What is Situated Evolution?

M.C. Schut E. Haasdijk A.E. Eiben

***Abstract*—In this paper we discuss the notion of situated evolution. Our treatment includes positioning situated evolution on the map of evolutionary processes in terms of time- and space-embeddedness, and the identification of decentralization as an orthogonal property. We proceed with a selected overview of related literature in the categories of our interest. This overview enables us to distill further detailes that distinguish the encountered methods. As it turns out the essential differences can be captured through the mechanics of selection and fertilization. These insights are aggregated into a new model called the *Situated Evolution Method*, which is then used to provide a fine-grained map of exisiting work.**

I. Background and objectives

The background of this paper is a research project[1] concerned with a group of robots that operate in a challenging environment and permanently adapt their controllers in order to increase their task performance. Evolution is chosen as the principal method of adaptation, hence evolutionary computing (EC) is expected to supply the technical machinery to enable successful adaptation on-the-fly. This choice draws our attention to evolutionary algorithms, expecting much existing work that can be used to drive the evolutionary mechanics in a group of evolving robots.

Looking around in EC soon reveals that there is a large variety of evolutionary algorithms, such as Evolutionary Programming, Evolution Strategies, Genetic Algorithms, Genetic Programming, Differential Evolution, Particle Swarm Optimization, and more. These 'dialects' show a great diversity in algorithmic details, but still fit under a unified model of evolutionary algorithms and are mainly used to solve optimization and machine learning problems [7], [4]. In general, evolutionary algorithms are typically applied as problem solvers pursuing an optimal solution in a search space defined by the problem at hand.

However, the collective robotics application we envision, has a very different look-and-feel than a genetic algorithm solving the travelling salesman problem. Intuitively, it is a different kind of evolutionary system, where the essence is not optimization in an abstract search space, but a population of reproducing agents that undergoes selection in a (physical or virtual) environment. For the time being, we will use the term *situated evolution* for such systems. Such systems have been studied within robotics, artificial life, adaptive multi-agent research etc., and by the different perspective and the corresponding terminology it is not clear how do they relate to mainstream evolutionary computing.

The inspiration for the present paper lies in the experience that on the one hand, systems based on situated evolution are clearly evolutionary, but on the other hand are very different from mainstream evolutionary algorithms. This forms an intellectual challenge, as we would like to understand the nature of these differences. Additionally, understanding the (dis)similarities has a concrete technical relevance. The more than 30 years of EC research has accumulated much knowledge on evolutionary algorithms (EAs). This knowledge could be used in applications where the evolutionary mechanism has the same essential properties. However, at the moment our application cannot be positioned with respect to mainstream EAs, because there is no conceptual framework and a corresponding vocabulary to express the similarities and differences. All in all, our main question could be formulated as follows:

> What is situated evolution and what distinguishes it from regular evolutionary algorithms?

Thus, this paper should give answer to the following particular questions: what is the differences between a genetic algorithm used for the Traveling Salesman Problem and evolution in Sugarscape?; what is the difference between evolution in Sugarscape and in evolutionary robotics?; and, what is the difference between evolution in evolutionary robotics and in embodied evolution [25], [28]? We return to these questions in the conclusion of this paper.

The main objectives of this paper are the following. First we want to identify the most important features along which traditional EAs and systems based on situated evolution can be distinguished. This will allow for a categorization of various evolutionary mechanisms. In other words, we are looking for a small feature set supporting a coarse grained map that allows us to define situated evolution more formally. Second, we present an overview of related work in those categories that are most relevant for our intended application. Third, based on our literature review, we want to distill descriptive features that allow a more fine grained map of existing mechanisms *within* the categories of our interest.

II. Distinguishing features for a coarse grained map

To illuminate our intuition, let us consider evolutionary robotics, where it is very common to use an evolutionary algorithm to develop robot controllers that are ported to the given robots and subsequently remain fixed [18]. In this appropach, the (evolutionary) development and the usage of robot controllers is separated. We, however, envisage evolutionary algorithms that not only allow us to program robot controllers off-line, but also provide continuous on-line adaptation. That is, our individuals, the robot controllers, are to undergo the development process *in vivo*, in the physical

{mc.schut,e.haasdijk,ae.eiben}@few.vu.nl, VU University Amsterdam.

[1]SYMBRION, EU FP7, grant no. 216342

978-1-4244-2959-2/09/$25.00 © 2009 IEEE

reality. Based on this, we identify the following two main features for categorization.

- The system, in particular, the individuals of the population, is embedded in space.
- The system, in particular, the individuals of the population, is embedded in time.

The role of space is quite straightforward and in fact twofold. First, in our robotic application individuals are real entities in physical space with a notion of 'location', 'distance' etc. This is obviously not the case for the huge majority of EAs where there is no distance between individuals other than in terms of fitness or genotypic variety. Second, the spatial/locational aspect allows for distinguishing 'containers' and 'contents'. For instance, in our robotic application the body of the robot (=container) is clearly different from its controller (=contents) and we can change the controller without changing the robot. Interestingly, this phenomenon is also known in conventional EAs. In particular, in a cellular GA, the cell or grid point where a bit-string individual is located can be seen as a container whose contents is the bit-string. Variation operators (crossover and mutation) can change the contents (the bit-string), but they do not change the container (location).

Secondly, there is a concept of time; time passes for our robots that make up the population between birth, mating and death. During this time they measure their fitness against the environment by acting in that environment. In most traditional evolutionary algorithms, individuals are not acting, but passively undergo fitness evaluation, selection and variation. In other words, these operations are instantaneous — at least from the point of view of the individuals of the population. A subset of EAs where individuals could be seen as active is memetic algorithms, where the EA is enriched with local search. This local search takes place between the birth and the death of the given individual and it does change its fitness. This holds both for Lamarckian and Baldwinian systems.

Figure 1 gives an overview of four categories, obtained by the combination of the features 'space embedded' and 'time embedded'.

Given this taxonomy, we can now specify more accurately what we mean by situated evolution: an evolutionary system that is both space embedded and time embedded. Note, that both space and time can be virtual here. Hence, our notion includes systems based on computer simulations, not only physical agents (robots) in the real world. This definition allows us to identify those areas within EC that are most interesting for our intended application. For instance, spatially structured EAs [22] are highly relevant, while evolution strategies in general are not.

In addition to the two main features discussed above, there are further properties that help to capture the essence of our envisioned system. The most prominent property is decentralization, motivated by reasons of scalability. In fact, decentralization means two things:

- The lack of a central authority that decides which agents reproduce and which agents are replaced.

	Not time embedded	Time embedded	
Not space embedded	**TYPE I** E.g., genetic algorithm, evolution strategy	**TYPE II** E.g., genetic algorithm + HillClimber	No container-contents distinction
Space embedded	**TYPE III** E.g., cellular genetic algorithm	**TYPE IV** E.g., evolving agent society, robots w/ on-line evolution	Containers and contents distinguished
	Passive individuals	Active individuals	

Fig. 1. Types and examples of evolutionary processes defined by the features 'space embedded' and 'time embedded'. NB. The vertical labels on the left-hand side (Space embedded vs. Not space embedded) define the same division as those on the right-hand side (Containers and contents distinguishable vs. No containers and contents distinction). The same holds for the horizontal labels: Not time embedded vs. Time embedded is the same as Passive vs. Active.

- The lack of an omniscient oracle knowing the fitness values of all individuals.

In our system we assume both of these aspects of decentralisation to be true. Consequently, the agents gauge their own (and each other's) fitness [2] in some way and it is the agents themselves who autonomously decide (based on their fitness information) when to mate and with whom. As already noted in [6]:

> [T]he key element here is the locally executable selection. Crossover and mutation never involve many individuals, but selection in EAs usually requires a comparison among all individuals in the population.

In other words, the operators that are specific for decentralized evolution (as opposed to a traditional, centrally controlled EA) are the selection operators: mate selection instead of parent selection and an individual survivor selection instead of a global replacement strategy. The variation operators need not be specific, they should only match the given representation, i.e., the structure of the controllers.

Finally, we are now capable of specifying the type of evolutionary system needed for a collective robotics application where controllers are evoled on-the-fly: it should be based on decentralized situated evolution.

III. Related work

In addition to the general topics and algorithms mentioned in the Section above (e.g., genetic algorithms, HillClimber), we found a number of papers related to situated evolution. We overview these papers here. It is explicitly not our aim to give a complete review of all related literature, but rather

[2]For the biologically inclined: we use the term 'fitness' in the sense of 'utility', i.e., how well the agent performs some task.

to give some examples of space- and time-embedded evolutionary processes in order to concretise the ideas presented in this paper better. Because of our problem domain (robotics), we have specifically looked for papers within this domain. Later in this paper, we make explicit how our ideas relate to each of the discussed papers.

A. Embodied evolution

In evolutionary robotics, there are surpisingly few forays into fully autonomous, localised evolution schemes. Most follow up –if sometimes only notionally– on Embodied Evolution, which was introduced by Watson, Ficici and Pollack [11], [28]. This introduces a truly autonomous scheme where agents broadcast (mutated) genes at a rate proportional to their fitness (Probabilistic Gene Transfer Algorithm, PGTA). This does mean that the agents can relate their fitness to an absolute maximum, so that they do not require comparison with the rest of the population.

Wischmann, Stamm and Wörgötter investigate the interplay between embodied evolution and individual learning by introducing a *maturation period* during which no mating/replacement can take place. This allows the agents to adapt using individual learning before feeling any selective pressure [30]. Although the reported results are inconclusive, the maturation age was shown to make a difference.

A common variation of embodied evolution uses time-sharing (running alternative controllers in rotation) to overcome issues of small population size when few robots are available. One such example is that of Elfwing et al. [9], [10]. There, the robots have to harvest batteries to update their energy level. If energy drops to 0, a robot dies and replaces its controller with a form of tournament selection from the fertilised eggs it carries. Utility is measured as the number of batteries captured.

Usui and Arita [26] also implement a time-sharing variant. In their model, each robot implements a complete evolutionary algorithm with the individuals evaluated in separate time slices. The actual evolutionary algorithm runs in a traditional manner within the robot itself. Locally created new individuals and individuals received from other robots – very similar to migration in island-based parallel evolutionary algorithms– are queued for evaluation, after which they are placed into the pool of the local algorithm, replacing the worst individual (unless they themselves turn out to be worse).

[19] describes another island-based approach where the robots run genetic programming locally and exchange individuals asynchronously "based on Microbial GA." It is unclear how the frequency of sending genetic material is decided; this seems to be once every time-step regardless of utility, but with the utility then attached to the broadcast material. Received material is incorporated into the worst local individual (if that is worse than the received fitness).

Simões and Dimond implement a fully panmictic evolutionary algorithm where the robots use radio to transmit their fitnesses; the best individuals survive and mate to create new controllers with which to reprogram the population. This implementation has all population members fully connected (by radio), which does imply issues with scaling to large populations [24].

Mahdavi *et al.* [13] present an evolutionary algorithm running on-board on a solitary robot. They call their algorithm 'embedded' to indicate that fitness is based not on simulated, but on actual achievement.

Another form of embodied evolution tends towards evolution strategies [3], either by having a very small number of robots (cf. [17]), or running an evolution strategy fully on-board with a time-sharing scheme as described above [27]. The latter provides a categorisation of similar work in evolutionary robotics. Walker *et al.* employ a rather traditional evolutionary algorithm with central control (the "training phase") to develop controllers in simulation. After transferring the result onto real robots, a particular flavour of single-parent (1/1+2) evolutionary strategy further refines the controller and adapts it to a changing environment. The evolutionary strategy seems to run on a single robot. They stress the need to evaluate performance in dynamic environments.

The approach of Nehmzow [17] is similar: the population consists of 2 robots that regularly switch to mating mode, seek each other out and exchange fitness and genome. The weaker of the two applies crossover, the stronger either mutates or takes a single bit from a cached 'best-so-far' genome. Note that this is a synchronised (albeit locally) scheme; the robots stop 'regular' behaviour after a fixed amount of time to reproduce.

B. Spatially structured EAs

Another research area that has to deal with the situatedness of the individuals that make up a population is that of *spatially structured* (or: cellular) evolutionary algorithms. Here, the elements that make up the population are statically located in a (virtual) grid, possibly with the nodes of this grid located on different CPUs. To minimise communication overhead, selection schemes that limit communication to relatively small neighbourhoods in the grid are preferable. Situateded evolution in robotics, with chance encounters providing the sampling mechanism, is equivalent to cellular evolutionary algorithms with continuous random rewiring of the connections.

[12] investigates spatially structured evolution strategies; experiments show that the spread of knowledge through the network is faster when connectivity extends in multiple directions (torus over ring, for instance) and is fastest for a panmictic approach. This is deemed due to increasing selection pressure and "is coupled with a decrease of the loss of variability of the gene pool".

De Jong and Sarma analyse three different mechanisms for local selection in a spatially structured evolutionary algorithm [5]. They find that local ranking as well as local binary tournament selection (providing constant selection pressure independent of actual fitness value) outperform local proportional selection, which is quite sensitive to the actual fitness value. In all cases, increasing the neighbourhood

size from 5 to 9 increases performance, but the gain for a further increase from 9 to 13 is negligible. They stress the importance of analysing the variance of selection schemes (typically by counting offspring for generation 0 from 100 samples). Their bottom line is: combine local tournament selection on small neighbourhoods with an elitist survivor strategy. Analysing selective pressure, they conclude that "higher variance is generally more strongly correlated with poor search performance when small population sizes are involved", so either decrease selection variance or increase population size if this is an issue.

Eklund tests various cellular evolutionary algorithm variations, comparing topologies and neighbourhood shapes [8]. He investigates various selection schemes: binary tournament, roulette, ranking, fitness uniform, best and random selection. The research confirms De Jong and Sarma's finding that local elitism is required for good performance[3]. Fitness uniform selection seems to work nicely, but note that this performs poorly visavis average fitness ([14]). They also find that larger neighbourhoods lead to faster convergence.

C. Miscellaneous

This Section contains more work that relates to our idea of situated evolution, but have no common ground as embodied or spatially structured EAs. We discuss work on island-based models, gossiping algorithms, social learning and cultural algorithms.

Alba and Tomassini [1] provide an overview of previous work from the perspective of parallel evolutionary algorithms; they differentiate between panmixia, island-based and structured models (and various hybrids). They list the following benefits (apart from the parallelism) of island-based and cellular approaches: "better sampling of the search space and improve the numerical as well as runtime behaviour of the basic algorithm in many cases". For the island model, high diversity and species formation are well-reported features.

Wickramasinghe et al. show that the gossiping algorithm can be employed to compare an agent's fitness to the population average without central calculation. In this evolutionary algorithm, the population size is dynamic: agents can die (if their fitness is too low), so the population can shrink. If two agents mate, a new individual is inserted whithout replacement, so the population can grow. Genetic material spreads through the population by gossiping [29].

In [25], Smith and Bonacina describe a social learning algorithm where agents volunteer as partners by sending 'plumage' containing 'eggs' at fixed intervals (it is unclear how recipients are selected). When 5 have been received, the best is chosen (which boils down to tournament selection with tournament size 5). They note that the effect of certain choices (e.g. elitism –only accepting eggs that have higher fitness than oneself– converges slower in terms of wall-clock time) is different in embodied evolutionary algorithm than for traditional evolutionary algorithm due to things such as communication overhead.

Reynolds [20], [21] describes Cultural Algorithms, where members of the population are connected to an explicit knowledge repository in which commonly available useful knowledge bits are stored and shared. He describes a rather sketchy, but apparently centralised *acceptance* and *influence* functions that actually implement selection based on global criteria. Kendall and Su [15] describe a similar approach for their imperfect evolutionary system. Such a central knowledge repository does limit the scalability (all agents in the population must be connected to the same repository) and reduces system robustness (by introducing a single point of failure; if the repository goes down, the whole adaptive system fails).

Alonso *et al.* provide an overview of multi-agent learning approaches that describes mechanisms for imitation-based social learning [2]. These mechanisms are classified into 5 types: contagious behaviour (act like others almost mechanistically), stimulus or local enhancement (follow a teacher or parent and learn from their example), observational learning (observe others and learn from their perceived benefit), matched-dependent behaviour (learning –individually– to match others' behaviour), cross-modal matching (exactly copying a conspecific's movement, sounds, e.g., bird-song) by mapping their action onto own action.

Schwarzer [23] describes experiments where robots impart their semen/eggs with a portion of their (virtual) energy. The amount of energy imparted determines which candidate wins (either to fertilise an egg or to lay an egg in a 'dead' individual). Individuals die if their energy runs out and another fertilised agent may then implant its egg, with attached dowry. The dead individual remains open for implantation for a certain period, after which the egg with the highest dowry is used to reprogram the controller. The fraction of energy invested is determined genetically. Note, that the energy level in these experiments bears no relation to either the battery or any task-related utility.

IV. Distinguishing features for a fine grained map

Based on the above overview of related work, we can distill features that distinguish the methods encountered in the literature. These feature enable a fine grained map of related work, i.e., they provide a system in which we can describe and position the essential mechanisms of the evolutionary engine in our robot application. We do not explicitly present such a map literally, but rather a *situated evolution* method that encapsulates such fine grained features.

We first present this method, and then relate the discussed work to it. After that, we take another generalisation step of the method, resulting in a very fundamental model, which even covers traditional EAs.

[3]This may be problematic in the context of robotics, as one cannot determine the fitness of a solution a priori —it can only be determined over time. Maybe this can be overcome by having two genomes during an 'elitism' phase and evaluating the proposed replacement in a time-sharing scheme before deciding which one to keep.

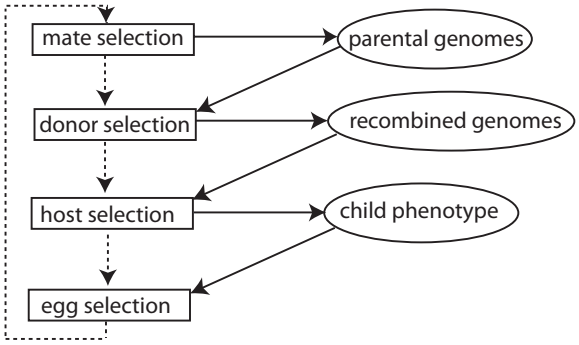

Fig. 2. The *situated-evolution* method.

A. *The* Situated-Evolution *Method*

Figure 2 shows a diagrammatic overview of the situated-evolution (SE) method, with processes on the left and datastructures on the right. Here, we explain the different processes and datastructures involved. As explained above, this method is specifically suited for space- and time-embedded evolutionary methods. Although the process may closely resemble the biological process of (human) reproduction, this is not our aim and we have made no attempt to completely comply with the biological counterpart of this process.

For illustration purposes, keep in mind the introduced problem domain (collective robotics) as a running example when we explain the method. To briefly recap, the problem at hand consists of a number of agents (here: robots) that have to perform some task and evolution is expected to deliver successful adaption on-the-fly.

Processes

- *mate-selection* – at this point, an agent chooses its *mates* with whom it potentially wants to reproduce. An agent may decide to reproduce instantly upon encountering another agent (which means that the subsequent selection processes also happen instantaneously). Alternatively, it may 'remember' the encountered agent and later, when it is ready for reproduction, deliberate if it wants to reproduce with this particular agent. If the latter is the case, then the agent stores information about the other agent (typically, its genome) in PARENTAL-GENOMES (i.e., the set of potential candidates). The to-reproduce-or-not decision may also take place at the moment of the encounter, which means that the met agent is not remembered (i.e., not stored in PARENTAL-GENOMES) after the encounter. A very important part of this stage is the ability of agents to *evaluate* other agents. Since there is no 'oracle' that knows who is good and who is not, the agent has to find this out itself. In one way or another, such "distributed selection" happens at the subsequent stages as well, and we discuss it in more detail below.
- *donor-selection* – after a set of potential mates is formed, an agent has to decide with which of its mates it wants to reproduce (i.e., generate an *egg*), which happens at this point. Based on the collected genomes (in PARENTAL-GENOMES), the agent performs recombination (typically with its own genome) and generates RECOMBINED-GENOMES. The latter can thus actually be considered to be fertilised *eggs*.
- *host-selection* – now that the agent possesses (one or more) fertilised eggs, a suitable host must be found. Trivially, this can be the agent itself – i.e., the container (agent) stays the same, but its content changes. But in other situations, the agent may 'donate' an egg(s) to other agents. In some collective robotics scenarios, agent controllers may 'die' as a result of insufficient performance, but their containers are still present (possibly indicating that they 'died'). If an agent with fertilised eggs observes such an agent, it can donate an egg to this other agent.
- *egg-selection* – finally, an agent that is presented an egg (either the agent itself or another agent, as discussed in the previous point), may or may not directly accept the egg. This last stage is similar to donor-selection, but rather than selecting a mate to procreate with, the agent accepts (or not) a complete genome to reprogram itself.

Datastructures

- PARENTAL-GENOMES – this is the storage location of the (original) genomes of the possible reproduction partners.
- RECOMBINED-GENOMES – this is the storage location of *eggs*: the recombinations of the parental genomes that were chosen as suitable parents. Typically, parental genomes are recombined with the genome of an agent itself, but not necessarily.
- CHILD-PHENOTYPE – this is the storage location of *fertilised eggs*, i.e., those eggs that made the move from genotype to phenotype and are ready to be 'implanted' on a host.

The SE algorithm

We briefly describe some characteristics of the overall algorithm here.

Firstly, one way to look at our SE method is as a traditional EA, but where a number of processes have been made explicit in order to cope with necessary time (passive/active) and space (container/content) requirements, as well as the decentralisation of the evolutionary process. Actually, the four processes are in this way *exhaustive*: we cannot think of any other processes relevant for situated evolution. Secondly, because of the decentralised nature of the evolutionary process, each of the four sub-processes may be carried out by different *actors*. As mentioned above, *egg-selection* is typically carried out by another actor than the other three selection processes. If we weaken the requirement of decentralisation a somewhat, then we can also push this actor-based model a bit further: each of the (four) sub-processes may not necessarily be carried out by each of the agents. Instead, it may be that *mate-selection* is performed by a single agent (or

some specific set of agents), whom may then function as an 'oracle' for the other agents. Exaggerating this and hypothetically speaking, it may be that all processes take place within one single agent. At that point of the spectrum, note that we actually have a (more or less) traditional EA again. Finally, we already mentioned in the *mate-selection* step that in each of the steps there is the issue of *decentralised selection*. In a traditional EA, the algorithm has knowledge about all the individuals and can apply centralised algorithms in order to select the best individuals, the elite or remove the worst individuals. Because our algorithm is distributed, none of the agents contains this 'global' knowledge. Instead, agents are faced with the problem of evaluating themselves and other agents. This issue opens up a new field where the concept of 'fitness' or 'utility' is to be reconsidered in this context.

B. Literature

Returning to the examples from the literature as discussed above, we can point out what parts of the SE method received attention in the different pieces of research.

Table I overviews the different selection processes (mate, donor, host and egg) that make up situated evolution, within the references from Section III. We can do a number of (remarkable) observations going through this Table. Firstly, there are relatively many 'cells' where a particular sub-process is implicit. While it is obvious that our newly introduced terms have not literally been used before, at some points it is surprising that works seem to not mention the (sub-)process itself. It is our hope that we make people aware of the conceptualisation of the introduced selection methods and that this leads to further and more focused development of these methods. Notwithstanding, we expect that it is not always necessary to (re)invent the wheel, as mentioned next. Secondly, we see that many 'traditional' techniques (tournament, elitism, roulette wheel, parent selection) are used for the different sub-processes. This is, first of all, not surprising, and, secondly, good news – it means that for implementation of our (conceptual) maps and methods, we may largely rely on existing techniques and methods. We consider furthering this observation in more detail as an important next step in following up this research. Thirdly, on the basis of our small survey we may conclude that *egg-selection* is most 'ignored' process in the sense that is not mentioned as a separate selection method. But having said that, one may also say that evolutionary strategies are actually *only* about *egg-selection*. A simplistic view on evolutionary strategies is that they consist of individual agents (or: solutions) applying mutations to themselves in an asexual way. In our terminology, this would be an agent that continuously or iteratively produces eggs (by applying mutation to its own genome) and subsequently performs *egg-selection* to see whether he wants to replace himself with this egg.

C. Sample and Select

Reconsidering the model introduced above, we can actually make one further step that makes this model still more general. The 4 sub-processes of the model can be considered as *sample-select-sample-select*. Here, *sample* refers to obtaining information about a set by examining one or more elements of this set; and *select* refers to picking an element from this set – typically based on the information about the set acquired by means of sampling.

In other words, a more general model would only contain these 2 steps and consist of a n number of iterations performing these two steps – in our case, 2 iterations. We briefly elaborate a bit further on this model, coined SASE^n, where SA refers to the sample step, SE to the selection step and n is the number of iterations.

Firstly, observe that with making different *actors* perform these steps, we can define a very wide range of different 'flavours' of evolutionary algorithms. On one extreme, all steps can be performed by one and the same actor, making it a traditional evolutionary method. In that case, everything is centralised and all information is contained within one individual. On the other hand, in a fully distributed system (like our scenario), different actors may perform sampling and selection. In addition, in a 'collectives' scenario (like collective robotics), one may explicitly consider letting specific individuals perform particular sub-processes (so-called sample- and select-agents) and communicate the acquired information. Secondly, in terms of more general types of actors, we can distinguish *senders* and *receivers* (for example, of the beforementioned *eggs*). If we consider the biological counterpart (i.e., the human reproduction process) of our work, then males are senders, and females are receivers. In a more fluid understanding of the model, these actor roles can be considerd 'modi' in which agents can be – at one point they are senders, at another time they are receivers. In a traditional EA, the receiver is by definition 'empty' (a new individual) and the senders are the selected parents. Finally, as mentioned, we can represent 'traditional' techniques in this model. We illustrate this by looking at *roulette wheel selection* that can be used for parent- and survivor-selection in a traditional EA: individuals are placed on a roulette-wheel proportional to their fitness and rotating the wheel results in a set of selected individuals (either for being parents or for populating a next generation). Dissecting this method, we see that one subsequently has to 1) calculate fitnesses, 2) allocate selection probabilities, and 3) perform selection operators given the selection probabilities. The first steps comprise our idea of *sampling* whereas the last step is the actual *selection*. In many works, the sampling step is left implicit or ignored and only the selection receives attention.

V. Conclusions

In this paper we elaborated on the notion of situated evolution. The inspiration came from our interest in evolutionary applications that involve a population of reproducing agents undergoing selection in a (physical or virtual) environment, and the observation that such an evolutionary process has a very different look-and-feel than most evolutionary algorithms used for optimization. Using the name 'situated

evolution' for this kind of applications we could formulate our main question as follows:

> What is situated evolution and what distinguishes it from regular evolutionary algorithms?

Our level one answer is based on three fundamental notions: time-embeddedness, space-embeddedness, and decentralization. We propose to use the first two to draw a map of four types of evolutionary processes and define situated evolution as time-embedded and space-embedded. We do not include decentralization in the definition. Instead, we see it as an orthogonal property, hence allowing both centralized and decentralized situated evolution.

Our level two answer is based on a selected overview of related literature that enabled us to distill further detailes that distinguish the encountered methods. As it turns out the essential differences can be captured through the mechanics of selection and fertilization. These insights are aggregated into a new model called the *Situated Evolution Method*, which is then used to provide a fine-grained map of exisiting work.

Also, we can now answer the example questions posed in the introduction very precisely in terms of Figure 1: a GA for the TSP is of type I; evolution in Sugarscape, evolution in evolutionary robotics and embodied evolution are of type IV - but differ in terms of the fine-grained map features.

Acknowledgements

This work was made partly possible by the European Union 7th Framework Programme, FET Proactive Intiative: Pervasive Adaptation funding the Symbrion project under grant agreement 216342. The authors would like to thank the partners in the Symbrion project for their constructive, insightful and inspiring discussions.

REFERENCES

[1] Enrique Alba and Marco Tomassini. Parallelism and evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 6(5):443 –462, October 2002.

[2] Eduardo Alonso, Mark d'Inverno, Daniel Kudenko, Michael Luck, and Jason Noble. Learning in multi-agent systems. *The Knowledge Engineering Review*, 16(3):277–284, 2001.

[3] Hans-Georg Beyer and Hans-Paul Schwefel. Evolution strategies – A comprehensive introduction. *Natural Computing*, 1:3–52, 2002.

[4] K.A. De Jong. *Evolutionary Computation: A Unified Approach*. The MIT Press, 2006.

[5] Kenneth De Jong and Jayshree Sarma. On decentralizing selection algorithms. In Larry Eshelman, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 17–23, San Francisco, CA, 1995. Morgan Kaufmann.

[6] A. E. Eiben, M. Schoenauer, J. L. J. Laredo, P. A. Castillo, A. M. Mora, and J. J. Merelo. Exploring selection mechanisms for an agent-based distributed evolutionary algorithm. In Lipson [16], pages 2801–2808.

[7] A.E. Eiben and J.E. Smith. *Introduction to Evolutionary Computation*. Natural Computing Series. Springer, 2003.

[8] Sven E. Eklund. A massively parallel architecture for distributed genetic algorithms. *Parallel Computing*, 30(5 – 6):647 – 676, May – June 2004.

[9] S. Elfwing, E. Uchibe, K. Doya, and H.I. Christensen. Biologically inspired embodied evolution of survival. In Zbigniew Michalewicz and Robert G. Reynolds, editors, *Proceedings of the 2008 IEEE Congress on Evolutionary Computation IEEE Congress on Evolutionary Computation*, volume 3, pages 2210–2216, Hong Kong, June 2008. IEEE Press.

[10] Stefan Elfwing. *Embodied Evolution of Learning Ability*. PhD thesis, KTH School of Computer Science and Communication, SE-100 44 Stockholm, Sweden, November 2007.

[11] S. Ficici, R. Watson, and J. Pollack. Embodied evolution: A response to challenges in evolutionary robotics. In J. L. Wyatt and J. Demiris, editors, *Proceedings of the Eighth European Workshop on Learning Robots*, pages 14–22, 1999.

[12] Martina Gorges–Schleuter. A comparative study of global and local selection in evolution strategies. In A.E. Eiben, Th. Th. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature – PPSN V*, volume Volume 1498/1998 of *Lecture Notes in Computer Science*, pages 367–377, Berlin / Heidelberg, 1998. Springer-Verlag.

[13] Siavash Haroun Mahdavi and Peter J. Bentley. Innately adaptive robotics through embodied evolution. *Auton. Robots*, 20(2):149–163, 2006.

[14] Marcus Hutter. Fitness uniform selection to preserve genetic diversity. *Submitted to IEEE Transactions Evolutionary Computation*, (IDSIA-01-01):13 pages, January 2001.

[15] Graham Kendall and Yan Su. Imperfect evolutionary systems. *IEEE Transactions on Evolutionary Computation*, 11(3):294 –307, 2007.

[16] Hod Lipson, editor. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2007)*. ACM, 2007.

[17] Ulrich Nehmzow. Physically embedded genetic algorithm learning in multi-robot scenarios: The pega algorithm. In C.G. Prince, Y. Demiris, Y. Marom, H. Kozima, and C. Balkenius, editors, *Proceedings of The Second International Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*, number 94 in Lund University Cognitive Studies, Edinburgh, UK, August 2002. LUCS.

[18] Stefano Nolfi and Dario Floreano. *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. The MIT Press, Cambridge, MA. / London, 2000.

[19] Anderson Luiz Fernandes Perez, Guilherme Bittencourt, and Mauro Roisenberg. Embodied evolution with a new genetic programming variation algorithm. *icas*, 0:118–123, 2008.

[20] Robert G. Reynolds. An introduction to cultural algorithms. In *Proceedings of the Third Annual Conference on Evolutionary Programming*, pages 131–139. World Scientific Press, 1994.

[21] Robert G. Reynolds. Cultural algorithms: Theory and applications. In David Corne, Marco Dorigo, and Fred Glover, editors, *New Ideas in Optimization*, pages 367–377. McGraw-Hill, London, 1999.

[22] E Sanchez and M Tomassini. *Towards Evolvable Hardware: The Evolutionary Engineering Approach*, volume 1062 of *Lecture Notes in Computer Science*. Springer, Berlin, Heidelberg, New York, 1996.

[23] Christopher Schwarzer. Investigation of evolutionary reproduction in a robot swarm. Master's thesis, Institute of Parallel and Distributed Systems, University of Stuttgart, 2008.

[24] Eduardo D. V. Simões and Keith R. Dimond. Embedding a distributed evolutionary system into population of autonomous mobile robots. In *Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference*, 2001.

[25] Robert E. Smith, Claudio Bonacina, Paul Kearney, and Walter Merlat. Embodiment of Evolutionary Computation in General Agents. *Evolutionary Computation*, 8(4):475–493, 2000.

[26] Y. Usui and T. Arita. Situated and embodied evolution in collective evolutionary robotics. In *Proceedings of the 8th International Symposium on Artificial Life and Robotics*, pages 212–215, 2003.

[27] Joanne H. Walker, Simon M. Garrett, and Myra S. Wilson. The balance between initial training and lifelong adaptation in evolving robot controllers. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 36(2):423–432, 2006.

[28] Richard A. Watson, Sevan G. Ficici, and Jordan B. Pollack. Embodied evolution: Distributing an evolutionary algorithm in a population of robots. *Robotics and Autonomous Systems*, 39(1):1–18, April 2002.

[29] W. R. M. U. K. Wickramasinghe, M. van Steen, and A. E. Eiben. Peer-to-peer evolutionary algorithms with adaptive autonomous selection. In Lipson [16], pages 1460—1467.

[30] Steffen Wischmann, Kristin Stamm, and Florentin Wörgötter. Embodied evolution and learning: The neglected timing of maturation. In *Advances in Artificial Life: 9th European Conference on Artificial Life*, LNAI, in press. Springer-Verlag, in press.

Reference	*Mate selection*	*Donor selection*	*Host selection*	*Egg selection*
Embodied evolution				
[11]	agents push own genetic material at at rate proportional to their fitness (with known maximum)	agents resist 'infection' proportionally to their fitness	implicit: an agent updates its own genetic material/controller	
[30]	equivalent to [11]			
[9]	unclear when and how it takes place (is it among the virtual agents in a robot, for instance), but it *is* a separate step	combined into a single replacement step when a virtual agent dies (presumably after a fixed amount of time)		
[26]	push model: individual to send (migrate) is selected through roulette wheel, then broadcast (presumably) with a probablility based on its relative fitness	fitness value from the sender is ignored	either the robot itself or –if seen at the level of the local GA's individuals– tournament with the worst locally available individual	
[19]	equivalent to [11]			
[24]	synchronised by internal timers; the robots emit a "mating call" over the radio, where they 'shout' their identification, fitness values, and chromosomes	best controllers mate (unclear how they are coupled) and survive, the worst robots are reprogrammed with the results; exact nature of selection is unclear		
[27]	implicit (as always for evolutionary strategies)	evolutionary strategy standard	asexual reproduction after tournament among the three on-board genomes	
[17]	implicit: there are only two robots	asymmetric: the stronger reproduces asexually, the weaker sexually (with the stronger as partner)	implicit	
Spatially structured EAs				
[12]	is called "parent selection" here: mates are the nodes in the neighbourhood	donors are either both randomly selected from the neighbourhood (local selection) or one is randomly selected and the other is the central node in a neighbourhood (centric selection). This procedure is repeated a number of times	implicit: the central node is replaced	best result is used (i.e., tournament)
[5]	as always for spatially structured EAs, is implied by the neighbourhood providing a pool of possible mates	among the mates, a donor is selected through a tournament	implicit: it is the centre of the neighbourhood	'elitism': a host is replaced only if the new egg outperforms the current one
[8]	neighbourhood	compared techniques	implied (central node)	elitism
Miscellaneous				
[29]	an agent determines if it is itself fertile (comparing its fitness to an approximation of the population average), and then its mate is chosen randomly from the neighbours in the overlay network	random from the set of mates	non-existent: a new agent is created to contain the new genome	not
[25]	agents send plumage objects	tournament when 5 plumages have been collected	implicit: agent replaces itself	not
[23]	random; attach dowry	highest 'dowry' wins	inseminate 'dead' robots that one encounters; attach dowry	highest 'dowry' wins

TABLE I
OVERVIEW OF THE DIFFERENT SELECTION METHODS (MATE, DONOR, HOST, EGG) IN THE RELATED REFERENCES FROM SECTION III.