Robotica (2017) volume 35, pp. 2330–2362. © Cambridge University Press 2017
doi:10.1017/S0263574717000017

Autonomous multirobot excavation for lunar applications

Jekanthan Thangavelautham†*, Kenneth Law‡, Terence Fu§, Nader Abu El Samid¶, Alexander D. S. Smith§ and Gabriele M. T. D'Eleuterio§

†*School of Earth and Space Exploration, Arizona State University, 781 E Terrace Mall, Tempe, AZ 85287, USA*
‡*David Schaeffer and Associates, Markham, ON, Canada. E-mail: ken.law@utoronto.ca*
§*University of Toronto, 4925 Dufferin Street, Toronto, ON M3H 5T6, Canada.*
E-mails: terence.fu@utoronto.ca, alexander.smith@utoronto.ca, gabriele.deleuterio@utoronto.ca
¶*MDA Space Missions, 9445 Airport Road, Brampton, ON L6S 4J3, Canada.*
E-mail: n.abuelsamid@utoronto.ca

(Accepted December 21, 2016. First published online: January 25, 2017)

SUMMARY
In this paper, a control approach called Artificial Neural Tissue (ANT) is applied to multirobot excavation for lunar base preparation tasks including clearing landing pads and burying of habitat modules. We show for the first time, a team of autonomous robots excavating a terrain to match a given three-dimensional (3D) blueprint. Constructing mounds around landing pads will provide physical shielding from debris during launch/landing. Burying a human habitat modules under 0.5 m of lunar regolith is expected to provide both radiation shielding and maintain temperatures of −25 °C. This minimizes base life-support complexity and reduces launch mass. ANT is compelling for a lunar mission because it does not require a team of astronauts for excavation and it requires minimal supervision. The robot teams are shown to autonomously interpret blueprints, excavate and prepare sites for a lunar base. Because little pre-programmed knowledge is provided, the controllers discover creative techniques. ANT evolves techniques such as slot-dozing that would otherwise require excavation experts. This is critical in making an excavation mission feasible when it is prohibitively expensive to send astronauts. The controllers evolve elaborate negotiation behaviors to work in close quarters. These and other techniques such as concurrent evolution of the controller and team size are shown to tackle problem of antagonism, when too many robots interfere reducing the overall efficiency or worse, resulting in gridlock. Although many challenges remain with this technology, our work shows a compelling pathway for field testing this approach.

KEYWORDS: neural-networks; excavation; multirobots; human-competitive; ISRU.

1. Introduction

Social insects such as a colony of ants excavating a network of tunnels or swarms of termites building towering cathedral mounds with internal heating and cooling shafts[11] show the potential of multiagent systems in building robust structures. These social insects, without any centralized coordination, produce emergent collective behaviors used to build these robust structures. Multiple individuals working in a decentralized manner offer some inherent advantages, including fault tolerance, parallelism, reliability, scalability and simplicity in robot design.[13]

Using this bio-inspired method, teams of autonomous robots can construct key elements of a human habitat on the moon (Fig. 1). They can work continuously in harsh environments making them

* Corresponding author. E-mail: jekan@asu.edu

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

Fig. 1. Artist impression of a lunar base and mining facility (courtesy NASA).

very productive, are fault tolerant to the failure of individual robots and are scalable depending on the task complexity and schedule. Robots do not require life-support infrastructure that would otherwise be required for a team of astronaut workers. Furthermore, robots may be required for certain tasks due to concerns of health and safety of the astronauts. Combining these factors, our study shows that use of teams of autonomous robots instead of astronauts can reduce launch cost by 50%[2] for lunar base construction. This can free ground operators from constantly tending to the multirobot team. In this architecture, it will be possible for operators on the ground to have total oversight over the activities of the robot team and intervene and recover from mishaps or unexpected events.

Constructing mounds around landing pads will provide physical shielding from debris during launch/landing. Burying a human habitat modules that under 0.5 m of lunar regolith is expected to provide both radiation shielding and maintain comfortable temperatures of $-25\ ^{\circ}$C (based Apollo 17 manual excavation experiments).[21]

Earth-based teleoperation systems have been proposed for control of robots on the moon.[26] Such systems have been demonstrated successfully with the Lunakhod 1 and 2 rover missions; however, latency (time delay) induced prolonged fatigue was a concern for the Lunakhod missions. Latency-induced operator fatigue is still a concern when coordinating actions with teams of robots over extended time.[29] Advancements have been made on coordination and control of multiple robots using teleoperation.[25,27] However these techniques have yet to be tested for spacecraft or robots at the moon.

The proposed ANT architecture allows ground control oversight and frees ground operators from mundane tasks while reserving all but the most delicate and mission critical tasks for human intervention, thus, reducing the chance of fatigue and human error. These factors make an autonomous robotic system with teleoperation capability more appealing than teleoperation alone. This approach permits having a base deployed and operational in time for astronauts to arrive from Earth. Major approaches to developing autonomous control systems utilize human knowledge, machine learning techniques or a combination of both.

Human knowledge-based control strategies rely on human input in the form of ad-hoc control rules, physical model-based planners, task-specific assumptions and human knowledge.[7,49] In many tasks that use multiple robots, it is often unclear of how best to plan the task, organize the group and determine the best interaction behaviors to complete the task. In lunar and planetary environments, task-specific assumptions may not be valid *in-situ*. The surface properties and material may vary from crater to crater. One or more robots may be disabled unexpectedly or need to perform tasks that have not been envisioned during mission planning and modeling stages. These factors make an adaptive, decentralized controls approach to reorganize and control a group of robots more appealing.

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

A novel, robotic learning controls approach is presented here that addresses these challenges. This approach requires much less human knowledge than conventional approaches. The controllers are homogenous (i.e., a single controller is replicated for use on each robot), decentralized and make use of local sensor data. Hardware implementations of the system utilize a shared resource such as an overhead camera for localization or TriDAR for three-dimensional (3D) mapping and hence the system is not truly decentralized but the system can be made decentralized by utilizing multiple shared resource. The approach learns to solve tasks through a process of trial and error and is given a global objective function, a generic set of basis behaviors, sensory input and a simplified training environment without detailed physical models of the system. The proposed approach requires an accurate localization system. This is possible by mounting cameras and lighting on a tower over the work area. Other options include use of radio beacons that are located from the main landing craft or structure. These two approaches can enable rover localization without requiring a Lunar GPS system.

This approach called the Artificial Neural Tissue (ANT)[39,42] combines a standard neural network with a novel coarse-coding mechanism inspired by the work of Albus and Hinton.[3,22] In ANT, coarse coding is used to perform regulatory control of networks.[39,41,42] The process occurs through simulated chemical diffusion[17,30] and enables segmentation of a network, through activation and inhibition of neuronal groups. This allows for modules of neurons to be added, removed and rewired during training facilitating self-organized task decomposition and task-allocation.[41] This method is shown to solve the sign-following task found to be intractable with conventional fixed and variable topology neural network architectures.[41,42]

Here, the capabilities of ANT are shown for multirobot excavation,[43,44] a difficult task, with a large, high-dimensional task space. Learning to solve the excavation task enables the robots to build berms, landing pads and excavate holes for burying the lunar habitat modules. The excavation task combines features of a typical foraging, grazing or cleaning task with ability to plan, interpret blueprints and perform coordinated excavation. Since little pre-programmed knowledge is given, ANT may discover creative solutions producing controllers that can interpret excavation blueprints, can successfully avoid obstacles, perform layered digging, leveling and avoid burying or trapping other robots. These innovative behaviors are discovered during training and are similar to behaviors used by social insects to perform group coordination.

In this work expanded from Thangavelautham et al.,[44] ANT is found to evolve superior solutions to the excavation task in fewer genetic evaluations than hand-coded and conventional neural networks solutions. The ANT solutions are found to be scalable and can be applied to unforeseen real world scenarios where one or more robots may become disabled or unavailable. Hand-coded solutions are found to work in single robot scenarios and show poor performance and robustness for increased number of robots, thus, lacking adaptivity. The required cooperative behaviors for all but the simplest of tasks are unintuitive and pose difficulty for humans programmers. Conventional neural networks can do better than hand-coded solutions but require an experimenter to manually decompose a complex task, determine a suitable network topology and activation function to make training tractable. ANT requires even less experimenter input and is useful for multirobot excavation tasks, where there is limited domain knowledge available. The controllers can generalize (interpolate) from limited training scenarios to handle unforeseen situations. ANT through a process of coarse-coding can segment high-dimensional tasks space more efficiently, performing automated task decomposition and simultaneously finding the required controller network topology, selecting optimal number of robots and coordination behaviors to complete the task. Neuronal activity and behavioral analysis of the controllers suggest that solutions emerge through a process of trial and error discovery, fine tuning and incremental assembly of "building-block" behaviors to solve tasks.

Using this approach, we show the feasibility of using multirobot excavation for site preparations tasks. The approach shows improved performance and scalability than conventional neural networks and hand-coded solutions. This facilitates finding creative behaviors that are not specified or encouraged by an experimenter. These creative behaviors verified in hardware include correctly interpreting blueprints, performing layered digging, obstacle avoidance and rocking behaviors to avoid getting stuck. This approach is shown to produce controllers that have improved scalability compared to conventional neural networks and hand-coded solutions. Furthermore, ANT can simultaneously evolve the desired controller and select for optimal number of robots for the task. This approach is shown as a possible solution to the problem of antagonism in decentralized multirobot control.

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

The evolved solutions have been analyzed in simulation and the best solutions have been ported onto real robots. Hardware experiments were performed on three different robotic platforms, including in the laboratory and under controlled field conditions. These experiments were used to validate individuals behaviors seen in simulation to verifying the overall excavation performance of the controllers. Laboratory hardware experiments and controlled field experiments produced promising results that show a promising pathway toward full implementation and demonstration in the field.

This article is organized as follows. Section 2 presents related work. Section 3 presents the ANT approach. Section 4 presents the excavation task used to demonstrate ANT's capabilities. This is followed by results and discussion in Section 5 and proof-of-concept experiments in Section 6.

2. Related Work

Previous work in autonomous excavation[36] has been limited to a single robot and separate loading/unloading vehicles. Digging is performed using hand-coded scripts that simplify repetitive excavation/truck loading cycles. These controllers are used to position and unload an excavator bucket relative to a dump truck using a suite of sensors onboard the vehicles.

These scripts are developed with input from an expert human operator and model vehicle specific limitations such as load handling capacity and latency. These systems incorporate adaptive coarse and refined planners to sequence digging operations.[33] Other works used coarse and refined planner containing a neural network approach to learn soil conditions and excavator capabilities during operation.[12] Such systems are comparable in efficiency to human operators. Other approaches are devoted to modeling kinematics and/or dynamics of the excavation vehicles and simulating their performance.[16] These techniques are designed for specific vehicle platforms and do not include scripts for every possible scenario in the task space, thus, requiring close monitoring and intervention by a human operator. This makes the approach unsuitable for fully autonomous operation of multiple robots on the moon.

Control systems such as LUCIE are more sophisticated and incorporate long-term planning.[10] Apart from identifying and automating excavation cycles, the system incorporates a whole sequence of intermediate goals that need to be achieved to complete a trench digging task. However, the system lacks autonomy because the task is decomposed and prioritized by a human operator.

Other techniques closely mimic insect in thir ability to build road ways and ramps using amorphous construction. A laboratory robot is used to heat, melt and deposit foam to produce a ramp and other complex structures.[31] More recent work by Halbach *et al.*[20] have performed simulations of multiple robots to perform excavation on Mars. The intent is to set up a permanent human base and utilize Martian resources for construction and *in-situ* resource utilization. The work has focused on human-assisted high-level planning required to locate a base and key facilities and the process of resource extraction. Human assistance is utilized in planning high-level tasks and giving execution orders to multiple robots. It is presumed that human astronauts are already located on Mars and can perform teleoperation on site (from a safe distance). Recent work by Halbach et al.[34] has show bucket wheels to be the most effective excavation platform for low gravity on the moon. As will be shown later, our results also show bucket wheels to be most efficient for excavation.

The construction of a human habitat on the moon will require multiple excavation robots working toward a given goal. Collective robotics is well suited because it incorporates multiple autonomous robots that work cooperatively toward a global goal. Some collective robotic controllers mimic mechanisms used by social insects to perform group coordination. These include the use of self-organization, templates and stigmergy. Self-organization describes how macroscopic behavior emerge solely from numerous interactions among lower level components of the system that use only local information[9] and is the basis for the bio-inspired control approach presented here. Templates are environmental features perceptible to individuals within the collective.[8] In robotic applications, template-based approaches include use of light fields to direct the creation of linear[37] and circular walls[47] and planar annulus structures.[48] Spatiotemporally varying templates (e.g., adjusting or moving a light gradient over time) have been used to produce more complex structures.[38]

Stigmergy is a form of indirect communication mediated through the environment.[19] Stigmergy has been used extensively in collective robotic construction, including blind bulldozing,[32] box pushing,[28] heap formation[6] and tiling pattern formation.[40]

However, conventional collective robotics control approaches have two limitations. First, they rely on either user-defined deterministic "if–then" rules or on stochastic behaviors. It is difficult to

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

design controllers by hand with cooperation in mind, as we show later in the paper, because there exists no formalisms to predict or control the global behaviors that will result from local interactions. Designing successful controllers by hand can devolve into a process of trial and error.

Second, these approaches can suffer from an emergent feature called *antagonism*[14] when multiple agents trying to perform the same task interfere with one another, reducing the overall efficiency of the group or worse, result in gridlock. This limits scalability of the solution to number of robots and size of the task area.

Because the approach presented here is evolutionary in nature, it "learns" to exploit the mechanisms described earlier to find creative solutions to a task. As with other evolutionary algorithms, the approach is stochastic and cannot guarantee a solution in finite time. However, as will be presented later, the controllers converge to solution with a probability of 93% at the optimal training settings. The presented method is able to mitigate the effects of antagonism, which is difficult to do with conventional approaches due to the lack of domain knowledge of a task at hand.

A means of reducing the effort required in designing controllers by hand is to encode controllers as behavioral look-up tables and allow a genetic algorithm to evolve the table entries. This approach is used to solve a heap formation task in Barfoot and D'Eleuterio[5] and a 2×2 tiling formation task in Thangavelautham et al.[40]

A limitation with look-up tables is that they have poor sensor scalability, as the size of the look-up table is exponential in the number of inputs. Look-up tables also have poor generalization. Neural network controllers perform better generalization since they effectively encode a compressed representation of the table. Neural networks have been successfully applied to multirobot systems and have been used to build walls, corridors, and briar patches[15] and for tasks that require communication and coordination.[46]

Neural network controllers have been also been used to solve the 3×3 tiling task.[45] Going from the 2×2 task to the 3×3 tiling formation task results in a search space of 10^{145} to 10^{1300}, respectively. This very large increase in search space prevents a lookup table from finding a suitable solution. However because a neural network can generalize better it finds a desired solution.

In standard neural networks, communication between neurons is modeled as synaptic connection (wires) that enable electrical signaling. Other fixed-topology networks, such as Gasnet, model both electrical and chemical signaling between neurons.[23] However, when using fixed-topology networks, the size of the network must be specified ahead of time. Choosing the wrong topology may lead to a network that is difficult to train or is intractable.[24,42]

Variable length neural network methodologies such as NEAT (NeuroEvolution of Augmenting Topologies) show the potential advantage of evolving both the neuron weights and topology concurrently.[35] It is argued that growing the network incrementally through (complexification) helps to minimize the dimensional search space and thus improves evolutionary performance.[35] This requires starting with a small topology and growing it incrementally through evolutionary training which can be slow.

The ANT framework presented here is a bio-inspired approach that simultaneously addresses both the problems in designing rule-based systems by hand and the limitations inherent in previous fixed and variable topology neural networks. Unlike previous models like Gasnet,[23] chemical communication within ANT enables it to dynamically add, remove and modify modules of neurons through coarse coding. This facilitates segmentation of the search space to perform self-organized task decomposition.[41] This also provides good scalability and generalization of sensory input.[42] ANT is more flexible than NEAT. It can be initialized with a large number of neurons without the need for incremental "complexification."[41]

As will be shown later, ANT does not rely on detailed task-specific knowledge or detailed physical models of the system. It evolves controllers to optimize a user-specified global objective (fitness) function. The evolutionary selection process is able to discover for itself and exploit templates, stigmergy and mitigate the effects of antagonism.

3. Artificial Neural Tissue

ANT[39,41,42] is a neural networks approach trained using evolutionary algorithms. ANT is applied in this paper as the controller for the robot exacavator(s). It consists of a developmental program encoded into an artificial *genome* composed of a set of genes to construct a 3D ANT. Inspired by neurobiology,

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

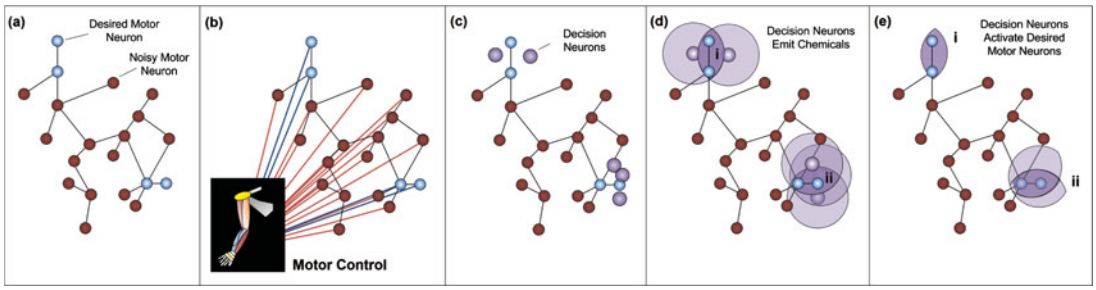

Fig. 2. In a randomly generated tissue, most motor neurons would produce spurious/incoherent output (a) that would "drown out" signals from a few desired motor neurons due to spatial crosstalk[24] (b). This can make training intractable for difficult tasks. Neurotransmitter (chemicals) emitted by decision neurons (c) selectively activate networks of desired motor neurons in shaded regions (i) and (ii) by coarse-coding overlapping diffusion fields as shown (d). This inhibits noisy motor neurons and eliminates spatial crosstalk (e).

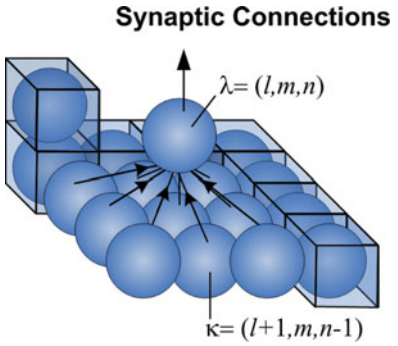

Fig. 3. Synaptic connections between ANT motor neurons from layer $l + 1$ to l.

ANT models chemical communication in addition to electrical communication along axons. Some neurons release chemicals that travel by diffusion and are read by other neurons, in essence serving as a "wireless" communication system to complement the "wired" one. This chemical communication scheme is used to dynamically activate and inhibit network of neurons. The tissue consists of two types of neural units, *decision neurons* and *motor-control neurons*, or simply motor neurons. Assume a randomly generated set of motor neurons in a tissue connected by wires (Fig. 2(a)). Chances are most of these neurons will produce incoherent/noisy output, whereas a few may produce desired functions. If the signal from all of these neurons is summed, then these "noisy" neurons would drown out the output signal (Fig. 2(b)) due to spatial crosstalk.[24]

Within ANT, decision neurons emit chemicals that diffuse omnidirectionally, shown in shaded regions in Fig. 2(d). Coarse coding is a distributed representation that uses multiple overlapping coarse fields to encode a finer field.[3,22] By coarse-coding multiple overlapping diffusion fields, the desired motor neurons can be selected and spurious motor neurons can be inhibited. With multiple overlapping diffusion fields (Fig. 2(d)) shown in shaded region (ii), there is redundancy and when one decision neuron is modified (i.e., due to a deleterious mutation) the desired motor neurons are still selected. In the following section, the computational mechanisms within the tissue are described first, followed by description of how the tissue is created.

3.1. Motor neurons

Motor neuron, $N_{\boldsymbol{\lambda}}$, occupies the position $\boldsymbol{\lambda} = (l, m, n) \in \mathbb{Z}^3$ (Fig. 3) and is arranged in a lattice structure. Depending on the activation functions used, the state $s_{\boldsymbol{\lambda}} \in \mathbb{S}$ of the neuron is either binary, i.e., $\mathbb{S}_{bin} = \{0, 1\}$ or can be real, $\mathbb{S}_p = [0, 1]$ or $\mathbb{S}_r = [-1, 1]$.

Each motor neuron $N_{\boldsymbol{\lambda}}$ receives input from neurons $N_{\boldsymbol{\kappa}}$, where $\boldsymbol{\kappa} \in \Uparrow(\boldsymbol{\lambda})$ is the nominal input set. These nominal inputs are the 3×3 neurons centered one layer below $N_{\boldsymbol{\lambda}}$; in other terms, $\Uparrow(\boldsymbol{\lambda}) = \{(i, j, k) \mid i = l - 1, l, l + 1;\ j = m - 1, m, m + 1;\ k = n - 1\}$.

The sensor data are represented by the activation (state) of the sensor input neurons $N_{\boldsymbol{\alpha}_i}$, $i = 1 \ldots m$, summarized as $A = \{s_{\boldsymbol{\alpha}_1}, s_{\boldsymbol{\alpha}_2} \ldots s_{\boldsymbol{\alpha}_m}\}$. The network output is represented by the activation (state) of

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

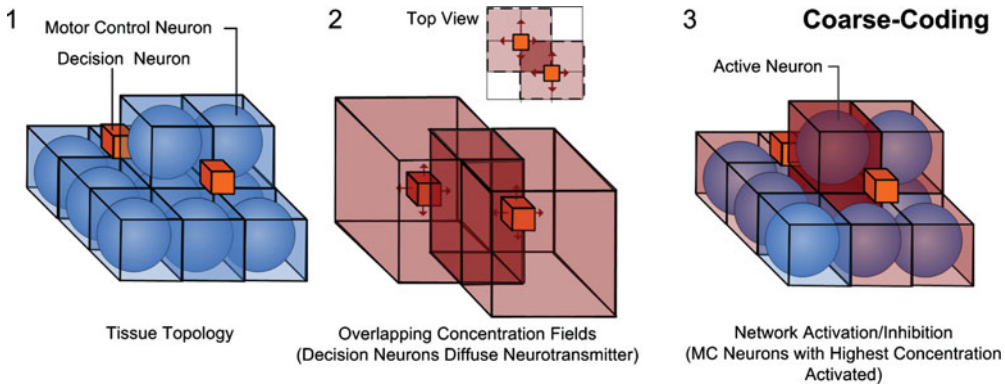

Fig. 4. Coarse coding regulation being performed by two decision neurons (shown as squares) that diffuse a chemical, in turn activating a motor neuron column located at the center (right).

the output neurons N_{ω_j}, $j = 1 \ldots n$, summarized as $\Omega = \{s_{\omega_1^1}, s_{\omega_2^1}, s_{\omega_3^2} \ldots s_{\omega_n^b}\}$, where $q = 1 \ldots b$ specifies the output behavior. Each output neuron commands one behavior of the robot. (In the case of a robot, a typical behavior may be to move forward a given distance. This may require the coordinated action of several actuators. Alternatively, the behavior may be more primitive such as augmenting the current of a given actuator.) If $s_{\omega_j^q} = 1$, output neuron $\boldsymbol{\omega}_j$ votes to activate behavior q; if $s_{\omega_j^q} = 0$, it does not. Since multiple neurons can have access to a behavior, an arbitration scheme is imposed to ensure that the controller is deterministic where $p(q) = \sum_{j=1}^{n} \gamma(s_{\omega_j^i}, q) s_{\omega_j^i} / n_q$ and $n_q = \sum_{j=1}^{n} \gamma(s_{\omega_j^i}, q)$ is the number of output neurons connected to output behavior q where $\gamma(s_{\omega_j^i}, q)$ is evaluated as follows:

$$\gamma(s_{\omega_j^i}, q) = \begin{cases} 1, & \text{if } i = q \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

and resulting in behavior q being activated if $p(q) \geq 0.5$. Once the behaviors are activated, they are executed in a *a priori* sequence.

3.2. The decision neuron

Decision neurons occupy nodes in the lattice as established by their genetic parameters (Fig. 4). These neurons excite into operation or inhibit the motor control neurons (shown as spheres) by excreting an activation chemical. Once a motor control neuron is excited into operation, the computation outlined in (3) is performed.

Each decision neuron can be in one of two states, either diffuse a neurotransmitter chemical or remain dormant. The state of a decision neuron $T_{\boldsymbol{\mu}}$, $s_{\boldsymbol{\mu}}$, is binary and determined by one of the activation functions (see Section 3.3). The inputs to $T_{\boldsymbol{\mu}}$ are all the input sensor neurons $N_{\boldsymbol{\alpha}}$; i.e., $s_{\boldsymbol{\mu}} = \psi_{\boldsymbol{\mu}}(s_{\boldsymbol{\alpha}_1} \ldots s_{\boldsymbol{\alpha}_m})$, where $\sigma_{\boldsymbol{\mu}} = \sum_{\boldsymbol{\alpha}} v_{\boldsymbol{\alpha}}^{\boldsymbol{\mu}} s_{\boldsymbol{\alpha}} / \sum_{\boldsymbol{\alpha}} s_{\boldsymbol{\alpha}}$ and $v_{\boldsymbol{\alpha}}^{\boldsymbol{\mu}}$ are the weights. The decision neuron is dormant if $s_{\boldsymbol{\mu}} = 0$ and releases a neurotransmitter chemical of uniform concentration $c_{\boldsymbol{\mu}}$ over a prescribed field of influence if $s_{\boldsymbol{\mu}} = 1$.

The decision neuron's field of influence is taken to be a rectangular box extending $\pm d_{\boldsymbol{\mu}}^r$, where $r = 1, 2, 3, \ldots$, from $\boldsymbol{\mu}$ in the three perpendicular directions. These three dimensions along with $\boldsymbol{\mu}$ and $c_{\boldsymbol{\mu}}$, the concentration level of the virtual chemical emitted by $T_{\boldsymbol{\mu}}$, are encoded in the genome.

Motor control neurons within the highest chemical concentration field are excited into operation by the decision neurons, whereas all others remain dormant. Owing to the coarse coding effect, the sums used in the weighted input of (2) are over only the set $\overline{\Uparrow}(\boldsymbol{\lambda}) \subseteq \Uparrow(\boldsymbol{\lambda})$ of active inputs to $N_{\boldsymbol{\lambda}}$. Likewise the output of ANT is $\overline{\Omega} \subseteq \Omega$.

3.3. Activation function

Each neuron, motor and decision uses a modular activation function. This allows selection among four possible threshold functions of the weighted input σ. The use of two threshold parameters allows

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

Tissue Gene

Specifier	Neuron Replication Prob.	Neuron Replication Ratios		Seed Address
D	T_r	n_d	n_m	T_s
Integer [0,2]	Real [0.001,0.1]	Integers [1,10]		Integer

Decision Neuron Gene

Specifier	Reference Address	Position			Diffusion Param.			Diffusion Concentration	Activation Function Parameters								Gene Activate
D	A	x	y	z	d_x	d_y	d_z	c	w_1	w_2	...	w_v	θ_1	θ_2	k_1	k_2	G
Integer [0,2]	Integer	Integers			Integers [1,3]			Integers [0,1]	Real [0,1]						Integers [0,1]		Binary

Motor Control Neuron Gene

Specifier	Reference Address	Position			Activation Function Parameters						Gene Activate	Cell Death	Replication Prob.	Output Behaviour	Reference Pointer
D	A	x	y	z	w_1	w_2	...	w_v	θ_1	θ_2	G	C	R	k	P
Integer [0,2]	Integer	Integers			Real [0,1]				Integers [0,1]		Binary	Binary	Real [0,1]	Integer [0,b]	Integer

Fig. 5. ANT gene map showing tissue, motor neuron and decision neuron genes.

for a single neuron to compute the XOR function, in addition to the AND and OR functions. For this activation function,

$$\begin{aligned}
\psi_{\text{down}}(\sigma) &= \begin{cases} 0, & \text{if } \sigma \geq \theta_1 \\ 1, & \text{otherwise} \end{cases} \\
\psi_{\text{up}}(\sigma) &= \begin{cases} 0, & \text{if } \sigma \leq \theta_2 \\ 1, & \text{otherwise} \end{cases} \\
\psi_{\text{ditch}}(\sigma) &= \begin{cases} 0, & \min(\theta_1, \theta_2) \leq \sigma < \max(\theta_1, \theta_2) \\ 1, & \text{otherwise} \end{cases} \\
\psi_{\text{mound}}(\sigma) &= \begin{cases} 0, & \sigma \leq \min(\theta_1, \theta_2) \text{ or } \sigma > \max(\theta_1, \theta_2) \\ 1, & \text{otherwise} \end{cases}
\end{aligned} \tag{2}$$

where θ_1, θ_2 are threshold parameters and $\theta_1, \theta_2 \in \mathbb{R}$.

The weighted input σ_λ for neuron N_λ is nominally taken as

$$\sigma_\lambda = \frac{\sum_{\kappa \in \Uparrow(\lambda)} w_\lambda^\kappa s_\kappa}{\sum_{\kappa \in \Uparrow(\lambda)} s_\kappa} \tag{3}$$

with the proviso that $\sigma = 0$ if the numerator and denominator are zero. Also, $w_\lambda^\kappa \in \mathbb{R}$ is the weight connecting N_κ to N_λ. These threshold functions are summarized as

$$\psi = (1 - k_1)[(1 - k_2)\psi_{\text{down}} + k_2\psi_{\text{up}}] + k_1[(1 - k_2)\psi_{\text{ditch}} + k_2\psi_{\text{mound}}] \tag{4}$$

where $k1, k2 \in 0, 1$. The activation function is thus encoded in the genome by k_1, k_2 the threshold parameters θ_1 and θ_2.

3.4. Evolution and development

A population of ANTs is evolved in an artificial Darwinian manner. The "genome" for a tissue contains a "gene" for each cell with a specifier D used to distinguish between motor control neuron, decision neuron and tissue. A constructor protein (an autonomous program) interprets the information encoded in the gene (Fig. 5) and translates this into a cell descriptor protein. The gene "activation" parameter is a binary flag resident in all the cell genes and is used to either express or repress the contents of the gene. When repressed, a descriptor protein of the gene content is not created. Otherwise, the constructor protein "grows" a cell. Each cell position is specified in reference to a seed-parent address. A cell-death flag determines whether the cell commits suicide after being grown. Once again, this feature in the genome helps in the evolutionary process with a cell committing suicide still occupying

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

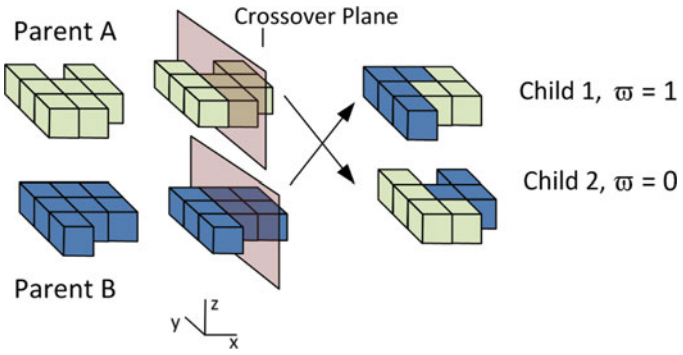

Fig. 6. A crossover operation between two ANT parents. "Compatible" neuron genes are interchanged as shown resulting in two offspring.

a volume in the lattice although it is dormant. Evolution can decide to reinstate the cell by merely toggling a bit through mutation.

In turn, mutation (manipulation of gene parameters with a uniform random distribution) to the growth program results in new cells being formed through cell division. The rate of mutation occurring on the growth program is specified for each tissue and is dependent on the cell replication probability parameter T_r. This probability parameter is used to determine whether a new cell is inserted. Cell division requires a parent cell (selected with highest replication probability relative to the rest of the cells within the tissue) and copying $m\%$ of the original cell contents to a daughter cell (where m is determined based on uniform random distribution). This models a gene duplication process with the first $m\%$ being a redundant copy of an existing gene and the remaining contents being malformed in which a daughter gene adopts some functions from its parent.

The tissue gene specifies parameters that govern the overall description of the tissue. This includes "neuron replication probability," a parameter that govern the probability additional cell genes are created through mutation and the "Neuron Replication Ratio," that determines the ratio of decision neuron genes to motor neurons genes in the tissue. The "Cell Type" of each new cell is determined based on the ratio of motor control neurons to decision neurons, a parameter specified in the tissue gene. The new neuron can be located in one of six neighboring locations (top, bottom, north, south, east, and west) chosen at random and sharing a common side with the parent and not occupied by another neuron. Furthermore, a seed address is specified, which identifies the seed cell gene which will be used to construct the first cell in the tissue.

3.5. Crossover and mutation

Within ANT, the genome is modular, with each gene defining each neuron's characteristics. Crossover is the exchange of genes between two parents to form a child. Before crossover, a parent affinity parameter, $\varpi \in \{0, 1\}$, is chosen at random for each child genome. The affinity parameter is used to establish if each child genome has closer "affinity" to one of its parents (either parent A or parent B). Thus, if $\varpi = 0$, then the genome has closer "affinity" to parent A, and $\varpi = 1$ if it has affinity with parent B. Each neuron has a unique position $\boldsymbol{\lambda} = (l, m, n)$ and a crossover is performed by drawing a plane (with a normal vector parallel to the x- or y-axes) separating the tissue. Cell genes on the parent genome located on the side of the plane closer to the origin are copied directly onto the child genome with the associated "affinity" parameter, and the remaining genes are exchanged between the parents based on a "compatibility criterion" (Fig. 6). Crossover operation does not result in arbitrary separation and exchange of gene contents.

The "compatibility criterion" imposes the following condition, that the gene for neuron $N_{\boldsymbol{\lambda}_1}$ from parent A and $N_{\boldsymbol{\lambda}_2}$ from parent B could be exchanged if $\boldsymbol{\lambda}_1 = \boldsymbol{\lambda}_2$, i.e., have the same position after development and only when *both* genes are expressed or repressed during development. Thus, child 1 with $\varpi = 0$ (affinity to parent A) assumes the gene for ${}^{B}N_{\boldsymbol{\lambda}_2}$ and child 2 with $\varpi = 1$ (affinity to parent B) assumes ${}^{A}N_{\boldsymbol{\lambda}_1}$. If the compatibility criterion is not met, then no exchange occurs, and thus ${}^{A}N_{\boldsymbol{\lambda}_1}$ is passed onto child 1 and ${}^{B}N_{\boldsymbol{\lambda}_2}$ is passed onto child 2. If ${}^{A}N_{\boldsymbol{\lambda}_1}$ is not expressed in parent A and ${}^{B}N_{\boldsymbol{\lambda}_1}$ is expressed in parent B, then this pair of genes fails the "compatibility criterion."

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

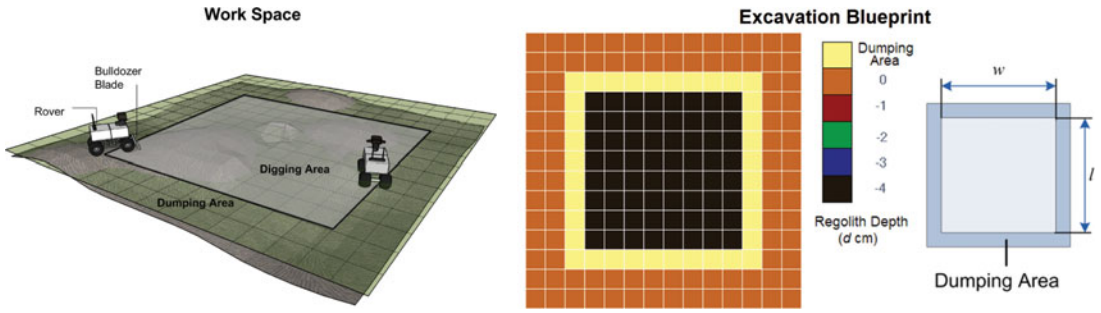

Fig. 7. A typical workspace for the excavation task (left). A corresponding excavation blueprint consisting of an excavation area surrounded by a dumping area (right). The blueprint is compared against the current workspace topology to compute a fitness that ranges from 0 to 1, where 1 corresponds to the desired topology as specified by the blueprint.

A detailed description and analysis of how this evolutionary approach utilizing coarse coding leads to task decomposition of complex task and discovery of novel behavior is presented in Section 5.2.

4. Excavation Task

The excavation task is intended to demonstrate the feasibility of autonomous teams of robots digging pits and clearing landing pads for lunar base construction. In these experiments, teams of robots are equipped with bulldozer blades that are used to push regolith. Self-organized task decomposition may be needed to accomplish the task given a global fitness function that does not give instructions on how to solve the task or bias for a particular solution strategy. A typical training environment is shown in Fig. 7. The workspace is modeled as a two-dimensional (2D) grid with each robot occupying four grid squares. For this task, the controller needs to possess several capabilities to complete the task, including interpreting excavation blueprints, performing layered digging and avoid burying or trapping other robots. The blueprint defines the location of the dumping area and target depth of the excavation area (Fig. 7). Each robot controller has access only to a local zone within the excavation blueprint at a time. Much like how insects have hard-coded genes to sense templates in their environment, the robots have pre-programmed capability (similar to hard-coded genes) to sense the various states of the goal map. However, this is insufficient in completing an excavation task and these various states need to be correctly interpreted to perform the correct actions. The fitness function f for the task is given as follows:

$$f = \frac{\sum_{j=1}^{J} \sum_{i=1}^{I} \vartheta_{i,j} \cdot e^{-2|g_{i,j} - z_{i,j}|}}{\sum_{j=1}^{J} \sum_{i=1}^{I} \vartheta_{i,j}} \tag{5}$$

where I and J are the dimensions of the entire area and $\sum_{j=1}^{J} \sum_{i=1}^{I} \vartheta_{i,j} > 0$ and $\vartheta_{i,j} = 1$ if grid square (i, j) is to be excavated and 0 otherwise; $g_{i,j}$ is the target depth and $z_{i,j}$ is the current depth. This objective function is used to train the robot controllers for the task at hand. The function produces a real value typically that can range from 0 to 1, where 1 indicatesthat the current excavation area matches the blueprint topology and 0 when it does not match the topology. This fitness function maybe used as a quantitative metric to compare the performance of various excavation systems.

4.1. Robot model

For these experiments, the inputs to controllers evolved using the ANT methodology (referred to as ANT controllers) are shown in Table I and their location is shown in Fig. 8. The robots have access to current position (x, y) from localization scans performed in simulation. The variable z is computed through estimated integration of changes in depth values. The discretized x and y coordinates are used to look up the goal depth $g_{x,y}$ from the excavation blueprint of each grid square region in front of the robot.

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

Table I. Excavation: Sensor inputs.

Sensor variables	Function	Description
$Z_1 \ldots Z_4$	Depth sensing relative to goal depth	Level, above, below, don't care, dump
E_1, E_2	Depth sensing relative to ground	Above, below or level
B_1	Blade position	Below, level, above, home
L_1	Blade force sensor	0–4
S_1	Front obstacle detection	Obstacle, no obstacle
D_1	Separation distance from nearest robot	0–3
H_1	Heading from nearest robot	North, east, west, south
R_1	Robot tilted downwards	True, false
U_1	Robot stuck	True, false
M_1	Memory variable	0, 1

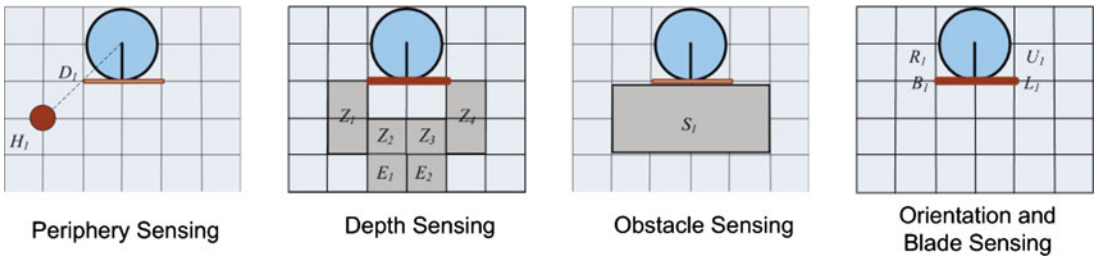

Fig. 8. Robot input sensor mapping for the simulation model.

All raw sensory input data are discretized and fed to the controller. $Z_1 \ldots Z_4$ and $E_1 \ldots E_2$ are obtained using simulated ground scans of the grid squares shown. For practical implementation, these ground scans would be obtained from a LIDAR or TRIDAR system. $E_1 \ldots E_2$ are obtained by comparing the soil depth in front of the rover to the wheel depth. A simulated sensor, S_1, is used to detect obstacles at the front. Robot tilt, heading of nearest robot, distance of nearest robot and whether a robot is stuck or not are represented using R_1, H_1, D_1 and U_1, respectively. The heading and distance to the nearest robot are used because front obstacle detection alone may not be sufficient to detect obstacle due to inherent blind spots. It is also important for the robot to determine whether it is stuck or not after executing its move behavior. This can be due to any number of reasons and provides the opportunity for the controller to react. In addition, the state of the attached bulldozer blade is provided. The bulldozer blade can be in one of the four positions, above ground, level, below ground and home position. This enables a robot to fill, push, dig or traverse over the soil, respectively. In addition, simulated force sensors are mounted to the blade and measure the net axial force against the blade. The robots can push a maximum of 24 units of soil and this is rescaled to between 0 and 4 for the blade force load, L_1. The robots also have access to one memory bit (M_1) that can be manipulated using several basis behaviors. Together there are $5^4 \times 3^2 \times 4 \times 5 \times 2 \times 4 \times 4 \times 2^3 = 8.6 \times 10^7$ possible combination of sensor inputs.

Table II lists the basis behaviors the robot can perform sequentially according to the given order within a single timestep. These basis behaviors are generic and can be used for many different tasks. It is up to the ANT controller to determine when to execute these basis behaviors (using sensory input) to solve the overall task. Furthermore, the excavation behaviors allow for actuation of a bulldozer blade. The blade can be raised or lowered to one of four positions as stated earlier.

4.2. Robot excavation model

The workspace is discretized into a 2D grid world, with each grid square having dimensions $l_x \times l_y$. In this model, the robots are equipped with a bulldozer blade that is used to dig and push regolith. The robot as explained earlier can only move forward and backward along cardinal directions and the new position after these behaviors is $(x_i + \Delta x, y_i + \Delta y)$ at time $t + \Delta t$, where $\Delta x, \Delta y \in \mathbb{Z}$ and Δt is the timestep required to execute the behavior. In addition, $|\Delta x| + |\Delta y| = 1$. A simplified soil interaction model is used to simulate excavation in the controller training environment. The simulation environment is used to track the soil height for each grid square, where $h(i, j, t)$ is the

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

Table II. Excavation basis behaviors.

Order	Behavior	Description
1	Throttle up	Set drive system to high power mode from default.
2	Move forward	Move one grid square forward
3	Move backward	Move one grid square backward
4	Random turn	Randomly turn 90° right or left.
5	Turn right	Turn 90° right
6	Turn left	Turn 90° left
7	Blade above	Set blade above ground d cm
8	Blade below	Set blade below ground d cm
9	Blade level	Set blade level to ground d cm
10	Blade home	Retract blade to home position (no contact with regolith).
11	Bit set	Set memory bit 1 to 1
12	Bit clear	Set memory bit 1 to 0

soil height at grid square (i, j) at time t. In the training environment, the robot occupies four grid squares. The digging apparatus, which is a bulldozer blade, occupies two grid squares next to each other and are at locations (x_1, y_1) and (x_2, y_2).

The soil interaction model assumes that the soil is incompressible and evenly distributed within each grid square. It is further assumed that the interaction between the soil is through a bulldozer blade that is used to push or fill soil according to the behaviors in Table II. The bulldozer blade, B_1, can interact with the soil when it is in one of the three states: below, level and above ground. This interaction with the soil is computed as follows:

$$h(x_i, y_i, t + \Delta t) = h(\tilde{x}_i + \Delta x, \tilde{y}_i + \Delta y, t) + h(\tilde{x}_i, \tilde{y}_i, t) - z_{wi} - b_h \cdot \varepsilon(V_{blade}, b_h) \quad (6)$$

where $i = [1, 2]$, (x_i, y_i) are the current positions and $(\tilde{x}_i, \tilde{y}_i)$ are the old positions of the blade. z_{wi} is the depth of front wheel i and b_h is blade position, where $b_h \in -1, 0, 1$, and $V_{blade} \geq 0$ and is the volume of soil in front of the blade. After the soil height against the blade is updated, the soil height underneath the front wheel (old position of the blade) is updated as follows:

$$h(\tilde{x}_i, \tilde{y}_i, t + \Delta t) = z_{wi} + b_h \cdot \varepsilon(V_{blade}, b_h) \quad (7)$$

where $\varepsilon(V_{blade}, b_h)$ is given as follows:

$$\varepsilon = \begin{cases} 0, & \text{if } V_{blade} = 0 \text{ and } b_h \geq 0 \\ 1, & \text{otherwise} \end{cases} \quad (8)$$

The volume of soil in front of the blade, V_{blade} is given as follows:

$$V_{blade} = \sum_{i=1}^{2} [h(x_i, y_i) - z_{wi} - b_h] \cdot l_x \cdot l_y \quad (9)$$

where l_x and l_y are the length and width of each grid square, respectively. When the blade is positioned below wheel depth, this results in scraping and accumulation of soil in front of the blade. When the blade is in level state, this results in pushing of already accumulated soil along the terrain and when the blade is positioned above ground, the accumulated soil is dislodged.

4.3. Training

Noting that each behavior in Table II can be triggered or not for any 1 of 8.6×10^7 possible combination of sensor inputs, there is a total of $2^{12 \times 8.6 \times 10^7} \approx 10^{3 \times 10^8}$ possible states in the search space! Task decomposition is often necessary to tackle very large search spaces and find desired solutions. ANT using its coarse-coding scheme described earlier is shown to perform task

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

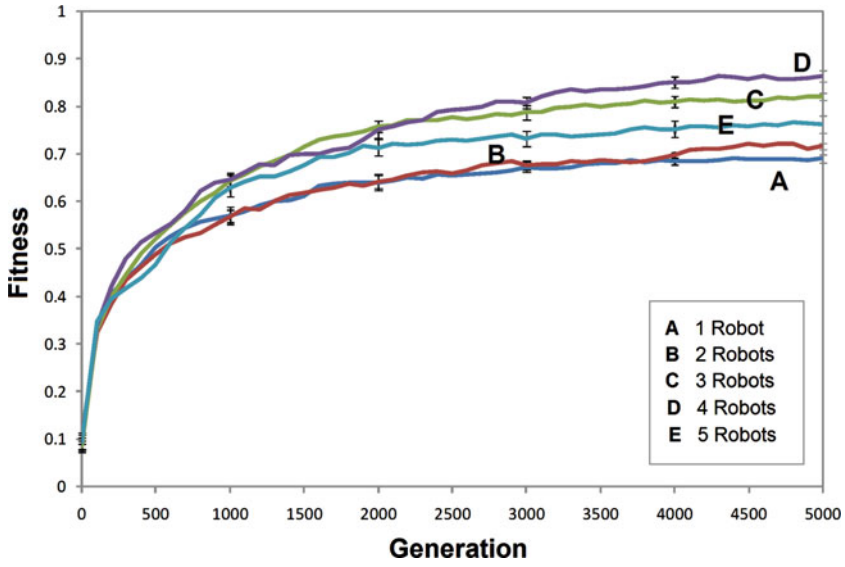

Fig. 9. Fitness comparison of ANT-based solutions during training for between 1 and 5 robots averaged over 30 runs.

decomposition[41] and is a good candidate to tackle this excavation task with its large task space. ANT controllers are first trained (evolved) in this simplified training environment. The genome for ANT is shown in Fig. 5.

Darwinian selection is performed using the given fitness function averaged over 100 different scenarios.[18] The population is randomly generated with an initial number of neurons ranging from 40 to 120 neurons. The neurons are feed forward and the second layer of neurons is fully connected to all the sensory input neurons. Furthermore, the ANT growth program is restricted to a maximum of four layers in total. The tissue however can grow in area. The fitness f for each controller is calculated for an excavation area, dug to a goal depth of d below ground and area spanning $l \times w$ squares. The work site area is 8×8 squares and a goal depth of $d = 1$, 2 or 3 units below ground. The robots are initialized at random positions on the work site. The population size for training is $P = 100$, with crossover probability $p_c = 0.7$, mutation probability $p_m = 0.025$ and a tournament size of $0.06P$. The fitness of the fittest individual from the population during each generation is taken to be the system fitness. Each training runs lasts 5000 generations and is repeated 30 times to obtain the average system fitness.

5. Simulation Results

Figure 9 shows the population best fitness of the system evaluated at each generation of the artificial evolutionary run. The system performance is affected by the number of robots per digging area (8×8 squares). A single robot is not as efficient as four robots working in parallel with each robot having a smaller area to cover. As will be shown later in Section 5.3, with more than four robots for a 8×8 area, the problem of antagonism arises when multiple robots trying to perform the same task interfere with one another and reduce the overall efficiency of the group.

First, we consider the effect of topology on controller training performance. The performance of standard fixed-topology neural networks and NEAT,[35] a variable topology neural network methodology, is compared in Fig. 10. For this comparison, topologies for NEAT and standard neural networks are randomly generated and contain between 40 and 120 neurons. For the fixed neural networks, the neurons are feed forward, and like ANT, restricted to a maximum of four layers that includes the sensor input layer and output layer. The transfer function used is the modular activation function.

ANT shows nearly a 30% improvement in fitness performance over the best of these conventional approaches. Importantly, ANT obtains a fit solution (fitness 0.9 or higher) with a nearly two-fold advantage over NEAT[35] (Fig. 11). In both cases, ANT shows a substantial improvement over standard

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

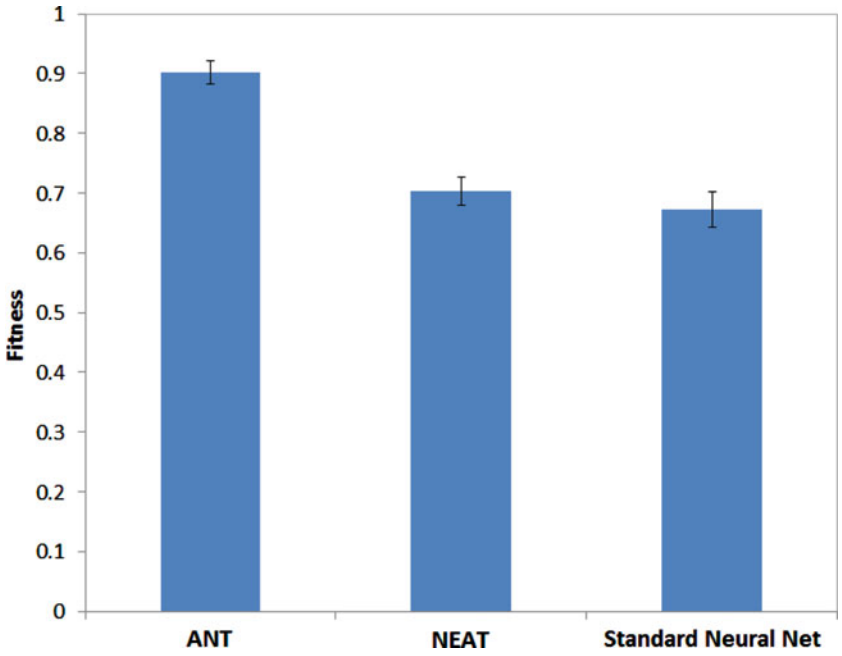

Fig. 10. Maximum fitness averaged over 30 training runs for ANT, NEAT and standard neural networks.

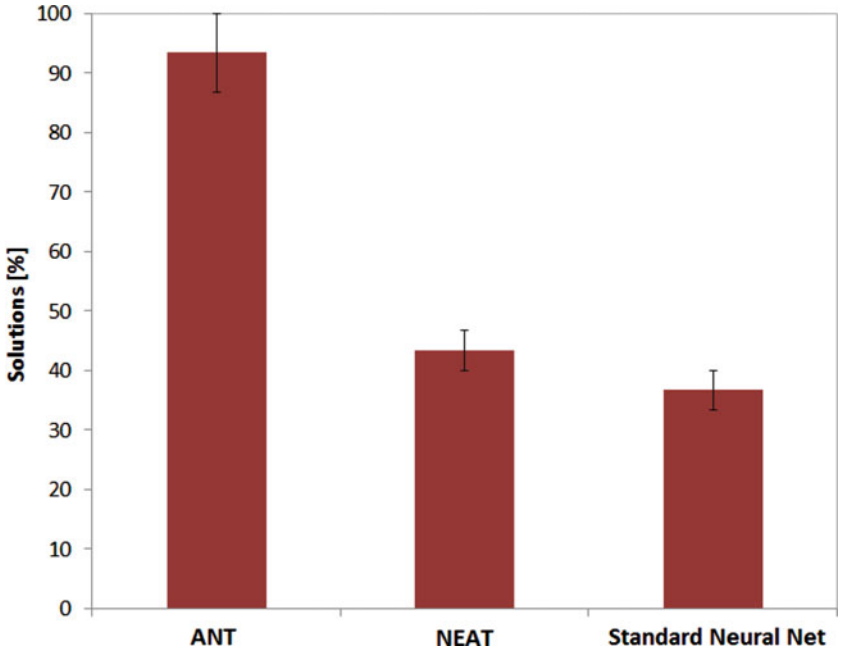

Fig. 11. Probability of finding a solution (fitness $\geq$0.9) averaged over 30 training runs for ANT, NEAT and standard neural networks.

neural networks. Further analysis of the robustness and scalability of conventional neural networks and ANT is presented in Section 5.3. These results show that ANT performs substantially better in terms of scalability and robustness than conventional neural networks.

In conventional fixed and variable topology networks, there tends to be more "active" synaptic connections present (since all neurons are active), and thus takes longer for each neuron to tune these connections to the sensory inputs. ANT is an improvement as the topology is evolved and decision neurons learn to mask out spurious neurons. The net result is that ANT produces fitter solutions in fewer genetic evaluations compared to conventional neural networks.

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

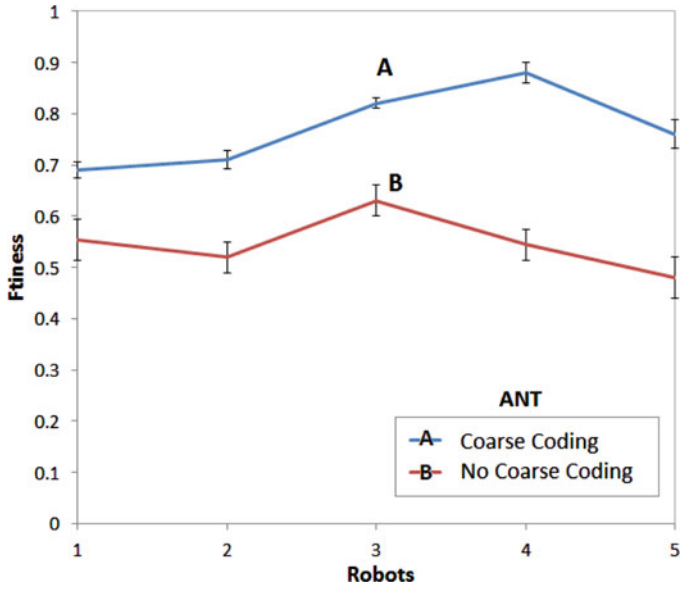

Fig. 12. Maximum fitness of ANT with and without-coarse coding for between 1 and 5 robots.

Further comparisons are performed to determine the key features within ANT that contribute to its improved performance over conventional neural networks. In this study, the coarse-coding functionality is turned off, to determine its contribution to ANT's overall performance. The results show a significant drop in performance (Fig. 12). The overall performance without coarse-coding is comparable to a variable topology architecture such as NEAT. This confirms that it is the coarse-coding functionality that provides ANT its advantage over conventional neural networks.

5.1. Evolution of behaviors through task decomposition

In this section, we present evidence suggesting how self-organized task decomposition occurs in ANT controllers. With human devised task decomposition, the first step is to devise the task objective and then figure out the necessary subtasks, followed by further partitioning of the subtasks until all components of the task are identified and readily solvable. With self-organized task decomposition, there is no supervisor to break up the objective function into the necessary subtasks. Instead modules form that without any central coordination and using local information learn to solve certain subtasks through a process of trial and error as will be shown here. These modules in turn interact and cooperate to solve the overall task.

Figure 13 shows a typical ANT training run for the excavation task. Statistics are obtained of key sensor–behavior combinations listed in Table A1 during a typical training run. In this scenario, four robots are used, using the standard environment described in Section 4.3. Figure 13(a) shows the controllers converge to a solution through a series of punctuated rises in several identified behaviors. These results give insight into how ANT solves the excavation task. The results suggests that the controllers evolve certain critical behaviors such as simple obstacle avoidance early during evolutionary training (Fig. 13(b)). This is followed by related behaviors, such as stuck avoidance (Fig. 13(e)) (i.e., avoiding getting stuck in soil). These behaviors are a requirement for the controllers to effectively move around the experiment area to perform excavation. Next, the controllers evolve to steadily improve the accuracy of performing correct dumping behaviors (Fig. 13(d)).

At around 3100 generation, a sharp rise in fitness is observed (Fig. 13(a)), rapidly converging toward a fitness of 0.9 (solution to the task). The controllers evolve to find critical behaviors to solve the task through a process of trial and error, after evolving prerequisite behaviors such as obstacle avoidance.[41] This evidence of performing trial and error to improve fitness can be seen through increased activity of cut-dig, which enables the controllers to randomly obtain a fitness advantage over rest of the population (Fig. 13(c)). Cut-dig requires the blade to be positioned below ground to perform a sharp incision cut into the soil. As the robot continues moving forward, a discretized

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

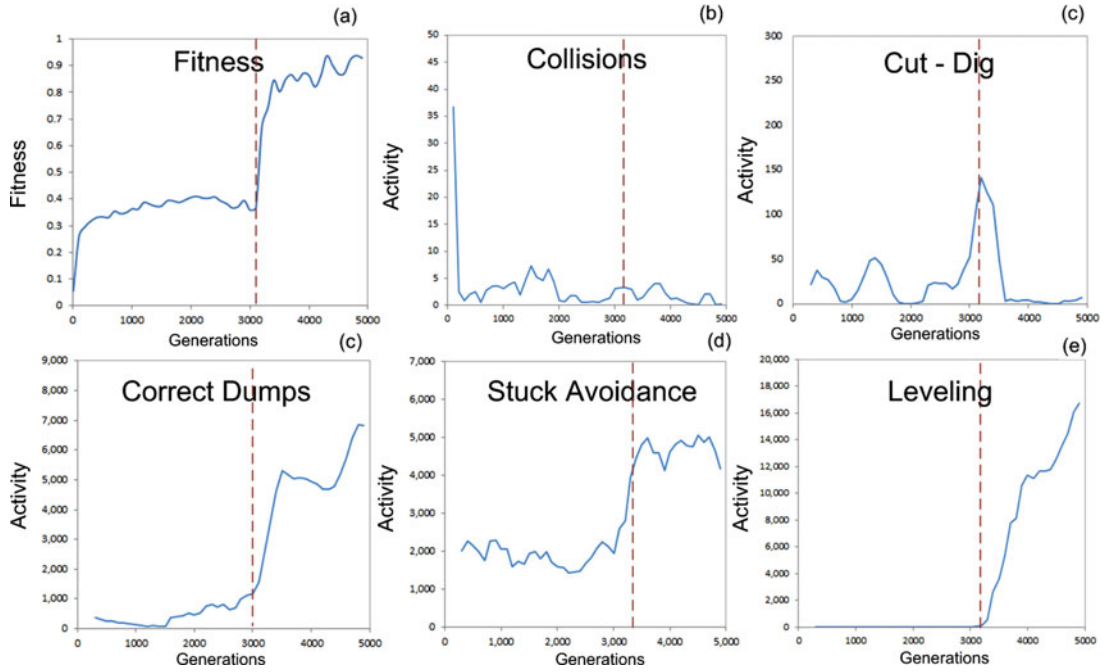

Fig. 13. (a) Fitness. (b)–(f) Evolution of important behaviors for the excavation task during a typical evolutionary training run of ANT. At generation, 3100 new behaviors are discovered foreshadowing a rapid rise in fitness and convergence to a solution.

sloped cut is obtained instead of the required level cut. Therefore a series of these cut-dig behaviors and back-fill behaviors are switched on and off for short distances to approximate a level cut.

This combined with improved dumping performance foreshadows improvement in fitness (Fig. 13(a), (c), and (d)). Although this digging approach is not every efficient, it increases fitness pressure to find more efficient ways. This results in the appearance of fine-tuned digging behaviors that require fewer, inefficient incision cut-dig behaviors and more level digging. Level digging requires the robot to first perform an incision cut and then continue pushing soil at a level depth.

Thus, with this improvement, the cut-dig behavior does not have to be used as often and can match blueprints with better accuracy (Fig. 13(c)). The cut-dig behavior acts as a scaffolding mechanism, which is used at a critical time to evolve efficient digging behaviors. However, for the controllers to achieve a high fitness, several other behaviors are acquired. Once the controllers achieve a high accuracy in following the excavation blueprint and performing digging and correct dumping behavior, this results in a significant improvement in fitness. Further refinement occurs with the appearance of leveling behaviors (Fig. 13(f)) that enables the robots to wander and correct small mistakes. This process of wandering and correcting mistakes enables even less accurate controllers that miss an area to achieve high fitness through a process of feedback. This is increasingly utilized because more time is available to perform these behaviors with increased digging accuracy. The net result is further improvement in fitness.

5.2. Neuronal analysis

Several different methods have been used to analyze ANT solutions to identify emergent behaviors that solve the excavation task. First, we analyze neuronal and behavioral activity of a evolved tissue solution with a fitness of 0.99. It is difficult to discern what the controllers are doing based solely on output behaviors unlike previous task such as sign following and resource gathering.[39,41] Hence, we have used sensor–behavior combinations (Table A1) to detect instances of complex behaviors that have evolved to solve the task. Further, we identify the location and activity of these behaviors in the tissue topology.

Figure 14 shows an ANT tissue with the location and activity of major behavioral centers. As with other tasks evolved using ANT,[39,41] the behaviors evolved as neurons modules are distributed within the tissue. Interestingly, most decision and motor neurons remain dormant and do not encode for

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

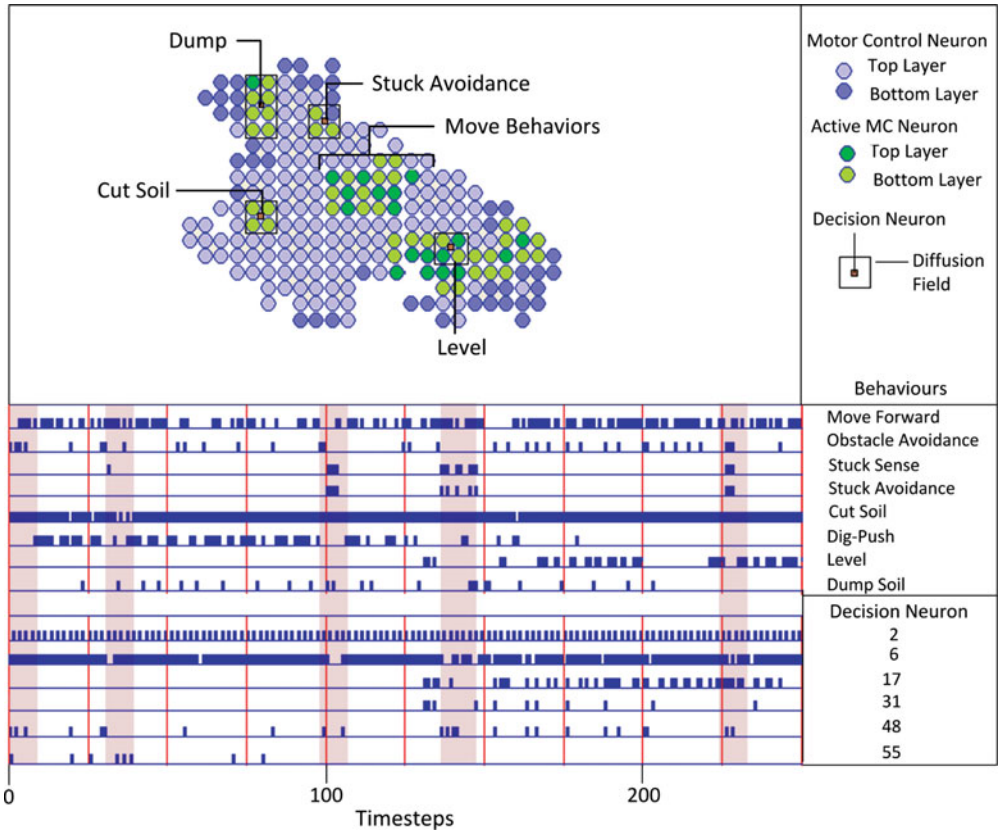

Fig. 14. ANT topology and neuronal activity identifying important behaviors evolved for the excavation task. Shaded columns highlight correlation in neuronal and behavioral activity during a typical simulation run. The controller was evolved using four robots in a 8 × 8 excavation area and has a fitness of 0.99.

any identified behaviors. This further suggests that these neurons remain neutral. This is beneficial in evolution, where the right combination of mutations can first encode a function neutrally followed by a trigger mutation that unlocks this behavior. The neutral neurons act as scaffolding for this transition. With so many neurons remaining dormant, this permits the process to occur in parallel throughout the tissue. Once a new behavior is turned on in a trial and error, explorative process, there is no guarantee that this behavior will provide a fitness advantage. If it does not, this individual has reduced chance of survival through selection. However, if this change results in a behavioral innovation that provides a significant fitness advantage, then the progeny multiplies and may then dominate the evolving population. This repeated process of trial and error continues, with innovations being accumulated much like building blocks until a high fitness is reached.[41]

These identified neurons modules have acquired specialized traits that are used to perform incision cuts (cut-dig), scout, level, dump, obstacle avoidance and perform stuck avoidance behaviors (Fig. 14). Analysis of decision neuron activity with sensor input and behavior output combination shows correlations. This suggests that the behaviors are distributed among one or more decision and motor neuron modules. These modules work cooperatively to encode tissue behavior. This has a significant advantage. For one, if a module is perturbed due to deleterious mutations, then another module that partly encodes for the behavior can recover some of this functionality. Further, if multiple modules encode for the exact same functionality, then deleterious mutations have minimal effect.

At the bottom of Fig. 14, key events in a typical run lasting 250 timesteps are shaded. Observing the move forward behavior as with the other 12 behaviors from Table II, it is difficult to ascertain what is occurring due to the complex interactions of multiple robots and their behaviors. Instead we obtain statistics of sensor–behavior output combinations from Table A1 to better understand the solutions.

Several correlations are identified here, including between stuck avoidance and decision neuron 6, leveling and decision neuron 31, obstacle avoidance and decision neuron 48, cut-dig and decision neuron 55. Interestingly, decision neuron 2 behaves like a clock oscillating between its two binary

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

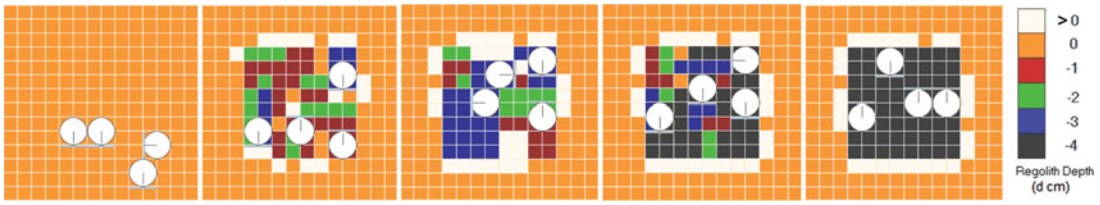

Fig. 15. Simulation snapshots of an excavation task simulation (four robots) after 0, 50, 75, 100, 170 timesteps.

states. This oscillator behavior has been observed in other ANT solutions for different tasks.[39,41] It is hypothesized that this is used to synchronize the various behavior modules distributed within the tissue.

5.3. Behavioral analysis

In a second approach, we analyze evolved solutions by observing robot controller behavior in the simulation environment (Fig. 15). The robots scan the area in front at each timestep to determine whether its above or below the goal depth. Once it is found to be above the goal depth, it starts digging. Because the robots are randomly positioned at the start, the excavated areas appear randomly throughout the work site. The robots start digging by lowering their blades one level below ground and move forward one grid square forward. Immediately afterwards, the controllers raise the blade to level and continue pushing soil at a level height until reaching a dumping area. Once the robots detects a dumping area in front, they push the soil forward, followed by a reverse and turn. This results in the accumulation of piles of soil (berms) at the dumping area.

The controllers learn to perform layered digging (Fig. 15). The controllers do not lower the blade immediately to reach the goal depth, instead they perform an incision cut and then push soil layer by layer until the target depth is reached. As one robot removes one layer, another cooperates and pickups where the last one left off. This process continues until the goal depth is reached. Multiple robots also cooperate by detecting one another through obstacle sensing and avoid getting in the way. Once another robot is sensed in front, the robot makes a turn. With sufficient number of these interaction, many of the robots end up being in parallel tracks, which minimizes obstructions from other robots. The dug areas that appear randomly scattered through the work site at the beginning merge into one large area at the end.

The process of self-organization occurs with individual robots controllers sensing and manipulating the environment using local information. The controllers can only sense local information. Although a global goal map may be present, each controller can only sense a localized region of the map at a time. The initial appearance of excavated areas randomly throughout the work site, followed by their merger into one excavated area that meets the goal depth specifications, is an example of the self-organization. This self-organization occurs due to cooperative behavior of multiple robots using local sensor input and performing only local actions. The work done by one robot is not undone by another.

Analysis of the solutions suggests that the controllers exploit templates by learning to correctly interpret dumping zones, “don’t care” regions, depth relative to the specified excavation blueprint. These controllers perform interpretation by taking the discernable sensor input and determining the proper action to perform, such as move forward or backward and whether to lower, level or raise the blade (Fig. 15). With the goal map, if the goal depth is lower than the current soil depth, then the interpreted action is to dig by lowering the blade below ground and pushing.

Communication between robots occurs through manipulation of the environment in the form of stigmergy. Manipulation of the environment occurs by excavating a region or dumping excavated material. Each robot interprets the blueprint and determines whether to deep, backfill or move to the next location. However, these robots cannot typically dig to the goal depth all at once and hence have to dig material layer by layer. By this, robots can only remove part of the soil at a time before having to dump at the dumping locations. It is up to another robot to determine that the goal depth has not been reached and pickup where the last robot left off, which is a form of implicit communication mediated through the environment. Furthermore, once the goal depth is reached at a grid square, any other robot that comes and sense the grid square interprets that the goal depth is reached.

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

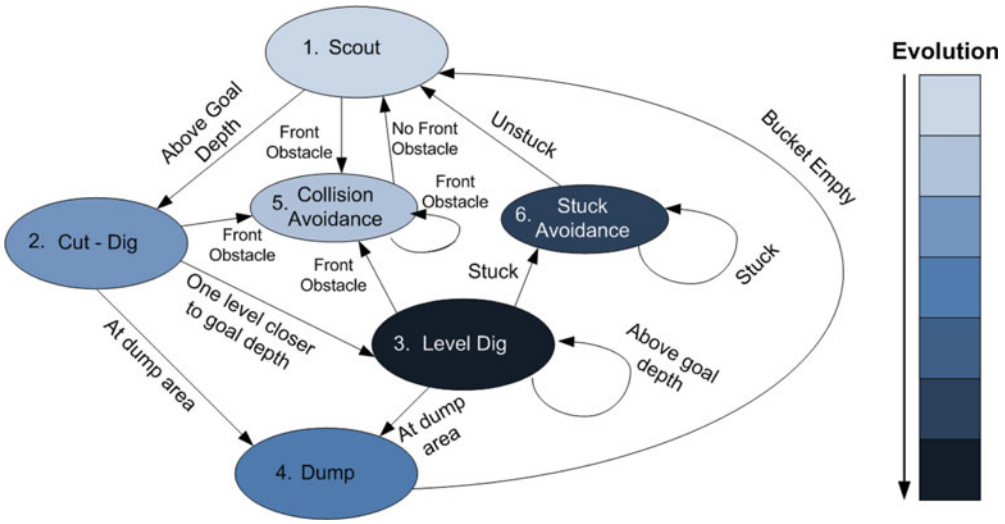

Fig. 16. Representation of major behavior states evolved by the ANT solution controllers. Shading indicates the order in which these complex behaviors are evolved.

Through this implicit communications, cooperative actions also occur. That is, the excavation work done by one robot is not undone by another. Initially, small digging areas form, but in time through cooperative actions these digging areas merge into one region that matches the goal depth. In addition, the ANT controllers exploit the ability to sense the depth of soil relative to wheel depth (Fig. 15). This enables each robot controller to sense whether it is excavating deeper or backfilling at the current depth. The ability to backfill while useful can also undo the effort of other robots excavating at different depths. Sensing and avoiding this scenario is a form of cooperation.

The robots also have the ability to sense the relative position of a nearby robot much like radar. This feature is exploited to to avoid collisions confirmed from Fig. 13(b), when a series of output behaviors such as "move forward" and "turn left" are applied in sequence. Although obstacles can be detected using front sonars, there exist blind spots to the extreme right and left, making it difficult to detect and react to obstacles when a sequences of behaviors are executed.

5.4. Summary on how the controller works

We used different techniques to analyze how complex behaviors emerge within the ANT controllers using neuronal analysis of controller that attains a fitness of 0.99. Within this tissue, we identify portions of the tissue that are active and find different specialized modules emerge to perform the required subtasks required to perform the overall task. We are also able to trace the evolution of these behaviors to a series of critical events, when the controllers discover new capabilities leading to substantial improvement in fitness performance. Here, we summarize our finding in Fig. 16 in the form of a finite state machine.

The controller during typical operation (1) scout (moves around) to find regions that are above the goal depth. Once found the blade is lowered to (2) cut/dig into the soil, moving forward and then transitioning into (3) "level digging." As the bucket is filled or if the robot nears a dumping area, the soil is dumped (4). Along the way, robots perform collision avoidance (5). This in fact appears to be a series of behaviors that minimize robot to robot interaction, enabling the robots to work in parallel and to maximize area coverage. This ability to minimize direct robot to robot interactions influences the scouting behavior state and decision of where to perform cut-dig. Along the way, the robot may also need to avoid getting stuck (6) when pushing too much regolith. As will be shown later, they take a non-greedy approach to excavation, where it is more efficient for individuals to temporarily give up when stuck (by unloading the pile of regolith being pushed) and resume later.

We also summarize how these behaviors have evolved. The ability to scout and perform basic collision avoidance is evolved first. This is followed by incision cuts. Initially, the incision cuts are performed randomly through trial and error, in addition to dumping the regolith. In turn, controllers evolve to interpret the signs correctly to dump at the designated locations. However, this process is

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

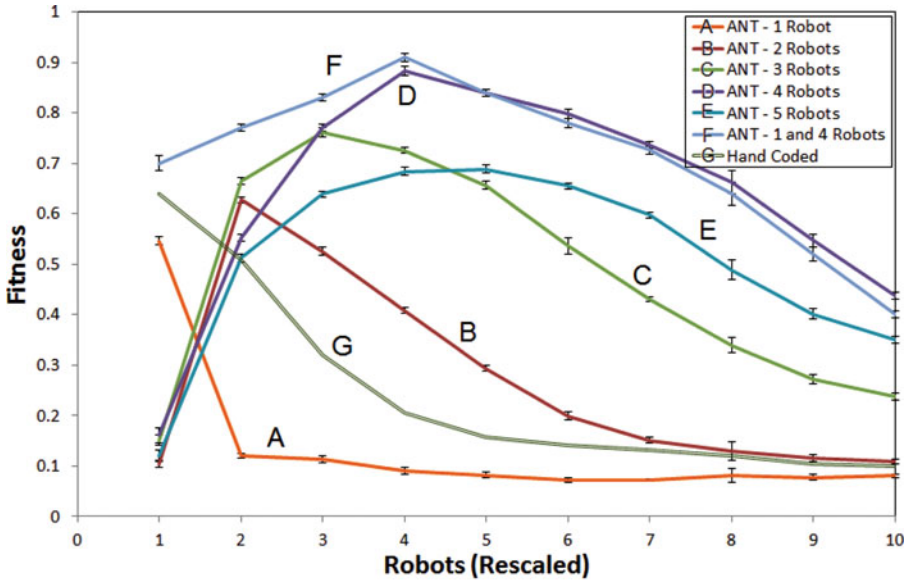

Fig. 17. Excavation performance of ANT-based solutions evolved using 1–5 robots (8 × 8 excavation area) applied to scenarios using 1–10 robots.

still inefficient, limiting the amount of regolith being excavated or causing to dig below the designated goal depth. With controllers evolving to excavate more regolith, they can get stuck pushing too much. The controllers in turn evolve stuck avoidance behaviors used to abort excavation when stuck and start over from a different direction. A critical step is reached when the controllers "discover" the level dig behaviour that allows pushing of regolith over longer distances and enables excavation and transport of significantly more regolith to dumping area resulting in a significant leap in fitness.

5.5. Scalability

First, we consider performance of an intuitive hand-coded robot controller (see Appendix A1 for program description) for varying number of robots. The hand-coded controller shows the best performance for a single robot scenarios, but steadily decreases in performance for increasing number of robots. For this controller, it is assumed that robots need to avoid one another and hence obstacle avoidance behaviors are included. The results show that this assumption is not optimal and results in poor performance for increased number of robots. The controller is not effective in exploiting parallelism. This suggests that cooperative behaviors are necessary to exploit increased number of robots, and do not just involve intuitive obstacle avoidance procedures.

The fittest ANT solutions from the simulation runs shown in Fig. 9 are applied to this new setting, by varying the number of robots while holding the digging area constant (Fig. 17). Taking the controller evolved for a single robot and running it on a multirobot system show a steep degradation in performance. This is expected since the single-robot controller lacks the cooperative behavior necessary to function well within a multirobot setting. For example, these controllers fail to develop "collision avoidance" behaviors. Similarly, a multirobot system scaled down to a single robot setting also shows a degradation in system performance. With the multirobot system, controllers have evolved to exploit and depend on cooperative actions to complete the task; thus, when the environment is abruptly changed, the controllers perform poorly. With more than four robots for a 8 × 8 area, the problem of antagonism arises when multiple robots trying to perform the same task interfere with one another and reduce the overall efficiency of the group. The key here is the number of robots selected during training. Proper selection of robots better enables the controllers to be scalable. Further, we extend the comparison by mixing number of robots scenarios during training. For example, mixing training scenarios of one and four robots shows better scalability performance than one and four robot solutions. In addition, this approach shows better performance than the intuitive hand-coded solution when using a single robot.

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

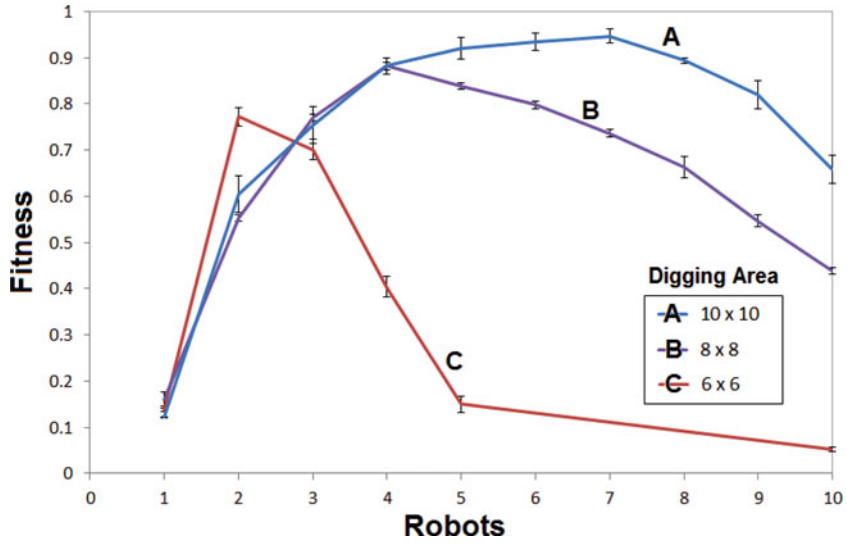

Fig. 18. Excavation performance ANT-based solutions evolved using four robots (8 × 8 excavation area) applied to resized excavation area and the number of robots.

This effect of antagonism on performance is further supported in Fig. 18 by varying excavation area and applying an ANT solution evolved using four robots on a 8 × 8 area to the 6 × 6 and 10 × 10 digging area. For the 6 × 6 area, peak performance is achieved using only two robots, but shows a larger drop in performance for increased number of robots, showing the increased effect of antagonism for a smaller area. In addition, increasing the area results in the peak performance reached with seven robots and performance drops in smaller increments. These two experiments confirm the problem of antagonism with these multirobot systems. Furthermore, the ANT solution show higher fitness than during training for the large digging area with the increased number of robots

As noted earlier, a key factor in developing adaptive controllers is scalability. Scalability is critical, because in real world scenarios, one or more robots may be disabled or unavailable and the approach needs to effectively complete the task. Taking the evolved standard neural network controllers and applying them on *n* robots are shown in Fig. 19. Comparing the results with Fig. 17, standard neural networks controllers show poor scalability performance. In comparison, ANT shows a two-fold performance advantage. Controllers evolved for four robots show highest fitness for a four-robot scenario. In addition, controllers evolved for a single robot show poor scalability applied on the increased number of robots. Alternately, controllers evolved with four robots show relatively better performance on average than other training scenarios. This indicates that in a real world scenario, ANT controllers are more robust to handle uncertainties in the number of active robots toward completing a task.

Focusing on the ANT controllers, it is interesting to note that the controllers trained with four robots for an 8 × 8 digging area perform considerably better on average than solutions trained for other number of robots. The optimal ratio of robots to digging area using solution trained with four robots is shown in Fig. 20. It is evident that this solution performs better for increased goal depth (Fig. 21) and shows better performance than other solutions for increased excavation area.

These simulation experiments suggest that there exists an optimal set of training conditions that enables controllers to evolve improved scalability. Although the controllers may be better adapted to antagonism under higher training densities with improved obstacle avoidance techniques, these behaviors may not be well tuned to completing the overall objectives effectively. This optimal condition is dependent on task duration. Furthermore, the optimal density is beneficial when the task is time limited. Given enough time, the suboptimal solution can attain the same fitness but consumes more energy.

5.6. Selection for multirobot behavior

Based on Figs. 17 and 20, the ANT controllers trained for a four-robot solution produces solutions that show better rescalability than other initial conditions for a 8 × 8 excavation area. Can evolutionary

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

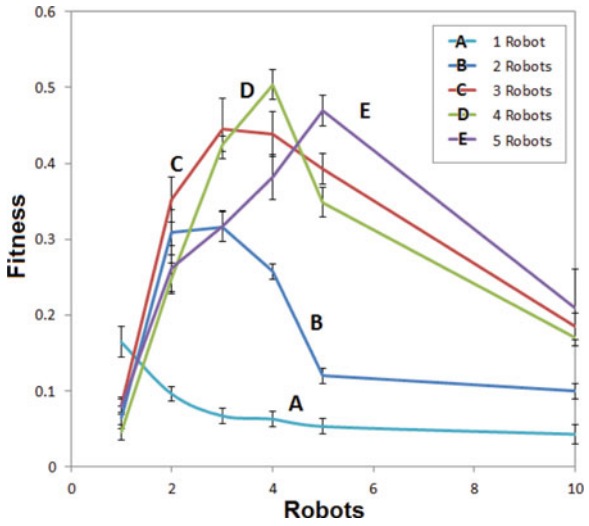

Fig. 19. Excavation performance of the fittest standard neural networks evolved using between 1 and 5 robots applied to scenarios using 1–10 robots.

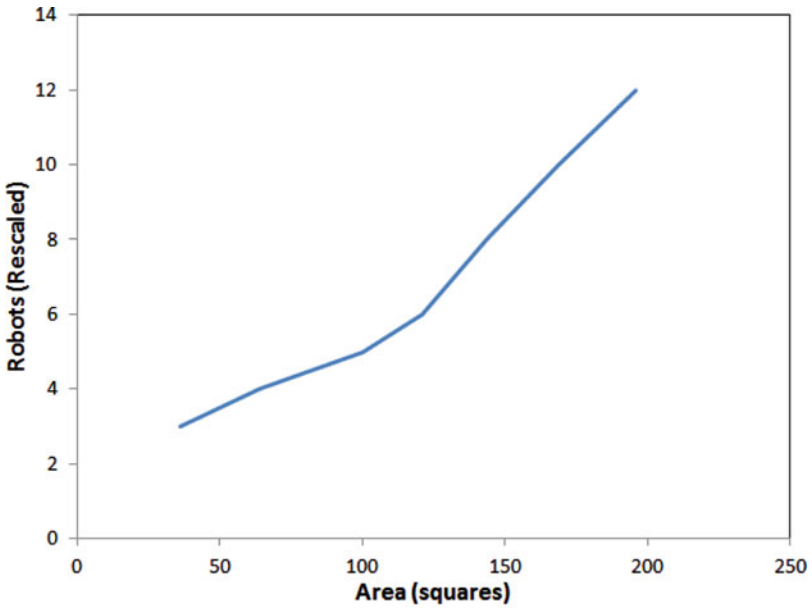

Fig. 20. Excavation performance of ANT solutions evolved using four robots on a 8 × 8 excavation area for varying excavation area.

techniques be used to determine both multirobot controller and optimal system parameters (such as number of robots) simultaneously? For this experiment, the number of robots, N, is introduced as an evolvable variable within ANT. This is akin to a population reproduction rate, where certain species have more children at a time than others. A histogram in Fig. 22 shows the distribution of N among the population best averaged over 30 runs. If there were no selection pressure on N, then the histogram should be a uniform distribution. The results suggests that evolutionary selection tends toward the observed optimal number of robots (Fig. 17).

The question then is why is the system tending toward the optimal number of robots per given area? It is not obvious that controllers gravitate toward an optimal setting and stay there. For one, if the optimal solution is sensitive to damaging mutations, then the population may gravitate to a more stable suboptimal solution. However, solutions evolved under the optimized condition rescale better. Both traits provide an advantage, helping to gravitate solutions toward this setting and remaining

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

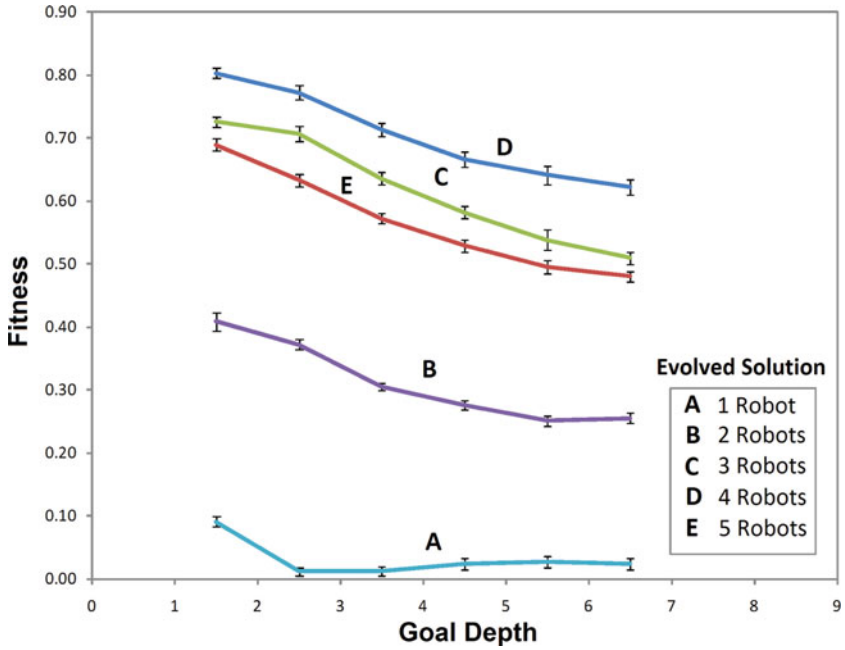

Fig. 21. Excavation performance of ANT-based solutions for varying depth taken after 10,000 timesteps.

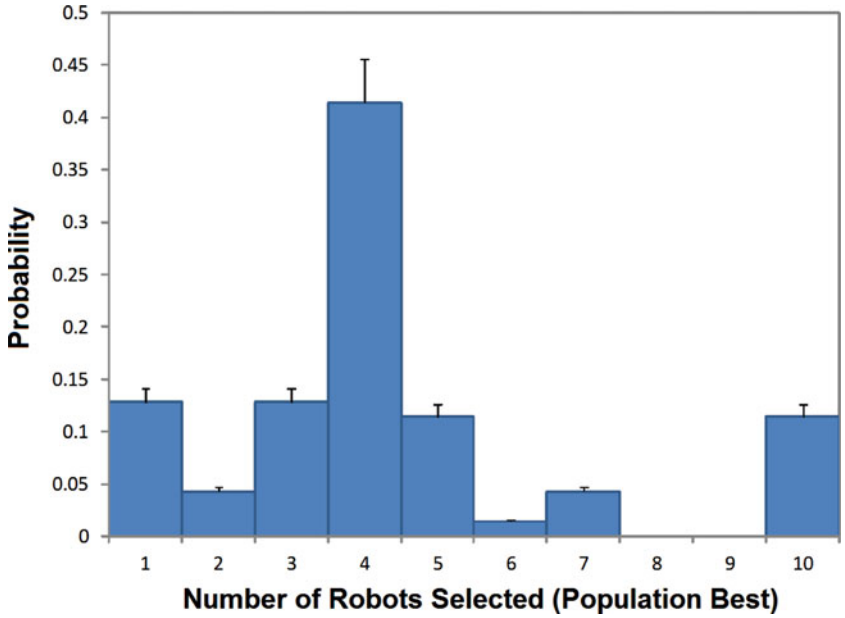

Fig. 22. Histogram of number of robots selected, where N the number of robots is evolved as a parameter (right).

there. An individual having adapted under this optimal setting is better adept at handling a deleterious mutation of N robots that could either result in increased or decreased number of robots.

This technique of evolving a controller and other system parameters concurrently reduces the need to analyze the rescalability performance and antagonism. Because the system tends toward the optimal number of robots, the problem of antagonism is limited. However, owing to the stochastic nature of the search process, the optimal number of robots can only be obtained with statistical certainty after repeated evolutionary runs.

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

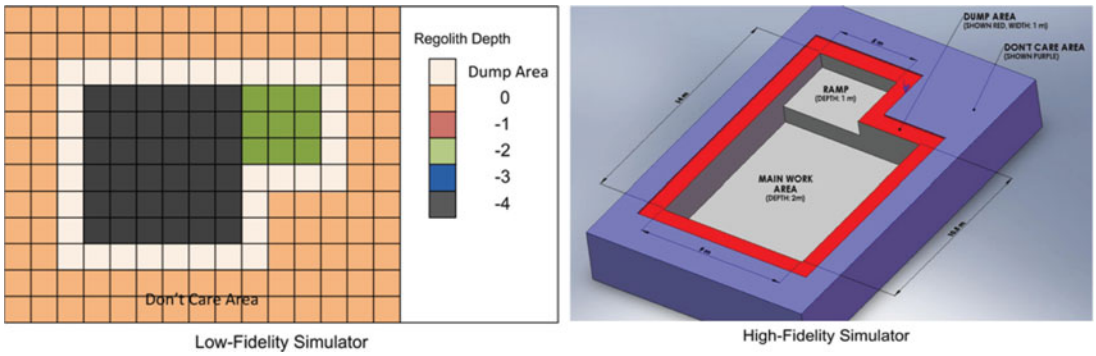

Fig. 23. Excavation blueprint of a landing pad. Low fidelity training simulator (top). High fidelity Digital Spaces^{TM} simulator (bottom).

6. Proof-of-Concept Experiments

6.1. High fidelity lunar simulator

This section describes work on porting ANT controllers onto excavation robots in the Digital Spaces^{TM} simulator. Digital Spaces^{TM} is an off-the-shelf, commercial, high-fidelity 3D simulator that simulates the low gravity lunar surface. The Balovnev Soil Interaction Model[1,4] is used to simulate the deformable terrain. Deformable terrain modeling allows for accurate regolith-tool and wheel–terrain interactions. This virtual approach facilitates prototyping and testing of alternative digging concepts and can potentially reduce hardware experiment costs especially of off-world environments.

The simulated robots are equipped with a front loader, are holonomic and deliver 300 W average power. The robot model shown in Section 4.1 is ported to the Digital Space environment. Details of the approach are found in Abu El Samid.[1] The (x, y) position, depth (z) and tilt (R_1) are built in variables. Using a discretized mesh of the surface terrain and (x, y, z) of the robots, $Z_1 \ldots Z_4$ and $E_1 \ldots E_2$ are computed. Relative position of the robots are used to compute the state of the front obstacle avoidance sensors (S_1), heading of nearest robot H_1 and distance to nearest robot, D_1. The blade positions are simulated on the front loader by defining four set positions to match the bull dozer blade settings, including above, level, below ground and home position. The blade load, L_1, is computed using the soil interaction model.[1,4]

To ensure consistency between the training simulations and the Digital Spaces virtual world, fitness is monitored for 1, 2, 3, and 4 robots applied to an excavation blueprint of a landing pad shown in Fig. 23. The results are shown in Fig. 24 (right). The excavation blueprint consists of a landing pad $9 \times 10.5 \times 2.0$ m^3 deep connected by a $5.0 \times 3.5 \times 1.0$ m^3 deep region for a ramp to exit/enter the landing pad. Comparison with the training simulator (Fig. 24) shows that the results correlate well. After reaching a peak fitness at about 200 timesteps for all except the single robot scenario, the fitness drops gradually in the Digital Spaces simulations. The training simulator results (Fig. 24) level off within 200 timesteps (except for the single robot scenario). In Digital Spaces simulations, the fitness then gradually drops after the final goal depth is reached. In this case, the robots continue to move around the work area with blade height set to level. Theoretically, this means the robots do not dig any regolith, but, in reality, they skim off small amounts of material, going slightly below the goal depth over time. Video snapshots of a typical simulation are shown in Fig. 25).

6.2. Controlled field experiments

This section describes work performed to test the ANT excavation controllers on real robots. The robots were tested inside a 50-m dome, containing loose sand. The robots were tested under low lighting conditions representative of a lunar region shadowed due to cratering. In addition, the ambient temperature was between -5 °C and -10 °C. The robot model shown in Section 4.1 is ported onto a team of UTIAS Argo class robots (Fig. 26). The robots are approximately 15-kg, non-holonomic and are four-wheel drive. Each robot is equipped with a 1-DoF servo actuated bulldozer blade system, a PC-104 Intel 80386 computer, an assortment of sensors and actuators. The drive

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

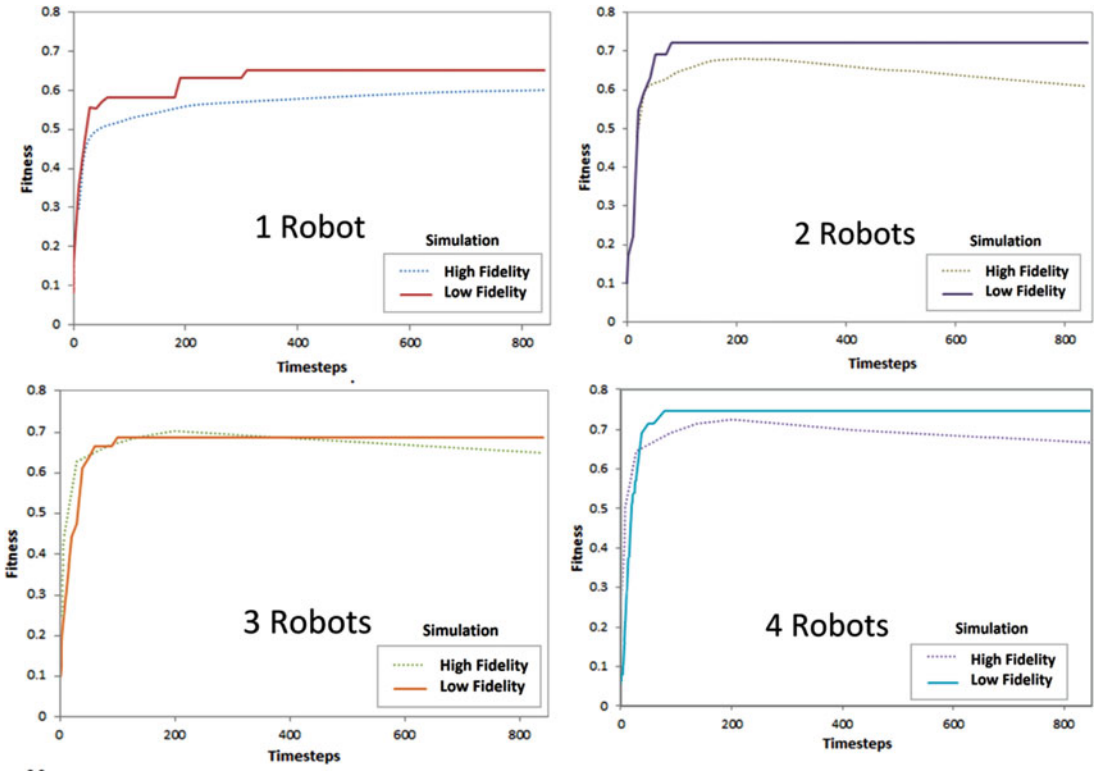

Fig. 24. Fitness comparison of ANT controllers in the low-fidelity training simulator and high-fidelity Digital Spaces^{TM} simulator.

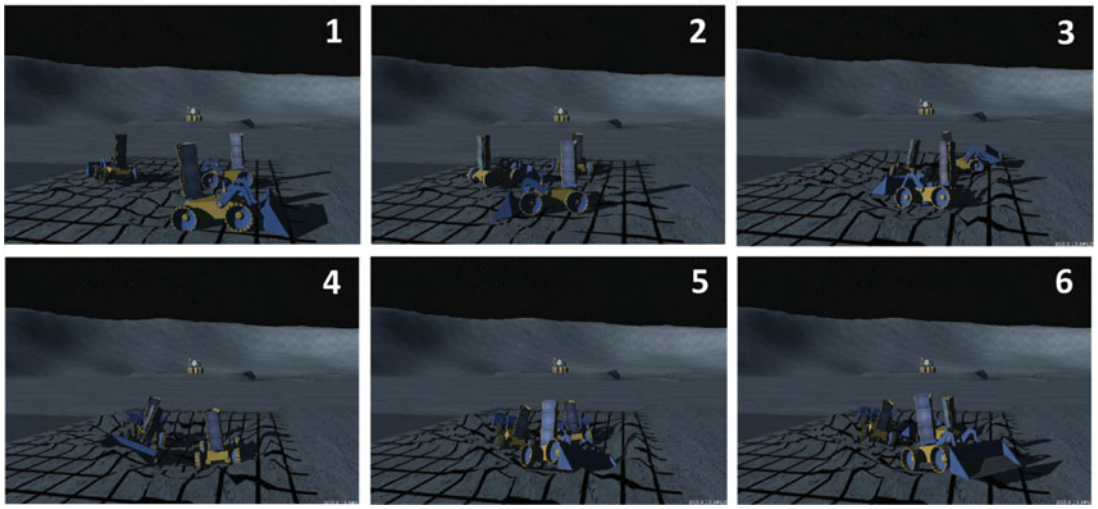

Fig. 25. Digital Spaces^{TM} simulation of three robots using a four-robot ANT controller solution to perform excavation on simulated lunar terrain. The robots unlike under training conditions use a front-loader bucket instead of a two-way bulldozer blade. The excavation blueprint includes a hole and a ramp for the robot to enter/exit surrounded by the dumping area.

and actuation system is powered by two 8-V, 4500-mAh lithium ion batteries. The electronics and computers are powered by four 7.5-V lithium ion batteries.

An overhead camera is used to determine the (x, y) location of the robots by tracking LED light beacons (Fig. 27). The variable z is estimated using an overhead laser range finder that performs periodic scans of the work site. Using the estimated z value, the robot equipped with a Leica laser range finder is used to compute $Z_1 \ldots Z_4$ and $E_1 \ldots E_2$. The onboard laser range finder and sonars are

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

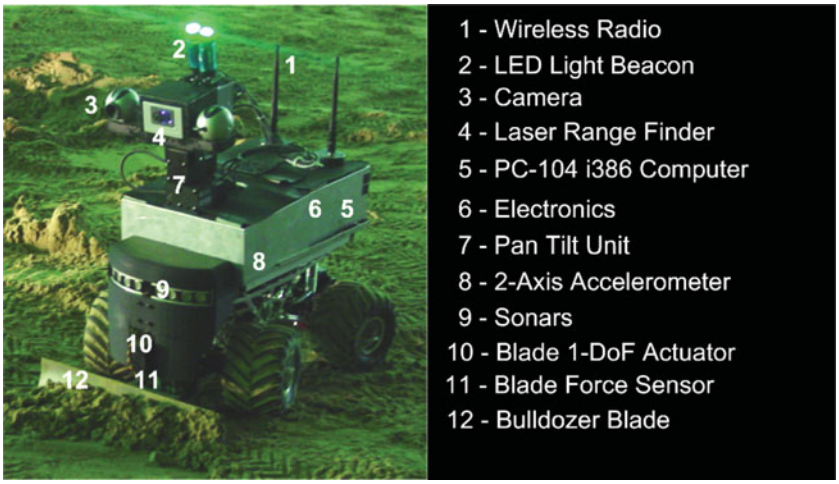

Fig. 26. An Argo robot equipped with a 1-DoF bulldozer blade, a laser range finder, a pair of Logitech® Quickcams™ affixed to the pan-tilt unit and an assortments of other sensors.

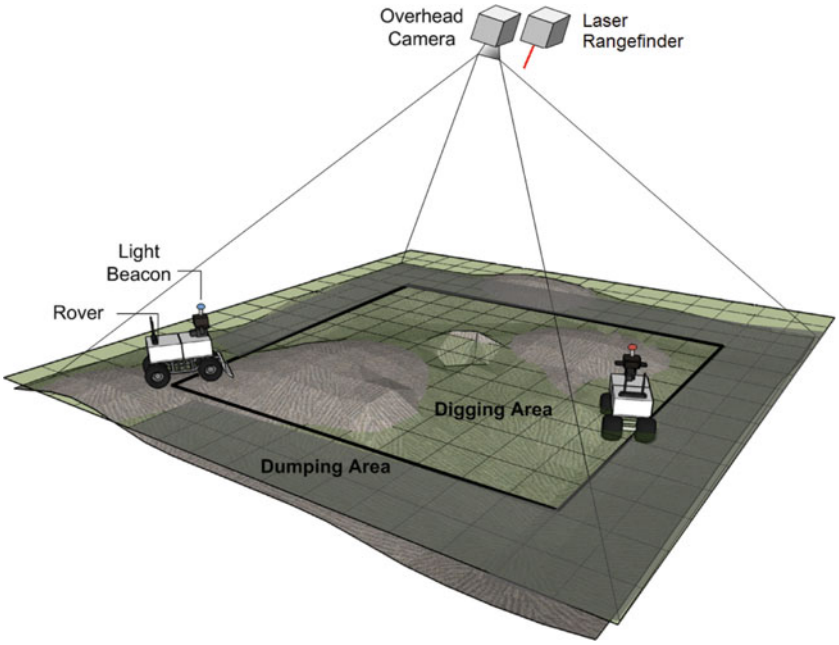

Fig. 27. Layout of the experimental system, showing excavation robots mounted with light beacons localized using an overhead camera system.

used to detect obstacles at the front (S_1). Heading and distance of nearest robot, H_1 and D_1 respectively, are computed using the overhead camera. R_1, the robot tilt, is measured using an onboard two-axis accelerometer. A webcam is used to determine U_1, i.e., whether the robot is stuck or not by simple comparison of consecutive images. On the robot, fine blade adjustments are made using PID control to ensure a constant force is met. The robot can handle a maximum of 10-N push force and this is rescaled to integer values between 0 and 4 for the blade load, L_1. The excavation blueprint used in shown in Fig. 7 (right).

Figure 29 shows 3D laser scans of the soil moved from a 5-h excavation experiment with two robots. Comparison of the laser scans shows that an estimated 0.7 m^3 of soil volume was displaced in the test area. This estimate is performed by averaging the regolith dug and dumped in the vicinity. This measurement is typically an underestimate, because the regolith at the start of the experiment is uncompressed, and after the experiment, with multiple rover traversal around, the regolith becomes quite packed. The robots start excavating holes randomly in the digging area. These holes in turn

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

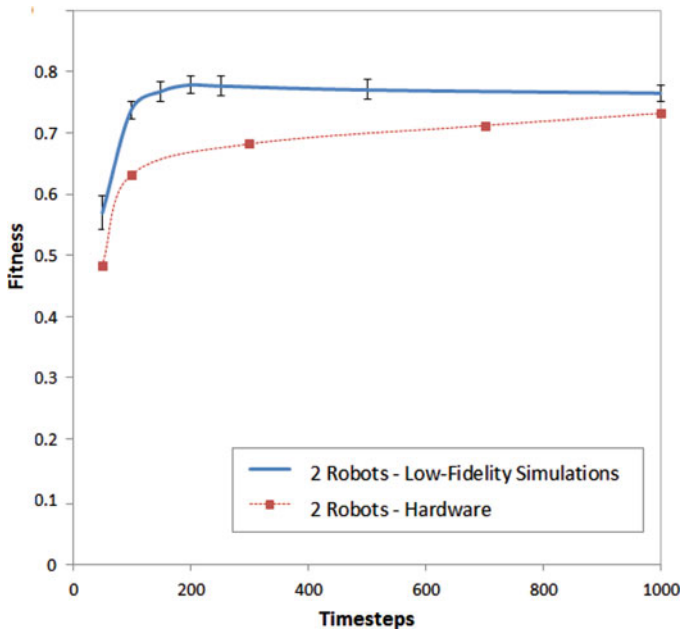

Fig. 28. Fitness during a typical excavation run of an ANT controller evolved using four robots (8 × 8 area) applied to two robots on the training simulator and real robots.

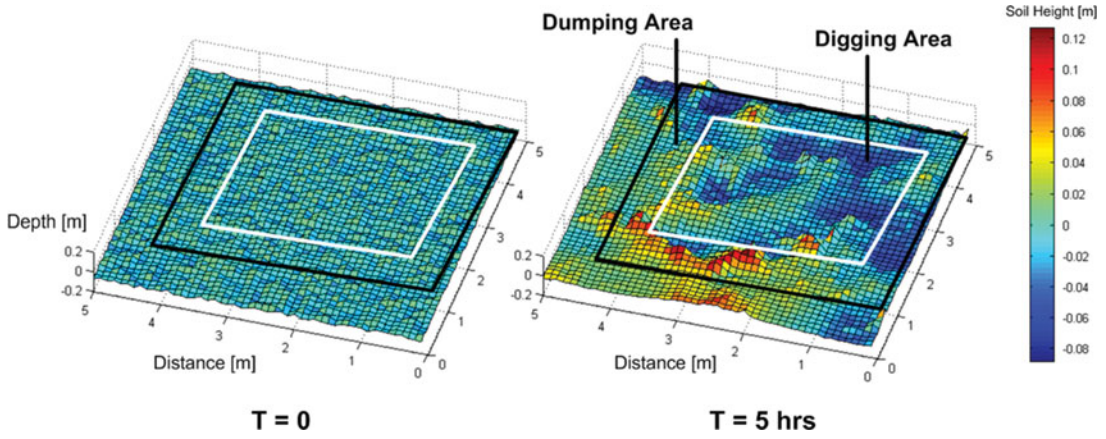

Fig. 29. An overhead laser scan showing the difference in soil height before and after the excavation experiment.

merge to form one big pit. In this excavation run, three sides of the pit wall have formed and are clearly visible from ground photographs and overhead 3D laser scan (Fig. 30).

Ground truth measurements show that the maximum depth reached is between the height of the berm and lowest point of the dig area to be 15 and 20 cm. At the target depth, the regolith is quite packed. This is despite the fact that the robots are underpowered for the task, resulting in the robots frequently getting stuck, backing out and attempt to restart digging, thus, slowing the overall excavation process. Fitness comparison between the low-fidelity training simulator and hardware experiments for the two-robot scenario is shown in Fig. 28.

On most occasions, the robots correctly interpret the excavation blueprints and dump dirt in the specified dumping areas as shown from the 3D laser scans (Fig. 29).

In the training simulator, the steady-state fitness is reached in shorter time than the hardware experiments. This is because the factors that reduce the efficiency of the real robots digging (due primarily to their design), such as side spillage of soil while pushing or frequently getting stuck while pushing, result in less soil being transferred to the intended destination. Furthermore, other *in-situ* factors may further reduce excavation efficiency of each robot on the lunar surface. As a result, each robot requires more time to reach a steady-state fitness. One solution would be to supply the optimal

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

Table III. Field experiment metrics.

Area	System response
72%	Correctly excavated towards goal depth
24%	Above goal depth
4%	Indeterminate

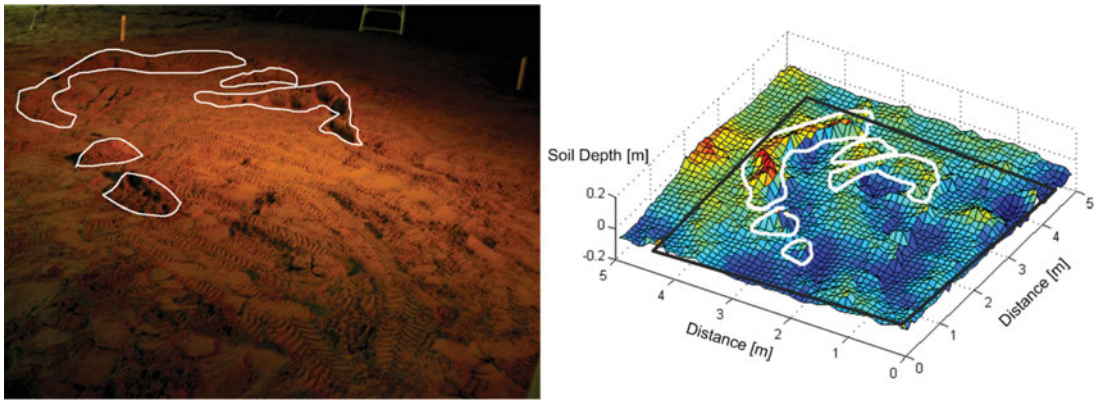

Fig. 30. Photo of the work site after excavation (left). Mounds of dirt (berms) highlighted. 3D laser scan of work site with berms highlighted (right).

number of robots for the given work area as described earlier. This would further parallelize the task and enable task completion in shorter time despite any limitations in design. Given higher power rovers, the robots could conceivably achieved the required goal map in shorter time. Nevertheless, our results show that a simple simulation environment, with low-cost proof of concept vision hardware, is sufficient to demonstrate the critical behaviors.

Table III shows that nearly 70% of the area is excavated toward the goal depth or material dumped to correct regions according to the goal map, whereas nearly 24% had material accumulated incorrectly (either regolith dumped at locations where it is supposed to be dug out or regolith dug out where it is to dumped). These results were achieved with low cost vision hardware, principally webcams and single-pixel laser rangers. The results suggest that the robot controllers correctly interpret and follow the goal map on most occasions. However, a series of real world conditions impart some error to the results particularly in dumping regolith off away from the dumping areas due to drift errors. This provides 1–3 cm error in position of the dumping areas. This accounts for a significant portion of the error. These drift errors while lowering fitness metrics are actually not a major concern for field applications. These field experiments would suggest that we introduce some margin distance between critical boundaries that can be easily included into the blueprints. Other factors that contributed to errors in the experiments include the overhead system at times loosing track of robot position.

The field experiments show several creative behaviors discovered by the controllers in solving the task. This includes correctly reading templates and cues in the environment such as the excavation blueprint, rocking and obstacle avoidance behaviors. The controllers evolve rocking behavior to get unstuck, a method of going back and forth to dislodge itself from sand traps and deep "pot holes." In this scenario, the controller learns to stop repeated excavation moves and "back out" when it is unfruitful to dig or when the robot is stuck moving forward (Fig. 31). This is accomplished using the memory variable to represent the 'stuck state' and prevent repeated attempts at digging. In addition, the controllers evolve obstacle avoidance/negotiation behavior. Figure 32 shows video snapshots of this behavior. Two robots are attempting to excavate and push dirt toward the dumping area. In frame 1, the robot in front (b) makes a left turn but is in the way of robot (a). Robot (b) backs out and scans in front (frame 2). It detects robot (a) and backs up further (frame 3). In frame 4, robot (b) attempts to move forward assuming the path is clear and stops having detected robot (a). In frame 5, robot (b) backs up again, makes a right turn and moves forward facing the dumping area as originally intended. The obstacle avoidance behaviors from high-fidelity simulations and hardware experiments show that

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

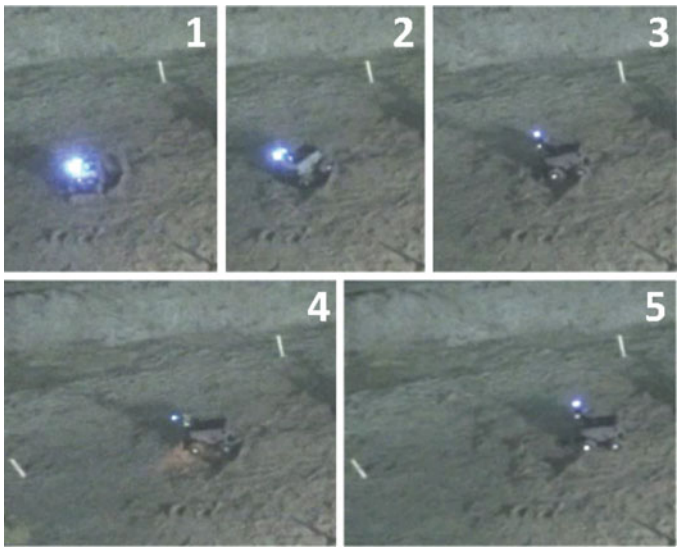

Fig. 31. Video snapshots of the stuck avoidance behavior, including a rocking behavior followed by a "backout" behavior triggered after the robot gets stuck trying to push the blade forward.

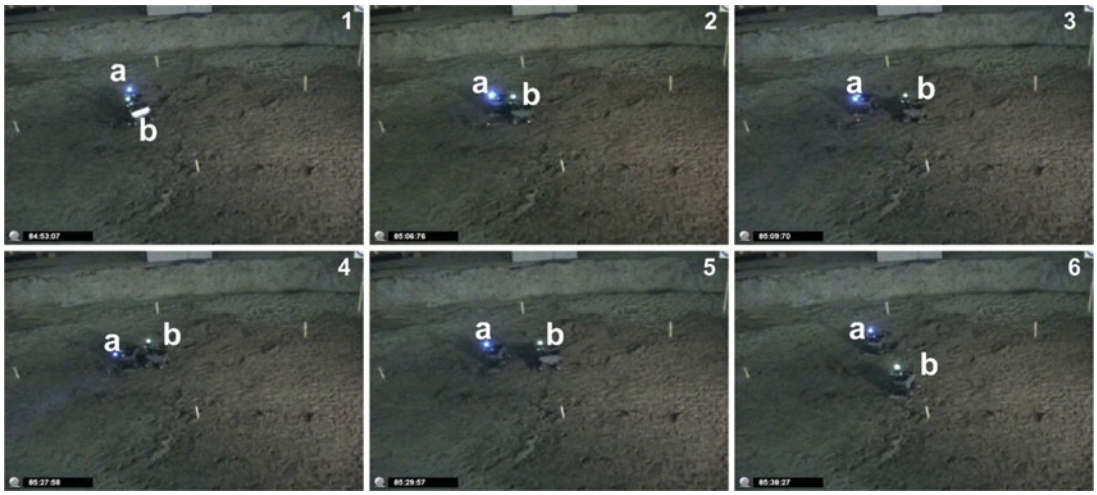

Fig. 32. ANT controllers evolve obstacle avoidance/negotation behaviors.

the robots avoid dumping soil on or next to each other, preventing the robots from getting stuck. This suggests that the controllers use proximity sensing to avoid such actions.

These field experiments demonstrate proof-of-concept feasibility of applying ANT on multirobot excavation for lunar applications using low-cost vision hardware consisting of webcams and laser range finder. However, significant work remains, particularly in demonstrating large-scale excavation. Work is underway in developing a team of large excavation robots for a field demonstration. Another solution is to use other excavation implements such as a bucket wheel or front loaders and is being actively explored using this approach.

7. Conclusion

The ANT framework, a neural network approach to robotic control, has been applied to multirobot excavation. Using this approach, we show the feasibility of using multirobot excavation for site preparations tasks. The approach shows improved performance and scalability than conventional neural networks and hand-coded solutions. This facilitates finding creative behaviors that are not specified or encouraged by an experimenter. These creative behaviors verified in hardware

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

include correctly interpreting blueprints, performing layered digging, obstacle avoidance and rocking behaviors to avoid getting stuck. This approach is shown to produce controllers that have improved scalability compared to conventional neural networks and hand-coded solutions. Furthermore, ANT can simultaneously evolve the desired controller and select for optimal number of robots for the task. This approach is shown as a possible solution to the problem of antagonism in decentralized multirobot control.

This approach shows that a machine learning method such as ANT is a machine-driven process to develop creative yet effective deterministic methods for multirobot excavation. ANT evolves efficient excavation methods such as slot dozing that would otherwise require experts. This is important in space operations, where the space system operational engineer may not be an excavation expert. It is fair to say that ANT is unlikely to outperform a trained excavation vehicle operator in the field. But the ability for an autonomous algorithm to approach that capability is significant for off world excavation, where it is prohibitively expensive to send a full team of astronauts to perform excavation. The proposed approach would rely on combination of teleoperation and autonomous control to perform excavation. The proposed approach would automate mundane excavation and regolith moving tasks. There would however be an important need for oversight, where a teleoperator monitors the excavation process and intervenes when there are unexpected failures in the field. Thanks to the use of the proposed autonomous control approach to perform most of the time consuming and mundane tasks; this can reduce the number of teleoperators and also reduce fatigue due to communication latency. However, significant work is required in infusing teleoperation-based control with autonomous control.

Future work is planned for full field excavation experiments using multiple robots. This work will combine both teleoperation and autonomous control. The proposed approach requires effective methods of surveying and localization to position the excavator and achieve correct dimensions of features in the work site. Our experiments demonstrate use of overhead cameras to perform localization. Use of overhead cameras is a viable approach on the moon. In addition, radio beacons may also be used for localization.

Overall, the proposed approach shows a credible pathway toward lunar excavation, ISRU and base construction. Further work is needed in fusing teloperation with the proposed autonomous controls approach. Challenges remain in scaling up the rovers and localization system to an appropriate size for a representative field demonstration.

Acknowledgments

This research has been supported by Natural Science and Engineering Research Council of Canada, NORCAT and EVC. The authors would gratefully acknowledge the contribution of Dale Boucher of NORCAT and Jim Richards of EVC. The authors would like to thank the reviewers for helping better identify the strengths and weaknesses in the proposed approach.

References

1. N. Abu El Samid, Lunar Excavation Methods and Evaluation *Master's Thesis* (University of Toronto, Institute for Aerospace Studies, Toronto, Canada, 2008).
2. N. Abu El Samid, J. Thangavelautham and G. D'Eleuterio, "Infrastructure Robotics: A Technology Enabler for Lunar in-situ Resource Utilization, Habitat Construction and Maintenance," *Proceedings of International Astronautic Conference* (2008).
3. J. Albus, "A theory of cerebellar function," *Math. Biosci*, **10**, 25–61 (1971).
4. V. Balovnev, *New Methods for Calculating Resistance to Cutting of Soil* (Amerind Publishing (Translation), New York, New York, 1983)
5. T. D. Barfoot and G. M. T. D'Eleuterio, "An Evolutionary Approach to Multiagent Heap Formation," *Proceedings of the 1999 Congress on Evolutionary Computation (CEC)*, (1999), pp. 427–435.
6. R. Beckers, O. E. Holland and J. L. Deneubourg, "From Local Actions to Global Tasks: Stigmergy and Collective Robots," *Proceedings of the 4th International Workshop on the Syntheses and Simulation of Living Systems* (MIT Press, 1994) pp. 181–189.
7. D. Bernard, G. Doraist, E. Gamble, B. Kanefskyt, J. Kurien, G. K. Man, W. Millart, N. Muscettolao, U. Nayak, K. Rajant, N. Rouquette, B. Smith, W. Taylor and Y. wen Tung, "Spacecraft Autonomy Flight Experience: The ds1 Remote Agent Experiment," *Proceedings of the AIAA*, (1999) pp. 28–30.
8. E. Bonabeau, M. Dorigo and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems* (Oxford University Press, New York, 1999).

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

9. E. Bonabeau, G. Theraulaz, J.-L. Deneubourg, S. Aron and S. Camazine, "Self-organization in Social Insects," *Trends Ecol. Evol.*, **12**, 188–193 (1997).
10. D. Bradley and D. Seward, "The development, control and operation of an autonomous robotic excavator," *J. Intell. Robot. Syst.* **21**, 73–97 (1998).
11. K. Bristow and J. Holt, "Can termites create local energy sinks to regulate mound temperature?" *J. Therm. Biol.* **12**, 19–21 (1997).
12. S. Cannon and S. Singh, "Models for Automated Earthmoving," *Proceedings from the International Symposium on Experimental Robotics* (1999).
13. Y. Cao, A. Fukunaga and A. Kahng, "Cooperative mobile robotics: Antecedents and directions," *Auton. Robots* **4**, 1–23 (1997).
14. F. Chantemargue, T. Dagaeff, M. Schumacher, and B. Hirsbrunner. Coopération implicite et performance. In Proceedings of the Sixth symposium on Cognitive Sciences (ARC), Villeneuve d'Ascq, France, December 10–12 (1996).
15. F. L. Crabbe and M. G. Dyer, "Second-Order Networks for Wall-Building Aagents," *Proceedings of the International Joint Conference on Neural Networks*, vol. 3 (1999) pp. 2178–2183.
16. M. Dunbabin and P. Corke, "Autonomous excavation using a rope shovel," *J. Field Robot.* **23**(6–7), 379–394 (2006).
17. J. Garthwaite, S. L. Charles and R. Chess-Williams, "Endothelium-derived relaxing factor release on activation of nmda receptors suggests role as intercellular messenger in the brain," *Nature* **336**(6197), 385–388 (1988).
18. D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning* (Addison-Wesley, NY, 1986).
19. P. Grassé, "La Reconstruction du nid les Coordinations Interindividuelles; la Theorie de Stigmergie," *Insectes Sociaux*, **35**, 41–84 (1959).
20. E. Halbach, V. Zhmud and A. Halme, "Simulation of Robotic Regolith Mining for Base Construction on Mars," *Proceedings of the 12th Symposium on Advanced Space Technologies in Automation and Robotics (ASTRA)* (2013) pp. 1–7.
21. G. Heiken, D. Vaniman and B. French, *Lunar Sourcebook: A User's Guide to the Moon* (Cambridge University Press, Cambridge, 1991).
22. G. Hinton, "Shape Representation in Parallel Systems," *Proceedings of the 7th International Joint Conference on Artificial Intelligence* (1981) pp. 1088–1096.
23. P. Husbands, "Evolving Robot Behaviours with Diffusing Gas Networks," *Proceedings of EvoRobots* (1998) pp. 71–86.
24. R. Jacobs, M. Jordan and A. Barto, "Task decomposition through competition in a modular connectionist architecture: The what and where vision tasks," *Cogn. Sci.* **15**(2), 219–250 (1991).
25. B. Khademian and K. Hashtrudi-Zaad, "Shared control architectures for haptic training: Performance and coupled stability analysis," *The International Journal of Robotics Research*, **30**(13), 1627–1642 (2011).
26. D. Lee and M. Spong, "Semi-Autonomous Teleoperation of Multiple Cooperative Robots for Human-Robot Lunar Exploration," *Proceedings of the AAAI Spring Symposium* (2006) pp. 1–8.
27. Z. Lu, P. Huang, Z. Liu and Z. Meng, "Stability conditions for asymmetric dual-user shared control method with uncertain time delay," IEEE, *2015 IEEE International Conference on Information and Automation*, pp. 1997–2002 (2015).
28. M. J. Matarić, M. Nilsson and K. T. Simsarian, "Cooperative Multi-Robot Box-Pushing," *Proceedings of the IEEE/RSJ IROS* (1995) pp. 556–561.
29. D. P. Miller and K. Machulis, "Visual Aids for Lunar Rover Tele-Operation," *Proceedings of 8th International Symposium on Artificial Intelligence: Robotics and Automation in Space* (2005).
30. P. R. Montague, J. A. Gally and G. M. Edelman, "Spatial signaling in the development and function of neural connections," *Cerebral Cortex* **1**(3), 199–220 (1991).
31. N. Napp and R. Nagpal, "Distributed Amorphous Ramp Construction in Unstructured Environments," *Proceedings of the Distributed Autonomous Robotic Systems (DARS '12)* (2012) pp. 1–15.
32. C. A. Parker, H. Zhang and R. C. Kube, "Blind Bulldozing: Multiple Robot Nest Construction," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* (2003) pp. 2010–2015.
33. P. Rowe and A. Stentz, "Parameterized Scripts for Motion Planning," *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems* (1997).
34. K. Skonieczny and D. Wettergreen, "Advantages of continuous excavation in lightweight planetary robotic operations," **35**, 1121–1139 (2016).
35. K. Stanley and R. Miikkulainen, "Continual Coevolution Through Complexification," *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2002)* (Kaufmann, 2002) pp. 113–120.
36. A. Stentz, J. Bares, S. Singh and P. Rowe, "A Robotic Excavator for Autonomous Truck Loading," *Proceedings of IEEE/RSJ International Conference on Intelligent Robotic Systems*, vol. 1 (1998) pp. 175–186.
37. R. Stewart and A. Russell, "Emergent Structures Built by a Minimalist Autonomous Robot using a Swarm-Inspired Template Mechanism," *Proceedings of the 1st Australian Conference on ALife (ACAL 2003)* (2003) pp. 216–230.
38. R. Stewart and A. Russell, "Building a Loose Wall Structure with a Robotic Swarm Using a Spatio-Temporal Varying Template," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* (2004) pp. 712–716.

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

39. J. Thangavelautham, A Regulatory Theory of Cortical Organization and Its Applications to Robotics *PhD Thesis* (University of Toronto, Toronto, Canada, 2008).
40. J. Thangavelautham, T. Barfoot and G. M. T. D'Eleuterio, "Coevolving Communication and Cooperation for Lattice Formation Tasks (Updated)," *Advances in Artificial Life: Proceedings of the 7th European Conference on ALife (ECAL)* (2003) pp. 857–864.
41. J. Thangavelautham and G. D'Eleuterio, "Tackling learning intractability through topological organization and regulation of cortical networks," *IEEE Trans. Neural Netw. Learn. Syst.* **23**, 552–564 (2012).
42. J. Thangavelautham and G. M. T. D'Eleuterio, "A Coarse-Coding Framework for a Gene-Regulatory-Based Artificial Neural Tissue," *Advances in Artificial Life: Proceedings of the 8th European Conference on ALife* (2005) pp. 67–77.
43. J. Thangavelautham, N. El Samid, P. Grouchy, E. Earon, T. Fu, N. Nagrani and G. D'Eleuterio, "Evolving Multirobot Excavation Controllers and Choice of Platforms Using an Artificial Neural Tissue Paradigm," *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)* (2009) pp. 258–265.
44. J. Thangavelautham, A. Smith, N. Abu El Samid, A. Ho, D. Boucher, J. Richard and G. D'Eleuterio, "Multirobot Lunar Excavation and ISRU Using Artificial-Neural-Tissue Controllers," *Proceedings of the Space Technology and Applications International Forum* (2008).
45. J. Thangavelautham, and G. M. T. D'Eleuterio, "A Neuroevolutionary Approach to Emergent Task Decomposition," *Proceedings of the 8th Parallel Problem Solving from Nature Conference*, vol. 1 (2004) pp. 991–1000.
46. V. Trianni and M. Dorigo, "Self-Organisation and Communication in Groups of Simulated and Physical Robots," *Biological Cybernetics*, vol. 95 (Springer, Berlin, Heidelberg, 2006) pp. 213–231.
47. J. Wawerla, G. Sukhatme and M. Matari, "Collective Construction with Multiple Robots," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* (2002) pp. 2696–2701.
48. M. Wilson, C. Melhuish, A. B. Sendova-Franks and S. Scholes, "Algorithms for Building Annular Structures with Minimalist Robots Inspired by Brood Sorting in Ant Colonies," *Autonomous Robots*, vol. 17 (2004) pp. 115–136.
49. E. Xidias, P. Zacharia and A. Nearchou, "Path planning and scheduling for a fleet of autonomous vehicles," *Robotica*, **34**, 2257–2273 (2016).

A. Appendix

A.1. Hand-coded controller

The hand-coded controller consists of 11 if–then statements executed at each timestep on each robot. The descriptions here mention which output behaviors are selected according to those given in Table II:

(1) Checks if the robot is stuck and selects behaviors 1, 4, 7, 10, 12.
(2) Otherwise, if the robot faces an obstacle, it selects behavior 4. Otherwise, the remaining if statements are executed as follows.
(3) If region Z_2 or Z_3 are in the "dumping zone" and the robot is pushing soil, $L_1 > 0$ and selects behavior 2, 3, 4.
(4) If region Z_2 or Z_3 are in the "don't care" and the robot is pushing soil, $L_1 > 0$ and selects behavior 3, 4, 12.
(5) If region Z_2 or Z_3 are in the "don't care" then selects behaviors 2, 4.
(6) If region Z_2 or Z_3 are in the "don't care" and $L_1 = 0$ then select behavior 3.
(7) If region Z_2 or Z_3 are at goal depth, then behaviors 2, 4 are selected.
(8) If region Z_2 or Z_3 are below goal depth, then behaviors 2, 7, 11 are selected.
(9) If region Z_2 or Z_3 are above goal depth, then behaviors 2 is selected.
(10) If region Z_2 or Z_3 are above goal depth and memory bit is 0, then behaviors 8, 11 are selected.
(11) If region Z_2 or Z_3 are above goal depth and memory bit is 1, then behaviors 2, 9 are selected.

All selected behaviors are executed sequentially according to Table II. The selections are then reset and control program executed on each robot for the next timestep.

A.2. Sensor-behavior detectors

Table A1 lists the sensor–behavior detectors used to analyze the evolution of behaviors in the ANT controllers.

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press

Table A1. Sensor-behavior detectors.

Detector	**Name**	**Logic**
1	Level	If $B_1 =$ Level and $L_1 > 0$ and Move Forward $= 1$
2	Collision avoidance	If $S_1 = 1$ and Move Forward $= 0$ and Random Turn $= 0$ and (Turn Right $\neq$ Turn Left or Move Backward $= 1$)
3	Stuck avoidance	If $U_1 = 1$ and Move Forward $= 0$ and (Move Backward $= 1$ or Random Turn $= 1$ or Turn Left $\neq$ Turn Right)
4	Cut dig	If $B_1 =$ Below and Move Forward $= 1$
5	Correct dump	If $Z_2 =$ Dump and $Z_3 =$ Dump and $L_1 > 0$ and Random Turn $= 0$ and Move Forward $= 1$ and Turn Left $=$ Turn Right or If $Z_2 =$ Dump and $Z_3 =$ Dump and $L_1 > 0$ and Random Turn $= 1$ and Random Turn $\longrightarrow$ Turn Left and Turn Right $= 1$ and Turn Left $= 0$ and Move Forward $= 1$ or If $Z_2 =$ Dump and $Z_3 =$ Dump and $L_1 > 0$ and Random Turn $= 1$ and Random Turn $\longrightarrow$ Turn Right and Turn Right $= 0$ and Turn Left $= 1$ and Move Forward $= 1$

https://doi.org/10.1017/S0263574717000017 Published online by Cambridge University Press