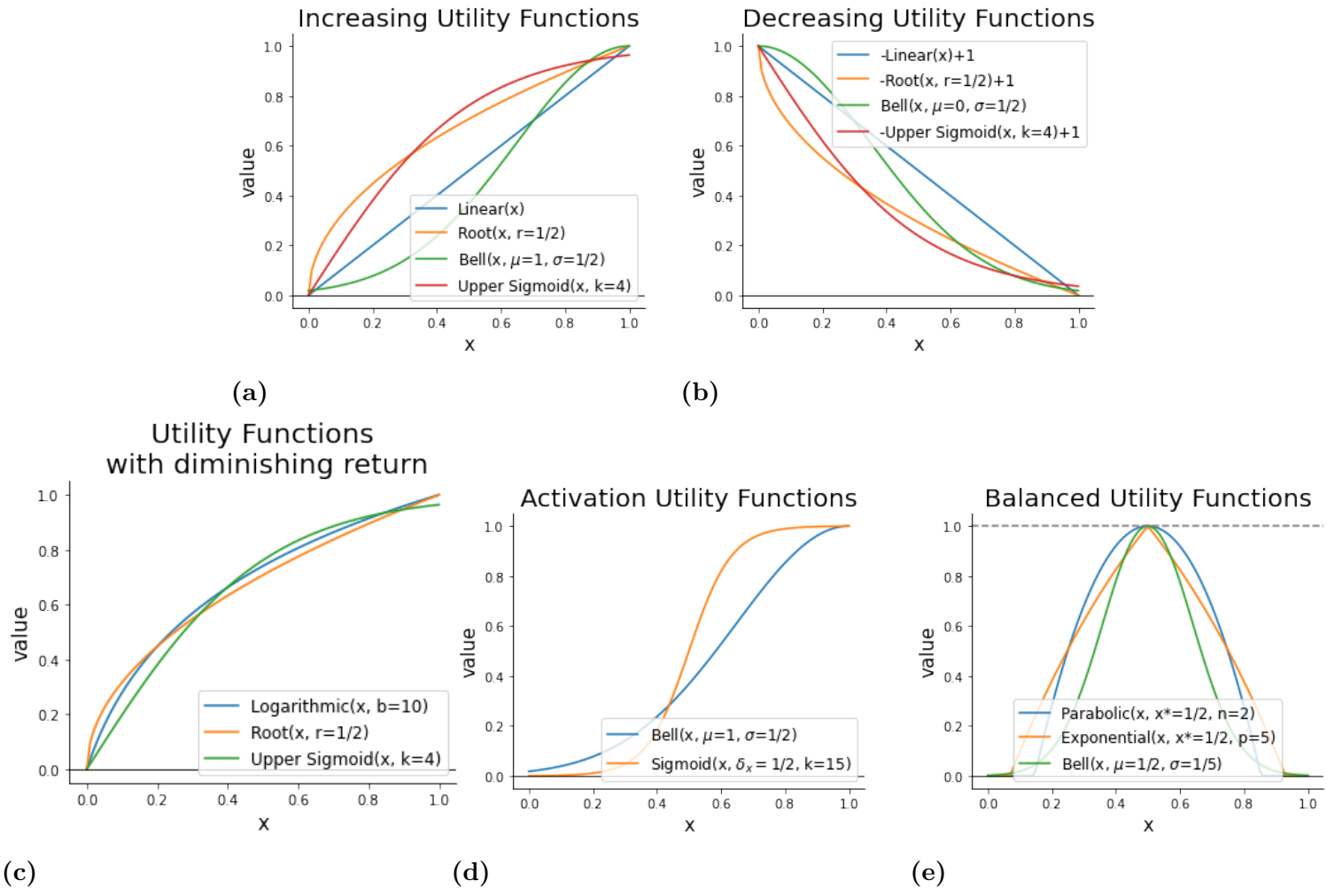


Figure 3.2: Examples of functions used to model different types of needs. Needs can be represented by a variable to be maximized (a) or minimized (b). Additionally, the need could have a utility with diminishing return (c) or a utility with distinct low-happiness and high-happiness regions, similar to activation functions (d). Finally, some needs are represented by a variable to be balanced in the inner part of an interval (e). All functions referenced can be found in Equation 3.6

3.1.6 Additive Utility Signals

Restricting happiness models to simple functions can be beneficial, as it is easier to apply intuition and obtain control over how one homeostatic variable impacts the happiness when all other variables are fixed. This can be preferable when doing advanced simulations, such as modelling ecosystems, both because it is easier to debug and because it is more intuitive and simple to design a happiness function considering each need separately. For instance, humans regulate their appetite against their satiety using two opposing signals [40]. Consider the following simplified functions signaling utility of *appetite*:

$$u_{appetite}(energy) = \log(0.1 + 20energy)$$

where an increase in energy is highly beneficial from a low energy state, but has diminishing returns for higher energy states, see Figure 3.3a, and *satiation*:

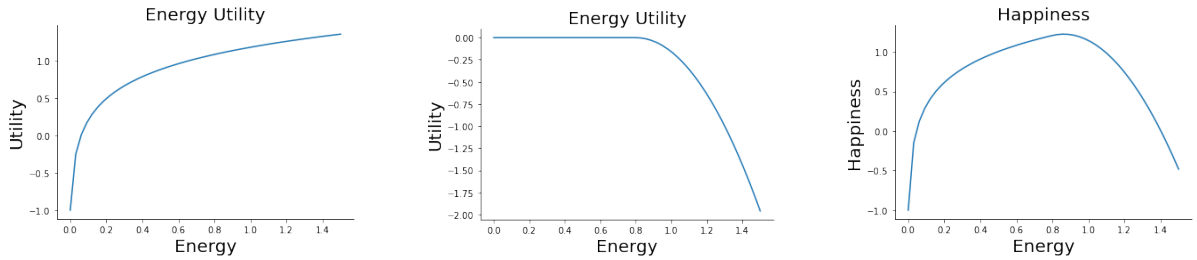
$$u_{satiation}(energy) = \mathbb{1}_{energy>0.8}[-4(energy - 0.8)^2]$$

where satiation sends a strong signal to stop overeating when energy is increased over 0.8, see Figure 3.3b.

The utility function for *energy*, and in this simple case the happiness value as well, will then be computed as the sum of the two opposing signals:

$$happiness_t = u_{energy}(energy_t) = u_{appetite}(energy_t) + u_{satiation}(energy_t)$$

This happiness function can be seen in Figure 3.3c.



(a) **(b)** **(c)**

Figure 3.3: Happiness function modelling a simplification of humans homeostatic regulation of appetite and hunger. (a) Utility of increasing energy modelled with diminishing returns. (b) Utility of increasing satiation modelled with a strong negative signal. (c) Happiness function constructed from utility of lowering appetite and increasing satiation.

3.1.7 Examples of Utility Functions

A list of utility functions defined on one happiness variable x is given:

$$
\begin{aligned}
&\text{Linear}(x) = x && x \geq 0; \\
&\text{Logarithmic}(x, b) = \log_b (1 + (b-1)x) && x \geq 0, b > 1; \\
&\text{Root}(x, r) = x^r && x \geq 0, r \in (0, 1); \\
&\text{Sigmoid}(x, \delta_x, k) = \frac{1}{1 + e^{-k(x-\delta_x)}} && x \geq 0, \delta_x \in \mathbb{R}, k > 0; \\
&\text{Upper Sigmoid}(x, k) = \frac{2}{1 + e^{-kx}} - 1 && x \geq 0, k > 0; \\
&\text{Parabolic}(x, x^*, n) = 1 - |x - x^*|^n && \forall x, x^* \in \mathbb{R}, n > 1; \\
&\text{Exponential}(x, x^*, p) = 2 - p^{|x-x^*|} && \forall x, x^* \in \mathbb{R}, p > 1; \\
&\text{Bell}(x, \mu, \sigma) = e^{-\left(\frac{|x-\mu|}{\sigma}\right)^2} && \forall x, \mu \in \mathbb{R}, \sigma > 0;
\end{aligned}
\tag{3.6}
$$

Multiple functions from this list can be used for computing happiness when multiple homeostatic variables are present according to Equation 3.3 or 3.5.

In general, it is possible to modify these functions through a weight w and an addend Y:

$$\text{Logarithmic}(x, b) * w + Y$$

For example, in order to retrieve a utility function yielding lower values in the proximity of 0.5 and higher values elsewhere, it is sufficient to apply a weight $w = -1$ and addend $Y = +1$ to a Bell curve. Nonetheless, in our framework, in Equations 3.3 and 3.5, all utility functions are designed to have their image between 0 and 1, and the weights are multiplied successively.

3.2 Drive-reduction Functions

Keramati et al. define their homeostatic regulation using the concept of *drive reduction* developed by Hull C.[41], where drive is the motivation to fulfill biological needs. As shown in Equation 3.9, the drive function in Keramati et al.'s model can be related to our definition of happiness through $drive = 1 - happiness$.

Given a set of homeostatic variables $h_1, h_2, ...$, let $H_t = (h_{1,t}, h_{2,t}, ...)$ denote the homeostatic state of the animat and $H^* = (h_1^*, h_2^*, ...)$ denote the desired homeostatic state. At each time step t the update of the homeostatic state is $K_t = (k_{1,t}, k_{2,t}, ...)$, see Figure 3.4.

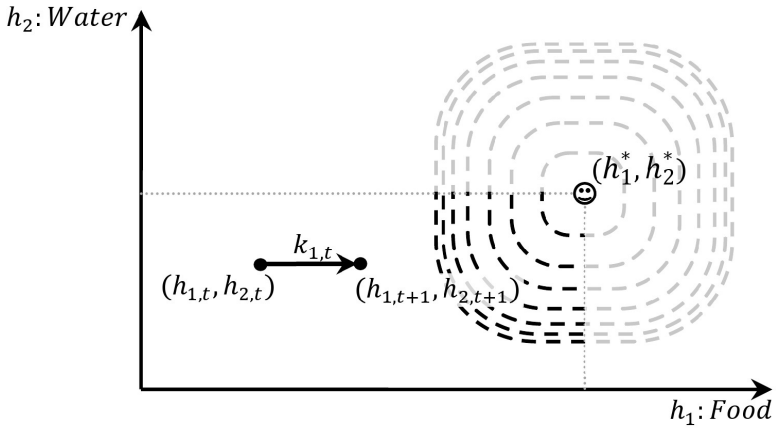
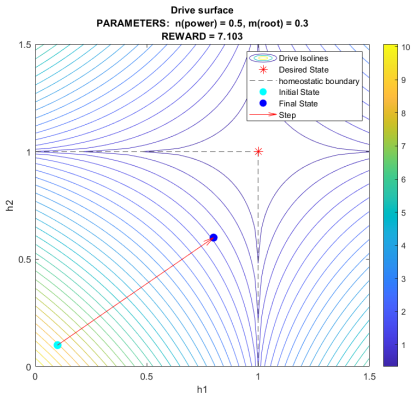


Figure 3.4: A simple example of homeostatic space for an animat that has to regulate two homeostatic variables, i.e. two different needs [5].

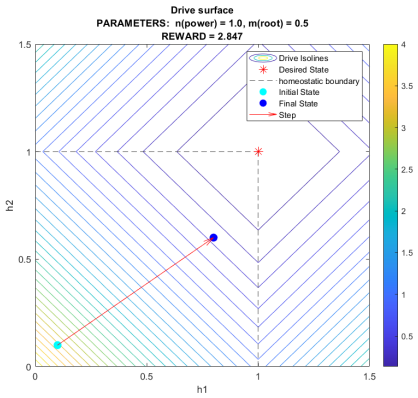
In the homeostatic space H, Keramati et al. define the *drive function*:

$$d(H_t) = \sqrt[m]{\sum_{i=1}^{N} |h_i^* - h_{i,t}|^n} \tag{3.7}$$

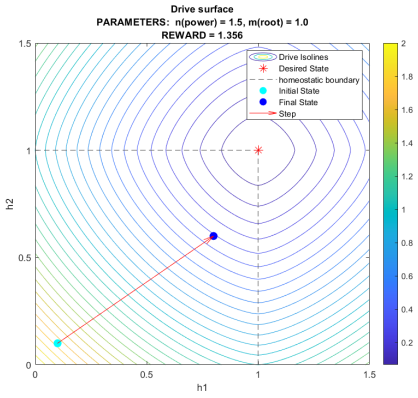
With fixed H^*, the function $d(\cdot)$ defines a surface in the homeostatic space H. When, for example, choosing $m = n = 2$, this function corresponds to the Euclidean distance. By varying the parameters n and m, different types of surfaces can be obtained, see Figure 3.5. Note that the drive function in Equation 3.7 can represent similar forms as the set of simpler functions considered in Section 3.1.5, e.g. choosing $m < n$, $m = 1$ yields a drive function similar to a sum of logarithmic functions and $m = n$, $m = 1$ similar to a sum of linear functions.



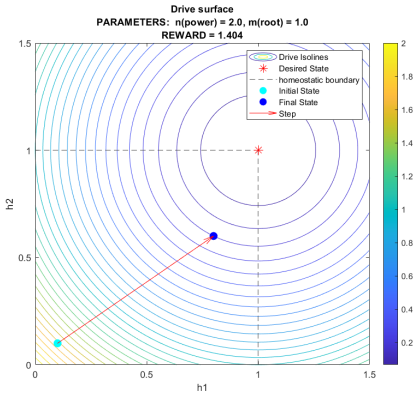
(a) $0 < n < 1$ yields star-shaped isolines.



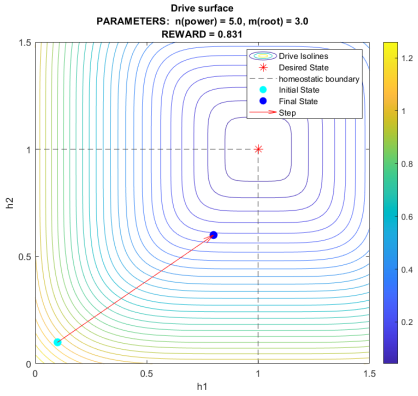
(b) $n = 1$ yields diamond-shaped isolines.



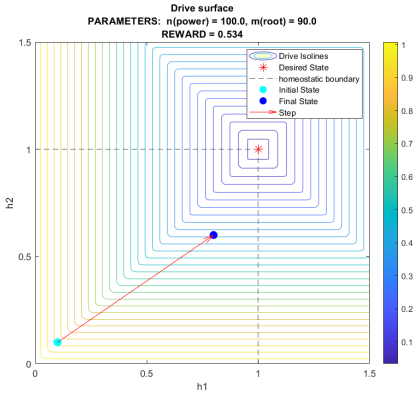
(c) $1 < n < 2$ yields isolines between a diamond-shape and a circle.



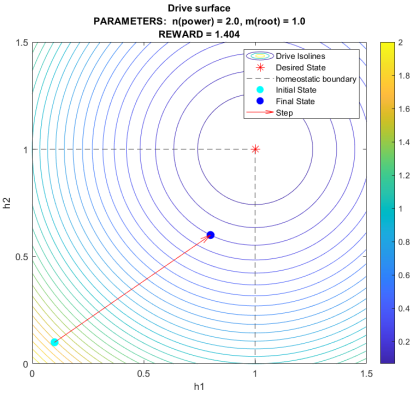
(d) $n = 2$ yields circular isolines.



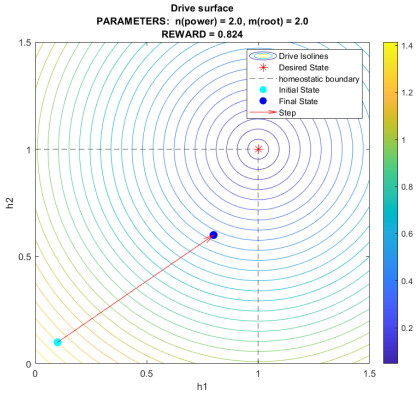
(e) $n > 2$ yields rounded square isolines.



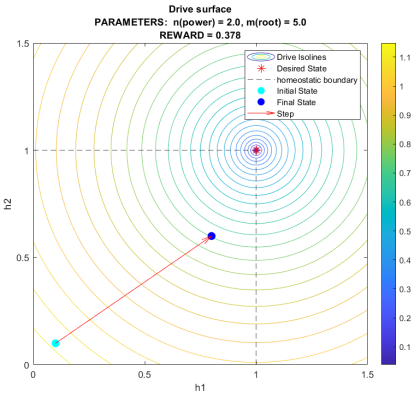
(f) $n \gg 2$ yields approximately squared isolines.



(g) Setting $m < n$, the drive surface has positive concavity.



(h) Setting $m = n$, the drive surface is a cone.



(i) Setting $m > n$, the drive surface is characterized by negative local concavity over the set $H \setminus H^*$with a singularity cusp in H^*.

Figure 3.5: Drive surfaces with different m, n.

On the surface defined by $d(\cdot)$, Keramati et al. define the reward function r as:

$$r(H_t, K_t) = d(H_t) - d(H_{t+1}) = d(H_t) - d(H_t + K_t) \tag{3.8}$$

Translating this into the framework of Equation 3.1 and 3.2 shows how the drive and happiness are simple functional transformations of each other:

$$r_{t+1} = d(H_t) - d(H_{t+1}) = happiness_{t+1} - happiness_t \iff d(H_t) = 1 - happiness_t \tag{3.9}$$

Keramati et al. show that conveniently, for $m < n$, the functional form $d(\cdot)$ leads to a few properties which correspond mathematically to behavioural phenomena observed in animals.

3.2.1 Reward Value Increases with Dose of Outcome

Given a state H_t and an update K_t such that:

$$K_t \cdot (H^* - H_t) > 0 \text{ and } |K_t| < |H^* - H_t|, \tag{3.10}$$

the derivative of the reward $r(H_t, K_t)$ on the magnitude of the homeostatic update $|K_t|$, where the starting state and the direction of the homeostatic update are fixed, is positive, indicating an increasing reward with an increasing dose of outcome. When the direction of K_t is equal to the direction of $(H^* - H_t)$, this result generalizes to any non-infinitesimal increment of the homeostatic update magnitude $|K_t|$, when the conditions above are satisfied, such that the update does not "overshoot" ($|K_t| > (H^* - H_t)$).

Keramati et al. cite two studies as the biological support of this property, which show that rats maintain a higher breakpoint when reinforced with larger appetitive outcomes, which reflect their higher motivation to obtain the outcome [42], [43].

3.2.2 Excitatory Effect of Deprivation Level

Given a homeostatic update K_t between the homeostatic states H_t and H_{t+1}, such that Equation 3.10 is satisfied, increasing the deprivation level, i.e. raising the value $(H^* - H_t)$, yields a larger output. This comes from the fact that the derivative of the reward $r(H_t, K_t)$ on the absolute distance of the starting homeostatic state from the desired state, $|(H^* - H_t)|$, is positive. This result generalizes any non-infinitesimal increase of the deprivation level when the conditions above are satisfied and the direction of $(H^* - H_t)$ is fixed.

Keramati et al. point to a biological support of this property in [43], which shows that rats increase their breakpoint in progressive ratio schedules with the increase of their food deprivation.

3.2.3 Inhibitory Effect of the Irrelevant Drive

Let us define $I = \{\text{homeostatic variables}\}$ with $N = |I|$.

Given an homeostatic state H_t such that:

$$\exists j \in I : \quad |h_j^* - h_{j,t}| > |h_i^* - h_{i,t}| \quad \forall i \neq j, i \in I$$

Let us define a set of homeostatic updates $\{K_i\}_1^N$ where $K_i = \alpha \hat{e_i}, \alpha \in \mathbb{R}, \forall i \in I$, where $\hat{e_i}$ is the unit-length vector in direction i, and such that $K_i \cdot (h^* - h_i) \geq 0$, $\forall i \in I$ and such that $|K_j| = |\alpha| \leq |h_j^* - h_{j,t}|$ and $|K_i| = |K_j| = \alpha \quad \forall i \in I$.
Then it follows that:

$$r(H_t, K_j) > r(H_t, K_i) \quad \forall i \neq j, i \in I$$

The above property corresponds to the fact that the best homeostatic variable to ameliorate, i.e. yielding the greatest reward, is the homeostatic variable which is farthest from the desired state.

Keramati et al. point to a review indicating a large set of studies in behavioural sciences which show that deprivation in one need has an inhibitory effect on the other, less deprived, needs. Intuitive examples of this pattern are the impediment of responses for food when thirsty as well as the impediment of responses for water when hungry or the lesser drive for mating when hungry. This inhibition effect is shown to affect needs at the chemical level as well. For example, a high deprivation from calcium is shown to suppress the need for phosphorus[44].

3.2.4 Risk Aversion

The particular set of drive surfaces we chose (with $m < n$) are united by the convexity of their restriction to a straight line passing through the desired homeostatic state. When the homeostatic state varies on such a straight, i.e. when K_t is proportional to the vector $(H^* - H_t)$, the reward, as a function of K_t, is concave. A concave utility function (our reward) in microeconomics is often translated to the concept of risk aversion. A simple explanation of this property is that if K_t "overshoots", i.e. the next state H_{t+1} is beyond the desired state H^* with respect to the initial homeostatic state H_t, it is possible that the new drive $d(H_{t+1})$ will be higher than the previous drive $d(H_t)$, thus yielding a negative reward, although the direction of the homeostatic update K_t was towards the desired state.

3.3 Generalized Homeostatic-sensory Regulation from Drive Reduction

The reward function introduced by Keramati et al. values all homeostatic variables equally. This is a limitation as e.g. the deprivation of sex might not affect hunger as much as the deprivation of food affects the drive for sex, since the need for food is, for most animals, a more relevant need. The need for food is also *critical*, i.e. it can lead to death by starvation if ignored. Therefore, it is sensible to associate different weights to different variables. Moreover, as already mentioned, we include sensory variables along homeostatic variables and have defined a variable of any of the two types as *happiness variable*.

We thus introduce two concepts, which are also present in Equations 3.3 and 3.5, to Keramati et al.'s framework: *happiness variables' weights* and *critical homeostatic variables.*

3.3.1 Weights for the Happiness Variables

We introduce weights for each happiness variable to easily control their impact on happiness relative to each other. A happiness variable's weight is a positive scalar value $w \in [0, 1]$. For each homeostatic function f_i in the set of homeostatic functions $\{f(H)\}_{i=1}^{N}$, we associate a weight w_i.

The homeostatic drive equation, which in Keramati et al.'s work[5] is Equation 3.7, with the addition of weights and the inclusion of sensory variables along homeostatic variables, as explained in Section 3.1, becomes:

$$d(V_t) = \sqrt[m]{\sum_{i=1}^{N_v} w_{v_i} |v_i^* - v_{i,t}|^n} \tag{3.11}$$

where $V^* = (v_1^*, \ldots, v_{N_v}^*)$ is the defined desired homeostatic-sensory state, $\{w_{v_i}\}_{i=1}^{N_v}$ is the set of weights relative to each variable v_i and n, m parameters greater than 0.

Note that the choice of $f(H)_{i=1}^{N}$ allows weights to be incorporated into the function itself. However, by using relative weights for each function the different functions impact and the choice of functions are decoupled. Meaning that the selection of utility functions $f(H_t)$ can be done by only considering the functional form of the phenomena.

Due to sensory variables belonging outside the domain of physiology, we argue that their relevance in computing happiness is generally lower compared to the importance of homeostatic variables. Thus, we always assign lower weights to sensory variables than to homeostatic variables.

3.3.2 Domain of the Homeostatic Variables

In our model, each homeostatic variable h_i in the set of the agent's homeostatic variables H has a domain which is an interval $[DL_i, DU_i]$.

As explained in Section 3.1.2, homeostatic variables, can be critical, i.e. can cause the death of the animat if their value is outside a defined interval $[SL_i, SU_i]$.

The function of SL_i, SU_i, DL_i and DU_i in the environment step update algorithm is briefly described:
At each step in the environment, after receiving the environment's updated state, the agent performs an action: If any homeostatic variable h_i is outside of its survival boundaries SL_i and SU_i, the agent is terminated. If instead the agent survives, its homeostatic variables are projected to their respective domain $[DL_i, DU_i]$ and a reward is calculated according to Equation 3.2.

4
Methods

4.1 Diel Sunlight

In several of the following environments, see Sections 4.4.4, 4.4.5, 4.4.6, 4.4.7, a diel light source is simulated through an oscillating *sun* object whose light intensity decreases with water depth. The light oscillation, along with the light dissipation towards the depths of the environment, is necessary to recreate the conditions for diel vertical migration, see Section 2.4.1.

The *sun* object oscillates following the equation:

$$y^{\text{sun}}(t) = H^{\text{sun}} + \frac{\Delta^{\text{sun}}}{2} \cos\left(\frac{t \cdot 2\pi}{T} + \pi\right) \tag{4.1}$$

where y^{sun} is the y coordinates of the sun light source, H^{sun} is the average height of the light source relative to the water surface and Δ^{sun} is the excursion length of the light source.

The source emits light whose intensity gradually dissipates towards the depths of the body of water. The light intensity, denoted *li*, depends on the vertical position and on time as follows:

$$li(y,t) = \max\left(1 - \left(\frac{y^{\text{sun}} - H^{\text{sun}}}{\Delta^{\text{sun}}} + 0.5\right) - \left(\frac{y^{\text{ws}} - y}{y^{\text{ws}}}\right), 0\right) \tag{4.2}$$

where y^{ws} is the y coordinate of the water surface. This creates a marine environment where with depth-dependent *li* ranging from 0 to 1 which, at any fixed moment, decreases with water depth, and in any fixed point, during a period T of time-steps, follows a diel cycle. *li* is maximal, i.e. equal to 1, at $y = y^{\text{ws}}$ and $t = \pi + 2k\pi$, $k \in \mathbb{Z}$.

4.1.1 Reflex Krill

In Environment E4, E5 and E6, predation is present through *reflex krills*, which do not learn their behaviour but instead act on reflexes, see Sections 4.4.4, 4.4.5, 4.4.6. Krills can move in the cardinal directions *up, down, left, right* but are not allowed to move to the deeper half of the environment, so that they can only live in the surface-most half. Finally, krills can *attack*, dealing damage to all neighbouring copepods, decreasing their energy levels.

Reflex krills perceive copepods through light-sensitive proto-vision, see Sections 4.2.1, 4.2.1.1. At each timestep they will thus perceive two observations u_{copepod} and v_{copepod} and will try to move in the direction $d = \max(\hat{u}_{\text{copepod}}, \hat{v}_{\text{copepod}})$, corresponding to one of the 4 cardinal directions above. A lethality parameter, denoted

let, affects the probability of the krill to move in direction *d*. It ranges from 0 to 0.9, increasing linearly in 10000 time-steps and stabilizing at 0.9 from time-step 10000 until the end of the episode. When $let = 0$, the probabilities of movement are not dependent on d and are: $p_{down} = p_{left} = p_{right} = 0.1, p_{up} = 0.7$, indicating krills' bias to live near the water surface. When lethality is increased, though, the movement direction will be increasingly more dependent on direction d, and less dependent on the upward bias and on randomness.

In particular, by indicating with $p_{\text{intentional}}$ the probability of selecting the direction as d, with p_{random} the probability of selecting the direction randomly between the 4 cardinal movements, and with p_{biased} the probability of selecting the upward movement direction due to the upward bias, the selection method of the effective movement direction, with a general lethality value, is:

$$p_{\text{intentional}} = \frac{let}{let + 0.1},\ p_{\text{random}} = \frac{0.04}{let + 0.1},\ p_{\text{biased}} = \frac{0.06}{let + 0.1}$$

When lethality is maximal, i.e. $let = 0.9$, the probabilities above will be $p_{\text{intentional}} = 0.9$, $p_{\text{random}} = 0.04$ and $p_{\text{biased}} = 0.06$.

The only exception when the krill will not move is if it perceives touching a copepod. In that case, it will *attack*.

4.1.2 Adaptive Krill

In environment EF, see Section 4.4.7, both prey and predators are reinforcement learning agents using the framework introduced in this thesis, although copepod prey constitute the focus of the experiment. The predators, in the form of krills, need to kill the prey, i.e. the copepods, in order to survive. Krills can idle and can move in the cardinal directions, but are not allowed to move to the deeper half of the environment, so that they can only live in the surface-most half. They have one homeostatic variable, *energy*, which is depleted with time and is replenished through the consumption of *protein* nutrients. Proteins are found in *meat* objects, which are generated in place of dead copepods, only when copepods die from a krill *attack*. Thus, in order to replenish their energy, a krill needs to attack a copepod and kill it. Finally, the krill needs to eat the newly generated meat object in order to replenish its energy.

Thus in environment EF, since krills are reinforcement learning agents as well and are expected to learn to detect and chase copepods, we expect to observe some form of co-evolution between copepods and krills.

Food name	**Protein**
Meat	0.4

Table 4.1: Initial nutritional content of meat, the only food type edible by krills in E4, E5, E6 and EF.

Additionally, meat objects are susceptible to senescence, a process of decay which gradually depletes their protein content. The decay process is described in the following pseudo-algorithm.

```
    initialProteinContent = 0.4;
    senescence = 0.0001;
    decayRate = 0.3;
    for (timestep t in Episode)
    {
        senescence += decayRate * senescence;
        if (senescence >= 1)
            destroy meat object;
        proteinContent = initialProteinContent - senescence/2;
    }
```

	energy
Proteins	1

Table 4.2: Mapping from nutrients to homeostatic variables for krill agents in environment EF, representing the effect that consuming a food containing 1 unit of a nutrient has on the animat's homeostatic variables. Such effect can be greater than the effective homeostatic variable shift, when the homeostatic variable goes outside of its domain.

Krill agents are rewarded applying Equation 3.2 to translate happiness into reward, through the following happiness function of variables energy $= en$ and light intesity $= li$ (see Equation 3.6 for functions used):

$$f_2^{krill}(en, li) = \text{Logarithmic}(en, b = 10) \cdot (1 + 0.5 \cdot \text{Linear}(li))\,; \tag{4.3}$$

which is displayed in Figure 4.1.

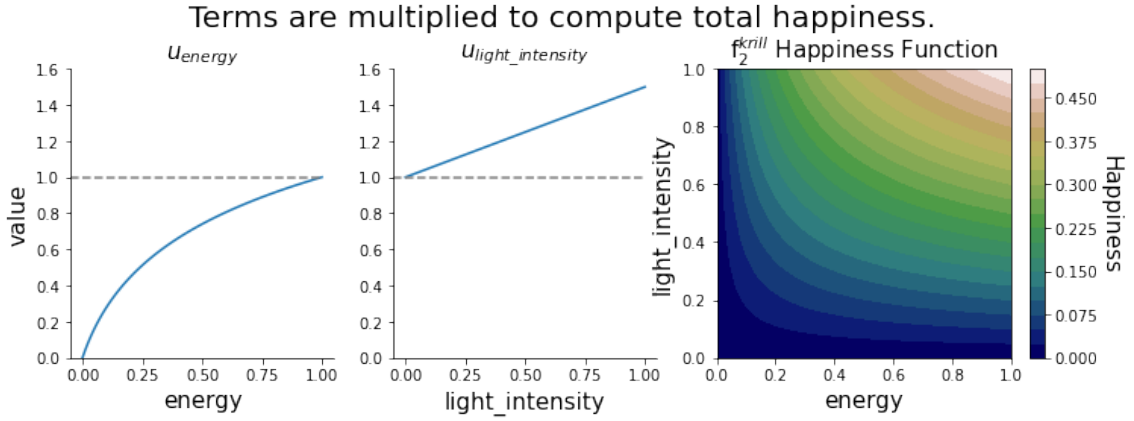


Figure 4.1: Happiness function f_2^{krill} of the homeostatic variables *energy* and *lightIntensity* used for krill agents in environment EF.

We use the f_2 happiness function for krill agents, multiplying the energy term and the light intensity term.

Both terms are monotonically increasing since krill animats should maximize their energy and are favored by an intense light. The term for energy was chosen

identical as the term used for copepods' energy, while the term for light intensity was simply chosen to be a linear function, with a weight of 0.5.

4.2 Perception

The animats have various senses that are carefully designed to be realistic. It is important to design the senses to be powerful enough to help the animats survive the environments yet not give them unrealistic powers.

4.2.1 Proto-Vision

Copepods have vision, but not accurate enough to motivate implementation of ray tracing. Their vision is quite murky. As such we implemented a limited vision and called it *proto-vision*.

For each object that can be sensed through proto-vision, the distance between the animat and the object is divided by its absolute value to the third power. This is equivalent to dividing the unit-length direction of the object by its distance squared, which is motivated by the inverse square law. This assigns a lower impact to farther away objects and a larger impact to closer objects. For each object type that the animat can sense through proto-vision, the values just calculated are summed between all objects of such type. The resulting vector is then decomposed into two fixed vector components, resulting in two scalar values. This yields two observation values for each type of objects that can be observed through proto-vision.

For each sensed type of objects *Type*, defining T the set of all objects in the simulation with type *Type*, given two unit-length vectors u and w defining the directions onto which to decompose the proto-vision signal, the 2 sensed proto-vision observations are:

$$S_{pv}^{u} = \sum_{obj \in T} \frac{obj.pos - agent.pos}{|obj.pos - agent.pos|^3} \cdot u;$$
$$S_{pv}^{w} = \sum_{obj \in T} \frac{obj.pos - agent.pos}{|obj.pos - agent.pos|^3} \cdot w.$$

where *obj.pos* is object *obj*'s position and *agent.pos* is the position of the agent sensing the signal.

4.2.1.1 Light Effect on Proto-Vision

The proto-vision supports light-sensitive vision by using the environments light intensity. Because our simulated environments are marine, the light intensity at some position in water is linearly scaled by the depth of that point, i.e. the distance from the surface level.

The light intensity is scaled by 1 at the surface and by 0 at the bottom. Light-sensitive proto-vision requires a certain threshold light intensity value in order to observe objects at all, which means that objects at a very deep point are not visible by light-sensitive proto-vision.

4.2.2 Smell

Given an environment E of size m by n grids, smell is represented by a smell matrix $M_s \in \mathbb{R}^{m \times n}$. In each time step t the environment increments the smell in each location (x, y) where there is an object that emits smell. The value of the increment depends on the size of the object. The smell is then dispersed using convolution over M_s with a Gaussian kernel $K_g \in \mathbb{R}^{7 \times 7}$:

$$(K_g * M_s)[m, n]$$

Finally, to decrease the smell intensity M_s is multiplied by a dampening scalar 0.98.

Note, this assumes that there is no force moving the smell such as wind in a terrestrial environment or streams in a marine environment. Moreover, this implementation also assumes that the smell spreads symmetrically in all directions, given that a Gaussian kernel is used. This strategy is inspired by kernel density estimation [45] and generalized to two dimensions.

A smell matrix can be visualized using a heatmap. Figure 4.2 shows the smell after a copepod spawned at location $(10, 10)$ and each time step moved 5 steps up the diagonal. Note how smell lingers on both previous positions, with lower intensity and higher variance.

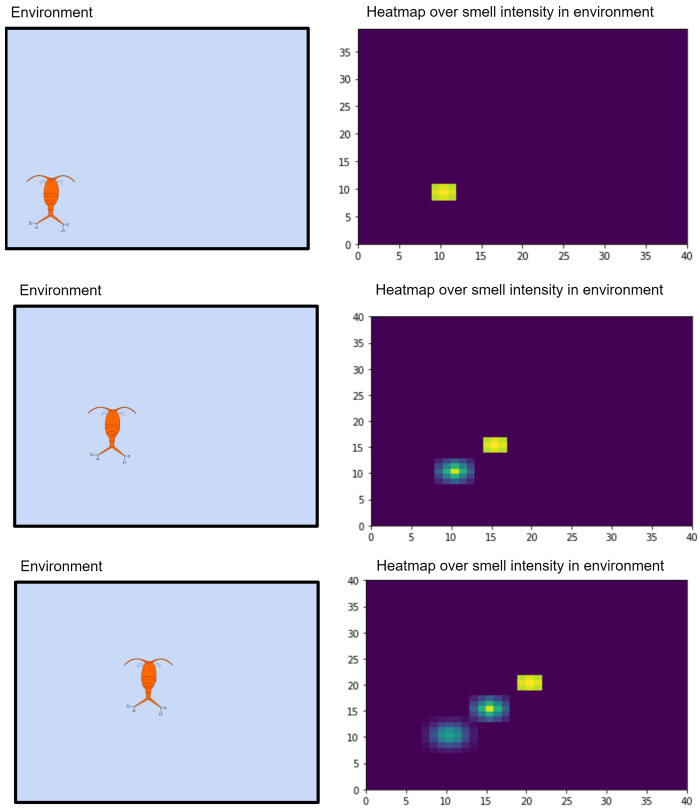


Figure 4.2: Heatmap of smell matrix during copepod movement along the diagonal.

4.2.3 Fluid Deformation Sense

As previously discussed in Section 2.4.5, Copepods can sense predators by sensing water deformation [38]. If an animat, either krill or copepod, moves in the simulations, they send a water deformation signal to all agents that can perceive water deformation, such as copepod agents in environment E5, within a radius determined by Equation 4.4 from the final position of the movement.

$$r = \sqrt{s^2 |\Delta_{\text{source}}|} \tag{4.4}$$

where r is the range of the signal, which is centered in the position of the animat producing it after the movement, s is the size of the animat producing the signal and Δ_{source} is the last step-movement of the animat producing the signal. The range of the perceived signal is in accordance to the distance of avoidance initiation observed by Haury et al. [38].

We assume that when an agent moves in a fluid, at each step, a signal is produced in the final position of the movement. The signal produced is proportional to a coefficient dependent on the size of the animat producing it. We set this coefficient to be $2.5s^2$. Additionally, the signal produced is proportional the agent's velocity $\Delta_{\text{source}}/d_{\text{source}}$ Finally, since the signal expands as an enlarging sphere [46], the inverse-square law says that a signal's strength is inversely proportional to the square of the distance from the source d_{source}. The perceived fluid deformation signal S_{fd} is therefore calculated as:

$$S_{fd} = \frac{s^2 \cdot \Delta_{\text{source}}}{d^3_{\text{source}}} \tag{4.5}$$

where d_{source} is the distance between the receiver of the signal and the center of the produced signal, or the position of the animat producing it.

Additionally, an animat senses the sum of all produced fluid deformation signals within reach in that time-step, and to introduce memory in the sensed fluid deformation signal, the animat finally senses an exponentially weighted moving average of the signal above. Thus, the update rule for the sensed signal at each time-step is:

```
1     S_fd = 0;
2     alpha = 0.95;
3     for (timestep t in Episode)
4     {
5         S_fd_tmp = 0;
6         for (source in AllMovingAnimats)
7         {
8             r = Sqrt(size^2 * |Delta_source|};
9             if (distance(animat, source) < r):
10                S_fd_tmp += size^2 * |Delta_source|^2 / d_source^3;
11        }
12        S_{fd} = alpha * S_fd_tmp + (1 - alpha) * S_fd;
13    }
```

Above, the *alpha* parameter controls the memorization of past fluid deformation signals, i.e. the moving average's inertia.

4.2.4 Touch

Animats sense their proximity to other objects in the simulations by tactile perception. These objects can be types of food or water, obstacles such as walls, rocks or trees, the boundary of the environment, agents of the same type or agents of another type.

Given a set of types of objects $S^{touch}_{animatType} = \{obj_1, obj_2, obj_3, \ldots\}$ that an animat of type *animatType* can touch, at each step all animats of type *animatType* will sense, for each object type obj_i in $S^{touch}_{animatType}$, whether they are in contact to at least one object of type obj_i or not.

4.3 Actions

All animats' set of actions in the performed experiments are a subset of the following set of actions: {Move up, Move down, move left, Move right, Idle, Dash, Eat, Attack}. Each of the 4 cardinal movement actions moves the animat by 1 unit of length in the specified direction, if such movement is allowed in the environment, for example, if the movement will not result in any interpenetration with another impenetrable object, such as a rock or a wall defining the environment's boundary. The Idle action will not produce any action for that particular time-step. The Dash action moves the animat by a long distance, as in a quick jump, within a cone in the opposite direction of the maximum perceived fluid deformation, see Section 4.3.1. The Eat action will eat one random food object that is in contact with the animat and that is edible by the animat, i.e. that contains at least one nutrient that the animat can consume. In the performed experiments krill predators can only eat *meat* objects while all other animats can eat all food objects except meat. Finally, the Attack action deals a *damage*, consisting of an energy value, randomly selected between 0 and the animat's *attack force*, to all other prey animats in contact with the agent performing the action. In the experiments, this action can only be performed by krill predators towards copepod prey. The attack force of each predator animat is found in the Appendix.

4.3.1 Dash

Dash can only be performed by copepod animats in environments E5 and EF, where copepods can perceive fluid deformation, see Section 4.2.3.

When performing a Dash action, firstly a cardinal direction d will be selected as the direction opposite to the direction of maximum perceived fluid deformation. If all 4 perceived fluid deformation observations are equal, a random cardinal direction will be chosen. Next, the distance of the jump along the cardinal direction of the dash, *dist*, will be selected, starting from 10 units of length and iteratively decreasing to 1 length unit. For each distance *dist*, a value *maxLateralShift* will be computed as $\lfloor dist \cdot \frac{4}{5} \rfloor$. Then it will be checked for an available position in a maximum

of $\lfloor maxLateralShift/2 \rfloor$ positions, laterally shifted by random values within the interval $[-maxLateralShift, maxLateralShift]$ from the position at distance *dist* in direction d from the copepod. If no available position is found at distance *dist*, *dist* will be iteratively decreased until an available position is found.

4.4 Environments

In this Section, the motivational model and six different experiments tested in this project are presented. Each experiment consists of a reinforcement learning environment where a specific animat behaviour is necessary in order to survive and succeed. As such, the performance of the motivation model proposed in this project can be measured by animats' survival time in each experiment. The animats in each experiment may have a different set of senses, possible actions and a different set of homeostatic variables.

In each experiment an additional animat with a simple reward signal for staying alive is used as baseline. The baseline animat and the animats using homeostatic regulation are pretrained in order to initiate their policy networks. In each experiment, animats use their pre-trained network and use individual lifelong learning, that is, each animat only updates its policy from its own experience within its own lifetime.

The aim of this thesis is to construct a framework of motivation that can be used to elicit the following six behaviours:

- B1: Regulation of hunger, only eat when not satiated.
- B2: Chemotaxis by selective eating by differentiating nutritious food and harmful food.
- B3: Chemotaxis by following scent trails.
- B4: Phototaxis by moving towards and away from light.
- B5: Barotaxis to escape close and quickly-approaching predators.
- B6: Chemotaxis to escape predators sensed by scent.

In order to test animats' ability to learn said behaviours, six environments E1-E6 are created, each specifically designed to elicit its corresponding Behaviour, e.g. Environment E1 is designed to elicit B1 and so forth. The environments are also designed so that its specific behaviour should increase the animats' probability of survival. Therefore the expected survival time serves as a success criteria of the specific model's capability to successfully learn each behaviour, see Section 6.2 for a detailed motivation.

In each environment five different types of animat models are evaluated, distinguished by the reward signal adopted. The models are named as the respective happiness function utilized in calculating the reward through Equation 3.2:

- f_1 - Happiness functions using sum as aggregation, see Equation 3.3.
- f_2 - Happiness functions using product as aggregation, see Equation 3.5.
- f_3 - Happiness function constructed as a generalization of the work by Keramati et al. [5] with weights and senses, see Equation 3.11.
- f_4 - Happiness function corresponding to Keramati et al.'s model. Happiness is calculated as $1 - d(\cdot)$, where $d(\cdot)$ is the drive function proposed in Keramati et al.'s work [5].

In environment $E1$ only one homeostatic variable, energy, is present when computing the happiness value. Not combining different happiness terms thus eliminates any difference between some models tested. For this reason, we name the tested happiness functions differently compared to the other environments. g_x^1 uses a Logarithmic function of energy while g_x^2 uses the inner function of Keramati et al. model[5], where in both cases the subscript x indicates the energy value which yields maximum happiness. E1 experiments thus compare performances of cases depending on 2 elements: the type of function utilized in a scenario where happiness depends on only 1 variable, and the defined energy value of peak happiness.

Each episode starts with an animat randomly spawning in the environment with optimal homeostatic values, and ends when the animat either dies or survives for 10000 timesteps. Environment E1-3 use one agent and resets the entire environment to its partially stochastic, initial settings, upon a new episode. Environment E4-6 use several animats. Each animat has individual episodes that are reset when the animat dies. Thus, environments E4-E6 are not reset when resetting an animat. In fact, these environments are never reset during a simulation.

Animats are trained for 1M timesteps. The trained models are then tested, by running the learned policy in inference mode and the survival function of each model is estimated over 300 episodes.

4.4.1 Environment 1

The purpose of Environment 1, E1, is to elicit the behaviour B1, to regulate hunger by only eating when not satiated. E1 consists of a 10 by 10 squared grid with 3 food objects at random locations and 1 animat, see Figure 4.3.

The animat's only need consists of its only homeostatic variable, energy, which can take values between 0 and 1. Energy depletes by a constant value at each timestep. Additionally, energy is consumed by any successful movement action. In this environment the movement actions are move up, move down, move left, move right. When energy reaches 0, the animat dies and its episode is terminated. The only way for the animat to replenish its energy is by eating the green spherical food objects, denoted *good food*, which can refill an amount of 0.4 in energy. Nonetheless, if energy surpasses the value 1 as a result of eating, it will be immediately set to 1, as its domain is $[0, 1]$.

The purpose of this experiment is to test whether the animat can learn that it will live a longer life when not wasting energy resources.

Using optimal regulation of hunger, i.e. by allowing the agent to eat only when eating would increase the energy level by the maximum amount, an agent can survive for approximately 1040 time steps (this value can vary by a small amount since the 3 food objects are generated in random positions and the movement actions cost energy), while if the agent does nothing it will die in at most 500 time steps (the agent can in fact die more quickly if it moves around without purpose, since moving costs energy).

Each episode in the environment is 1040 time steps long, as such is approximately the survival time of an animat with optimal regulation hunger regulation. In this setting a greedy overeating behaviour results in the agent depleting its food

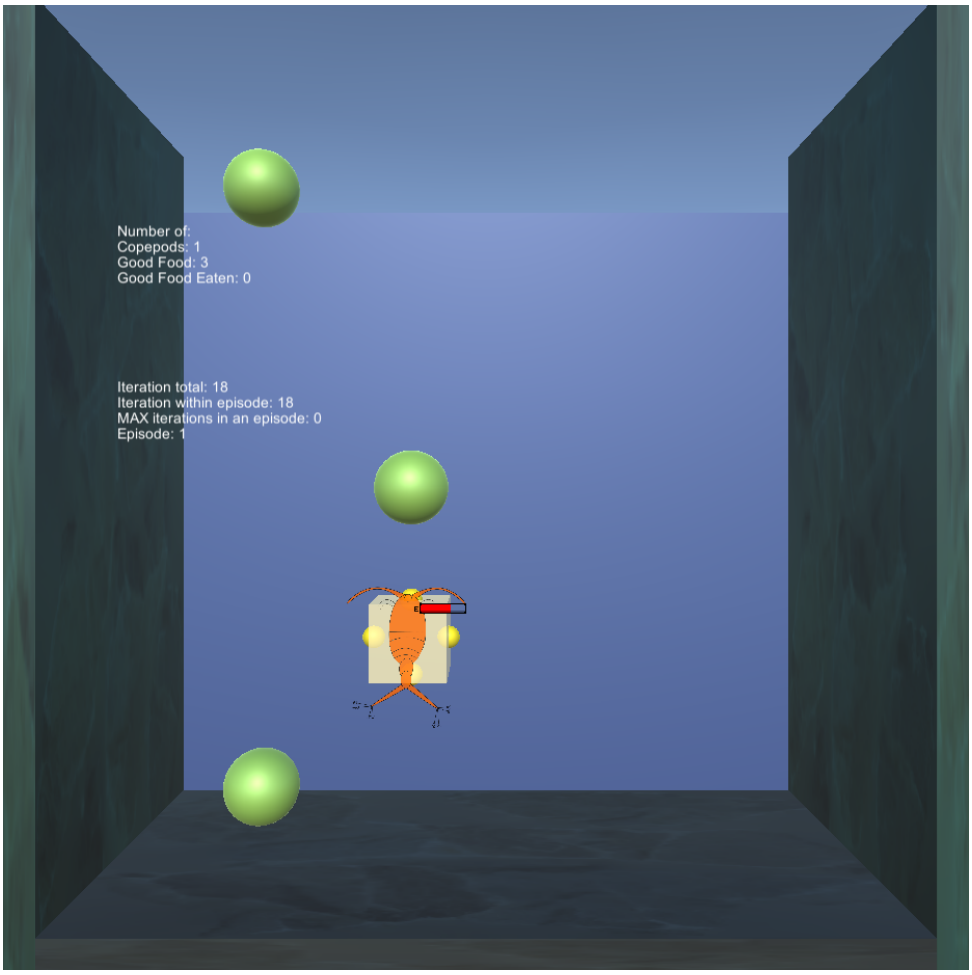


Figure 4.3: Illustration of environment 1.

sources too quickly.

The animat in Environment 1 has the following senses, or observations, which define the states for the reinforcement learning algorithm:

Senses:

- Touch (Mechanoreception - chemoreception): tactile perception of food and walls. 1 observation for good food and 1 observation for environment boundaries (4.2.4).
- Proto-Vision (Photoreception): food direction (4.2.1).
- Energy level.

The animat has 1 homeostatic variable: *energy*, and a set of actions:
$A = \{\text{Move up, Move down, Move left, Move right, Idle, Eat}\}$.

We test the following happiness functions defined on 1 homeostatic variable, $energy = en$, applying Equation 3.2 to translate happiness into reward (see Equation 3.6 for

functions used):

$$
\begin{aligned}
g_1^1(en) &= \text{Logarithmic}(en, b = 10) \\
g_{0.8}^1(en) &= \text{Logarithmic}(en, b = 10) + \mathbb{1}_{en>0.8}[-20(en - 0.8)^2] \\
g_1^2(en) &= 1 - d(en), \qquad en^* = 1, n = 3, m = 1.5 \\
g_{0.8}^2(en) &= 1 - d(en), \qquad en^* = 0.8, n = 3, m = 1.5
\end{aligned}
$$

where $d(\cdot)$ is the drive function with 1 variable from Equation 3.7.

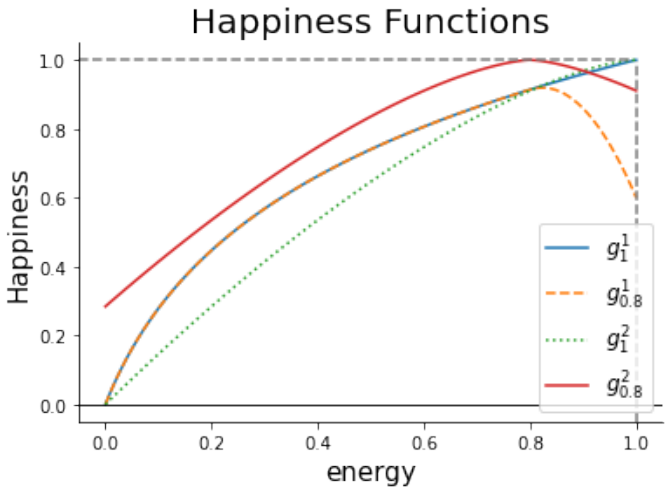


Figure 4.4: Happiness functions tested in E1.

The happiness functions tested are shown in Figure 4.4. In particular, function $g_{0.8}^1$ is composed of two terms: a simple sigmoid term (see Equation 3.6 and a quadratic penalty term $\mathbb{1}_{en>0.8}[-20(en-0.8)^2]$ which penalizes energy exceeding 0.8. Similarly, function $g_{=.8}^2$ has its desired energy $en^* = 0.8$. The 2 functions mentioned make use of the knowledge that to succeed in this environment, i.e. survive for a longer time, the animat should not aim at maximizing its energy, but at eating a food object when its energy level is below 0.6

On the other hand, g_1^1 and g_1^2 are very similar: both consist of a monotonic increasing function, which assigns the highest happiness to the highest energy within the domain. Moreover, in all 3 functions the derivative is decreasing, thus an increase in energy yields a higher reward the lower the energy.

4.4.2 Environment 2

The purpose of Environment 2, E2, is to elicit the chemotaxis behaviour B2: selective eating by differentiating between different types of nutritious food, each containing a different needed nutrient. In this scenario the animat has 2 needs: *energy* and *vitamins*. These 2 needs correspond to 2 homeostatic variables with the respective name. The homeostatic variables are constantly depleted with time passing and can be replenished by eating the appropriate nutrients, see Table 4.4, contained in the appropriate food, see Table 4.3. Thus, a successful animat should be able to balance these 2 needs by searching for the particular type of food that contains the needed

nutritional element. In addition, an harmful food type is present, which deplenishes *energy* if eaten. The agent thus needs to differentiate between food types using chemotaxis. A preliminary version of environment E2 can be seen in Figure 4.5.

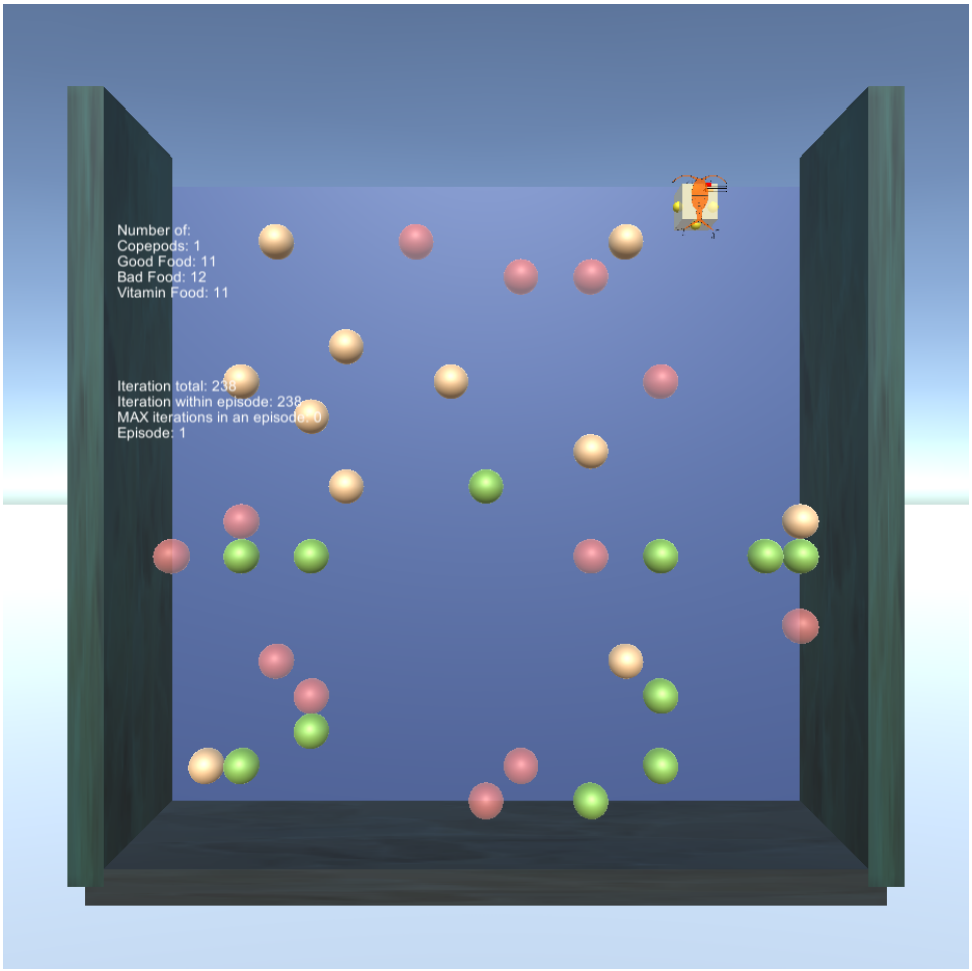


Figure 4.5: Illustration of environment 2 with various types of food.

In this environment, copepod animats need to learn to balance two nutrient needs, where some are needed in small amounts and toxic in larger.

Food name	Glucose	Toxin	Vitamins
Sugar Food	0.4	0	0
Toxic Food	0	0.4	0
Vitamin Food	0	0	0.4

Table 4.3: Nutritional content of the 3 food types used in E2.

Senses:

- Proto-Vision (Photoreception): food direction, see 4.2.1;
- Touch (Mechanoreception - chemoreception): tactile perception and discrimination of food. 4 observations, 1 for each food type and 1 for environment boundaries, see 4.2.4);

	energy	**vitamins**
Glucose	1	0
Toxin	-1	0
Vitamins	0	1

Table 4.4: Mapping from nutrients to homeostatic variables for the agent in environment E2, representing the effect that consuming a food containing 1 unit of a nutrient has on the animat's homeostatic variables.

- Energy and vitamins levels (2 observations).

The animat has 2 homeostatic variables: *energy* and *vitamins*, and a set of actions: $A = \{$Move up, Move down, Move left, Move right, Idle, Eat$\}$.
We test various happiness functions defined on 2 homeostatic variables, $energy = en$ and $vitamins = vit$, applying Equation 3.2 to translate happiness into reward.

In this environment happiness functions depend on 2 variables. We discriminate such functions depending on how they combine the terms dependent on the different variables. In the following, we will distinguish between 4 types of aggregation functions: f_1 combines the terms through a summation, f_2 combines them through a product, f_3 uses Equation 3.11 to combine the various terms while f_4 uses Equation 3.7. The main difference between f_3 and f_4 is the use of weights in the former, as explained in Section 3.3.

Since the weights used for the two variables are both 1, we do not consider f_3 in this particular environment. The functions tested are the following (see Equations 3.6 for functions used):

$$
\begin{aligned}
f_1^{E2}(en, vit) &= \text{Logarithmic}(en, b = 10) + \text{Bell}(vit, \mu = 0.5, \sigma = 1/3); \\
f_2^{E2}(en, vit) &= \max\Big(\text{Logarithmic}(en, b = 10) + \mathbb{1}_{en>0.8}[-20(en - 0.8)^2], 0\Big) \cdot \\
&\quad \cdot \max\left(\text{Bell}(vit, \mu = 0.5, \sigma = 1/3), 0\right) \\
f_3^{E2}(en, vit) &= 1 - D(en, vit), \qquad en^* = 1, vit^* = 1, w_{en} = 1, w_{vit} = 2, n = 3, m = 1.5 \\
f_4^{E2}(en, vit) &= 1 - d(en, vit), \qquad en^* = 1, vit^* = 1, n = 3, m = 1.5
\end{aligned}
$$

where $D(\cdot)$ is the drive function with weights in Equation 3.11 and $d(\cdot)$ is the drive function with 2 variables from Equation 3.7.

In this environment, animats should try to maximize energy and balance vitamins. Since these two needs are fundamentally different, we used increasing functions for the energy term and hill-like functions with the peak at 0.5 for the vitamins term.

Since one of the two homeostatic needs is the same as in E1, we use the same functions for the energy term as we did in E1, respectively, for the 4 tested happiness functions.

An additional term for vitamins is then introduced. In f_1 and f_2 the vitamins term is a Bell function centered in the desired vitamins level of 0.5. In this way, the animat is encouraged to stabilize its vitamin level around the state 0.5.

Lastly, the 4 happiness functions can be distinguished by the way in which the 2 terms of each homeostatic variable are aggregated to form the happiness function.

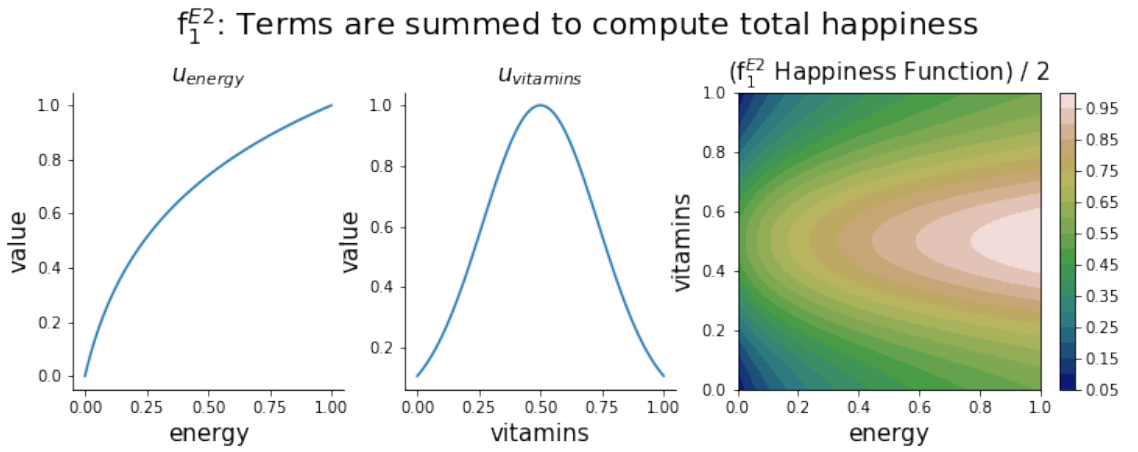


Figure 4.6: Illustration of the happiness function f_1^{E2} and its 2 terms. The two terms are added to compute f_1^{E2}.

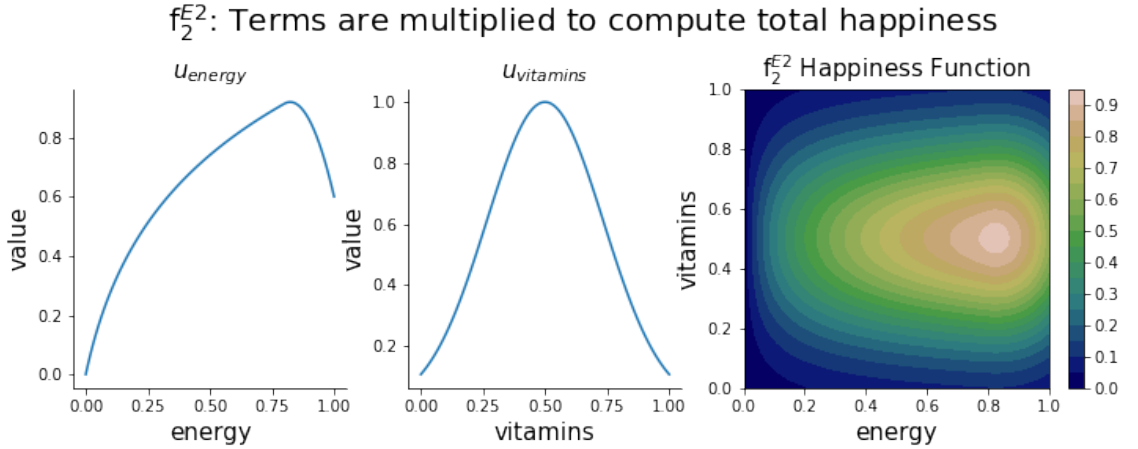


Figure 4.7: Illustration of the happiness function f_2^{E2} and its 2 terms. The two terms are multiplied to compute f_2^{E2}.

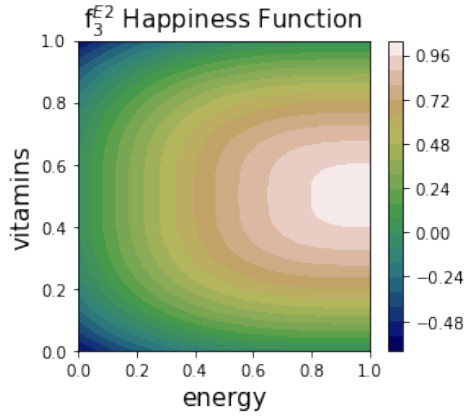
The approach in f_1 sums the two terms: thus each term can contribute independently to the total happiness value.

The approach used for f_2 multiplies the two terms, while assuring they are non-negative through a $\max(\cdot)$ function. The motivation behind the design of this function is the homeostasis concept that all variables should be balanced at the same time. Since the 2 homeostatic variables are both critical, happiness should be 0 if either one of those is 0, and should be very low if either one of these is very low, thus the use of the product to aggregate the terms.

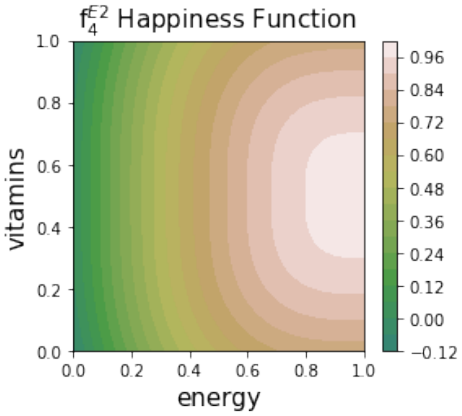
Lastly, f_3 uses $D(\cdot)$ as in Equation 3.11 with desired states and homeostatic weights given in the equation above, while f_4 uses $d(\cdot)$ as in the general Equation 3.7 with desired states specified in above.

In function f_3, since $vit^* = 0.5$, the respective weight w_{vit} is set to 2 in order to obtain the same happiness range as for energy. Instead, in the other 3 functions tested we assign the same weight to the energy term and to the vitamins term.

While the amount of terms aggregated or the weights given to the different



(a) Contour plot of happiness function f_3^{E2}. Terms are combined through drive formula with weights D, see Equation 3.11



(b) Contour plot of happiness function f_4^{E2}. Terms are combined through drive formula without weights d, see Equation 3.7

terms may vary in other environments, the distinctive aggregation approach between the 3 happiness functions is consistent across all 6 environments.

4.4.3 Environment 3

The purpose of Environment 3, E3, is to elicit the chemotaxis behaviour B3: following scent trails to detect food.

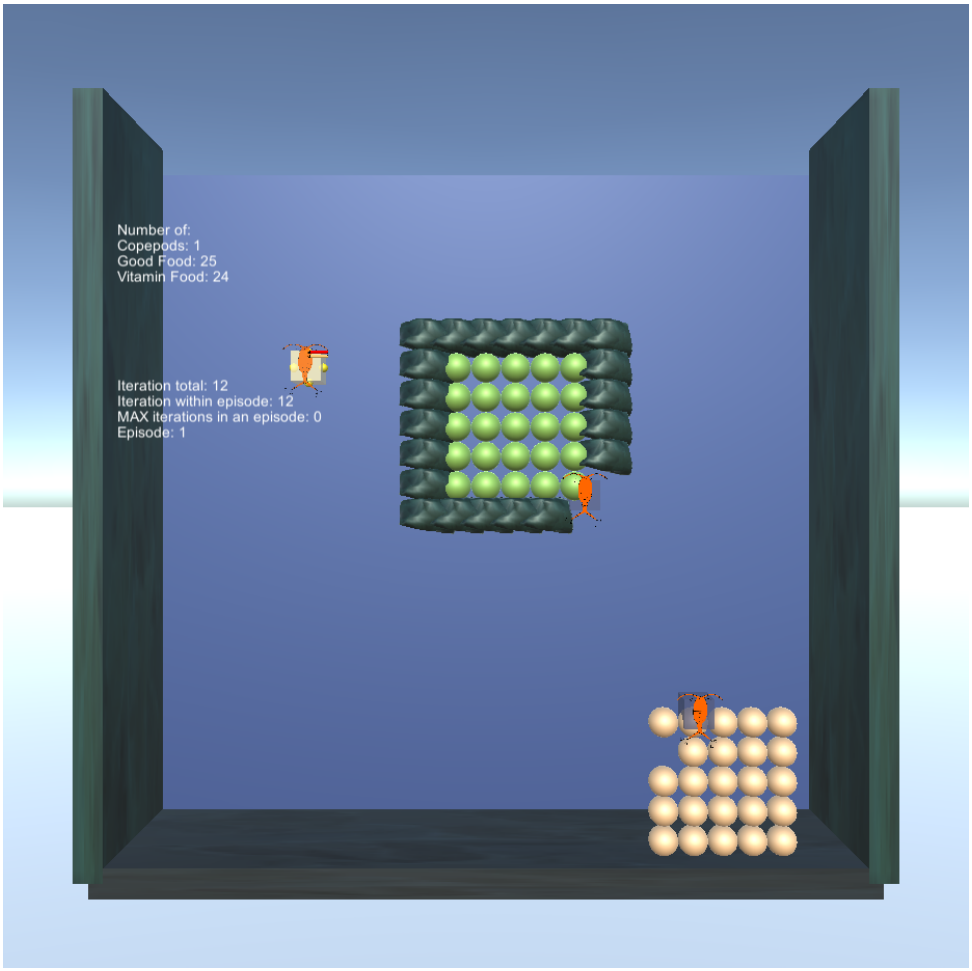


Figure 4.9: Illustration of environment 3.

The experiment focuses on a single animat with energy and vitamins as needs, as in the previous environment. Here, though, food sources for energy are all grouped in one restricted area, enclosed by rocks, with one small opening, and food sources for vitamins are grouped in a different, not enclosed, area. A group of non-agent copepods with predefined path continuously travels between the two areas, leaving a scent trace of their passing, which is dissipated in time and space, see Smell section 4.2.2. A visualization of environment 3 with the smell of copepods produced after 10, 25, 50, 75 and 100 steps is given in Figure 4.10.

The location of the two food sources changes randomly every 2000 timesteps, thus the animat cannot rely on learning the location of the two food sources a single time during its lifetime. Behaviour B3 infact consists in associating the smell of other (hardcoded) animats to the path between the two food sources and then eating in order to satisfy the homeostatic needs.

Senses:

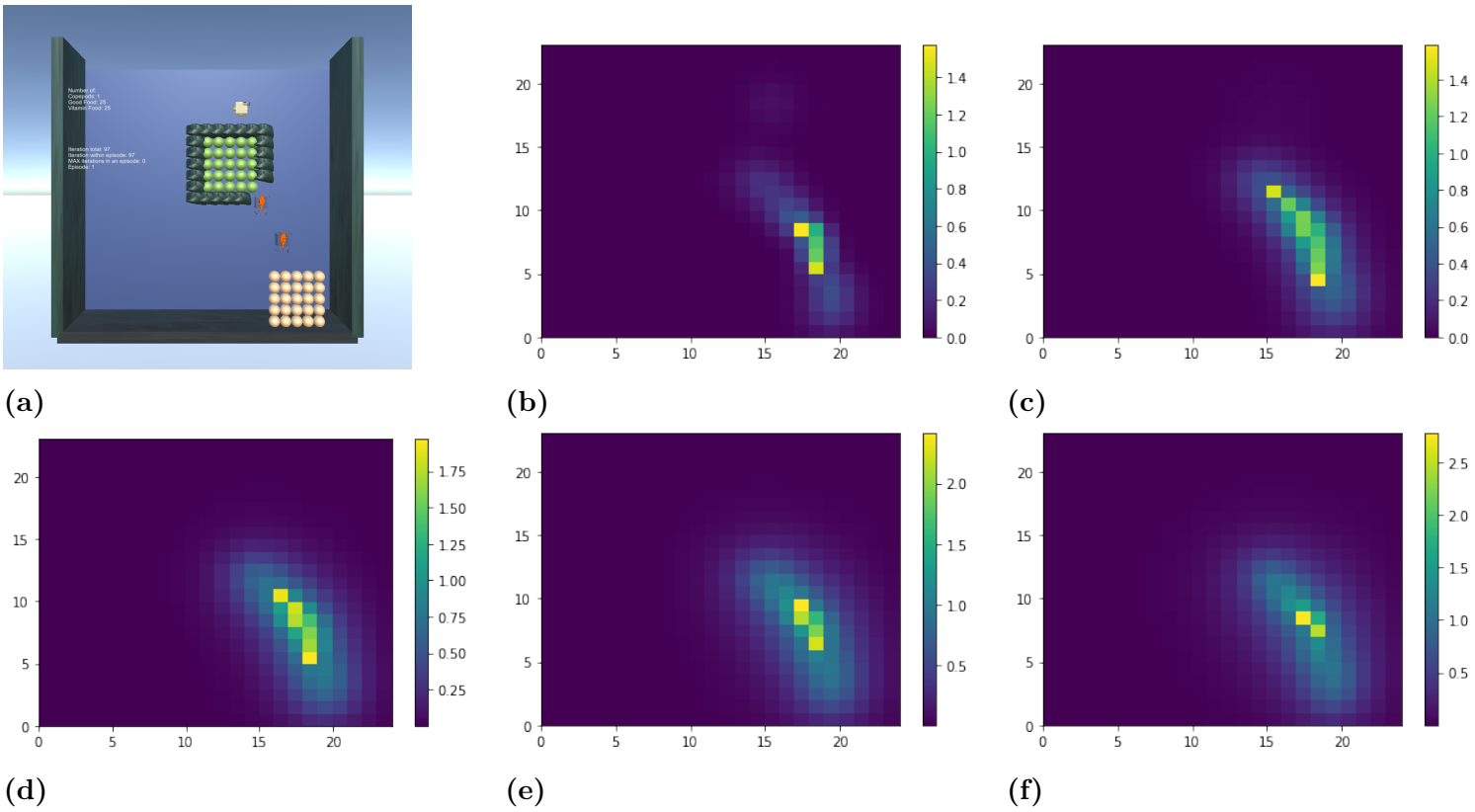


Figure 4.10: One instance of environment E3 (a) with 2 non-agent copepods leaving a smell trail between the two food sources. Visualizations of the copepod smell present in the environment at different times is given through heatmaps generated after 10 timesteps (b), 25 timesteps (c), 50 timesteps (d), 75 timesteps (e) and 100 timesteps (f).

- Touch (Mechanoreception - chemoreception): tactile presence of food and walls. 3 observations: 1 for each type of food and 1 for walls which can be environment boundaries or the rocks enclosing food, see 4.2.4;
- Smell (Chemoreception): pheromone intensity of other animats, see 4.2.2;
- Energy and vitamins levels (2 observations).

The animat has 1 homeostatic variable: *energy*, and a set of actions:
$A = \{\text{Move up, Move down, Move left, Move right, Idle, Eat}\}$.

In this environment the happiness functions build on the happiness functions used in E2, with the addition of a term for the sensory variable indicating the smell intensity perceived.

The happiness functions tested, defined on 3 variables energy = en, vitamins = vit and copepod smell = $smellCP$, applying Equation 3.2 to translate happiness into reward (see Equation 3.6 for functions used), are the following:

$$
\begin{aligned}
f_1^{E3}(en, vit, smellCP) &= \text{Logarithmic}(en, b = 10) + \text{Bell}(vit, \mu = 0.5, \sigma = 1/3) + \\
&\quad + 0.2 \cdot \text{Logarithmic}(smellCP, b = 10); \\
f_2^{E3}(en, vit, smellCP) &= \max\Big(\text{Logarithmic}(en, b = 10) + \mathbb{1}_{en>0.8}[-20(en - 0.8)^2], 0\Big) \cdot \\
&\quad \cdot \max\left(\text{Bell}(vit, \mu = 0.5, \sigma = 1/3), 0\right) \cdot \\
&\quad \cdot (1 + 0.2\text{Logarithmic}(smellCP, b = 10)); \\
f_3^{E3}(en, vit, smellCP) &= 1 - D(en, vit, smellCP), \qquad n = 3, m = 1.5; \\
f_4^{E3}(en, vit, smellCP &= 1 - d(en, vit, smellC), \qquad n = 3, m = 1.5
\end{aligned}
$$

where $D(\cdot)$ is the drive function with 3 variables from Equation 3.11 and $d(\cdot)$ is the drive function from Equation 3.7. In both cases the desired homeostatic states are: $en^* = 1, vit^* = 0.5, smellCP^* = 1$ and in $D(\cdot)$ the weights used are: $w_{en} = 1, w_{vit} = 2, w_{smellCP} = 0.2$.

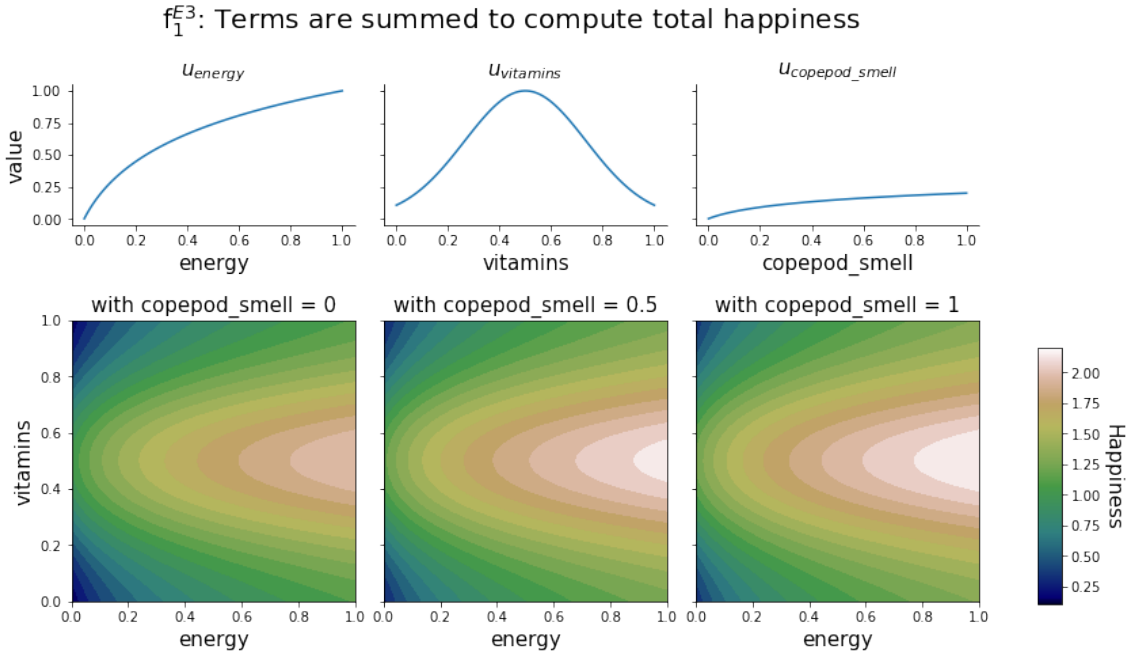


Figure 4.11: Illustration of the 3 terms composing the happiness function f_1^{E3} (above) and countour plots of f_1^{E3} for energy and vitamins and fixed values of copepod smell (below).

In f_1 and f_2 we adopted the Root function for the smell term which is a monotonically increasing function, representing the preference of the animat for being closer to other animats and following smell paths.

According to the approach used to design f_2, explained in section 4.4.2, any critical variable in a "danger zone" should determine a low happiness. Nonetheless, this should not apply for non-critical variables such as sensory variables: no value of the smell variable should force happiness to be low. Therefore, the smell term is increased by 1 before being multiplied to the other 2 terms.

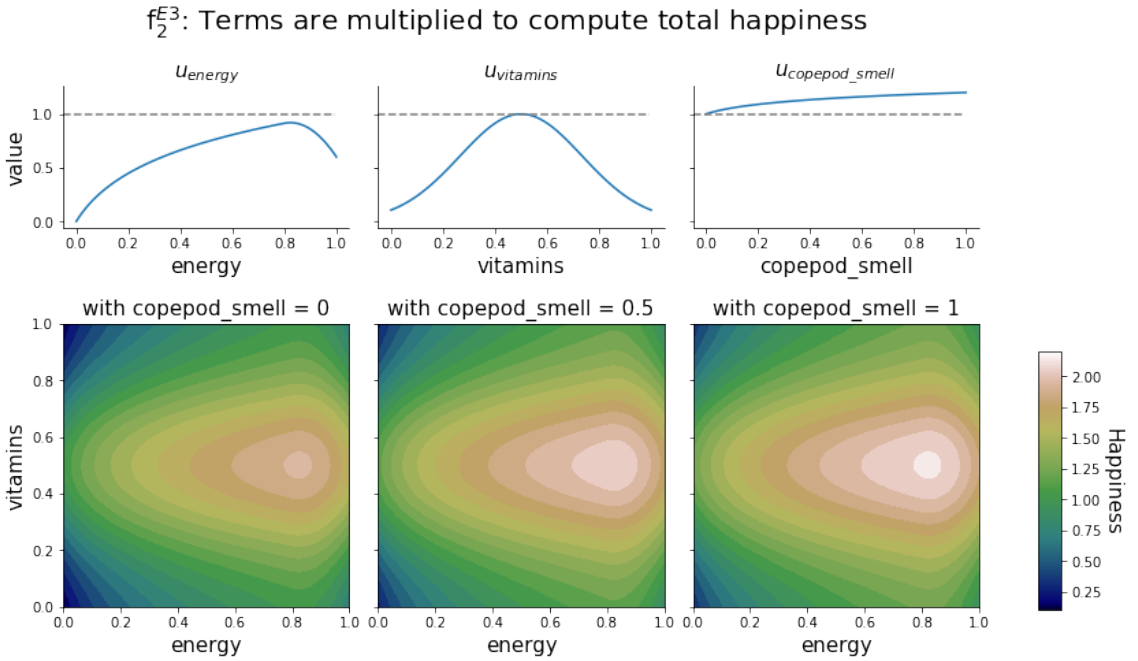


Figure 4.12: Illustration of the 3 terms composing the happiness function f_2^{E3} (above) and countour plots of f_2^{E3} for energy and vitamins and fixed values of copepod smell (below).

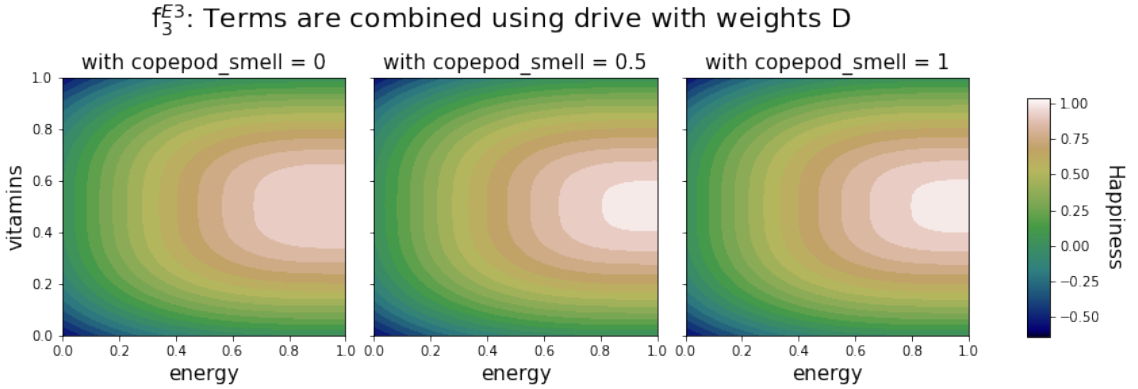


Figure 4.13: Countour plots of the happiness function f_3^{E3} for energy and vitamins and fixed values of copepod smell.

This approach will be maintained in other environments for non-critical variables, i.e. their term will be summed to 1 before multiplication in f_2.

Lastly, the weights used for the non-critical sensory variable are set to 0.2, reflecting the lower effect that such sensory variable has on happiness compared to physiological variables.

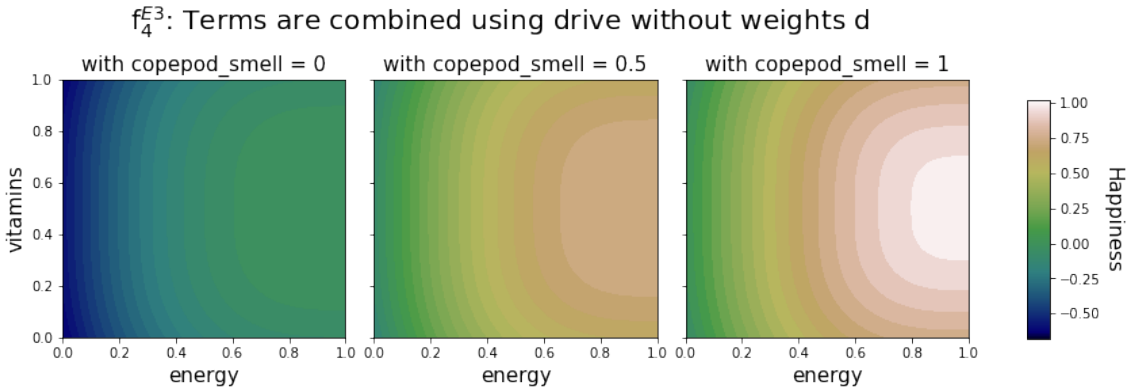


Figure 4.14: Countour plots of the happiness function f_4^{E3} for energy and vitamins and fixed values of copepod smell.

4.4.4 Environment 4

E4 simulates a basic ecosystem for free-living planktonic copepods that feed on phytoplankton [37] which grow close to the water surface. The purpose of Environment 4, E4, is to elicit the phototaxis behaviour B4, or diel vertical migration (DVM), see Section 2.4.1, which makes use of graviception and photoreception in order to move towards and away from the water surface (graviception), depending on the light intensity (photoreception). Light intensity, on the other hand, is simulated with an oscillating *sun* light source as described in Section 4.1

This environment is constructed as follows: a diel sunlight source is moved closer and away from the water surface following a cosine wave with period $T = 1000$ time-steps, see Equation 4.1, generating a depth-dependent light intensity which is computed as in Equation 4.2.

The environment is populated by copepod animats, with an energy need that can only be satisfied by eating phytoplankton objects. Nonetheless, phytoplankton only spawn near the surface, thus copepods need to venture near the surface in order to survive. Predation is introduced via non-agent krill animats, whose ability to perceive the copepod animats is proportional to the intensity of light, see Sections 4.2.1.1 and 4.1.1.

Copepod animats have the following characteristics:

Senses:

- Proto-Vision (photoreception): food direction, see 4.2.1;
- Touch (mechanoreception - chemoreception): tactile perception of food (1 observation), copepods (1 observation), krills (1 observation) and walls (1 observation), see 4.2.4;
- Photoreception: light intensity;
- Energy level.

Each copepod animat has 1 homeostatic variable: *energy*, and a set of actions:
$A = \{$Move up, Move down, Move left, Move right, Idle, Eat$\}$.
For copepod animats we test the following happiness functions defined on 2 variables: energy $= en$ and lightIntensity $= li$, applying Equation 3.2 to translate happiness

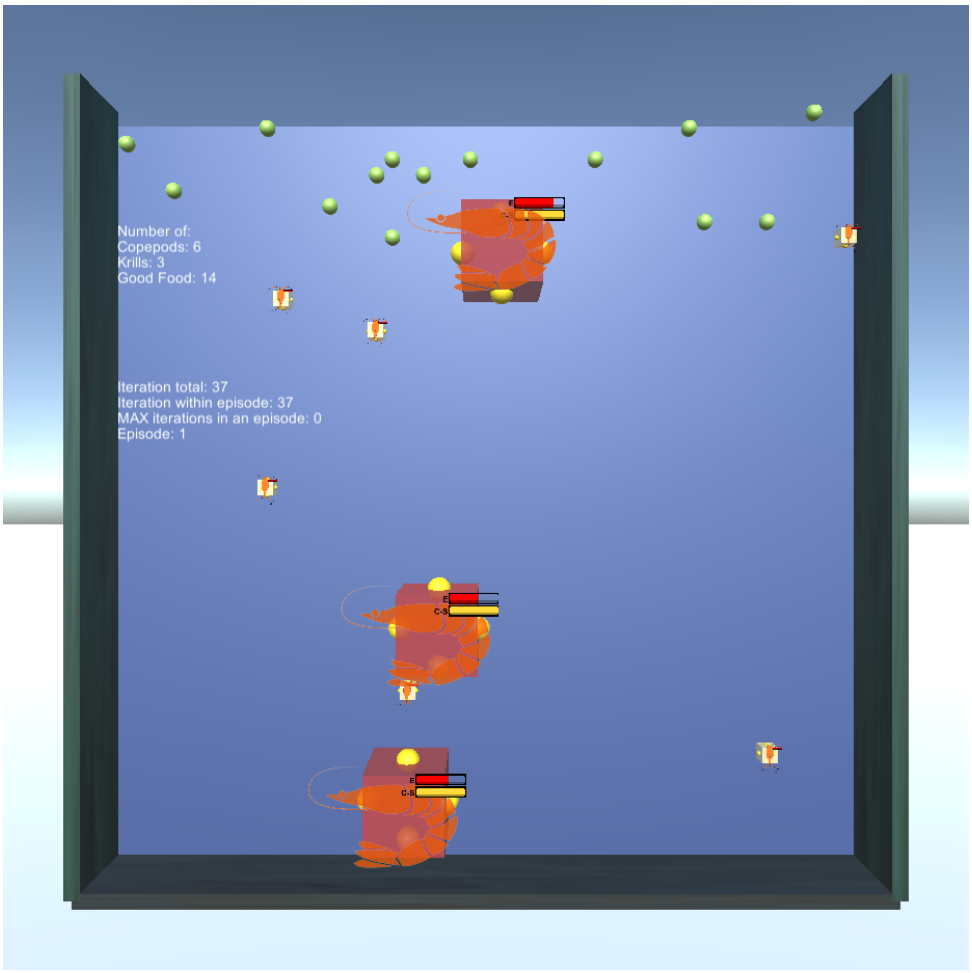


Figure 4.15: Illustration of environment 4.

into reward (see Equation 3.6 for functions used):

$$
\begin{aligned}
f_1^{E4}(en, li) &= \text{Logarithmic}(en, b = 10) + 0.2 \cdot \text{Bell}(li, \mu = 0, \sigma = 1/3); \\
f_2^{E4}(en, li) &= \max\Big(\text{Logarithmic}(en, b = 10) + \mathbb{1}_{en>0.8}[-20(en - 0.8)^2], 0\Big) \cdot \\
&\quad \cdot (1 + 0.2 \cdot \text{Bell}(li, \mu = 0, \sigma = 1/3)); \\
f_3^{E4}(en, li) &= 1 - D(en, li), \qquad en^* = 1, li^* = 0, w_{en} = 1, w_{li} = 0.5, n = 3, m = 1.5; \\
f_4^{E4}(en, li) &= 1 - d(en, li), \qquad en^* = 1, li^* = 0, n = 3, m = 1.5;
\end{aligned}
$$

where $D(\cdot)$ is the drive function from Equation 3.11 and $d(\cdot)$ is the drive function from Equation 3.7.

Copepods' happiness is designed to favor higher energy and lower light intensity, thus we use a Bell function with mean in 0 for the light term in happiness functions f_1 and f_2, and we set the desired light state in 0 for f_3 and f_4.

Since light intensity is a sensory variable its weight is set to 0.5, whereas energy's weight is set to 1. Moreover, in f_2 the light intensity term is added to 1 before multiplication, as in E3 with the smell variable.

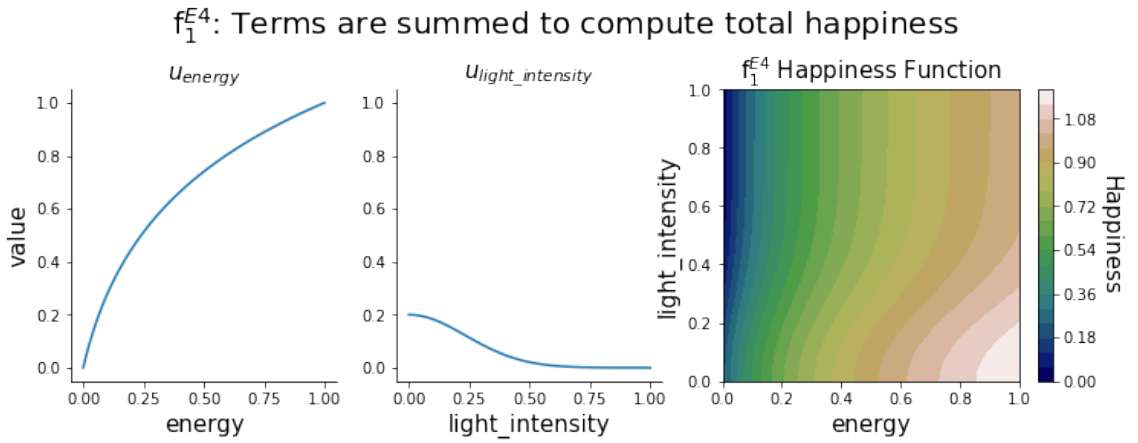


Figure 4.16: Illustration of the happiness function f_1^{E4} and its 2 terms. The two terms are added to compute f_1^{E4}.

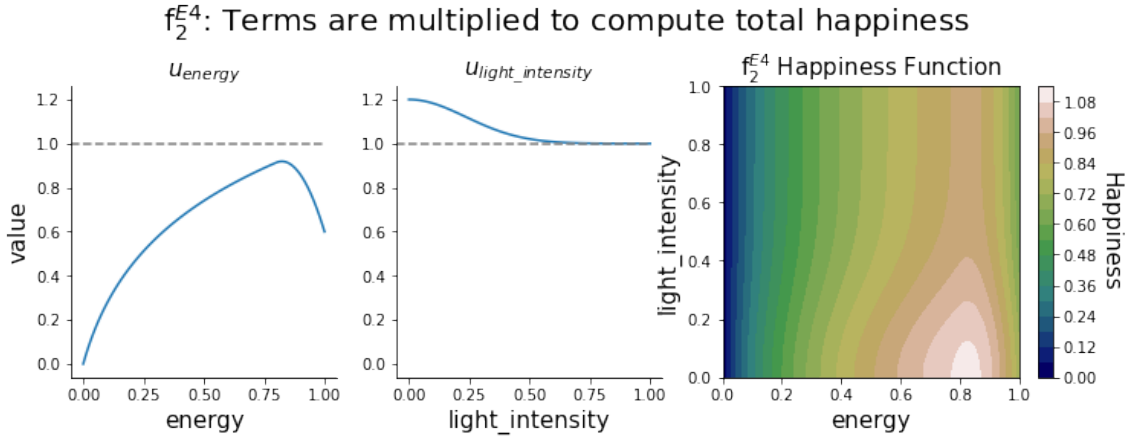


Figure 4.17: Illustration of the happiness function f_2^{E4} and its 2 terms. The two terms are multiplied to compute f_2^{E4}.

4.4.5 Environment 5

The purpose of Environment 5, E5, is to elicit the barokinesis behaviour B5, which refers to the escape reaction of copepods when sensing close and fast-approaching predators, see Sections 2.4.2, 4.3.1.

In this environment, copepods' perception includes the sensing of fluid deformation in their surroundings. Non-agent krill predators are present in the environment, see Section 4.1.1. The movement of agents produces such fluid deformation signal in their surroundings, the intensity of which depends on the animat's size. The signal is perceived by other pressure-sensing animats through the mechanism described in Section 4.2.3.

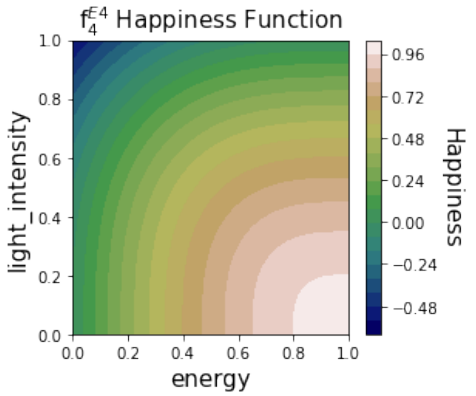
An example of Environment 5 can be seen in Figure 4.19.

Copepod animats in this environment have the following characteristics:

Senses:

- Mechanoreception - baroception: fluid deformation signal, see 4.2.3;

(a) Contour plot of happiness function f_3^{E4}. Terms are combined through drive formula with weights D, see Equation 3.11



(b) Contour plot of happiness function f_3^{E4}. Terms are combined through drive formula with weights d, see Equation 3.7

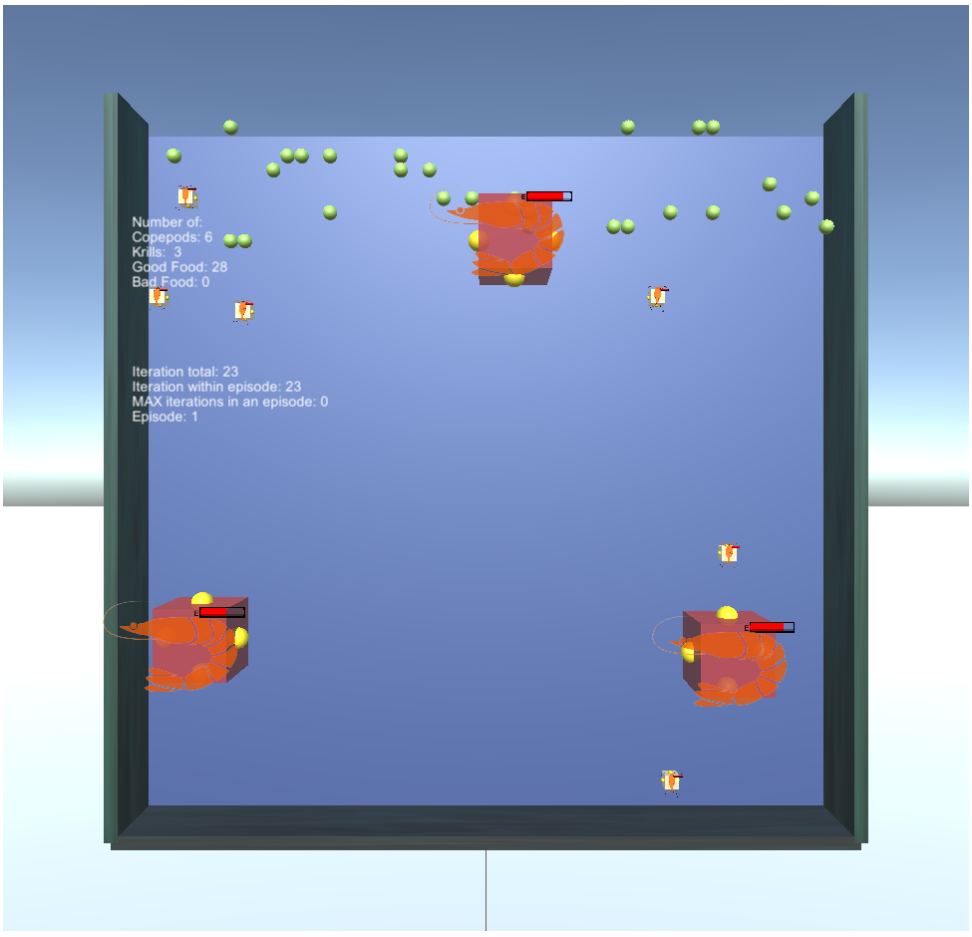


Figure 4.19: Illustration of environment 5. Note the difference in size between the predator krills and prey copepods, which determines larger fluid deformation signals emitted by krills.

- Photoreception: light intensity, see 4.2.1.1;

- Proto-vision (photoreception): food direction, see 4.2.1;
- Touch (mechanoreception - chemoreception): tactile perception of food (1 observation), copepods (1 observation), krills (1 observation) and walls (1 observation), see 4.2.4.

Each copepod animat has 1 homeostatic variable: *energy*, and a set of actions: $A = \{\text{Move up, Move down, Move left, Move right, Idle, Eat, Dash}\}$.
We test the following happiness functions defined on 3 variables:
energy $= en$, lightIntensity $= li$ and fluidDeformation $= fd$, applying Equation 3.2 to translate happiness into reward (see Equation 3.6 for functions used):

$$
\begin{aligned}
f_1^{E5}(en, li, fd) &= \text{Logarithmic}(en, b = 10) + 0.2 \cdot \text{Bell}(li, \mu = 0, \sigma = 1/3) + \\
&\quad + 0.5 \cdot \text{Bell}(fd, \mu = 0, \sigma = 1/3); \\
f_2^{E5}(en, li, fd) &= \max\Big(\text{Logarithmic}(en, b = 10) + \mathbb{1}_{en>0.8}[-20(en - 0.8)^2], 0\Big) \cdot \\
&\quad \cdot (1 + 0.2 \cdot \text{Bell}(li, \mu = 0, \sigma = 1/3)) \cdot (1 + 0.5 \cdot \text{Bell}(fd, \mu = 0, \sigma = 1/3)); \\
f_3^{E5}(en, li, fd) &= 1 - D(en, li, fd), \qquad n = 3, m = 1.5, w_{en} = 1, w_{li} = 0.5, w_{fd} = 0.5; \\
f_4^{E5}(en, li, fd) &= 1 - d(en, li, fd), \qquad n = 3, m = 1.5
\end{aligned}
$$

where $d(\cdot)$ is the drive function from Equation 3.7 and $D(\cdot)$ is from Equation 3.11; the desired homeostatic states in both cases are: $en^* = 1, li^* = 0,\ fd^* = 0$.

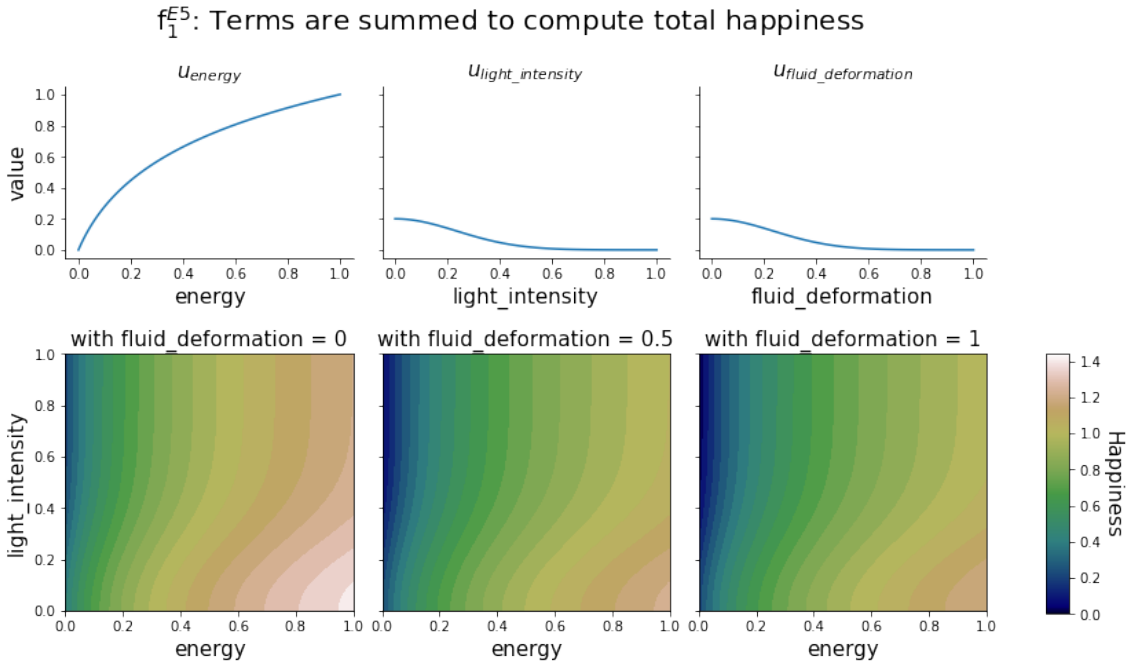


Figure 4.20: Illustration of the 3 terms composing the happiness function f_1^{E5} (above) and countour plots of f_1^{E5} for energy and light intensity, with fixed values of fluid deformation (below).

From the perspective of a copepod, sensing a fluid deformation signal is a threat to its life, thus in our model the maximum happiness from the fluid deformation term is obtained when such variable is zero. In f_1 and f_2, this term is a Bell curve centered in 0, and in f_3, the desired state for this variable is at 0.

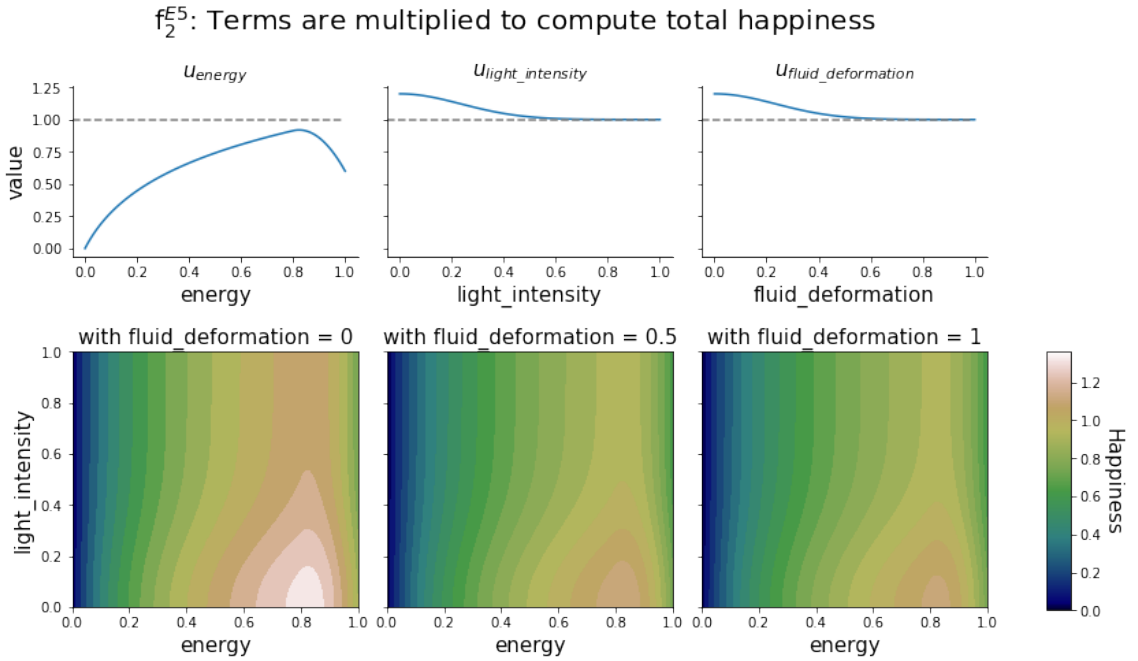


Figure 4.21: Illustration of the 3 terms composing the happiness function f_2^{E5} (above) and countour plots of f_2^{E5} for energy and light intensity, with fixed values of fluid deformation (below).

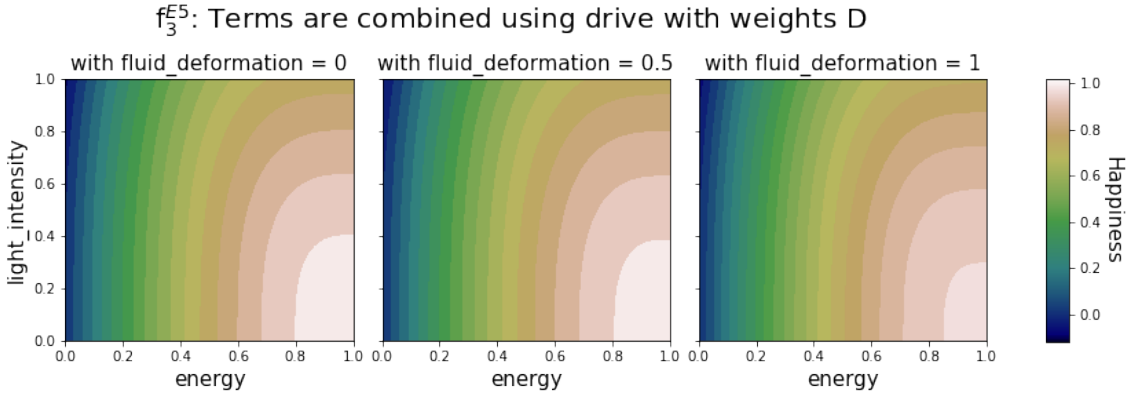


Figure 4.22: Contour plots of happiness function f_3^{E5} for energy and light intensity, with fixed values of fluid deformation. Terms are combined through drive formula with weights D, see Equation 3.11

For the sensory variable lightIntensity, the weight choice was of 0.2 in reward functions f_1 and f_2 and of 0.5 in reward function f_3, while for the sensory variable fluidDeformation the weight choice was 0.5 for all f_1, f_2 and f_3. The energy weight is consistently 1.

As in previous environments, terms for sensory variables are added to 1 before multiplication for computing f_2.

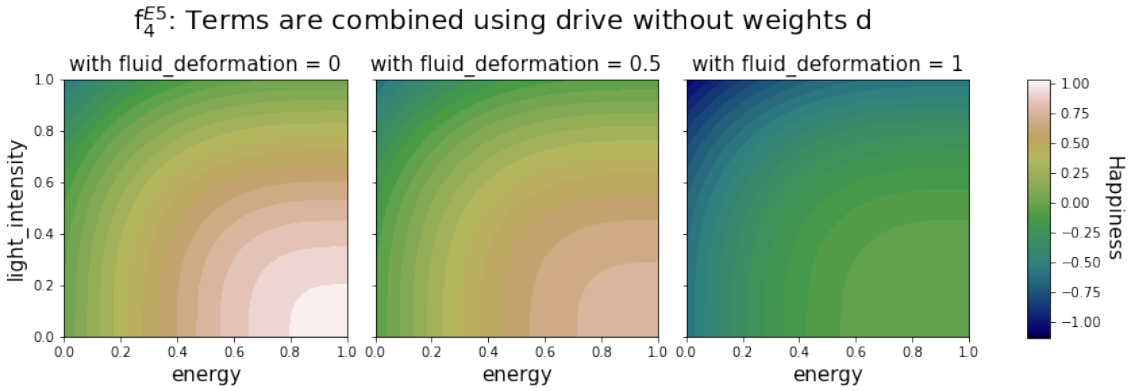


Figure 4.23: Contour plot of happiness function f_4^{E5} for energy and light intensity, with fixed values of fluid deformation. Terms are combined through drive formula with weights d, see Equation 3.7

4.4.6 Environment 6

The purpose of Environment 6, E6, is to elicit the chemotaxis behaviour B5, to escape predators sensed by scent. As in E5, this environment is inhabited by prey animats and non-agent predators, see Section 4.1.1. All animats leave a dissolving smell trace along their path, as described in Section 4.2.2, and the scent of copepods is distinct from the scent of krills. Copepod animats, which can perceive the smell present in their position and distinguish its type, are expected to learn to associate the smell of other copepods as non threatening, and on the other hand the smell of krills as threatening. The smell of copepods could possibly even be perceived as advantageous, if herd behaviour increases the chances of survival.

Moreover, food objects emit a distinct smell too, and that can be perceived by copepods. In order to replenish their energy, which constantly decreases, copepods need to eat the food objects.

As in the previous environments, krills' proto-vision is affected by light intensity, which is emitted by a diel source.

To summarize, a copepod exhibiting behaviour B6 should generally feed near the surface at night, detecting the good food with its smell perception, and stay in the depths during daytime. Moreover, it should avoid krills by moving away from the direction of the gredient of the krill smell.

The copepod prey are equipped with the following senses:

Senses:

- Smell (Chemoreception): krills smell intensity
- Smell (Chemoreception): copepods smell intensity
- Smell (Chemoreception): food smell intensity
- Touch (mechanoreception - chemoreception): tactile perception of copepods (1 observation), krills (1 observation), food (1 observation) and walls (1 observation);
- Energy level.

Each copepod animat has 1 homeostatic variable: *energy*, and a set of actions: $A = \{$Move up, Move down, Move left, Move right, Idle, Eat$\}$.

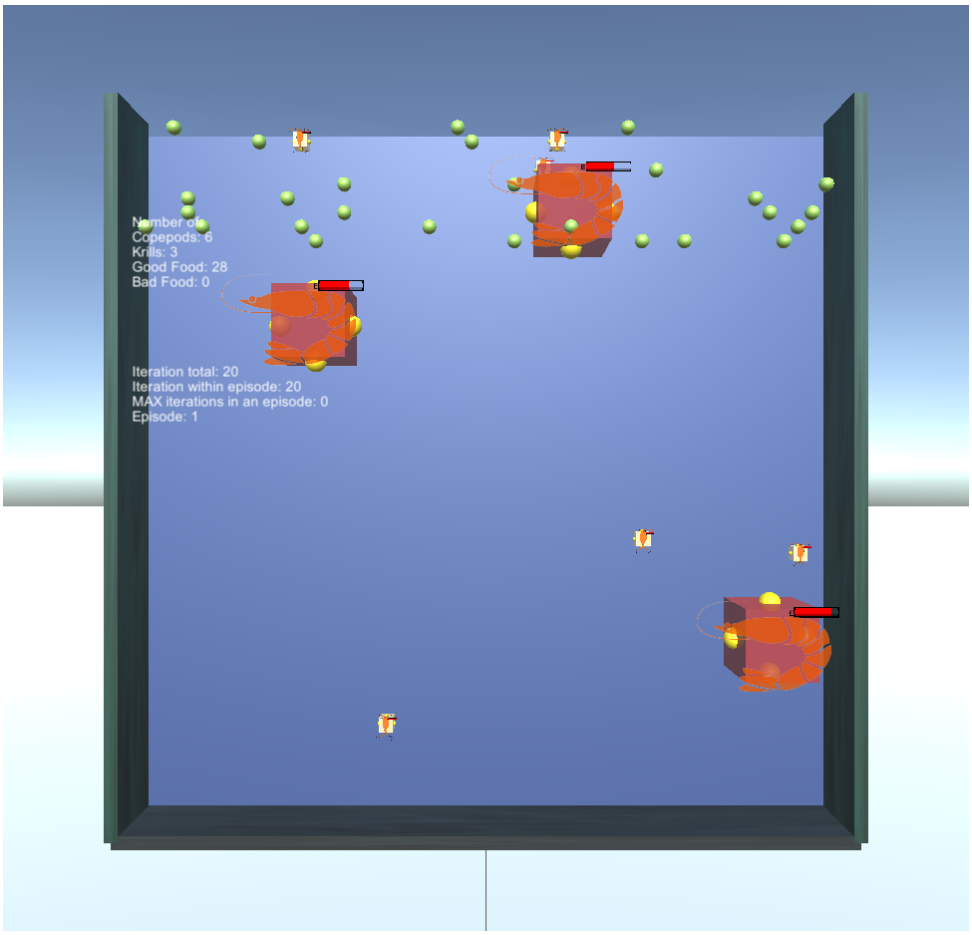


Figure 4.24: Illustration of environment 6.

We test the following happiness functions defined on 3 variables:
energy $= en$, lightIntensity $= li$ and krillSmell $= ks$, applying Equation 3.2 to translate happiness into reward (see Equation 3.6 for functions used):

$$
\begin{aligned}
f_1^{E6}(en, li, ks) &= \text{Logarithmic}(en, b = 10) + 0.2 \cdot \text{Bell}(li, \mu = 0, \sigma = 1/3) + \\
&\quad + 0.2 \cdot \text{Bell}(ks, \mu = 0, \sigma = 1/3); \\
f_2^{E6}(en, li, ks) &= \max\Big(\text{Logarithmic}(en, b = 10) + \mathbb{1}_{en>0.8}[-20(en - 0.8)^2], 0\Big) \cdot \\
&\quad \cdot (1 + 0.2 \cdot \text{Bell}(li, \mu = 0, \sigma = 1/3)) \cdot (1 + 0.2 \cdot \text{Bell}(ks, \mu = 0, \sigma = 1/3)); \\
f_3^{E6}(en, li, ks) &= 1 - D(en, li, ks), \qquad w_{en} = 1, w_{li} = 0.5, w_{ks} = 0.5, n = 3, m = 1.5; \\
f_4^{E6}(en, li, ks) &= 1 - d(en, li, ks), \qquad n = 3, m = 1.5
\end{aligned}
$$

where $d(\cdot)$ is the drive function from Equation 3.7 and $D(\cdot)$ is the drive function from Equation 3.11. The desired homeostatic states used in f_3 and f_4 are: $en^* = 1, li^* = 0, ks^* = 0$.

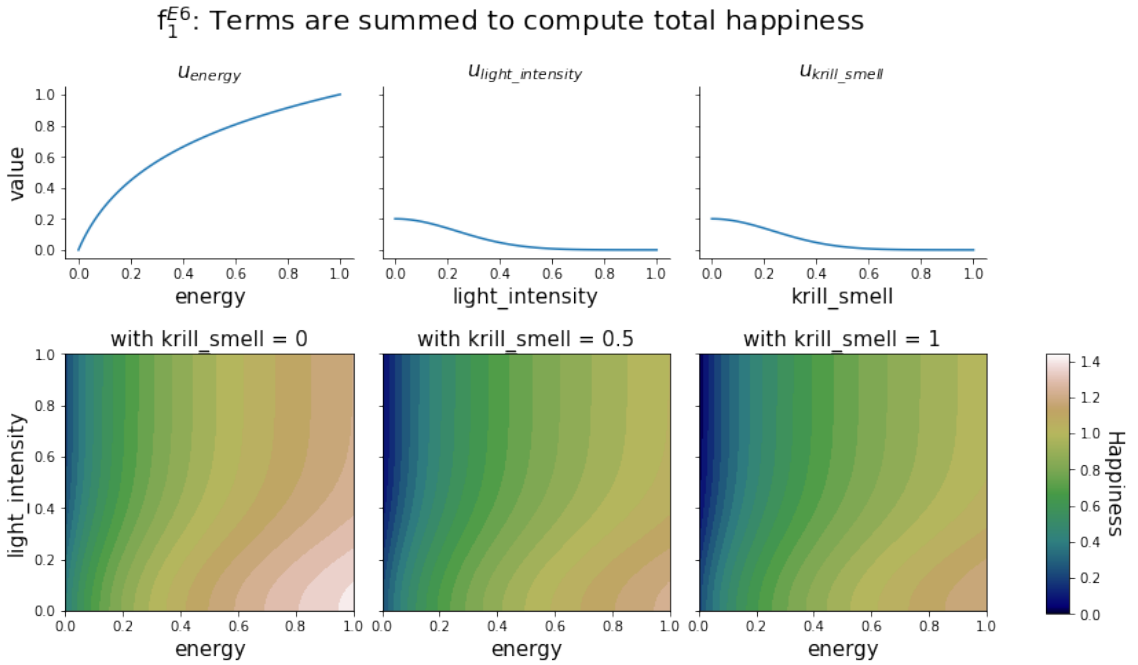


Figure 4.25: Illustration of the 3 terms composing the happiness function f_1^{E6} (above) and countour plots of f_1^{E6} for energy and light intensity, with fixed values of krill smell (below).

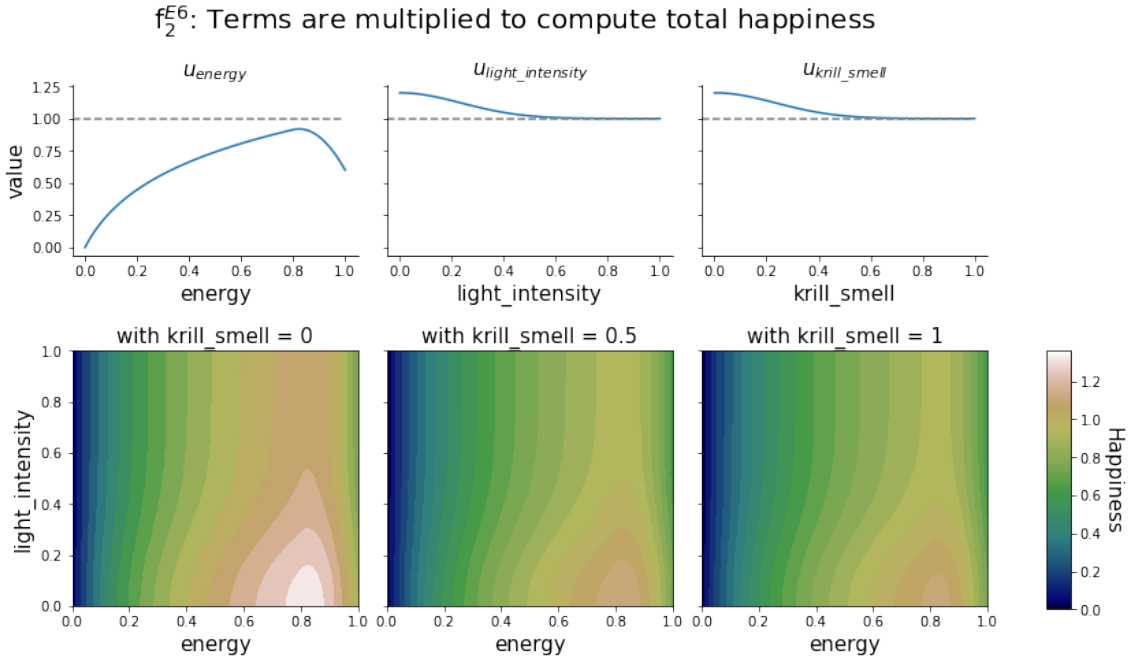


Figure 4.26: Illustration of the 3 terms composing the happiness function f_2^{E6} (above) and countour plots of f_2^{E6} for energy and light intensity, with fixed values of krill smell (below).

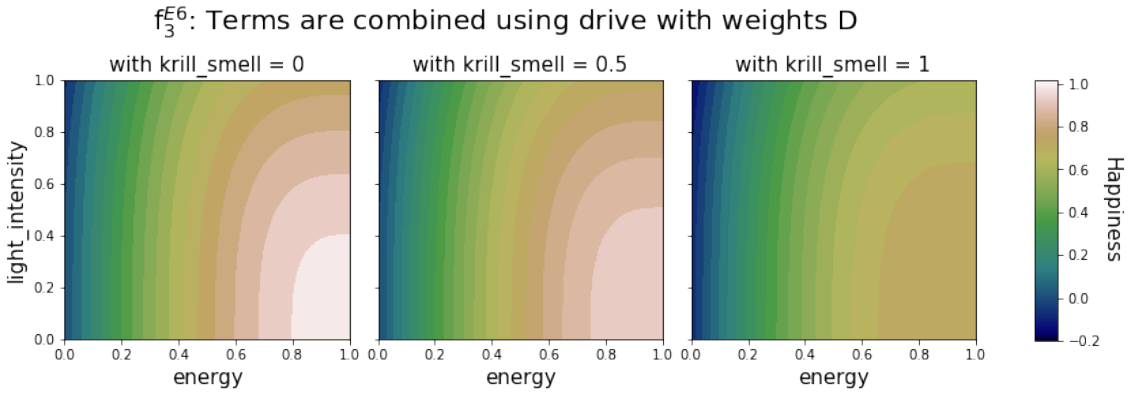


Figure 4.27: Contour plot of happiness function f_3^{E6} for energy and light intensity, with fixed values of krill smell. Terms are combined through drive formula with weights D, see Equation 3.11

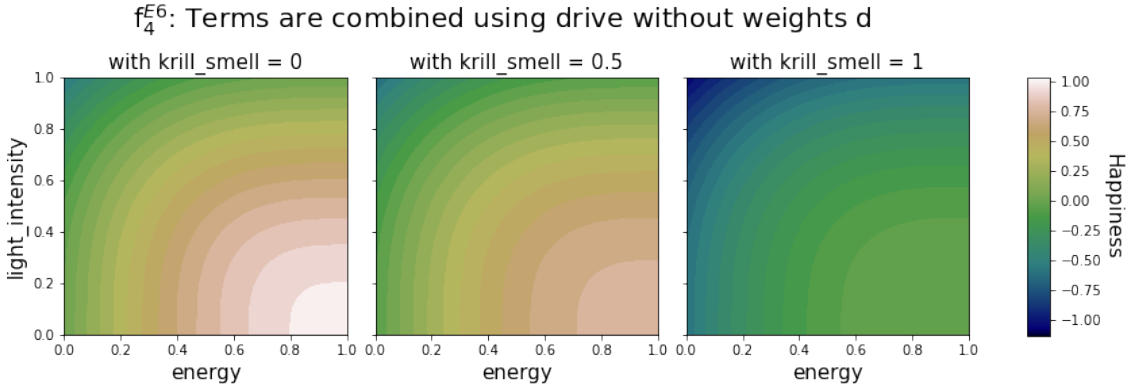


Figure 4.28: Contour plot of happiness function f_4^{E6} for energy and light intensity, with fixed values of krill smell. Terms are combined through drive formula with weights d, see Equation 3.7

4.4.7 Final Environment

Environment EF is an aggregation of all elements present in previous environments, and as such it is a general simulated marine environment whose purpose is to test the models' ability in surviving in a complex environment where multiple behaviours can be adopted for survival.

In EF copepod animats are present and their characteristics detailed below. Additionally, krill predators are present. Krills can only move in the upper-most region of the environment, above the half point in depth. Krills emit a scent which dissipates in the surroundings as in E6, end emit a fluid deformation signal when moving as in E5. Both krill-smell and fluid deformations can be perceived by copepod agents, providing multiple indications of krill presence or krill approach. In EF krills' perception, in the form of Proto-Vision, see Section 4.2.1, is affected by light intensity as in E4-6. Thus, copepod survival is greatly facilitated by DVM behaviour. Light is produced in the same manner as in E4-6. Finally, krills can attack copepods and eat *meat* objects generated when a copepod is killed by a krill,

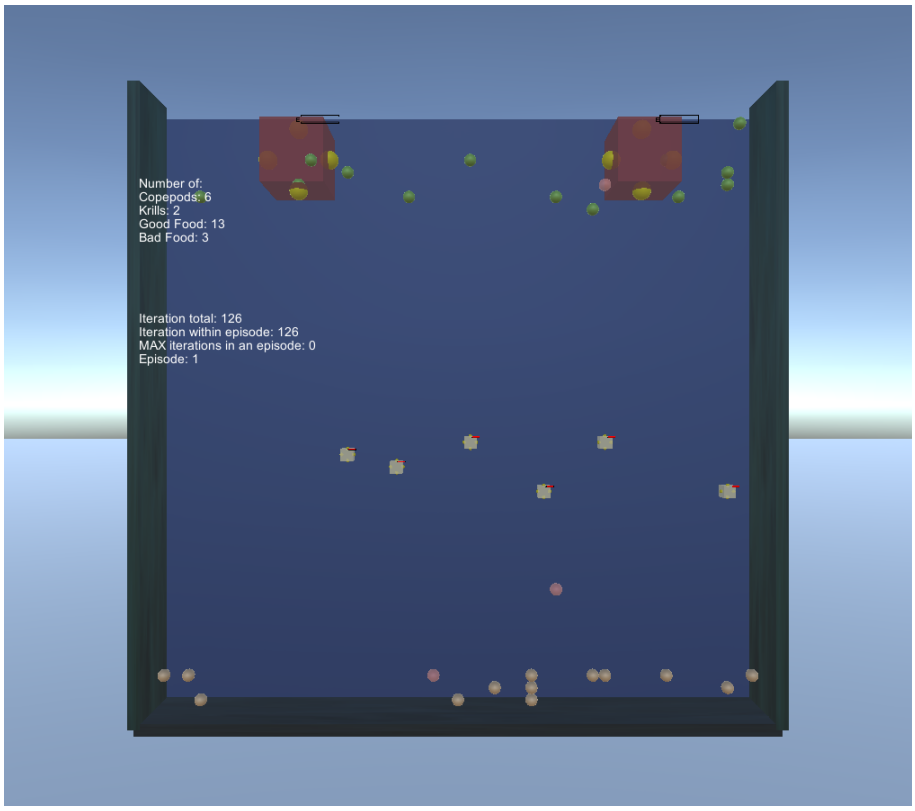


Figure 4.29: Illustration of the final environment EF. Red cubes are krill predators, yellow cubes are copepod prey, green spheres are good food, yellow spheres are vitamins food and red spheres are bad food.

as explained in E4.

In EF, copepod animats possess 2 homeostatic variables: *energy* and *vitamins*, as in environments E2 and E3. These homeostatic variables are replenished by eating respective food objects: *good food* and *vitamins food*, and as in E2 and E3, energy leads to death when 0 while vitamins leads to death when either 0 or 1, needing to be balanced in the inner of its domain. In this environment good food objects, or phytoplankton food, is found near the surface, introducing the need for exploration of the dangerous surface region, while vitamins food are found at the bottom of the environment, separating the two sources of food.

Copepod animats are equipped with the following senses:

Senses:

- Touch (mechanoreception - chemoreception): tactile perception of copepods (1 observation), krills (1 observation), good food (1 observation), vitamins food (1 observation) and walls (1 observation);
- Proto-Vision (photoreception): food direction, see 4.2.1;
- Photoreception: light intensity, see 4.2.1.1;
- Mechanoreception - baroception: fluid deformation signal, see 4.2.3;
- Smell (Chemoreception): krills smell intensity
- Energy level;
- Vitamins level.

Copepod animats have 2 homeostatic variables: *energy* and *vitamins*, and a set of actions:
$A = \{$Move up, Move down, Move left, Move right, Idle, Eat, Dash$\}$.
In this final environment we only test happiness functions f_2, defined on 3 happiness variables:
energy $= en$, vitamins $= vit$ and lightIntensity $= li$, applying Equation 3.2 to translate happiness into reward (see Equation 3.6 for functions used):

$$f_2^{EF}(en, vit, li) = \max\Big(\text{Logarithmic}(en, b = 10) + \mathbb{1}_{en>0.8}[-20(en - 0.8)^2], 0\Big) \cdot \\ \cdot \text{Bell}(vit, \mu = 0.5, \sigma = 1/3) \cdot (1 + 0.2 \cdot \text{Bell}(li, \mu = 0, \sigma = 1/3)) \,;$$

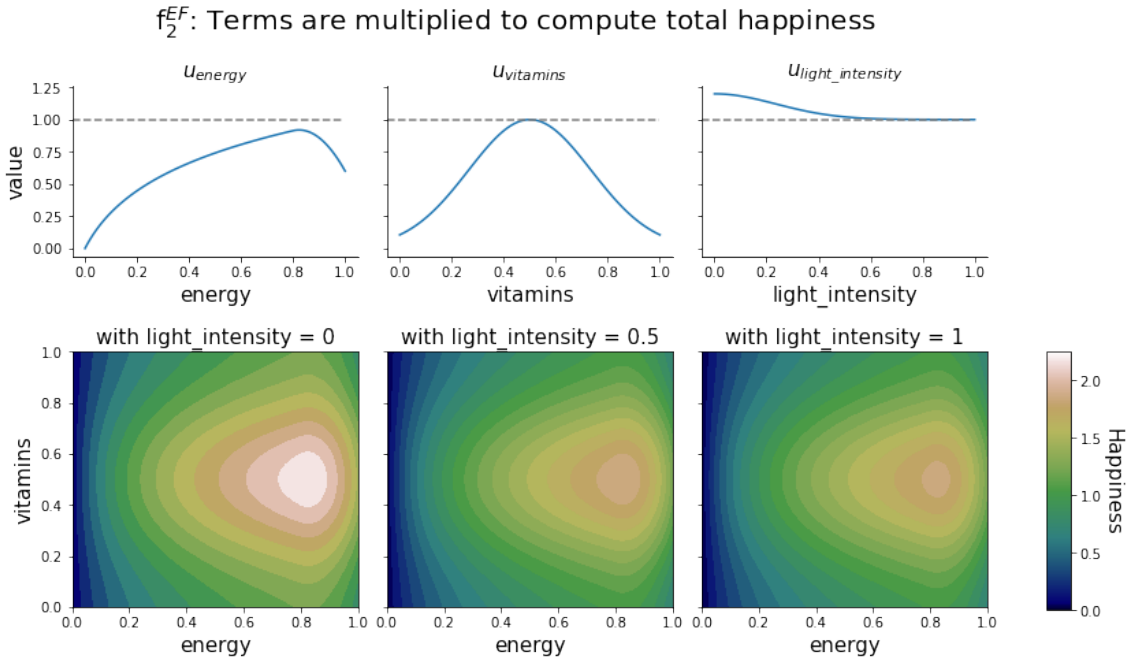


Figure 4.30: Illustration of the 3 terms composing the happiness function f_2^{EF} (above) and countour plots of f_2^{EF} for energy and vitamins, with fixed values of light intensity (below).

4.5 Tools

The experiments was initially based on code from Strannegård et al. [47] and was then reworked and extended to allow for the environments outlined above. The experiments used Unity to build the simulations and Unity together with mlagent-learns reinforcement framework and implementation of PPO for training [48].

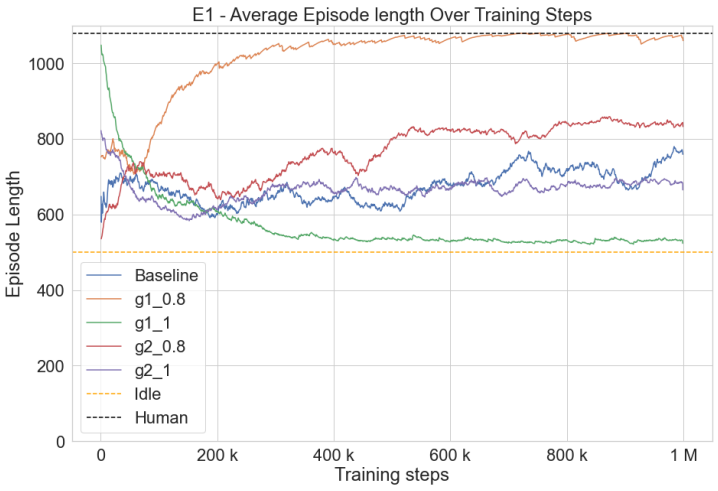
The simulations were run on hardware from Swedish National Infrastructure for Computing [49] using compute nodes with Nvidia Tesla T4 GPUs and in excess of 3000h computed.

5
Results

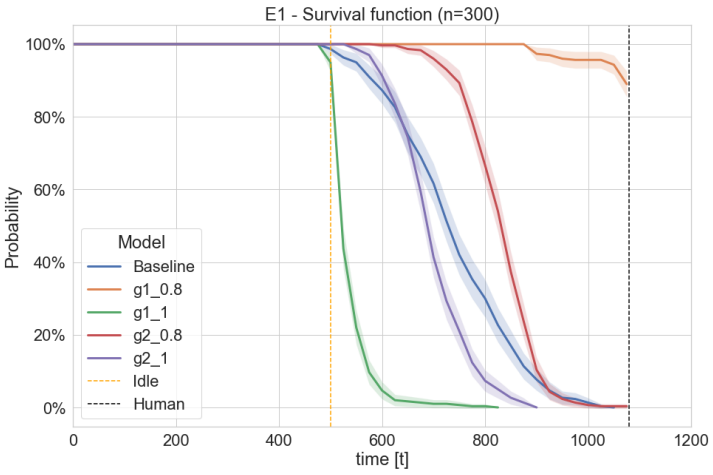
The results of the different experiments for each environments are presented below. Each experiment is designed to elicit a certain behaviour, and following said behaviour should result in a longer expected survival time. Therefore, all environments share the same evaluation tool for the models' performance: the survival function of each model.

5.1 Experiment 1

The average episode length throughout the training of each model can be seen in Figure 5.1a together with reference lines showing episodes length for an agent controlled by a human and an idling agent. Results display fast learning for $g^1_{0.8}$, medium learning for $g^2_{0.8}$ and g^2_1, comparable to the baseline model's performance, and a decreasing performance for g^1_1, failing to learn any behaviour.



(a) Episode length during training for reward functions in E1. Reference performance lines for a human controlling the animat (serving as a proxy of optimal behaviour) and an idling agent can be seen in black and yellow dashed lines.



(b) Survival function of each model evaluated in Environment E1 together with 95 % confidence bands.

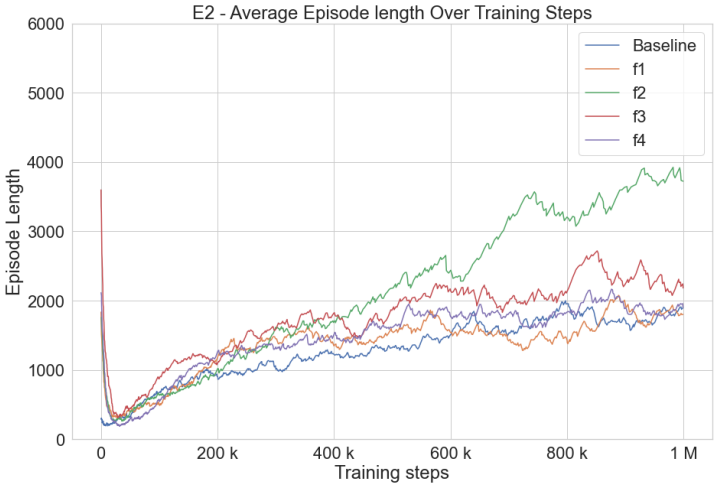
Figure 5.1: Comparative results of different reward functions tested in E1.

The survival function together with 95% confidence bands over n=300 episodes using each trained model can be shown in figure 5.1b. Mirroring the learning per-

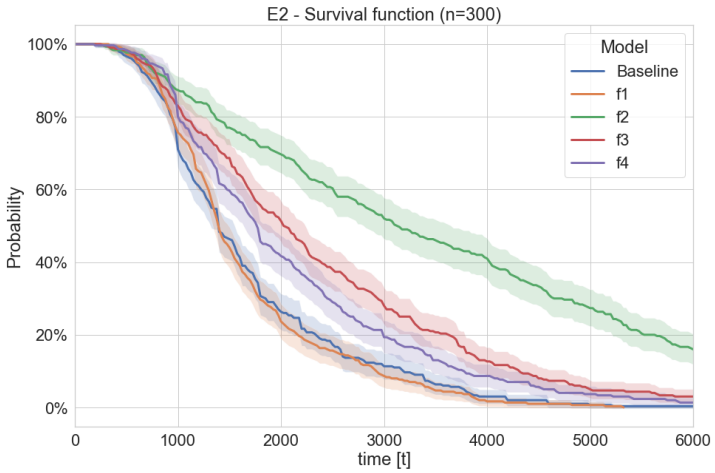
formance during training, $g_1^{0.8}$ excels in Environment 1, whereas g_1^1 performs just slightly better than an idling agent.

5.2 Experiment 2

Models performance in E2, as average episode length, can be seen in Figure 5.2a, where all models, including the baseline, show an upward trend in learning. However, model $f2$ excels over the other models that appear to converge to a lower point, while $f2$ shows a constant upward trend in learning within the 1M time-steps of training.



(a) Training-phase learning for the different reward functions tested in E2, showing that all models perform well in Environment E2



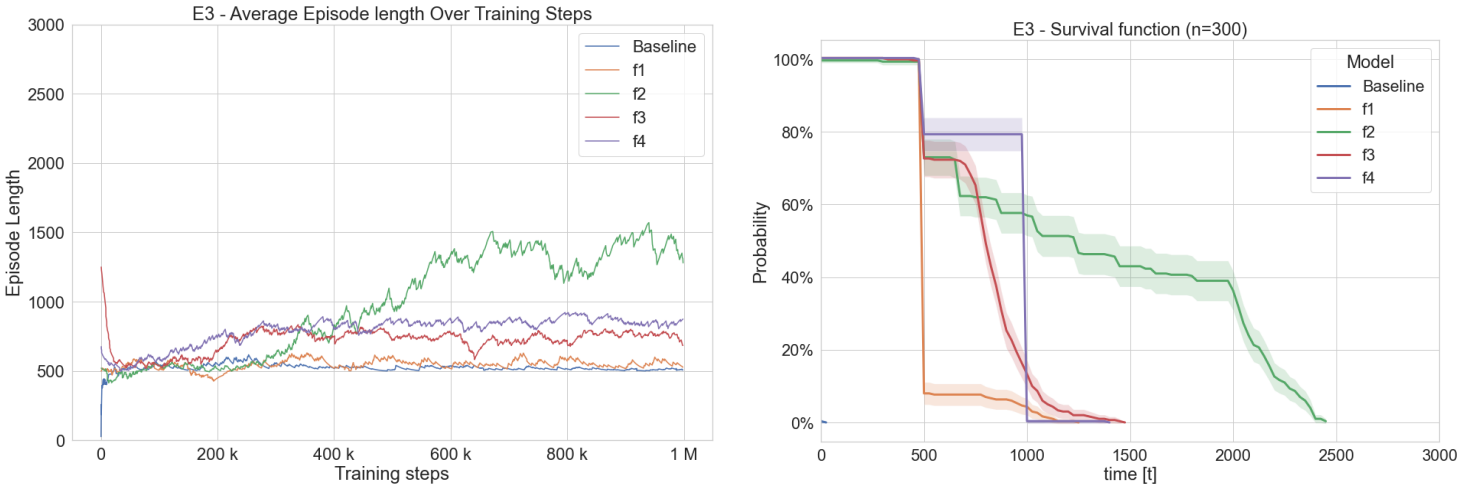
(b) Survival at testing-phase with different reward functions used in E2, with 300 sampled animats.

Figure 5.2: Comparative results of different reward functions tested in E2.

Figure 5.2b shows the survival function induced from 300 samples for each model tested. Similarly to the learning performance, f_2 show the highest probability of survival.

5.3 Experiment 3

Figure 5.3a show the average episodes length in E3 during training. In this environment, surviving longer than 1000 time-steps requires the agent to manage foraging food for both energy and vitamins. As seen in Figure 5.3a only f_2 seems to learn to forage both types of food during training.



(a) Training-phase learning for the different reward functions tested in E3, where only f_2 learns to survive over 1000 time-steps.

(b) Survival function at testing-phase of each model evaluated in Environment E3, with 95 % confidence bands. 300 animats were sampled.

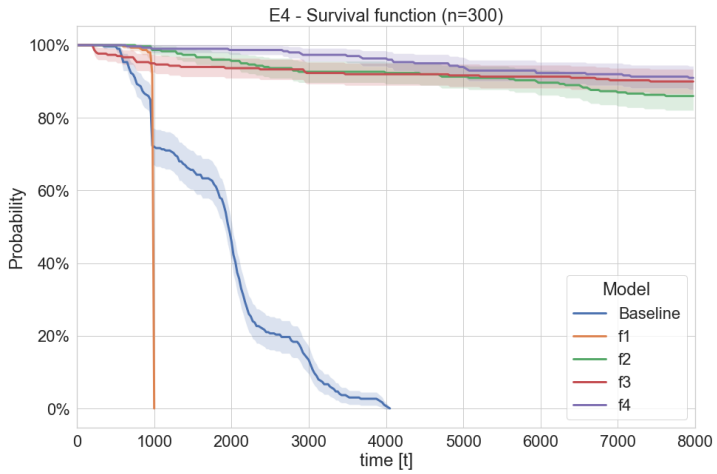
Figure 5.3: Comparative results of different reward functions tested in E3.

The need to regulate both vitamins and energy through foraging is clearly shown in the survival function of the trained models in Environment 3, see Figure 5.3b. For example, note that f_4 has learned to regulate one need while ignoring the other, thus quickly dying after 1000 time-steps, whereas f_2 focuses on both needs.

5.4 Experiment 4

Environment E4 contains six agents training in parallel. Therefore 6 training-steps in Figure 5.4a correspond to one time-step in the environment. Additionally, the models are trained with curriculum learning where 2 immortal reflex krills, spawns after 1.000 time-steps, equivalent to 6.000 training-steps, and become gradually more lethal up to 11.000 time-steps, equivalent to 66.000 training-steps, when they are fully lethal, navigating towards visible copepods and attacking them, see Section 4.1.1. Each animat's episode in the environment can be a maximum of 8.000 time-steps long. The Figure 5.4a shows the trend of the average survival time, indicated as Episode Length, during training.

(a) Episode length during training for reward functions in E4, where only f_2 learns to survive over 2000 timesteps.



(b) Survival functions of different models evaluated in environment E4, with 95 % confidence bands. 300 sampled agents were used.

Figure 5.4: Comparative results of different reward functions tested in E4.

Only functions f_2, f_3 and f_4 show any significant ability to learn the behaviour B4 needed to survive in Environment 4, see 5.6a. This is also reflected in the survival function during the testing-phase, where these models show high probabilities of survival, see Figure 5.6b. Figure 5.7 shows that f_2 clearly masters behaviour $B4$, Diel Vertical Migration, after training for 1M time-steps.

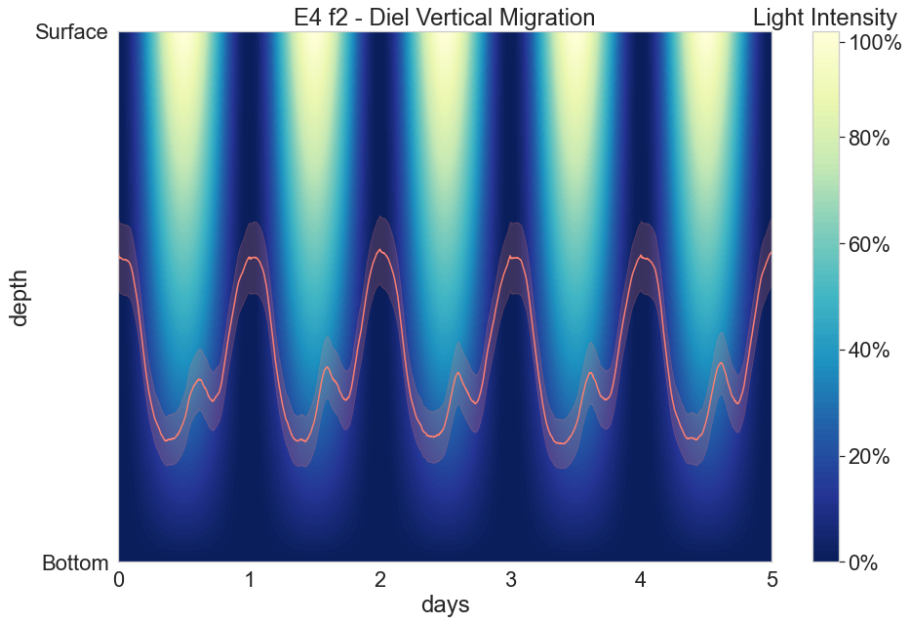


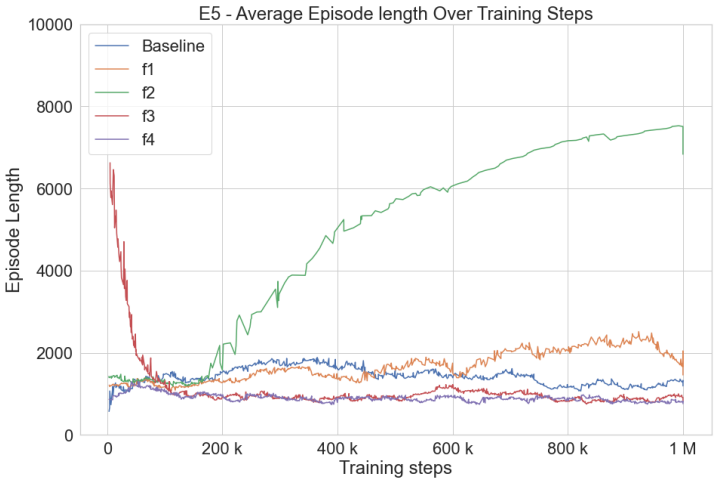
Figure 5.5: Diel Vertical Migration behaviour performed by copepod animats trained with f_2 happiness function in environment E4, during the testing-phase. The distribution over time shows the mean vertical position of 6 different copepod animats, with 95% confidence bands. The displayed time is obtained as:
$\text{days} = (\text{timestep} \mod 5T)/T$, where the period $T = 1000$ time-steps.
The background's colour indicates light intensity as a function of depth and time.

5.5 Experiment 5

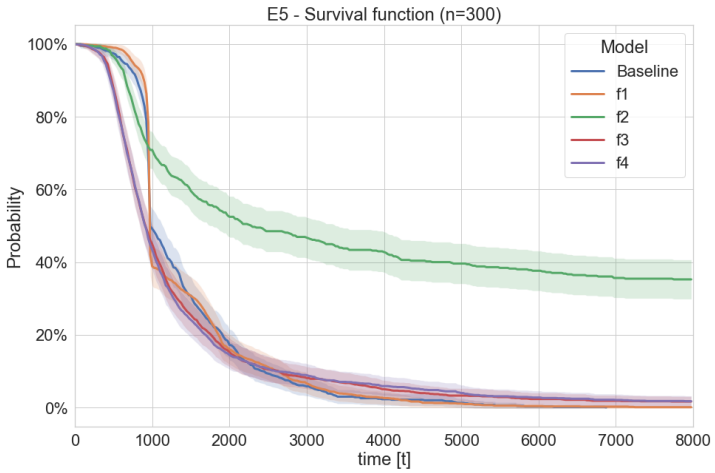
In environment E5, a number of elements are identical to environment E4, such as the diel light cycle, the non-agent predators, the presence and distribution of food and the number of copepod animats in the environment. Environment E5 is distinct in the ability of copepods to perceive fluid deformation produced by other animats, see Section 4.2.3, and in their ability to dash, moving multiple units of length in one step instead of 1 unit. Each animat's episode in the environment can be a maximum of 8.000 time-steps long. The Figure 5.6a shows the trend of the average survival time, indicated as Episode Length, during training.

In Figure 5.6 it is clear that copepod animats with happiness function f_2 are able to learn how to survive in the environment, outperforming the other models, which do not learn to survive, on average, for more than 2000 timesteps. As in previous environments, by always idling, the agents will deplete their energy resources in exactly 1000 time-steps. In this case, only the model with happiness function f_2 does not converge to the always-idle behaviour, shown by all other models.

Figure 5.7 shows the Diel Vertical Migration behaviour, during the testing-phase, of copepod animats trained for 1M training-steps with happiness function f_2.



(a) Episode length during training for reward functions in E5, where f_2 successfully learns to survive over 6000 time-steps.



(b) Survival functions of different models evaluated in environment E5, with 95 % confidence bands. 300 sampled agents were used.

Figure 5.6: Comparative results of different reward functions tested in E5.

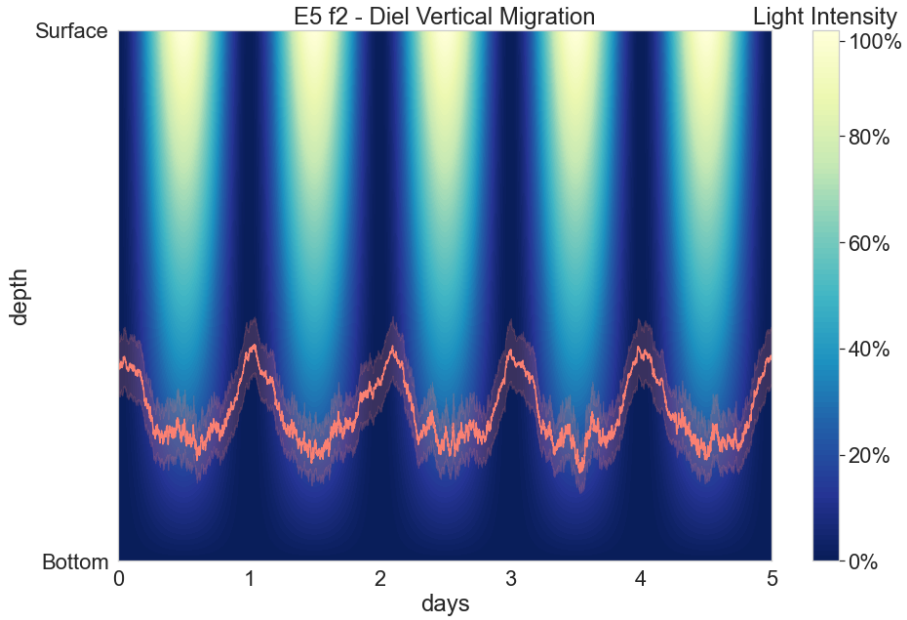
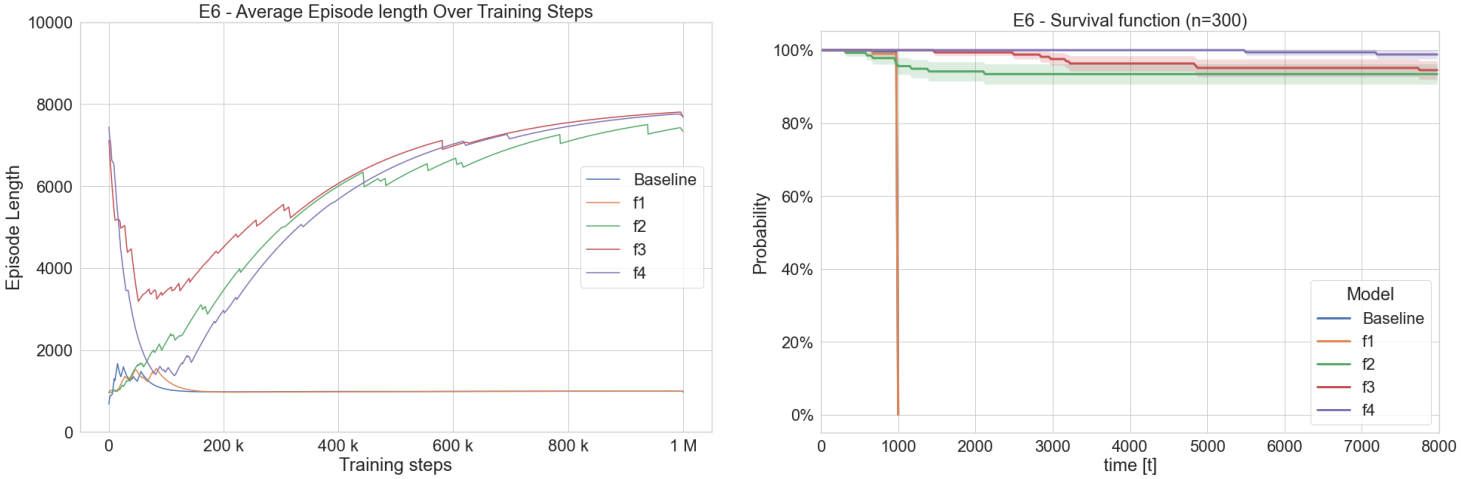


Figure 5.7: Diel Vertical Migration behaviour performed by copepod animats trained with f_2 happiness function in environment E5, during the testing-phase. The distribution over time shows the mean vertical position of 6 different copepod animats, with 95% confidence bands. The displayed time is obtained as:
$\text{days} = (\text{timestep} \mod 5T)\,/T$, where the period $T = 1000$ time-steps.
The background's colour indicates light intensity as a function of depth and time.

5.6 Experiment 6

Experiment 6 tests copepod animats with different reward models in environment E6, where DVM behaviour and the avoidance of predators through smell cues are tested. Each animat's episode in the environment can be a maximum of 8.000 time-steps long. The Figure 5.8a shows the trend of the average survival time, indicated as Episode Length, during training.



(a) Training-phase learning for the different reward functions tested in E6.

(b) Survival at testing-phase with different reward functions used in E6, with 300 sampled animats.

Figure 5.8: Comparative results of different reward functions tested in E6.

Figure 5.8, shows a clear learning split between models f_2, f_3, f_4, and the rest. During the training-phase, the 3 mentioned models show an upward trend in learning for the whole duration of 1M training-steps, equivalent to 166.666 environment-steps. All other models, instead, quickly converge to the always-idle behaviour, since their episode lengths converge to 1000 time-steps. Training results are reflected during the testing-phase as seen in Figure 5.8b.

Figure 5.9 shows the Diel Vertical Migration behaviour, during the testing-phase, of copepod animats trained for 1M training-steps with happiness function f_2.

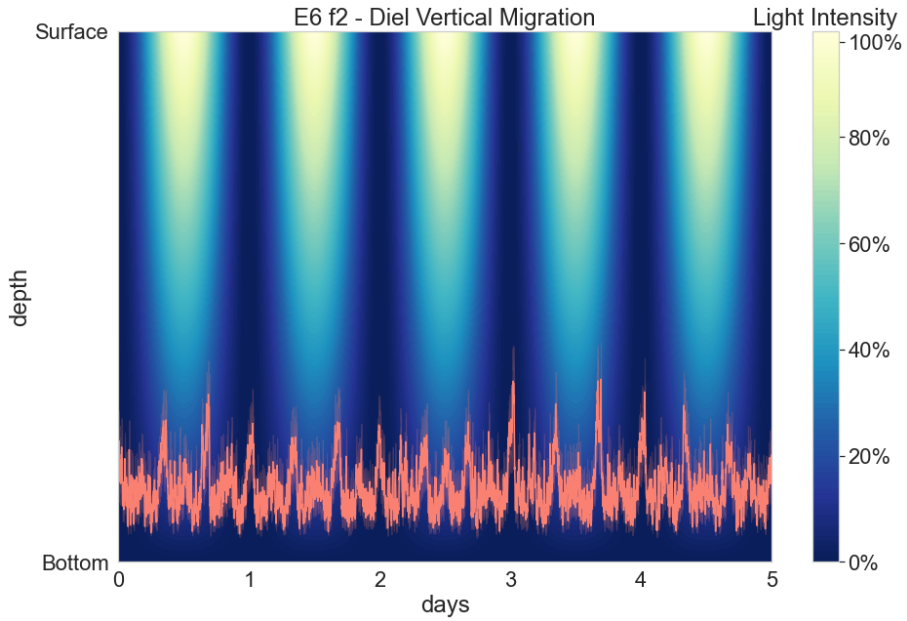
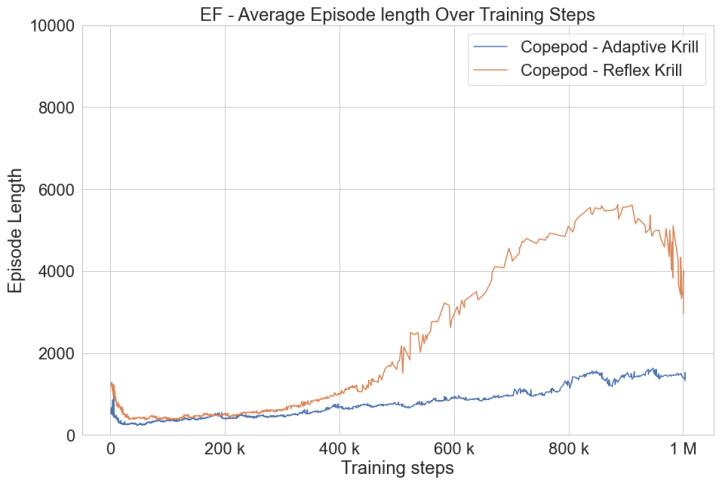


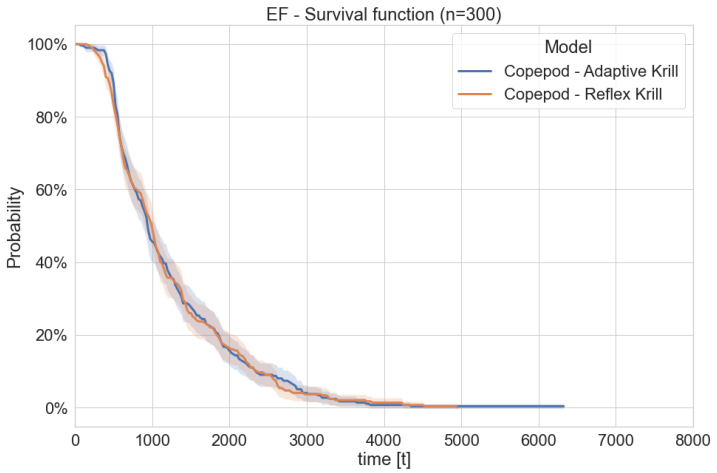
Figure 5.9: Diel Vertical Migration behaviour performed by copepod animats trained with f_2 happiness function in environment E6, during the testing-phase. The distribution over time shows the mean vertical position of 6 different copepod animats, with 95% confidence bands. The displayed time is obtained as:
$\text{days} = (\text{timestep} \mod 5T)/T$, where the period $T = 1000$ time-steps.
The background's colour indicates light intensity as a function of depth and time.

5.7 Experiment 7 - Final Environment

For the seventh and final experiment, the complete copepod model with happiness function f_2, detailed in Section 4.4.7, is trained in two similar environments, where only the predators present differ. In one environment the predators are adaptive krills, whereas in the second environment the predators are non-agent krills, or *reflex krills*, moving in the direction of copepods sensed through light-sensitive proto-vision, identical to the ones in E4, E5 and E6. In both environments, each animat's episode can be a maximum of 8.000 time-steps long, but episodes are terminated when an animat dies. The Figure 5.10a shows the trend of the average survival time, indicated as Episode Length, during training. Results for each environment, can be seen in Figure 5.10.

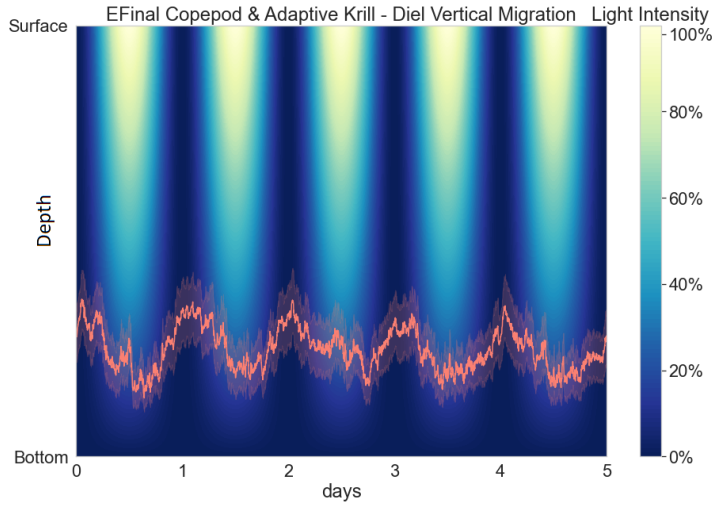


(a) Training-phase learning for copepod animats with happiness function f_2, tested in 2 EF environments with different types of krill predators.

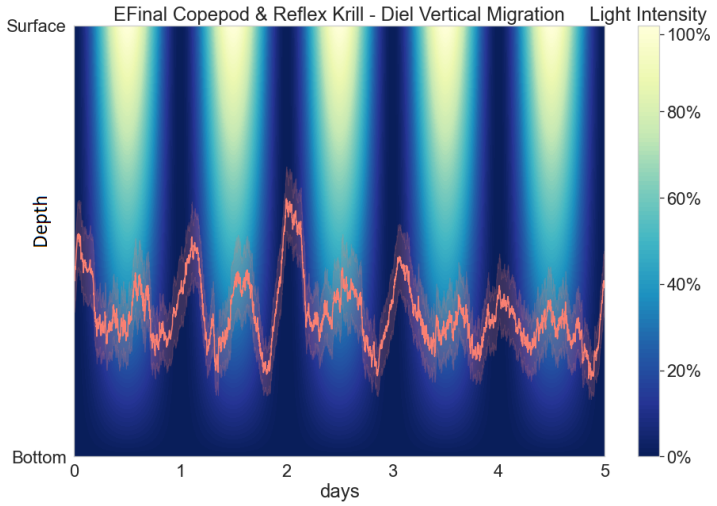


(b) Survival at testing-phase for copepod animats with happiness function f_2, tested in 2 EF environments with different types of krill predators, with 300 sampled animats.

Figure 5.10: Comparative results of copepod animats with happiness function f_2 in 2 different environments tested in EF.



(a) DVM behaviour in EF environment with adaptive krills.



(b) DVM behaviour in EF environment with reflex krills.

Figure 5.11: Diel Vertical Migration behaviour performed by copepod animats trained with f_2 happiness function in environment EF with 2 kinds of krill predators, during the testing-phase. The distribution over time shows the mean vertical position of 6 different copepod animats, with 95% confidence bands. The displayed time is obtained as:
$\text{days} = (\text{timestep} \mod 5T)\,/T$, where the period $T = 1000$ time-steps.
The background's colour indicates light intensity as a function of depth and time.

6

Discussion

6.1 Comparison to Nature

The environments and animats in this thesis project emulate aspects of real ecosystems. Environments have positional characteristics such as various smell types' intensities and light intensity. Animats possess various perception senses. They can interact with each other through smell perception, touch perception and attacks. Additionally, animats' physiology is defined through a set of homeostatic variables which are affected by the environment, time and food nutrients. Homeostatic variables serve to simulate animats' motivation using a reward function appropriately defined on the homeostatic space. As such, we developed a framework which allows for the construction of environments and animats resembling real ecosystems and real animals, sufficiently realistic to elicit behaviour B1-B6, defined in Section 1.5.

At the same time, it is useful to underline differences between the proposed framework and real ecosystems, especially in the perspective of possible future advancements in this topic, aiming at more realistic ecosystems.

To begin with, in experiments E1-E6 and EF environmental complexity aims at being minimal, limiting variability and thus focusing on one particular mechanic of interest in each experiment, additionally saving time resources. Environments are thus all squared, flat and enclosed with impassable borders, which are not typical characteristics of real environments.

In addition, both spatial and time scale differ from real ecosystems. Firstly, our marine environments are 2-dimensional instead of 3-dimensional. This choice limits the amount of animats that are required in the environment to get the same amount of interactions, limiting the computational resources required; moreover, we argue that behaviours B1-B6 are not greatly affected by the dimensional reduction: diel vertical migration for example,

Chosen sizes of environments are small in comparison to the animats' size, with respect to real environments which animals inhabit. Moreover, relative sizes between animats and their food, and between predators and prey are within the same order of magnitude. The main reason behind this simplification is to improve visualization of the dynamics. The time scales in the experiments differs from real ecosystem, e.g. animats spend one time-step to consume food, while spending the same amount of time to move 1 spatial unit, corresponding to the size of copepod animats or the size of food itself.

6.2 Feasibility and Success Criterion in Learning Target Behaviours

Behaviour B1, the regulation of hunger when food is scarce, requires an environment with limited, exhaustible and easily reachable food supplies, where time depletes the animat's energy resources. At the same time, the animat has to be able to perceive the proximity to food, which enables it to eat the food object by performing an additional action of eating. The animat has to perceive its internal energy level at all times, which allows it to infer how much energy is replenished by one food object. These elements are present in environment E1 and allow an agent to purposefully eat a food object when its energy level is below a certain threshold, i.e. when it is not satiated. Since energy depletes by a constant amount at each time-step and food is limited, greedy eating, i.e. eating when energy is above said threshold, is a direct cause to a shorter lifespan, along with unmotivated movement, which costs energy. Thus, a long life-span, reflected in a long episode length, is only possible when purposeful movement and regulation of hunger are both performed. Figure 5.1 clearly shows that the only successful animat in this environment was the one using happiness function f_2, reaching close to optimal performance in a relatively short training time of around 300k time-steps, which indicates successful learning of both motivated movement and behaviour B1.

Behaviour B2, selective eating by differentiating food types, requires the animat to locate and distinguish between different types of foods, each with a different effect on the animat's physiology, along with the animat's ability to perceive proximity to a type of food and sensing its own physiological values. If the animat eats the wrong type of food, some critical homeostatic variable might go out of bounds causing its death. Therefore, in order to survive for a longer time, the animat has to not eat toxic food, which can cause energy to drop below zero, and balance intake of vitamin food so that its inner vitamins value does not become too high or too low, causing death. Finally, the animat has to satisfy the need for energy from good food, which increases the animat's energy. Good food respawns at a low rate, requiring the animat to regulate its hunger in order to survive between food-spawns. Figure 5.2 shows that animats using happiness function f_2 learn behaviour B2 the fastest, and have the highest chances of survival in the environment during testing-phase.

Environment E3 is designed to elicit Behaviour B3, i.e. finding food sources by following scent trails. In this environment, in order to survive, the agent has to find good food (replenishing energy) and vitamins food (replenishing vitamins), which are found at two separate sources, in order to satisfy and balance energy and vitamins intake for their respective homeostatic variables not to go out of bounds, causing death. The position of food sources changes with time, which does not allow the animat to only learn the various food sources' positions once for the entirety of the episode. The animat is thus forced to forage food through scent trails in order to reliably secure food through longer periods of time. Thus, a larger survival time implies the learning of behaviour B3. Figure 5.3 shows that animats equipped with reward function f_2 learn behaviour B2 the fastest, and have the greatest chances of survival in environment E3.

The results from Environment E2 and E3 show that animats, in particular

when using happiness function f_2, manage to balance multiple vital needs, where one of them also needs to be balanced at 0.5, as vitamin levels of outside the interval $(0, 1)$ cause death.

While Behaviours B1-3 are general behaviours of interest for animats in a simulated ecosystem, mostly adopted with the purpose of testing the proposed model's capabilities, behaviours B4-6 can be observed in real copepods [12, 11, 9, 8, 13].

Behaviour B4, is a type of phototaxis shown by copepods moving up and down depending on the perceived light intensity. In order to elicit this behaviour, environment E4 uses a light source emitting oscillating light intensity with a diel cycle, which gradually dissipates the further from the source, i.e. the deeper down in the marine environment. In this environment predators, i.e. krills, spawn close to the surface and their vision is affected by light intensity, causing their sight distance to be shorter during night and at depths. This makes copepods' journey to the surface safer at night and more dangerous during daytime. These elements favor a Diel Vertical Migration behaviour, or DVM, consisting in copepods moving towards the surface during night in order to eat and staying at the bottom during day in order to lower the risk of predation. As such, DVM behaviour, or B4, greatly increases the probabilities of survival. It is in fact the behaviour which increases the probability of survival the most, since it optimizes the environment's features. Figure 5.4 shows that the happiness functions f_2, f_3 and f_4 yield the fastest learning and the highest survival chance during testing, while the other 2 tested functions do not manage to reach satisfactory results within training time. It is noticeable that f_1 and baseline functions converge to an episode length of 1000 time-steps, corresponding to the energy depletion time when idling. Idling copepod animats thus optimized the local maximum of idling in order to not consume energy resources with movement, possibly because of lack of exploration and never experiencing the states successive to eating food objects.

Environment E5 elicits behaviour B5, where animats escape close and fast-approaching predators based on barokinesis. All animats in environment E5 produce a fluid deformation signal, when moving, within a certain radius of an intensity which is proportional to the animat's size squared. The intensity of the signal produced by a krill is thus much greater than that produced by a copepod. Note that since fluid deformation is calculated based on both size and speed the source, and all fluid deformation signals perceived are summed before being sensed, it might be possible on rare occasions that a school of dashing copepods produce a krill-level fluid deformation signal. Copepod animats can dash at a large energy expense, it is thus only convenient for copepods to dash when sensing intense fluid deformation, most likely produced by a close predator. The fluid deformation signal, together with touch, is the only copepods sense of predator presence in environment E5.

It can be seen in Figure 5.6 that f_2 model excels in environment E5. Thus, behaviour B5 is observed in Environment 5 using when using f_2, as it dashes away from pressure signals.

While f_1 does not manage to display an average survival time larger than 2000 time-steps, corresponding to 2 days, the other models visibly converge to 1000 time-steps, corresponding to the time it takes for their energy to deplete when idling.

Behaviour B6, a type of chemotaxis defined by escaping predators sensed by

scent, is elicited in environment E6 by having krill predator and copepod animats emit distinct scent and allowing copepod animats to perceive and distinguish both types of scent. Since copepods can only perceive krills through smell and touch, which arguably is perceived at a late stage for escaping, copepods' best chance of survival is by generally moving in the direction of the negative gradient of krill smell intensity, when such intensity is great enough to indicate closeness to a predator. This movement, along DVM behaviour, indicates behaviour B6. Thus, behaviour B6 greatly increases the probabilities of survival, increasing the average survival time. In Figure 5.8 it is clear that models f_2, f_3 and f_4 manage to learn to not idle. These models also display a good performance during the testing-phase, generally leading to a survival across multiple days.

Interestingly, the models that succeed in E6 are also the models that succeed in E4, but f_3 and f_4 do not succeed in E5. E5 might thus constitute a harder environment to adapt to. This might be explained by the copepods' availability to the costly dash action in E5, which constitutes a more easily learnable alternative for decreasing the chances of death, compared to the DVM behaviour. Dashing whenever a predator is sensed can thus be seen as a easier local optima for copepods.

6.3 DVM Behaviour

In environment E4 copepods have no way of sensing predators. The only behaviour that yields long expected survival time is thus to travel to the surface during night when krills can only touch them but not sense them through proto-vision, see Section 4.2.1.1. Environments E5 and E6 extend E4 by introducing a dashing action and fluid deformation or animats' smell diffusion and smell perception, which can be used to sense and avoid predators. Any agent that manages to survive in E4 would thus manage to survive in E5 and E6 by adopting the same behaviour. Therefore, survival time might not only be correlated with B5 and B6 in Environment E5 and E6, but also with B4, i.e. DVM.

DVM does not only increase the chances of survival in E5 and E6, but is necessary for enduring in those environments, since foraging food during daytime is highly risky as during that time predators are effective at chasing and killing copepods.

However, when introducing additional rewards for fluid deformation and smell together with a costly dashing action, local optimas of minimizing fluid deformation and smell together with high expense movement is introduced. Those local optima might be easier to learn compared to the DVM behaviour, but only favor short term survival.

6.4 Happiness Functions' Design

6.4.1 Baseline reward function

Experiments E1-E6 were designed to elicit specific behaviours. An agent that successfully learn the behaviours should have a longer expected survival time. There-

fore, the most natural and general choice of reward would be to reward the agent for surviving. Survival time is also the end success criteria measured in each experiment in order to know if the model performed well.

The baseline reward signal rewards the agent for staying alive by giving it a constant reward for each time step. Maximizing the cumulative reward $J(\pi_{Baseline})$, which is the agents' learning objective means maximizing the success criteria measured in each experiment. Therefore, the choice of baseline model in this project is highly sensible and would perhaps be the reward signal chosen by researchers focusing on other components of ecosystems, along with heuristic rewards.

Since the learning objective in each environment corresponds to a behaviour which maximizes the episode length, the episode length's trend over time-steps serves as a proxy for measuring the learning during training. The pre-training plot thus shows and compares learning speeds and eventual behaviour convergence, when the experiment's duration is large enough. Conversely, the testing-phase plots display survival functions for each reward function tested, sampling 300 episodes as explained above. More precisely, a survival function assigns, for each episode-length t, the percentage of samples with larger or equal episode duration.

Theoretically, in a stationary environment, good RL models would be expected to perform optimally given the baseline reward signal. However, this can not be observed in the results, see the Baseline performance in all environments in which it was tested, with the exception of E2 and E4 where animats using the Baseline reward manage to survive 2 days on average, see Figures 5.2, 5.4. The animats are not allowed infinite exploration as this is infeasible for ecosystem simulations and this project in particular. In addition, the choice of reinforcement learning algorithm further limits the amount of exploring that animats can perform, as discussed below.

6.4.2 f_1 and f_2 Happiness Functions in Comparison

Two types of happiness functions, which generate the reward through Equation 3.2, have been introduced, along with an additional reward function, generalization from the reward function proposed by Keramati et al.[5].

The happiness function calculated through summation of different terms is denoted f_1. In f_1, each single utility term has a linear effect on happiness. In other words, the value of a happiness variable doesn't affect the effect of any other happiness variable on happiness. This happiness function is highly intuitive but the limited interconnection between homeostatic variables drastically lowers the useful effect of inhibition of the irrelevant drives, explained in Section 3.2.3, which might explain its poor performance. In other words, animats using f_1 as reward function are rewarded to focus on particular aspects of the environment and might fail to see the bigger picture.

When the happiness value is calculated through the product of the different terms, the reward function is denoted f_2. f_2 is generally the best performing happiness function for the animat model adopted, between the happiness functions tested, as seen from the consistently good performance, superior to the other models in Figures 5.2, 5.3, 5.6, or on par with other well-performing models, see Figures 5.4, 5.8. f_2 happiness functions make great use of the inhibition of irrelevant drives concept,

as each single homeostatic variable can cause happiness to drop to zero, whereas sensory variables can only increase the total happiness value computed by multiplying the homeostatic terms first, see Equation 3.5. Another point of view of this feature is that the happiness gain when improving the value of an irrelevant need is greatly diminished when a homeostatic variable is in an high-drive state. This feature helps ranking the needs depending on their value.

6.4.3 The Zero-sum Problem of Cumulative Reward

The reward is calculated using the difference in happiness, see Equation 3.2. The cumulative reward of a trajectory with homeostatic-sensory states spanning over a closed loop in the homeostatic-sensory space $H \times S$ is equal to zero, without regard to the shape of the happiness surface. Because of this, the expected return when an episode ends with the death of the animat is negative, since the animat's homeostatic state is initialized in a good state, usually in the state with peak happiness, and death always corresponds to a state with low happiness. The PPO algorithm maximizes the expected discounted return, using the discount factor γ. Thus, rewards that are further in future time possess a lower weight compared to rewards that are closer in future time. Higher discounting through a lower γ parameter therefore leads to a model more focused on maximizing the present reward, which in our model is equivalent to achieving homeostasis. This mitigates the zero expected return problem, at the cost of introducing bias. The default setting $\gamma = 0.99$ used in the project was sufficient to enable this, as shown by the efficient learning of f_2 throughout the environments.

Generally, the framework theorized by Keramati et al. [5] denoted as f_4 performed quite poorly, with the exception of $E4$ and $E6$. In these settings the agents managed to explore the environment sufficiently to find the policy leading to survival. When the agent survives the cumulative reward is not zero as the closed loop is broken, i.e. the homeostatic-sensory states are not depleted. This is quite a large flaw which makes the Keramati et al. model infeasible to use when agents are expected to die, which is common in ecosystems.

6.4.4 Information-rich Happiness Functions

In environment E1, the best performing models use a penalty term for satiety in the happiness function. This design choice was inspired by Nature's way of providing multiple signals, see Section 3.1.5, making the agent receive a reward signal which more closely reflects the long term effect of the eating action: when overeating, happiness will diminish and the reward will be negative. In addition, this signal provides a strong signal for the agent to follow. The agent can infer from the reward received whether it is hungry, receiving negative rewards when energy depletes, or full, receiving positive rewards when energy depletes. Inspirations from Nature, e.g. in the form of a penalty term for satiety, exemplifies the ease in modeling basic phenomena such as contrasting signals with a trade-off effect.

6.4.5 Idling as a Locally Optimal Behaviour

In Environments E4, E5 and E6 the predators are constructed to perfectly chase and kill copepods when their perception is sharp; with high light intensity they constitute a great threat to copepods. Thus, before learning that food can be found in the same region where predators are present, a locally optimal behaviour might be to idle at the bottom of the environment in order to be safe from predation. Therefore, a copepod animat has to rely on exploration at the beginning of learning in order to eat food and increase the survival time over the threshold which can be reached by idling, without eating. Note that the baseline, which is a really good reward signal in other projects, also fails to break out of this locally optimal behaviour.

Note that all functions can try to optimize the local optimum with low energy, but in f_1 this local optimum is higher compared to the other cases because of the summation between terms, see Equation 3.3. Therefore, f_1 animats cannot differentiate between hunger and the need to follow scent or stay in the dark. This leads to greedy optimization by the models trained by f_1, often not deprioritizing the irrelevant needs at the cost of the vital ones. In particular, this can be seen in experiments 4-6 where the agents choose to idle at the depth of the environment, minimizing exposure to sun and slowly dying due to lack of energy. Any exploration led to lower reward from light exposure and low risk of survival, not being worth more than any slight potential of increasing its energy.

6.4.6 Inhibition of Irrelevant Needs

Contrary to f_1 and baseline models, f_2 models manage to break out of greedy optimization by inhibiting non-vital needs such as light when a vital need is in a critical state, e.g. in a moment of lack of energy. This motivates the better performance of f_2 models.

In contrast to the Keramati et al. inspired models f_3 and f_4, the f_2 happiness function has more flexibility by allowing the selection of different univariate functions per variable. This enables larger flexibility in designing the interactions between different variables and thus different needs. In particular, Keramati et al. do not allow for conditionally ignoring non-critical variables, which might be one of the elements causing happiness function f_2 to outperform the other models.

6.4.7 Happiness Functions' Flexibility

It should be noted that f_3 and f_4 have a much smaller number of parameters: f_3 has n weights and 2 hyperparameters n and m while f_4 only has the two parameters n and m. f_1 and f_2 on the other hand, allow to incorporate distinct signals for distinct needs, inspired by nature, and thus enable the model to rely more on instinct and less on exploration, enabling it to overcome the stability issues from only one agent.

The added flexibility in the proposed network is critical for ecosystem simulations, as this enables modeling specific hand-designed rewards, such as in experiment 1 where animats are forced to not over-consume food, or in experiment 3 where animats have to follow scent trails in order to find food sources. Being able to hand-design such features is critical for ecosystems as this enables training

specie-specific behaviours and using behaviours observed in Nature, as opposed to completely learning the behaviour. This can be preferable as phenomena selected by evolution can be quite powerful. More accurate design of the reward function greatly lowers exploration needs, which can otherwise be large, as animats are placed in complex environments. Finally, reward shaping allows for controlling what behaviours different species can learn. The more general a reward function, the more arbitrary the behaviour it learns. The agents might learn to exploit designs in the experiment that are not present in reality, which may mitigate the need for exploration, by providing a richer reward signal.

However, happiness functions' design can still be non-trivial. Early designs of experiment 4 used the krill seen in the final environment, but without any restrictions on movement. When krills and copepods learn to behave depending on the skills and behaviours developed by other agents, in a fashion similar to co-evolution, it was noted that krills could easily kill copepods by traveling to the bottom of the environment and searching the corners, although not being able to use proto-vision in a dark ambience, see Section 4.2.1.1. The environment's corners exist for feasibility of the simulation, as explained above, and in reality the ocean's bottom would be further down. It is thus far more complex to simulate ecosystems and a great deal of restrictions needs to be carefully set. The fact that krills learned from copepods' behaviour is yet another reason why a heuristic model was selected for krills in environments E4-6, as this enables constant predation throughout the various experiments, so that a comparison between copepod models is supported, as they face the same constant threat.

6.5 Reinforcement Learning Considerations

6.5.1 Single-agent and Multi-agent Environments

In experiments 1-3, the models were trained using only one active agent. This creates problems for PPO in particular as the samples are highly correlated and thus makes it harder to train a neural network as controller [50], discussed further below. This might be another reason for the generally worse performance of f_3 and f_4 models in E2 and E3 in comparison to E4 and E6. This is made worse by the fact that f_1, f_3 and f_4 are symmetric, meaning that the advantage estimations all point to the same negative reward function. This is equal to running several epochs of updates on a neural network using MSE with 99.9% negative samples and one positive: the loss function will gain very little in its objective function to fit the 0.01 % rare event. The advantage estimator, which relies on an underlying neural network to estimate the state value function, is fitted with a MSE loss function, see Equations 2.2, 2.3, on data where only 3 out of 1000 observations contain slight positive values, due to the rate of eating in those environments. The MSE loss function handles such anomalies poorly, as the improvement in fit is negligible when trying to fit the 0.3% positive rewards. However, when training several agents in E4-E6, samples are given throughout the lifetime of the agents, breaking the correlation between samples.

6.5.2 Exploration in PPO

Policy gradient methods have a tendency for their exploration to collapse as they optimize a loss function that depends on its current policy, and not the optimal policy. Therefore to fit to the current policy can mean overfitting to a potential local optima.

The main idea of PPO is to control the policy updates and only allow small and sensible updates of the policy through its clipping mechanism in its surrogate objective function, see Equation 2.4. This, together with an entropy term added in the loss function, keeps the policy from local optima. Moreover, the use of A3C architecture makes PPO a clear choice for on-policy models when using a neural network as controller.

However, PPO only mitigates these effects, the tendency of exploration to collapse is still present, albeit lower. This can be clearly seen in Experiments E3-E6, see Figures 5.3, 5.4, 5.6, 5.8, and in particular for the baseline model in Figure 5.4. As PPO is an online policy method, and since it uses an objective function based on current policy, it can not reuse samples. Therefore, the policy can be updated to forget about previous experience. In experiment 4-6 curriculum learning was used by placing a dormant predator with random movement, yet lethal when close, that starts to gradually become more lethal with time. This facilitated the models to explore and the animats to learn that there is food in their environment, providing an energy source. Without curriculum learning in E4-E6, when placing a fully functional predator in the environment, all models collapsed into local exploration. It can be seen that the baseline model do learn to explore for food but once the predator becomes sufficiently dangerous it completely collapses into the local optimum of idling and forgets previous experiments.

Using a large beta hyperparameter for PPO in ecosystem simulations, giving a larger entropy regularization and therefore allowing sufficient exploration, can be a sensible choice. This however comes at the cost of PPO learning more slowly. Note that this restricts the policy by not allowing it to fit to its objective, therefore it might lead to exploration but the experience might not be accurately saved to the policy and in addition it might be forgotten at later updates given a sufficiently complex environment. By introducing richer reward functions that are not as sparse as traditional reward functions such as the baseline are in this project, less exploration is needed by the model as the cost of possibly larger bias.

7

Conclusion

7.1 Summary

Previous work by Keramati et al.[5] did not produce satisfying results, except in environments E4 and E6, where the animats managed to survive and break out of the closed loop zero cumulative reward. This suggests that the model by Keramati et al. is not suitable to model animats that do not survive. Additionally, as the model by Keramati et al. lacks the ability to completely ignore non-vital needs it is more prone for greedy optimization than the framework introduced in this thesis.

The proposed f_1 and f_2 happiness functions' flexibility allows for modeling multiple, possibly contrasting, utility signals from each need. In Experiment 1 results show that introducing more information in the reward signal through a satiation term can quicken the learning and improve the performance, see Figure 5.1. A more flexible, but at the same time intuitive, reward framework can thus facilitate modeling animats' motivation through need-specific utility terms.

Our results show that at least some of the proposed happiness functions were successful in multiple environments, with the reward function f_2 being successful in every tested environment. f_2's superiority is thus reinforced by the wide gamma of environments and mechanics tested. f_2 solves both issues with the model introduced by Keramati et al. as it allows conditionally ignoring non-vital needs and using additional penalty terms it can break the closed loop zero cumulative reward problem.

Thus, we can conclude that homeostatic regulation is a feasible generic model of motivation in artificial animats based on reinforcement learning, and we assess reward function f_2 as the best performing between the functions tested.

Happiness function f_2 is characterized by an efficient inhibition of irrelevant needs, when compared to the other tested functions, since each critical homeostatic variables can lower the happiness value to zero, whereas sensory variables cannot, in our framework. This feature thus seems to be determinant for the model's adaptability to the environment within an homeostatic-sensory regulation setting.

Moreover, since environments E4, E5 and E6 were specifically aiming at replicating specific copepods' behaviours in simulated marine environments, having successfully observed behaviours B4, B5 and B6 respectively in those environments, we can conclude that a model of motivation based on homeostatic regulation, in particular of the form utilized by f_2, is adequate enough for replicating said copepod behaviours.

Finally, we observed that exploration plays an important role for animats'

learning to adapt to a simulated ecosystem, especially when said ecosystem is complex, such as in E4-E6 and EF. When utilizing the PPO algorithm as we do, one way to increase exploration is by increasing the beta parameter, i.e. the entropy regularization parameter. Nonetheless, this causes PPO to decrease learning speed, which is one of its distinctive positive traits[27]. Through a richer reward function, less exploration is needed by the model as the cost of possibly larger bias.

With this we have shown that our framework solves issues with previous theoretical work and further that PPOs are prone to collapsing into local optima. As shown by the results achieved by f_2 model, it is possible, using homeostatic regulation, to produce a generic model of motivation that can elicit the six artificial animat behaviours B1-B6. In particular, the framework replicates copepods behaviours B4-B6 in the complete model tested in EF, as seen by the results in Section 5.10.

7.2 Future work

While the simplifications adopted in this project, discussed in Section 6.1 were useful in order to limit the scope and focus on particular animat behaviours within a simple environment, allowing more malleable mechanics, such as different animat speeds, a wider array of objects' sizes and 3-dimensional environments would be a useful advancement towards more complex simulated ecosystems.

Bibliography

[1] Stewart W Wilson. "Knowledge growth in an artificial animal". In: *Adaptive and Learning Systems*. Springer, 1986, pp. 255–264.

[2] S.W. Wilson. *The Animat Path to AI*. 1991.

[3] Computing Machinery. "Computing machinery and intelligence-AM Turing". In: *Mind* 59.236 (1950), p. 433.

[4] Satinder Singh et al. "Intrinsically motivated reinforcement learning: An evolutionary perspective". In: *IEEE Transactions on Autonomous Mental Development* 2.2 (2010), pp. 70–82.

[5] Mehdi Keramati and Boris S Gutkin. "A reinforcement learning theory for homeostatic regulation". In: *Advances in neural information processing systems* 24 (2011), pp. 82–90.

[6] Walter B Cannon. "Organization for physiological homeostasis". In: *Physiological reviews* 9.3 (1929), pp. 399–431.

[7] Arthur Juliani et al. "Unity: A General Platform for Intelligent Agents". In: *CoRR* abs/1809.02627 (2018). arXiv: 1809.02627. URL: http://arxiv.org/abs/1809.02627.

[8] Rebecca J Waggett and Edward J Buskey. "Escape reaction performance of myelinated and non-myelinated calanoid copepods". In: *Journal of Experimental Marine Biology and Ecology* 361.2 (2008), pp. 111–118.

[9] David M Fields and Jeanette Yen. "The escape behavior of marine copepods in response to a quantifiable fluid mechanical disturbance". In: *Journal of Plankton Research* 19.9 (1997), pp. 1289–1304.

[10] David W Pond and Geraint A Tarling. "Phase transitions of wax esters adjust buoyancy in diapausing Calanoides acutus". In: *Limnology and Oceanography* 56.4 (2011), pp. 1310–1318.

[11] N Sören Häfker et al. "Circadian clock involvement in zooplankton diel vertical migration". In: *Current Biology* 27.14 (2017), pp. 2194–2201.

[12] W Charles Kerfoot. "Adaptive value of vertical migration: comments on the predation hypothesis and some alternatives". In: *Contrib. Mar. Sci. Suppl.* 27 (1985), pp. 91–113.

[13] Lars-Anders Hansson. "Plasticity in pigmentation induced by conflicting threats from predation and UV radiation". In: *Ecology* 85.4 (2004), pp. 1005–1016.

[14] Alfred James Lotka. *Elements of physical biology*. Williams & Wilkins, 1925.

[15] Vito Volterra. "Variazioni e fluttuazioni del numero d'individui in specie animali conviventi". In: (1926).

[16] Yan Wei et al. "The population dynamics of bacteria in physically structured habitats and the adaptive virtue of random motility". In: *Proceedings of the*

National Academy of Sciences 108.10 (2011), pp. 4047–4052. ISSN: 0027-8424. DOI: 10.1073/pnas.1013499108. eprint: https://www.pnas.org/content/108/10/4047.full.pdf. URL: https://www.pnas.org/content/108/10/4047.

[17] Villy Christensen and Carl J Walters. "Ecopath with Ecosim: methods, capabilities and limitations". In: *Ecological modelling* 172.2-4 (2004), pp. 109–139.

[18] Donald L DeAngelis and Volker Grimm. "Individual-based models in ecology after four decades". In: *F1000prime reports* 6 (2014).

[19] Daniel B Botkin, James F Janak, and James R Wallis. "Some ecological consequences of a computer model of forest growth". In: *The Journal of Ecology* (1972), pp. 849–872.

[20] Richard M Sibly et al. "Representing the acquisition and use of energy by individuals in agent-based models of animal populations". In: *Methods in Ecology and Evolution* 4.2 (2013), pp. 151–161.

[21] Thorsten Wiegand et al. "Assessing the risk of extinction for the brown bear (Ursus arctos) in the Cordillera Cantabrica, Spain". In: *Ecological monographs* 68.4 (1998), pp. 539–570.

[22] Alfred J Lotka. "Analytical note on certain rhythmic relations in organic systems". In: *Proceedings of the National Academy of Sciences* 6.7 (1920), pp. 410–415.

[23] TM Mitchell. "The need for biases in learning generalizations (Rutgers Computer Science Tech. Rept. CBM-TR-117)". In: *Rutgers University* (1980).

[24] Gerard J Tortora and Bryan H Derrickson. *Principles of anatomy and physiology*. John Wiley & Sons, 2018.

[25] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. Second. The MIT Press, 2018. URL: http://incompleteideas.net/book/the-book-2nd.html.

[26] Volodymyr Mnih et al. "Asynchronous Methods for Deep Reinforcement Learning". In: (2016). eprint: arXiv:1602.01783.

[27] John Schulman et al. *Proximal Policy Optimization Algorithms*. 2017. eprint: arXiv:1707.06347.

[28] John Schulman et al. *High-Dimensional Continuous Control Using Generalized Advantage Estimation*. 2015. eprint: arXiv:1506.02438.

[29] T. Horton et al. *World Register of Marine Species (WoRMS)*. =http://www.marinespecies.org. Accessed: 2021-04-26. Apr. 26, 2021. URL: http://www.marinespecies.org.

[30] Petra H. Lenz et al. "Sensory specialization along the first antenna of a calanoid copepod, pleuromamma xiphias (crustacea)". In: *Marine and Freshwater Behaviour and Physiology* 27.2-3 (1996), pp. 213–221. DOI: 10.1080/10236249609378966. eprint: https://doi.org/10.1080/10236249609378966. URL: https://doi.org/10.1080/10236249609378966.

[31] John Mauchline. *Adv. Mar. Biol. 33: The biology of calanoid copepods*. 1998.

[32] Karl Banse. "On the vertical distribution of zooplankton in the sea". In: *Progress in oceanography* 2 (1964), pp. 53–125.

[33] Mark D Ohman. "The demographic benefits of diel vertical migration by zooplankton". In: *Ecological Monographs* 60.3 (1990), pp. 257–281.

[34] MG Mazzocchi and G-A Paffenhöfer. "Swimming and feeding behaviour of the planktonic copepod Clausocalanus furcatus." In: *Journal of Plankton Research* 21.8 (1999).

[35] Thomas Kiørboe. "What makes pelagic copepods so successful?" In: *Journal of Plankton Research* 33.5 (2011), pp. 677–685.

[36] Brad J Gemmell, Jian Sheng, and Edward J Buskey. "Morphology of seahorse head hydrodynamically aids in capture of evasive prey". In: *Nature communications* 4.1 (2013), pp. 1–8.

[37] Cristian A Vargas and Humberto E González. "Plankton community structure and carbon cycling in a coastal upwelling system. I. Bacteria, microprotozoans and phytoplankton in the diet of copepods and appendicularians". In: *Aquatic Microbial Ecology* 34.2 (2004), pp. 151–164.

[38] Loren R Haury, Douglas E Kenyon, and James R Brooks. "Experimental evaluation of the avoidance reaction of Calanus finmarchicus". In: *Journal of Plankton Research* 2.3 (1980), pp. 187–202.

[39] Thomas Kiørboe and Andre W Visser. "Predator and prey perception in copepods due to hydromechanical signals". In: *Marine Ecology Progress Series* 179 (1999), pp. 81–95.

[40] Rexford S Ahima and Daniel A Antwi. *Brain regulation of appetite and satiety.* 2008. URL: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2710609/`.

[41] Clark Hull. *Principals of Behavior.* Vol. 33. 2. 1945, pp. 293–299.

[42] Paul Skjoldager, Peter J Pierre, and Guy Mittleman. "Reinforcer magnitude and progressive ratio responding in the rat: effects of increased effort, prefeeding, and extinction". In: *Learning and Motivation* 24.3 (1993), pp. 303–343.

[43] William Hodos. "Progressive ratio as a measure of reward strength". In: *Science* 134.3483 (1961), pp. 943–944.

[44] A Dickinson and Balleine BW. *The role of learning in motivation. Volume 3 of Steven's Handbook of Experimental Psychology: Learning, Motivation, and Emotion, ed Gallistel CR.* 2002.

[45] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning.* Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[46] Rod Nave. *Inverse Square Law, General.* URL: `http://hyperphysics.phy-astr.gsu.edu/hbase/Forces/isq.html` (visited on 04/21/2021).

[47] Claes Strannegård et al. "Combining evolution and learning in computational ecosystems". In: *Journal of Artificial General Intelligence* 11.1 (2020), pp. 1–37.

[48] Arthur Juliani et al. *Unity: A General Platform for Intelligent Agents.* 2018. eprint: `arXiv:1809.02627`.

[49] *Swedish National Infrastructure for Computing.* URL: `https://www.c3se.chalmers.se/`.

[50] Volodymyr Mnih et al. "Asynchronous methods for deep reinforcement learning". In: *International conference on machine learning.* PMLR. 2016, pp. 1928–1937.

A

Appendix 1

Table A.1: Environment E1

environment size	10
n. animats	1
episode length	1040
initial Good Food	3
Good Food spawn probability[1]	0
initial Bad Food	0
Bad Food spawn probability[1]	0
initial Vitamins Food	0
Vitamins Food spawn probability[1]	0

Table A.2: E1 - Animat

step energy decrement	0.002
movement energy decrement	0.0001

Table A.3: Environment E2

environment size	20
n. animats	1
episode length	10000
initial Good Food density	0.025
Good Food spawn probability[1]	0.00002
initial Bad Food density	0.025
Bad Food spawn probability[1]	0.00001
initial Vitamins Food density	0.025
Vitamins Food spawn probability[1]	0.00001

Table A.4: E2 - Animat

step energy decrement	0.002
movement energy decrement	0.0001
step vitamins decrement	0.001

Table A.5: Environment E3

environment size	24
n. animats	1
n. non-agent animats	2
episode length	10000
initial Good Food density[2]	0.05
Good Food spawn probability[1]	0
initial Bad Food density	0
Bad Food spawn probability[1]	0
initial Vitamins Food density[2]	0.05
Vitamins Food spawn probability[1]	0
food sources reset period	2000

Table A.6: E3 - Animat

step energy decrement	0.002
movement energy decrement	0.0001
step vitamins decrement	0.001

Table A.7: Environment E4

environment size	50
n. animats	6
n. non-agent krills	2
episode length	8000
initial Good Food density[2]	0.001
Good Food spawn probability[1]	0.00001
initial Bad Food density	0
Bad Food spawn probability[1]	0
initial Vitamins Food density	0
Vitamins Food spawn probability[1]	0

Table A.8: E4 - Animat 1: Copepod

step energy decrement	0.001
movement energy decrement	0.0001

Table A.9: E4 - Animat 2: Krill

attack force[3]	0.2
light intensity perception threshold[4]	0.5

Table A.10: Environment E5

environment size	50
n. animats	6
n. non-agent krills	2
episode length	8000
initial Good Food density[2]	0.001
Good Food spawn probability[1]	0.00001
initial Bad Food density	0
Bad Food spawn probability[1]	0
initial Vitamins Food density	0
Vitamins Food spawn probability[1]	0

Table A.11: E5 - Animat 1: Copepod

step energy decrement	0.001
movement energy decrement	0.0001
dash energy decrement	0.01

Table A.12: E5 - Animat 2: Krill

attack force[3]	0.2
light intensity perception threshold[4]	0.5

Table A.13: Environment E6

environment size	50
n. animats	6
n. non-agent krills	2
episode length	8000
initial Good Food density[2]	0.001
Good Food spawn probability[1]	0.00001
initial Bad Food density	0
Bad Food spawn probability[1]	0
initial Vitamins Food density	0
Vitamins Food spawn probability[1]	0

Table A.14: E6 - Animat 1: Copepod

step energy decrement	0.001
movement energy decrement	0.0001

Table A.15: E6 - Animat 2: Krill

attack force[3]	0.2
light intensity perception threshold[4]	0.5

Table A.16: Environment EF

environment size	50
n. animats	6
n. non-agent krills	2
episode length	8000
initial Good Food density[2]	0.005
Good Food spawn probability[1]	0.00001
initial Bad Food density	0
Bad Food spawn probability[1]	0.00001
initial Vitamins Food density[2]	0.005
Vitamins Food spawn probability[1]	0.000005

Table A.17: EF - Animat 1: Copepod

step energy decrement	0.001
movement energy decrement	0.0001
step vitamins decrement	0.0005

Table A.18: EF - Animat 2: Reflex Krill

attack force[3]	0.2
light intensity perception threshold[4]	0.5

Table A.19: EF - Animat 3: Adaptive Krill

attack force[3]	0.2
light intensity perception threshold[4]	0.5

[1]Probability of the corresponding food type spawning per each block (1 square unit area) per each time-step.

[2]Density within defined boundaries for food spawning.

[3]Maximum amount of energy decrease applied to the copepods receiving an attack.

[4]Minimum light intensity necessary for enabling the animat's proto-vision perception.