mathematics

Article

Emergent Intelligence via Self-Organization in a Group of Robotic Devices

Konstantin Amelin [1,2], Oleg Granichin [1,2], Anna Sergeenko [1,2,*] and Zeev V. Volkovich [3]

[1] Faculty of Mathematics and Mechanics, Science Educational Center "Mathematical Robotics and Artificial Intelligence", Saint Petersburg State University, 199034 St. Petersburg, Russia; k.amelin@spbu.ru (K.A.); o.granichin@spbu.ru (O.G.)
[2] IPME RAS, 199178 St. Petersburg, Russia
[3] Software Engineering Department, ORT Braude College, Karmiel 21982, Israel; vlvolkov@braude.ac.il
* Correspondence: a.sergeenko@spbu.ru

Abstract: Networked systems control is a known problem complicated because of the need to work with large groups of elementary agents. In many applications, it is impossible (or difficult) to validate agent movement models and provide sufficiently reliable control actions at the elementary system components level. The evolution of agent subgroups (clusters) leads to additional uncertainty in the studied control systems. We focus on new decentralized control methods based on local communications in complex multiagent dynamical systems. The problem of intelligence in a complex world is considered in connection to multiagent network systems, including a system named airplane with feathers, load balancing, and the multisensor-multitarget tracking problem. Moreover, the new result concerning the emergency of intelligence in a group of robots is provided. All these methods follow the paradigm of the direct reaction of each element (agent) of the system to its sensory data of current situation observations and the corresponding data from a limited number of its neighbors (local communications). At the same time, these algorithms achieve a mutual goal at the macro level. All of the considered emergent intelligence appearances inspire the necessity to "rethink" the previously recognized concepts of computability and algorithm in computer science.

Keywords: emergent artificial intelligence; multiagent technologies; consensus; robot

Citation: Amelin, K.; Granichin, O.; Sergeenko, A.; Volkovich, Z.V. Emergent Intelligence via Self-Organization in a Group of Robotic Devices. *Mathematics* **2021**, *9*, 1314. https://doi.org/10.3390/math9121314

Academic Editor: António M. Lopes

Received: 16 April 2021
Accepted: 1 June 2021
Published: 8 June 2021

1. Introduction

Consistently appearing in the modern age, multifaceted problems encourage reconsidering the role of computer-based tools and methods in various applications. Many contradictory purposes and approaches are naturally being combined, in this connotation, giving rise to the need to establish novel mobile artificial intelligence generation. Smart embedded computing devices will soon be able to resolve many tasks not imagined in the recent past. Thus, two main directions in artificial intelligence evolution crystallize in the late twentieth century. There are intentions to construct methods aiming to attain a specific result and the brain activity modeling within generalization and self-organization in social communication.

However, the traditional perception of the computer structure is continually changing. These circumstances lead to revolutionary transformations in the application and programming of computational devices. This innovative computing paradigm initiates changes in the computer's architecture toward concurrent asynchronous interacting dynamical systems (functional elements). Stochastic, hybrid, asynchronous, and cluster types of system conduct become more evident and dominant with the absence of rigid centralization and dynamic classification of related models.

Elementary computational elements (gates) are currently approaching the atom scale in their sizes, so that at this gradation, the quantum laws become increasingly crucial. Simultaneously, due to Heisenberg's uncertainty principle, precise answers about a current system state cannot be principally reached. Therefore, the upcoming computing system hybridization is understood as a necessity to operate with combinations of continuous and discrete processes, including considering the evolution of the continuous patterns and possible switch from one model to another.

Noticeable computing devices' performance and miniaturization unavoidably suggest dealing with "transient" processes. A crucial limitation of the classical computation model is a partition into isolated bits of the memory.

Principally, reducing the clock cycle time (strobe impulse) and the distances between bits makes it impossible to isolate bits due to the quantum mechanical laws [1]. It looks natural, in the future, to switch from primitive operations with the classical bits to operations outlined by specific micro-dynamic models operating with the related fragments. A usage denial of unified, simple computing elements leads to problems with performance of different stage-managed components of significantly distinct physical characteristics. This single "tact" event is expressed as unsolvability of the continuum Zermelo—Fraenkel hypothesis [2].

Model grouping in the known multiagent systems consuming the agents' connections topology time evolution is an incredible tool for studying complex stochastic systems. Here, the agent notion may correspond to some dynamical model (a system component) or a specific set of models. Without rigid centralization, these structures can successfully treat complex problems by splitting them into modules reallocating the agents' dynamical partitions. Such a system can effectively perform under significant uncertainties, demonstrating so-named "emergent intelligence". This notion signifies an intellectual resonance or a swarm intelligence consisting of the manifestation of unexpected properties of the whole system not inherent to any simple element.

The critical feature of emergent intelligence consists of the dynamics and unpredictability of the decision-making process. In practice, this means that the solution is achieved at the expense of hundreds and thousands of untraceable interactions. The desirable protocols are generated and executed by the agents themselves. At each step, they ponder the system inputs and respond to unpredictable events such as delays and crashes. Thus, emergent intelligence is a new superior unique factor arising, as it were, "out of thin air" due to many hidden or explicit conditions spontaneously appearing and disappearing in the system.

The rest of this paper is organized as follows. Section 1 provides a review of related works. The main idea behind this paper and the correlation between intelligence and multiagent network systems are introduced in Section 2. Some examples of multiagent network systems are shown in Section 3. The main new result concerning the emergency of intelligence in a group of robots is provided in Section 4. Section 5 concludes the paper.

Related Works

The phenomenon of the emergence of order from chaos is a popular research field. Due to the development of computer technology, the topic has actively been studied within the problem of mobile group interaction (particularly, a group/swarm of robots) [3]. One of the most critical issues in the communication systems theory is picking or creating a procedure capable of adapting to a permanently changing environment using centralized and decentralized algorithms. Problems of control and distributed interaction in dynamic network systems have attracted significant attention in the last decade due to the widespread applications of the multiagent systems methodology in fields such as control [4,5], distributed sensor networks [6], and mobile robotics [7]. Unlike the classical attitude, when a search is implemented for some (deterministic) algorithm discovering the best solution, in multiagent technologies the solution is automatically obtained as a result of the interactions of many independent targeted software modules (agents). The traditional methods are often inapplicable in actual practice, such as resolving the enterprise management problem in an unpredictable dynamic environment of modern business, requiring a huge settlement volume. On the whole, distributed systems are increasingly used parallelly, dividing a batch of tasks between several computing threads (devices) [8–11]. Moreover, real-world problems lead to restrictions on communications channels in the strategies intended to solve this type of problem effectively [12].

The intelligence needed for solving specific problems interpreted as "complexity" is considered in our model according to [13] in two aspects. The first one is related to subjective complexity appearing in a person's mind due to its limited abilities for the perception of the actual problems. The factual complexity causes the second one. These types are inherently different but can often coexist together, leading to distinguishing, sometimes contradictory, interpretations of complexity and complex systems apportioning, for example, the "ontological" and the "semiotic" simplicity-complexity. The "ontological" one is more associated with the complexity of the material (physical) world.

The relationship between the mentioned types of complexity determines the systemic continuum in which the concept of "intelligence" is generated, so that the traditional interpretation of intelligence as a sum of locally owned cognitive apparatus of a person abilities (instrumental capabilities of an artificial system in case of artificial intelligence) has to be changed by general integral properties, conditioned by the environment and the system operating. Indeed, the functions and abilities of a system existing and coexisting in an environment can be useless and may be harmful in another environment. Intelligence is determined, to a large extent, by the control degree on a system in a particular environment.

2. Motivation

The present vital tendencies in computer technologies suggest the permanent development and application of artificial intelligence methods. The people's way of thinking is currently radically different from that of a computer strictly following a predefined arrangement of operations. This circumstance means that a human can behave in problematic situations adequately and adapt to the environment changing through active imagination and "to learn from mistakes". Such as for an ant colony, typical swarm behavior can clarify new approaches connected to multiagent technologies and randomized algorithms. On the other hand, every single ant does not demonstrate smartly proper functioning, but an ant colony exhibits surprising performance considered to the same extent as intelligent. As was mentioned earlier, such manifestation of emergent intelligence expresses a system's unexpected properties as a whole but not features of its elements. This summarizing effect of "intellectual resonance" is called "swarm intelligence" [14], where an agent group is comparable to a team composed of members who cooperate in the decision-making process.

Consider the ant algorithm suggested in [15]. Suppose a bounded area containing the anthill and the food source and deem, for simplicity, a hypothetical colony of "artificial ants" composed of two ant types. The first one, "the mother", is responsible for the general strategies and producing new worker artificial ants. In the second worker ants category, they strive for two objectives: discovering food and bringing it home. The food delivery process is crucial in the colony living processes. Traditionally, in a centralized approach (the primary device), some artificial super brain controls everything and everyone. If it has super abilities, it is possible to exploit space satellite data to allocate the delivery. As a result, the described system could be too complicated and expensive. Although there is another significant drawback. If something went wrong during the mission, the strategy must be promptly updated that suggests involving substantial computational resources.

In this regard, it seems likely to enquire if it is feasible to build a much simpler system resolving the assigned task without attributes such as space satellite data, supercomputing, maps, GPS navigation, and actual communication with the super brain? Sure, we desire that this structure stills perform well even under future unforeseen conditions. A multiagent approach could be a "cheaper" version of such a system. In this connotation, "the mother" is responsible for producing the worker ants without being the super brain, and each one of the worker ants is responsible for at least two missions:

- to find food;
- to bring it to the anthill.

It is additionally assumed that we study a grid area with reasonably small square cells such that a worker ant can relocate just in one unit at any given time at the same instance as

- marking the path with pheromones;
- discovering neighboring cells marked by the freshest pheromones.

This model operates with two kinds of pheromones. The first type marks cells during moving out without food, while the second during moving out with food. Such a system appears to be much cheaper and more feasible than the centralized one, being semi-optimal under unpredictable uncertainties [15]. A crucial circumstance is that there is no need to wait for an ant returning with information on the nourishment location because we assume that the information comes immediately back via the created by the ant chain.

Let us suppose that a new way of food supply is needed since the anthill neighborhood is destroyed through some catastrophic event. A strategy intended to resolve this problem can consist of ants spreading in all possible directions chosen, for example, randomly (a randomized strategy). Let us explain an example shown in Figure 1. Line A shows the situation when there are no obstacles on the way from the anthill to the food. On line B, the barrier (obstacle) appears. For example, it can be a fallen stone. In this case, ants start to avoid the obstacles in two ways, as shown in line C. After some time, according to the algorithm described in [15], all ants start to move only as is illustrated in line D since it is the shortest way. This scheme makes it possible to discover and optimize new food paths. Thus, the system possesses emergent intelligence properties.

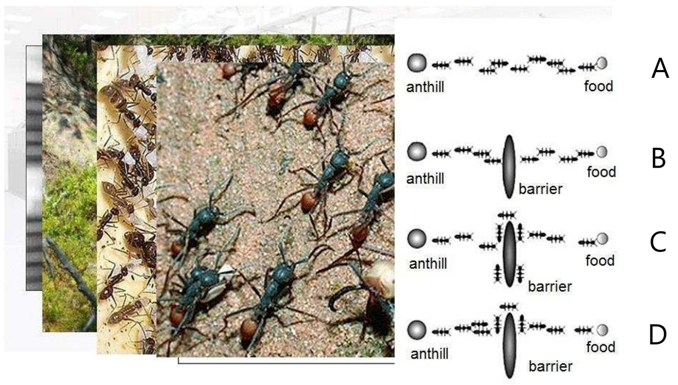

Figure 1. An illustration of an ant colony conduct. Lines A, B, C and D show different time stages.

Multiagent behavior can also manifest itself in the communities of other organisms, i.e., bacteria. A recent scientific breakthrough revealed (see, for example, [16]) that bacteria can interconnect with each other. This fact has radically changed the general perception of the existence of the simple organisms inhabiting the world. Bacteria apply signaling molecules to measure population concentration. Nowadays, the term "quorum sensing" (QS) describes the process of signaling molecules, allowing a single cell to sense bacteria's cell density. In nature, different kinds of bacteria located in the same environment use diverse signaling molecules that make it possible to converse with other various types of bacteria. Today, these quorum-sensing systems are intensively studied for various bacteria categories. Recently, extraordinary advances in understanding the genetics, genomics, biochemistry, and signal diversity of QS were achieved, including information about the connections between QS and bacterial sociality [17]. The behavior and evolution of these bacteria communities are comprehended as natural examples of multiagent systems because the interactions between bacteria occur locally and bacterial density can be measured with no need to gather the whole data in one data center. These bacteria solve the global task (to measure the density of the population) using only local communications (QS) and, overall, this is an example of multiagent technologies.

Another essential application of self-organization arises in DNA (deoxyribonucleic acid) computing. Aiming to explain the concept of DNA computing, we consider a known combinatorial problem of finding a Hamiltonian path in a graph. As is well known, the Hamiltonian path problem consists of discovering a path in an undirected or directed graph precisely visiting each node once and starting with a given vertex and ending with another specified one. The most popular branch-and-bound method can hardly be used in many situations due to its exponentially growing complexity. The DNA computations based on the principles of self-organization inherent in nature [18] are able to resolve such problems in linear time-consuming.

We remind that the DNA molecules usually form a double helix consisting of nucleotides containing a phosphate group, a sugar group, and a nitrogen base. There are four different nucleotides: adenine (“A”), thymine (“T”), guanine (“G”), and cytosine (“C”). In the helix, the nucleotides join in pairs according to a fundamental complementarity rule: “A” is always opposite to “T”, and “G” is always opposite to “C” [19]. DNA computing allows modeling in a biological laboratory dealing with double-stranded DNA, which is able to “split” into two separate strands (by heating) or to do the reverse action annealing with the help of the particular enzyme *ligase*.

Let us consider a directed graph $G = (\mathcal{N}, E)$ with $|\mathcal{N}| = n$ nodes. Random DNA sequences consisting of 20 nucleotides represent nodes (vertexes). An edge is associated with DNA sequence obtained as concatenation of the complement of the second half of the starting node and the complement of the first half of the finishing node.

For example, if the first node has the DNA code

$$\begin{gathered} TCAGTACCAG \quad TACAGTCACA \\ complement : (AGTCATGGTC \quad ATGTCAGTGT), \end{gathered}$$

where A is adenine, T is thymine, G is guanine, C is cytosine, the second node has the DNA code

$$\begin{gathered} TAGGTATGCT \quad CAGATAAAGG \\ complement : (ATCCATACGA \quad GTCTATTTCC), \end{gathered}$$

and there is an edge from the first node to the second, then that edge is coded by

$$ATGTCAGTGT \quad ATCCATACGA.$$

This procedure is repeated with every available edge. Note that the directions of DNA strands are ignored for simplicity.

Further, all described DNA representations are created in a lab and mixed in a single annealing reaction connecting the molecules according to the complementary rule. All available pathways are simultaneously in parallel created in one probe. An example of a short molecule is shown in Figure 2.

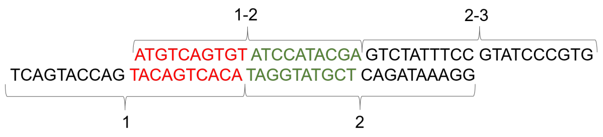
1-2
2-3
ATGTCAGTGT ATCCATACGA GTCTATTTCC GTATCCCGTG
TCAGTACCAG TACAGTCACA TAGGTATGCT CAGATAAAGG
1
2

Figure 2. An example of a molecule after annealing.

An analogy with multiagent technologies is undoubtedly evident in this process. By forming strands and units into massive DNA molecules, as if they were agents within a cascade process, nucleotides provide a solution to the problem. The subsequent steps intend to check if the molecules encoding the Hamiltonian path are within the strand or not. The time consumption of DNA computing results from local interactions and clustering growing linearly as the number of nodes. Subsequently, in [20], this characteristic is compared with the same one appearing in the branch-and-bound method. It turns out that DNA computing performs much better for the adjacency matrix sparsity parameters equaling 0.85, 0.9, and 0.8 and the number of graph nodes arising from 37, 43, and 45, respectively.

3. Multiagent Network Systems

Creating an emergent intelligence system suggests many efforts compared to traditional elaboration, but its maintenance is significantly cheaper. Moreover, the multiagent systems are considerable reliable because being significantly decentralized across the strong hierarchy organized traditional architecture. This fact causes priority for their massive applications in the business area, where the "actors" theory has appeared as an essential alternative to the conventional business models.

This section considers several illustrative dynamic network systems composed of intelligent collaborating sensors (agents). Without loss of generality, agents in the network system are numbered. Given a network consisting of n nodes, let the interaction between nodes be described by the directed graph $\mathcal{G}_A = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N} = \{1, \dots, n\}$ is a set of nodes (vertices) and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ is a set of edges. Denote by $i \in \mathcal{N}$ an identifier of i-th node and $(j, i) \in \mathcal{E}$ if there is a directed edge from node j to node i. The latter means that node j is able to transmit data to node i. For a node $i \in \mathcal{N}$, the set of *neighbors* is defined as $\mathcal{N}^i = \{j \in \mathcal{N} : (j, i) \in \mathcal{E}\}$. The *in-degree* of $i \in \mathcal{N}$ equals $|\mathcal{N}^i|$. Here and after, $|\cdot|$ is the cardinality of a set, and the identifier of i-th node is used as a superscript and not as an exponent. Let $\forall i \in \mathcal{N}\ \bar{\mathcal{N}}_t^i \subset \mathcal{N}^i$ be a subset of agents, which send information to agent i at time instant t (its neighbors). The complexity of all presented methods is proportional to the cardinality of this subset of agents $|\bar{\mathcal{N}}_t^i|$.

3.1. Airplane with Feathers

Control of turbulence is a known hardly treated problem. From a conventional standpoint, this problem appears to be very difficult because unpredictable changes in the current environment do not allow application control action prepared during a long process. An alternative framework consists of using a set of agents as an investigation system possessing complexity comparable to one of the studied phenomena. This section aims to demonstrate that adaptation in complex systems to external disturbances can be made similarly to multiagent systems with internal self-organization. This approach is illustrated with a special experimental testbed. Similarly to the earlier anthill model, there is no central powerful computational unit (super brain), and each element is a cheap computational unit communicating only with its neighbors.

Starting to deal with discussed in this subsection problem, we firstly cover the airplane wings with hundreds of small rotating units endowed by pressure sensors. This idea is bio-inspired cause all birds have feathers and all sharks have similar scales. The system's key feature is a massive amount of local feedback where each feather's orientation changes according to comparison with pressure deviations of neighbors. If the sum of pressure moments deviations of neighbors is positive, then the feather rotates, increasing the pressure; otherwise, it rotates to decrease pressure.

Initially, all plates are under the same pressure. If turbulence occurs, then they undergo different wind pressure. To solve the problem of equating the influence we apply a so-called Local Voting Protocol to disturb forces on different plates of the wing and transforming airplane flow to a mode close to laminar. It has a multiagent structure and processes in a real-time fashion.

Let x_t^i be the deviation of the integrated pressure force on the "feather" a^i, $i \in \mathcal{N}$, at time instant t. Each "feather" at time instant t sends the following information about its state to the neighbors $\bar{\mathcal{N}}_t^i$:

$$y_t^i = x_t^i + \xi_t^i, \tag{1}$$

where ξ_t^i is a noise.

Consider the Local Voting Protocol. Here, the control for each node is defined as a weighted sum of the differences between information about its state and information about the state of its neighbors:

$$u_t^i = \alpha \sum_{j \in N_t^i} b_t^{i,j} (y_t^j - y_t^i), \tag{2}$$

where α is the step-size parameter, $b_t^{i,j}$ are the weights. The consensus is reached when $x_t^i \approx x_t^j$ for all $i, j \in \mathcal{N}$.

Each feather gets the data from its own pressure sensor and communicates with neighbors receiving information about their plates' current pressure. Then it compares its own pressure with pressures on neighbors. If its pressure is less than the summary comparison, the feather changes its angles to increase the pressure. Otherwise, the angles are changed to decrease the pressure. As a result, the whole system comes to a consensus [21].

In the intermediate stages, clusters with similar behavior of the feathers are formed. Even if the system cannot come to a consensus, cluster control is still a much easier task than managing a considerable array of heterogeneous elements. Many simulations show the typical process of equalizing the pressures [22].

3.2. Load Balancing

Multiagent technology allows solving, on a finite time horizon, a typically hard problem of load balancing. In a huge network, multiagent technology can resolve this problem reasonably compared to the traditional mathematical optimization tools consuming Local Voting Protocol. Let $\mathcal{N} = \{1, \ldots, n\}$ be a set of agents (nodes) with states

$$x_t^i = \frac{q_t^i}{p_t^i}, \tag{3}$$

where q_t^i is a task queue length of agent i, $i \in \mathcal{N}$ and p_t^i is an agent i productivity.

The dynamics of each agent i, $i \in \mathcal{N}$ is described by

$$q_{t+1}^i = q_t^i - p_t^i + z_t^i + u_t^i, \tag{4}$$

where z_t^i is the new job received by agent i at time t, u_t^i is the result of jobs redistribution between agents, which is obtained by using the selected protocol of jobs redistribution. Each agent $i \in \mathcal{N}$ at time t can receive noisy observations about its own and its neighbor's queue length:

$$y_t^{i,j} = q_t^j + \xi_t^{i,j};\ j = i \text{ or } j \in \mathcal{N}_t^i, \tag{5}$$

where $\xi_t^{i,j}$ are noises. The agent also receives information about productivity p_t^i and about its neighbors' productivities p_t^j, $j \in \bar{\mathcal{N}}_t^i$. The objective is to balance the load such that the overall implementation time $T_t = \max_{i \in \mathcal{N}} x_t^i$ can be minimized.

An approximate consensus problem in a decentralized stochastic dynamic network with incomplete information about the states of nodes, delaying in measurements, and a changing connections structure is discussed in [23]. A solution can be found there using the mentioned earlier protocol with non-vanishing steps. In a situation of a switching topology, noisy, and delayed information about agent states, this protocol is also suitable. In [6], the problem of optimizing the protocol parameters is studied. In [23], general theoretical results are applied to the load balancing problem in stochastic dynamic networks with incomplete information and changing configuration. There, a network of 1024 agents connected in the circle is considered, and the number of incoming jobs. An agent's assigning of the next job is performed randomly following the uniform distribution. In addition, there are n (number of all nodes) randomly changing connections between agents on each iteration. The performance of the adaptive multiagent strategy with the redistribution of jobs among "connected" neighbors is significantly better than the performance of the strategy without redistribution.

The optimal distributed node scheduling in a multihop wireless network is discussed in [24]. The proposed algorithm tries to semi-equalize the load (defined as the ratio of the queue length over the number of allocated slots) through slot reallocation based on local information exchange. The approach stems from the fact that the shortest delivery time or delay occurs for the load being semi-equalized throughout the network. It is also demonstrated that with Local Voting, the network system converges asymptotically toward the optimal scheduling. Simulation results show that the proposed algorithm achieves better performance than other distributed algorithms in terms of average delay, maximum delay, and fairness [24].

3.3. Multisensor-Multitarget Tracking Problem

Consider a distributed network system of n sensors collaborating to track m targets. Decentralized algorithms have many advantages over centralized algorithms, especially when agents only can exchange information locally with their neighbors [25]. The main difference is that decentralized algorithms assume only local interaction between sensors, while the centralized algorithms require governing data to a fusion center. If positions of all m targets need to be computed, then $m(d-1)$ measurements have to be collected simultaneously, where d is the space dimension. Such a procedure is often impractical. There is no mutual data processing or communication center in a decentralized approach, so the information exchange and all computations are performed in a decentralized way (Figure 3). However, besides topological constraints (each sensor can interact only to a few adjacent nodes of a network), communication between sensors can be restricted due to, e.g., the limited capacity of communication channels, delays, and data distortions. A possible solution to the multisensor-multitarget tracking problem, in this case, is to apply a distributed stochastic approximation procedure to track the targets and Local Voting Protocol to exchange the target position assessments [6]. Such an approach appears to be workable even in the case of random, independent choice of sensors tracking at a particular time instant.

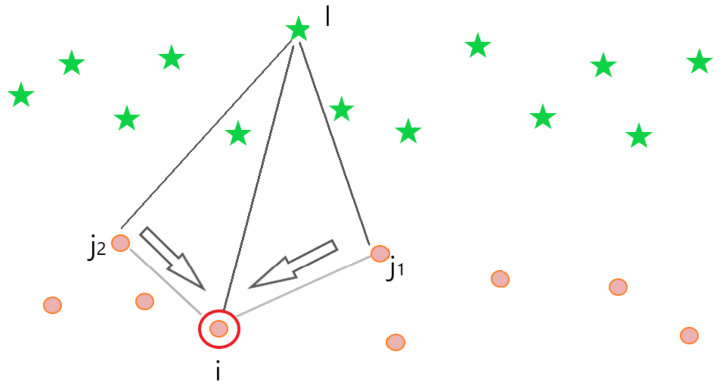

Figure 3. Sensor $i, i \in \mathcal{N}$ estimates the state of target l at time instant t. The sensor also collects the distances to the same target measured by its neighbors $j_1, j_2 \in \bar{\mathcal{N}}_t^i \subset \mathcal{N}^i$.

3.4. Synchronization in Networks of Kuramoto Oscillators

Yoshiki Kuramoto proposes a straightforward yet versatile nonlinear model [26]. This approach appears to be a successful tool for an appropriate description of various biological systems [26] and robotics [27] describing oscillatory dynamics of coupled oscillators, affecting each other. Dynamics a network of n agents with one degree of freedom the following describes each one (often called a phase of an oscillator) system of differential equations:

$$\dot{\theta}_t^i = w^i + \sum_{j \in \mathcal{N}^i} K_{i,j} \sin(\theta_t^j - \theta_t^i), \tag{6}$$

where θ_t^i is a phase of agent $i, i \in \mathcal{N}$, $K_{i,j}$ is a weighted adjacency matrix of the network, and w^i is a natural frequency. Agents have to synchronize their frequencies ($\dot{\theta}_t^i = \dot{\theta}_t^j$, $\forall i, j \in \mathcal{N}$) or phases ($\theta_t^i = \theta_t^j$, $\forall i, j \in \mathcal{N}$) under certain conditions on w^i and $K_{i,j}$.

A new approach proposed in [28] to treat complex multiagent systems makes it possible to analyze processes such as local agent interactions synchronization and clustering from a new standpoint. For example, mesoscopic control inputs for clusters associate with an algorithm of microscopic local interactions between agents, described by the Kuramoto model [28].

Agents' group synchronization arises during their local interactions. A clusterization process acts much faster in comparison to the usual system lifecycle when the control objectives are established by the system behavior within considerable time intervals. If these periods are longer than the time needed for partition forming, then the created clusters can be considered as new meso-scale objects with dynamics in the so-called "slow time", ignoring the short-term effects. These objects are called "mesoscopic" because their live scale lies between the entire system (macro-level) and the individual agent's (micro-level) time levels [28]. Comprehensive models of complex systems with many simple components (agents) are hardly controlled because of their large dimension and technological obstacles arising in this context of barely resolved tasks. A system cannot suitably track the individual agents' traffic due to their small sizes and the high-frequency control effects. New mesoscopic variables (clusters) can work with smaller amounts of inputs, assigning them separately to each cluster.

4. Robots Communication without Routing

The central new upshot of this paper is a novel decentralized algorithm developed in the general frame of the emerging intelligence, where robots as a whole system avoid obstacles without gathering data about the whole field. Each one of these robots is a simple device with a limited number of technical tools. Control of the robot community is based on algorithms implementing group movement in an unspecified area, e.g., locations and shapes of obstacles are unknown in advance together with a possible direction. There is no camera with a general view and no traditional navigation system for motion tracking.

Self-organization of the system is provided by applying computationally efficient autonomous navigation algorithms (such as the consensus Local Voting Protocol) [29]. A vital ingredient is a paradigm suggesting an immediate response of a robot (agent) to the data about the current situation collected by itself and a sufficiently small number of its neighbors (local communications). At the macro level, these algorithms can guarantee, in some extension, achieving the overall purpose.

We study problems of space robots aligning without traditional navigation sources. A typical framework suggests increasing the number of sensors aiming to gather more relevant information. It obviously leads to more complicated arrangements assuming meaningful implementation outlays.

4.1. Swarm of Wheeled Robots

We consider a group of wheeled robots with the mission of moving to a given light source when each robot has only two driving wheels and one ball bearing capable of moving forward, backward, left, right, rotate in place to the left and the right. It also has four light sensors, one rangefinder observing the front side towards the movement direction, two cheap microcomputers, and a communication unit (see Figure 4). The robot's movement is implemented by the rotation of the motors set by the control signal. The robot's length, height, and width are less than 10 cm each.

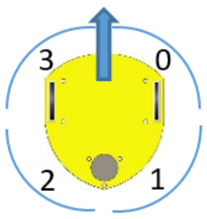

Figure 4. Scheme of a robot with 3 wheels, 2 servo motors connected to two wheels, 4 light sensors, 1 distance-sensor observing the front side towards the movement direction, and 1 magnetometer.

Missions are performed as follows.

1. In the beginning, all robots stand along one wall in a room where obstacles with a height exceeding three times the height of the robots are randomly placed.
2. One or two light sources are located at other walls, and the robots start to move in the direction of a light source when one of them turns on. If there are two light sources, they are never turned on at the same time: if one is turned on, the other immediately turns off.
3. Each robot has a collision-avoidance system. Robots start splitting into groups when they try to overcome obstacles (see Figure 5).
4. The direction of moving may be changed if one light source turns off and another one turns on.
5. The mission completes when all robots gather near a luminous light source.

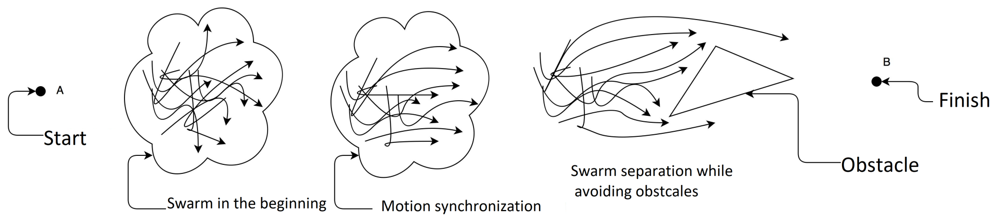

Figure 5. Movement of a swarm from point A to point B using the Local Voting Protocol.

Each robot has no information about its and its neighbors' locations in any global coordinate system. A robot has its own local orthogonal coordinate system with the origin at the center of mass, and the first axis is directed along with the movement of the robot. The second axis is orthogonal to first one and directed from left to right. The last axis is orthogonal to the plane defined by two first axes and has direction to up. The location of the four light sensors in the base plane is given in Figure 4. A distance sensor is situated at the front of a robot, measuring a distance to obstacle with the first axis when it is in a range from 10 to 100 cm. The magnetometer of each robot determines the angle between the first axes and the direction to the Nord.

As before, denote by $\mathcal{N}_t^i$ a set of all neighboring agent of an agent i at time instant t and by $\bar{\mathcal{N}}_t^i$ ($\bar{\mathcal{N}}_t^i \subset \mathcal{N}_t^i$) the set of neighbors whose data the robot i will receive over the network. The maximal number of elements $\bar{\mathcal{N}}_t^i$ is selected based on the performance of the data transmission module, but it cannot be more that $\mathcal{N}_t^i$. If $|\bar{\mathcal{N}}_t^i| = |\mathcal{N}_t^i|$, then the data of all neighbors are taken into account; if $|\bar{\mathcal{N}}_t^i| < |\mathcal{N}_t^i|$, then the neighbors are chosen randomly. There are several algorithms intended to the searching technique used to detect and select the nearest neighbor (see [30]). In this work, we apply the following approach employing the strength of the received signals:

1. sort all visible signals according to the power level from highest to lowest;
2. cut off those neighbors whose signal strengths are lower than a certain threshold;
3. select the prespecified number of neighbors randomly;
4. receive the data from these neighbors.

If there is no light source, then robots do not move. Otherwise, each robot travels with a given speed in the source direction, intending to reach it, bypassing encountered obstacles. More precisely, let $d_{\min} > 10$, $d_{\max} < 100$ and be fixed numbers, $\alpha \in [0,1]$ and $\beta \in [0,1]$ be common predefined parameters of navigation protocols for each robot, and ϕ be the operator of vector rotation by angel 0.1π. The distance-sensor measures the distance d_t^i to the nearest obstacle (along the axis of the motion), and the light sensors provide the data $l_t^{i,p}, p = 0,1,2,3$, about illuminations. Each light sensor is oriented to the direction of corresponding corners and has a range 0.75π. At each time instant t, robot i calculates a path direction vector $\mathbf{s}_t^i$ to the light source by light sensors data and magnetometer data (azimuth—the angle between the first axes and direction to the Nord). Let m_0, m_1, m_2, m_3 be an order of the light sensors $p = 0,1,2,3$ according to decreasing order of light power $l_t^{i,p}$.

Path direction $\mathbf{s}_t^i$ is calculated by the magnetometer data and the proportional difference between two largest values of light values (l_t^{i,m_0} and l_t^{i,m_1}) with the corresponding shift to the sector defined by m_0 and m_1. For example, if the most significant light values are equal, then the robot moves straightforward.

The weighted actual path direction vector $\mathbf{r}_t^i$ of robot i at time instant t defines with a weight equals the mean of the rotation speed of two wheels. Depending on the distance to the obstacle d_t^i and the front light sensors values $l_t^{i,0}$, $l_t^{i,3}$, each robot i computes its own confidence parameter q_t^i (a potential of its chosen direction):

$$q_t^i = (1-\beta)\left(1 - \left|\frac{l_t^{i,0} - l_t^{i,3}}{l_t^{i,0} + l_t^{i,3}}\right|\right) + \beta\frac{\min\{d_t^i, d_{\max}\}}{d_{\max}}, \tag{7}$$

which indicates the influence of the data light and range sensors for confidence. The case $\beta = 0$ excludes the impact on the certainty of the data of obstacle sensor, and the case $\beta = 1$ excludes the data of light sensors. Weighted actual path direction $\mathbf{r}_t^i$ and confidence q_t^i are state parameters of robot i, which it broadcasts in the network. Robot i receives from neighbors their states $\mathbf{r}_t^j$, q_t^j, $j \in \bar{\mathcal{N}}_t^i$. Based on the whole obtained information, the robot chooses the next movement direction $\mathbf{r}_{t+1}^i$, which is selected according to one of three possible scenarios depending on the value d_t^i of the distance sensor, which are listed below.

1. If $d_t^i \geq d_{\max}$, then there are no visible obstacles along the path direction, and the final path direction is calculated as follows:

$$\mathbf{r}_{t+1}^i = \mathbf{s}_t^i(1 - \delta_t^i\gamma_t^i) + \alpha\frac{\gamma_t^i}{|\bar{\mathcal{N}}_t^i|}\sum_{j\in\bar{\mathcal{N}}_t^i}\mathbf{r}_t^j q_t^j - \mathbf{r}_t^i q_t^i, \tag{8}$$

   where
$$\delta_t^i = \frac{\alpha}{|\bar{\mathcal{N}}_t^i|}\sum_{j\in\bar{\mathcal{N}}_t^i} q_t^j - q_t^i,\ \gamma_t^i = 1/(q_t^i + \delta_t^i).$$

2. If $d_{\min} < d_t^i < d_{\max}$, then robot i makes correction rotation R by $\pm\phi$ to previously computed $\mathbf{s}_t^i$. The sign of rotation is selected based on appropriate comparisons light sensors data. After that, the new path direction $\mathbf{r}_{t+1}^i$ is computed by the same formula (8) with updated value $\mathbf{s}_t^i := R\mathbf{s}_t^i$.
3. If $d_t^i \leq d_{\min}$, then the robot goal is to find a direction of movement in which $d_t^i > d_{\min}$, and it does not use data from neighbors and light sensors. In this mode, the robot randomly selects the direction of rotation and rotates in place until the distance $d_t^i \leq d_{\min}$ is more than $d_{\min}$.

The described algorithm demonstrates that each robot operates according to a well-defined procedure without knowing its location and the location of its neighbors in the global system. A robot is only aware of its neighbors' states, their path direction, and confidence correcting their own path direction resting upon these data. An idea of this algorithm is shown in Figure 6. An example of three robots group exhibits that the robot moving last bypasses obstacles along the optimal route. Note that it does not recognize these obstacles but is guided only by its neighbors' data (Figure 6).

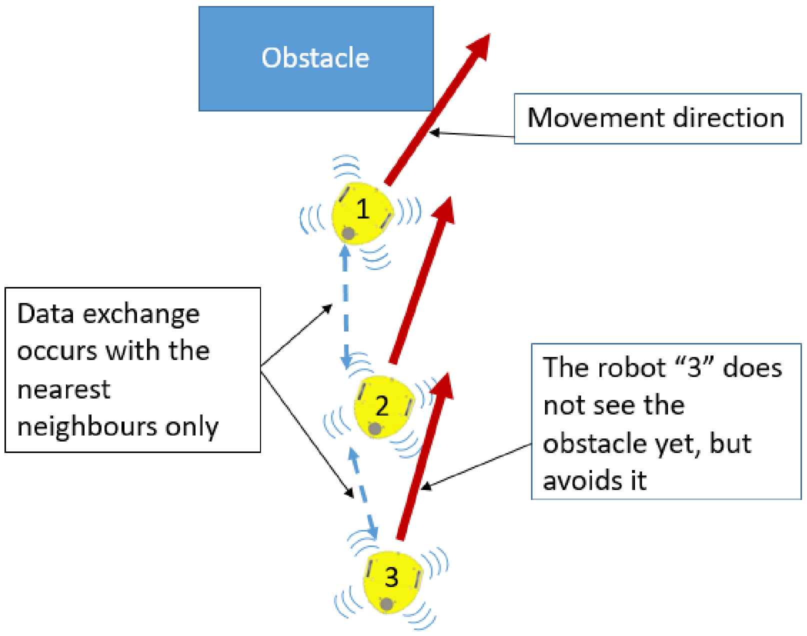
Obstacle
Movement direction
1
2
3
Data exchange occurs with the nearest neighbours only
The robot "3" does not see the obstacle yet, but avoids it

Figure 6. Example of three robots group movement.

4.2. Simulation Results

The system is simulated in the Webots simulator [31] on a laptop with an AMD Ryzen 5 4600HS processor, Geforce 1650 graphics card, 4 GB memory, 24 GB RAM, and SSD.

There is an area of 40 $\times$ 22 squares (rectangle cells) and one light source located at its border (Figure 7). A robot fits one square. Two obstacles are randomly placed on the field. The height of these obstacles exceeds the size of a single robot by 5 times, and the width by 9 times. Obstacles cover 50% of the sight line of the light source.

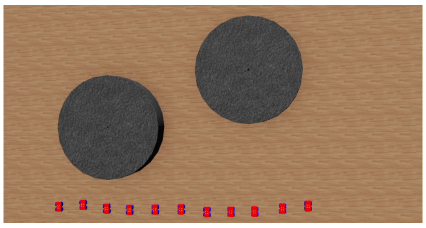

Figure 7. Initial simulation state. Robots are depicted as red rectangles, obstacles as gray circles.

A group of 10 identical robots is considered. Each robot either stands still or moves at a given speed $v_i(t)$ measured as the number of squares per unit time. The speed $v_i(t)$ changes by ± 1 per clock cycle and can reach a maximum value of 2. The number of the neighbors at each time instant t $|\bar{\mathcal{N}}_t^i|$ is 5. The aim of each robot is to move towards the light source, avoiding collisions with obstacles.

The uniform robot controller algorithm presented in Section 4.1 is implemented in the Python programming language. The same type of controller is used for all robots. As was said before, robots do not know their location and the configuration of the obstacles and receive data of their nearest neighbors' states. Data transmission in a group is implemented using a receiver/transmitter system. To control the number of neighbors at receiving devices, the visibility zone is limited to 2 squares. The obstacles are observable at a distance of 3 squares.

The scenario for the experiment is as such.

1. Robots stand in a row at the same wall.
2. One light source is turned on the opposite side.
3. Robots start moving towards the light source and avoid obstacles.
4. The experiment is finished when all robots reach the opposite wall.

The step-size parameter α plays an essential role in the algorithm. Therefore, we check its possible values lying in the interval $[0, 1]$ with step 0.05 and start point 0 and evaluate the appropriate times needed for all robots to reach the opposite wall. For the purity aim, for each given value of α, ten experiments are provided, and the average time is calculated.

Figure 8 shows the simulation results with different values of α. $\alpha = 0.8 \pm 0.05$ can be recognized as the optimal value. During the simulation, it is found that at high values of the parameter α, the robots begin to rely too much upon the noisy data received from their neighbors and even turn away from a light source. At lower values of α, the robots are guided mostly by their own sensors and choose non-optimal trajectories to avoid obstacles. As a result, they do not benefit from working in a group. At the optimal value, the robots do not interfere with each other and work in a group to achieve a common goal.

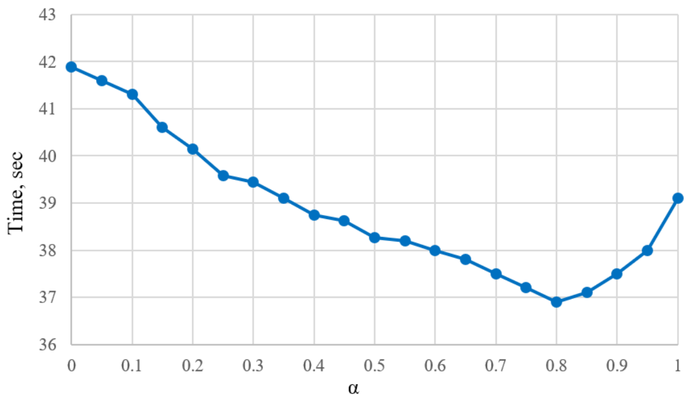

Figure 8. Dependence of the experiment time on step-size parameter α.

The simulation results are shown in Figure 9. The left part of the figure provides the robot movement without the usage of the Local Voting Protocol. In this case, robots move considering only the knowledge of the direction to the source and the distance to the obstacle. It is clearly observed that some of the robots might collide in a small number of next steps. The right part of this figure demonstrates the robot movement according to the novel algorithm provided in Section 4.1 with optimal step-size $\alpha = 0.8$. The robots avoid collisions with the obstacles and between themselves as they use the Local Voting Protocol to obtain the data from their neighbors.

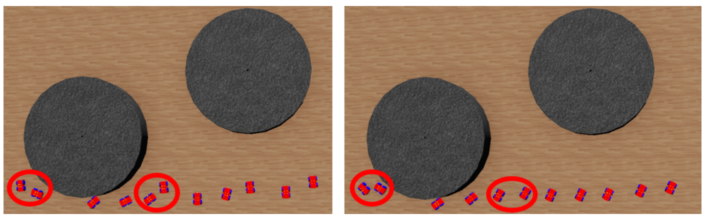

Figure 9. Screenshots of the simulations. The robots are depicted as red rectangles, obstacles as gray circles. The left side presents the simulation without Local Voting Protocol ($\alpha = 0$). In this case, robots do not keep the distance between them. The right side presents the simulation with step-size parameter $\alpha = 0.8$ of the algorithm from Section 4.1, where the robots keep the distance and avoid collisions.

It is important to mention that the Local Voting Protocol for related resource allocation problems was previously considered and evaluated in [32]. The current article is intended to describe the functionality of approach modifications. Theoretical investigations of the problem are planned to be published in future works.

5. Conclusions

The method proposed in the article is distinguished by its simplicity of implementation and potentially low cost. Alternative techniques suggested in the literature are based on the idea of the SLAM (simultaneous localization and mapping) type and require significantly more considerable information amounts from sensors together with massive computation resources. It leads to high systems complexity preventing obtaining a solution at a suitable time.

The appearance of emergent intelligence in the discussed examples encourages the essential "rethink" of the common notion of algorithms computability, which is traditionally considered the feasibility of implementing a given function as a combination of meanings from a basic set. Many practical problems can hardly be resolved by algorithmic fashion with a reasonable time conducting "brute force"-like approaches for checking almost every suitable input. A more widespread technique now is using networked systems with asynchronous nodes consisting of

$$\{\text{sensors, actuators, computing units}\}$$

which can be considered as a set of models. If the network complexity is similar to the complexity of a solving problem, then this problem can be appropriately treated for a finite (not huge) time. Applying the emergent intelligence methodology makes it possible to propose new perspectives for the study of such systems. The considered examples also exhibit appearance of the mentioned self-organization schemes in multiagent arrangements, aspiring to cluster structures.

Author Contributions: Conceptualization, O.G.; investigation, K.A., A.S. and Z.V.V. The authors have equally contributed to the writing, editing, and style of the paper. All authors have read and agreed to the published version of the manuscript.

Funding: The Sections 1–3 were supported by the St. Petersburg State University (project No. 73555239). The Section 4 was supported by the Ministry of Science and Higher Education of the Russian Federation (Project No. 075-15-2021-573, IPME RAS).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors are grateful to the reviewers for constructive and useful comments, which made it possible to significantly improve the presentation of the results obtained, and also to Veronika Revyakina for help in obtaining experimental results.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Johnson, K.G.; Neyenhuis, B.; Mizrahi, J.; Wong-Campos, J.D.; Monroe, C. Sensing Atomic Motion from the Zero Point to Room Temperature with Ultrafast Atom Interferometry. *Phys. Rev. Lett.* **2015**, *115*, 213001. [CrossRef] [PubMed]
2. Cohen, P.J. The independence of the continuum hypothesis. *Proc. Natl. Acad. Sci. USA* **1963**, *50*, 1143–1148. [CrossRef] [PubMed]
3. Vicsek, T.; Zafeiris, A. Collective motion. *Phys. Rep.* **2012**, *517*, 71–140. [CrossRef]
4. Olfati-Saber, R.; Fax, J.A.; Murray, R.M. Consensus and cooperation in networked multi-agent systems. *Proc. IEEE* **2007**, *95*, 215–233. [CrossRef]
5. Hwang, K.-S.; Tan, S.-W.; Hsiao, M.-C.; Wu, C.-S. Cooperative multiagent congestion control for high-speed networks. *IEEE Trans. Syst. Man Cybern. Part B* **2005**, *35*, 255–268. [CrossRef]
6. Granichin, O.; Erofeeva, V.; Ivanskiy, Y.; Jiang, Y. Approximation-based Consensus for Tracking under Unknown–but–Bounded Disturbances. *IEEE Trans. Autom. Control* **2021**. [CrossRef]
7. Savla, K.; Notarstefano, G.; Bullo, F. Maintaining limited-range connectivity among second-order agents. *SIAM J. Control Optim.* **2009**, *48*, 187–205. [CrossRef]
8. Blackman, S.S. Multiple hypothesis tracking for multiple target tracking. *IEEE Aerosp. Electron. Syst. Mag.* **2004**, *19*, 5–18. [CrossRef]

9. Tugnait, J.K. Tracking of multiple maneuvering targets in clutter using multiple sensors, IMM, and JPDA coupled filtering. *IEEE Trans. Aerosp. Electron. Syst.* **2004**, *40*, 320–330. [CrossRef]
10. Chen, X.; An, L.; Bhanu, B. Multitarget Tracking in Nonoverlapping Cameras Using a Reference Set. *IEEE Sens. J.* **2015**, *15*, 2692–2704. [CrossRef]
11. He, S.; Shin, H.; Tsourdos, A. Multi-sensor multi-target tracking using domain knowledge and clustering. *IEEE Sens. J.* **2018**, *18*, 8074–8084. [CrossRef]
12. Grachev, S.; Skobelev, P.; Mayorov, I.; Simonova, E. Adaptive clustering through multi-agent technology: Development and perspectives. *Mathematics* **2020**, *8*, 1664. [CrossRef]
13. Sergeev, S. Prerequisites for the creation of cooperative artificial intelligence systems. *Stoch. Optim. Inform.* **2020**, *16*, 13–30.
14. Beni, G.; Wang, J. Swarm Intelligence in Cellular Robotic Systems. In *Robots and Biological Systems: Towards a New Bionics*; Dario, P., Sandini, G., Aebischer, P., Eds.; Springer: Berlin/Heidelberg, Germany, 1993; Volume 102.
15. Dorigo, M.; Maniezzo, V.; Colorni, A. Ant system: Optimization by a colony of cooperating agents. *IEEE Trans. Syst. Man Cybern. Part B* **1996**, *26*, 29–41. [CrossRef] [PubMed]
16. Rutherford, S.; Bassler, B. Bacterial quorum sensing: Its role in virulence and possibilities for its control. *Cold Spring Harb. Perspect. Med.* **2012**, *2*. [CrossRef]
17. Whiteley, M.; Diggle, S.; Greenberg, E. Progress in and promise of bacterial quorum sensing research. *Nature* **2017**, *551*, 313–320. [CrossRef] [PubMed]
18. Adleman, L. Molecular computation of solutions to combinatorial Problems. *Sci. New Ser.* **1994**, *266*, 1021–1024. [CrossRef]
19. Watson, J.; Crick, F. A structure for deoxyribose nucleic acid. *Nature* **1953**, *171*, 737–738. [CrossRef]
20. Sergeenko, A.; Yakunina, M.; Granichin, O. Hamiltonian path problem solution using DNA computing. *Cybern. Phys.* **2020**, *9*, 69–74. [CrossRef]
21. Proskurnikov, A.; Granichin, O. Evolution of clusters in large-scale dynamical networks. *Cybern. Phys.* **2018**, *7*, 102–129. [CrossRef]
22. Granichin, O.; Khantuleva, T. Adapting wing elements ("feathers") of an airplane in a turbulent flow with a multiagent protocol. *Autom. Remote Control* **2017**, *78*, 1867–1882. [CrossRef]
23. Amelina, N.; Fradkov, A.; Jiang, Y.; Vergados, D.J. Approximate consensus in stochastic networks with application to load balancing. *IEEE Trans. Inf. Theory* **2015**, *61*, 1739–1752. [CrossRef]
24. Vergados, D.J.; Amelina, N.; Jiang, Y.; Kralevska, K.; Granichin, O. Toward optimal distributed node scheduling in a multihop wireless network through local voting. *IEEE Trans. Wirel. Commun.* **2018**, *17*, 400–414. [CrossRef]
25. Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; Eckstein, J. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.* **2011**, *3*, 1–122. [CrossRef]
26. Acebrón, J.A.; Bonilla, L.L.; Pérez Vicente, C.J.; Ritort, F.; Spigler, R. The Kuramoto model: A simple paradigm for synchronization phenomena. *Rev. Mod. Phys.* **2005**, *77*, 137–185. doi:10.1103/RevModPhys.77.137. [CrossRef]
27. Xu, Z.; Egerstedt, M.; Droge, G.; Schilling, K. Balanced deployment of multiple robots using a modified kuramoto model. In Proceedings of the American Control Conference, Washington, DC, USA, 17–19 June 2013; pp. 6138–6144. [CrossRef]
28. Granichin, O.; Uzhva, D.; Volkovich, Z.V. Cluster flows and multiagent technology. *Mathematics* **2021**, *9*, 22. [CrossRef]
29. Amelin, K.; Amelina, N.; Granichin, O.; Putov, V.V. Task allocation algorithm for the cooperating group of light autonomous unmanned aerial vehicles. *IFAC Proc. Vol.* **2013**, *46*, 152–155. [CrossRef]
30. Cavoretto, R.; De Rossi, A.; Dell'Accio, F.; Di Tommaso, F. Fast computation of triangular Shepard interpolants. *J. Comput. Appl. Math.* **2019**, *354*, 457–470. [CrossRef]
31. Cyberbotics. Webots. Available online: https://cyberbotics.com/ (accessed on 2 June 2021).
32. Amelina, N.; Granichin, O.; Granichina, O.; Jiang, Y. Differentiated consensuses in decentralized load balancing problem with randomized topology, noise, and delays. In Proceedings of the 53rd IEEE Conference on Decision and Control, Los Angeles, CA, USA, 15–17 December 2014; pp. 6969–6974.