Austin Journal of Robotics & Automation

Open Access

Austin Publishing Group

Review Article

Towards Unified Human-Robotic Societies

Peter Simon Sapaty*
Institute of Mathematical Machines and Systems, National Academy of Sciences, Kiev, Ukraine

***Corresponding author:** Peter Simon Sapaty, Institute of Mathematical Machines and Systems, National Academy of Sciences, Glushkova Ave 42, 03187 Kiev, Ukraine

Received: November 28, 2017; **Accepted:** December 12, 2017; **Published:** December 26, 2017

Abstract

Large numbers of robotic facilities have been accumulated worldwide, but existing robots still remain specialized devices rather than intelligent collaborators for humans. To effectively integrate massive robotics into human societies, radically new and much more universal approaches are needed. A semantic level model supported by special pattern-based knowledge processing language is described. It expresses operations and decisions in distributed spaces in a very compact and mobile form, with traditional system organization and management essentially shifted to automatic language interpretation in cooperative networked environments. Communicating language interpreters, associated with humans and robots, can form holistic goal-driven teams under unified command and control, where humans and robots, if needed, can seamlessly substitute each other at runtime, during scenario execution. The paper provides details of the proposed ideology and technology, especially of its basic scenario language with distributed networked implementation. Examples of solving practical problems from different fields with the effective use of multiple cooperative robotic facilities under the approach offered are exhibited too.

Keywords: World dynamics; Human-robotic systems; Spatial grasp technology; Networked language interpretation; Self-evolving patterns

Introduction

The world is changing dramatically for the last decades, with numerous conflicts and crises emerging frequently and everywhere, which include terrorism, ethnic, religious and military conflicts, endless floods of refugees, economy collapses, and so on. To withstand such unfortunate situations, new system ideologies, approaches, and management technologies are desperately needed. Of particular interest and effectiveness may be those allowing for seamless embedment of massive robotics into human societies, with robots taking care of dangerous and critical situations while acting cooperatively with humans and among themselves under global goals and unified control. But in many areas and cases the existing robots still remain as specialized devices rather than full-scale collaborators for people.

The paper is describing a new and quite unusual approach for human-robot integration which is not pursuing and developing further the traditional and overwhelmingly used interoperability [1,2] ideology and practice, but rather creating a much higher, "over-operability" [3,4] layer in the form of supreme (i.e. standing above humans and robots) spatial intelligence. This layer expresses top semantics of what should be done in distributed spaces and main decisions to be taken in complex situations.

Under this approach, it becomes extremely easy to assemble any teams with any ratio between humans and robots, which can substitute each other at runtime without interrupting system missions while always preserving global goal orientation and mission capabilities. Expressing complex spatial operations at this level allows us to automate most of organization and management routines for large human-robotic teams, including sophisticated command and control.

The paper briefs the related networking ideology and technology (Chapter 2) that can express operations and top decisions in physical, virtual and executive environments regardless of who (humans) or what (robots) should perform them, and in which quantities. This allows us to make an effective implementation in dynamic environments where manned and/or unmanned resources may not be known a priori but rather defined at runtime, depending on circumstances. The key element of this approach, a recursive Spatial Grasp Language (SGL), is described together with the structure and organization of its distributed networked interpreter which can form universal spatial machines operating with both information and physical matter.

Chapter 3 contains elementary examples of programming in SGL of different computational, knowledge processing, networking and control tasks, all in the same recursive syntax, showing the possibility of using SGL as a universal high level system management language. Chapters 4 to 9 provide examples of practical application of SGL for organizing integral human-robotic teams on different levels, coastal waters cooperative patrol, conducting swarms-based aerial warfare, effective use of driverless cars with autonomous collective solutions on roads, and distributed operations on social networks. Chapter 10 concludes the paper.

Spatial Grasp Technology (SGT)

Key ideas of the developed ideology and technology suitable for high-level organization and management of advanced human-robotic teams and its possible networked implementation are revealed in brief.

Self-evolving spatial patterns

Within SGT, a high-level scenario for any task to be performed in a distributed world is represented as an active self-evolving pattern

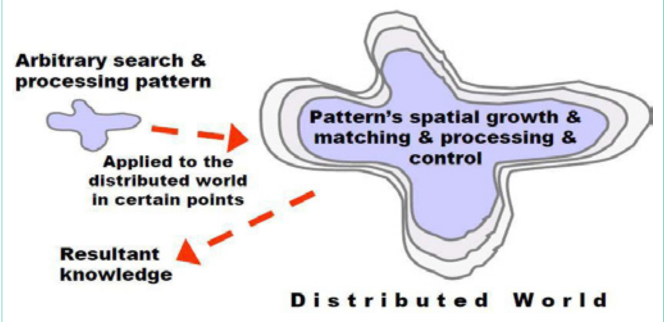
Arbitrary search & processing pattern
Pattern's spatial growth & matching & processing & control
Applied to the distributed world in certain points
Resultant knowledge
Distributed World

Figure 1: Spatial pattern growth & coverage & matching.

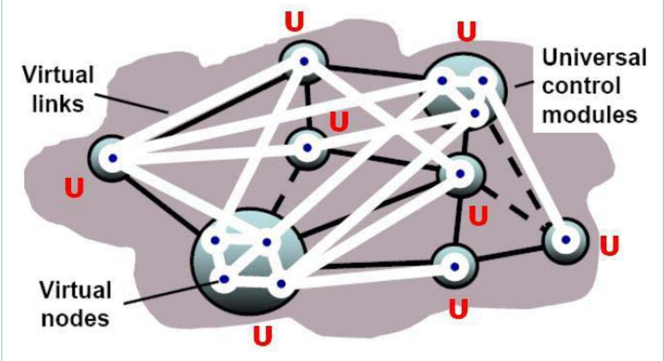

Figure 2: Creating distributed knowledge infrastructures.

rather than traditional program, sequential or parallel, inheriting holistic and gestalt [5,6] ideas rather than of communicating agents [7]. This also reflects integral style of human thinking and brain activity [8,9] when directly perceiving complex images as a whole, while treating parts and their sense within this whole rather than vice versa. The self-evolving pattern, written in high-level Spatial Grasp Language (SGL) expressing top semantics and key decisions of the problem to be solved and starting from any point, spatially propagates, grows, replicates, modifies, covers, interlinks and matches the distributed world, as shown in Figure 1.

The self-spreading & matching patterns can create knowledge infrastructures arbitrarily distributed between system components (humans, robots, sensors), as in Figure 2 (where SGL interpreters are shown as universal control modules U). Covered subsequently or simultaneously by same or other patterns with operations and control, these knowledge infrastructures can effectively support distributed databases, command and control, situation awareness, and autonomous decisions, also simulate any other models, both sequential and parallel like, for example, Petri nets or neural networks.

Spatial grasp language (SGL)

SGL [10-12], the core of the approach, allows us to directly move through, observe, and make any actions and decisions in fully distributed environments (whether physical, virtual, executive, or combined). It has universal recursive structure, shown in Figure 3, capable of representing any parallel and distributed algorithms operating over spatially scattered data or other, lower level, distributed systems of arbitrary natures.

SGL main features are in brief as follows:

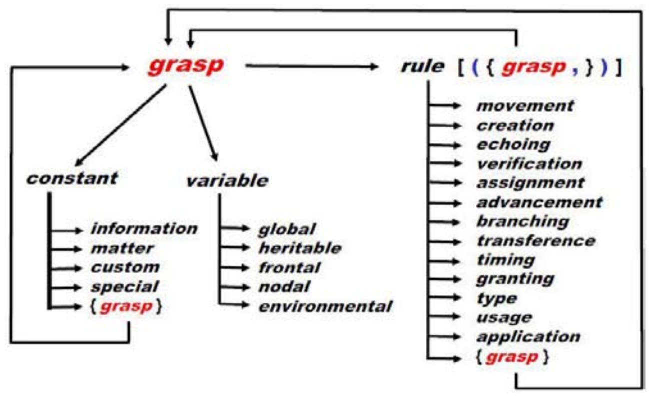

Figure 3: SGL recursive syntax.

An SGL scenario develops as parallel transition between sets of progress points (or props), with self-modified and self-replicating scenario code freely moving in distributed spaces. Starting from a prop, an action may result in new props (which may be multiple) or remain in the same prop. Each prop has a resulting value, which may be arbitrarily complex, and resulting state (one of: thru, done, fail, and abort). Different actions may evolve independently or interdependently from the same prop, splitting and parallelizing in space. Actions may also spatially succeed each other, with new ones applied sequentially or in parallel from all props reached by the previous actions.

Elementary operations can directly use states and values of props reached by other actions whatever complex and remote they might be. Any prop can associate with a position in physical, virtual, executive or combined world, sharing local information at them. Staying with the world points, it is possible to directly access and impact local world parameters in them, whether virtual or physical.

Overall organization and control of the breadth and depth space navigation and coverage is provided by SGL rules, which may be nested and can, for example, be as:

- Elementary arithmetic, string, or logic operation.
- Hop in a physical, virtual, execution, or combined space.
- Hierarchical fusion and return of (remote) data.
- Distributed control, both sequential and parallel.
- A variety of special contexts for navigation in space influencing embraced operations and decisions.
- Type or sense of a value or its chosen usage, guiding automatic interpretation.
- Creation or removal of nodes and links in distributed knowledge networks.
- A rule can be a compound one, integrating a number of other rules; it can also be defined in a result of local or global operations of arbitrary complexity.

Working in fully distributed physical, virtual, or executive environments, SGL has different types of variables, called spatial, effectively serving multiple cooperative processes:

- **Heritable variables** – these are starting in a prop and serving all subsequent props, which can share them in both read & write operations.

- **Frontal variables** – are an individual and exclusive prop's property (not shared with other props), being transferred between the consecutive props and replicated if from a single prop a number of other props emerge.

- **Environmental variables** – are accessing different elements of physical and virtual words when navigating them, also a variety of parameters of the internal world of SGL interpreter.

- **Nodal variables** – allow us to attach an individual temporary property to different world nodes, accessed and shared by all activities currently associated with these nodes.

These types of variables, especially when used together, allow us to create spatial algorithms working in between components of distributed systems rather than in them, allowing for flexible, robust, and self-recovering solutions. Such algorithms can freely replicate, spread and migrate in distributed environments (partially or as an organized whole), always preserving global integrity and overall control.

To simplify SGL programs, traditional to existing programming languages abbreviations of operations and delimiters can be used too, substituting certain rules as in the examples throughout this text, but always remaining within the general syntactic structure shown in Figure 3.

Networked SGL interpreter

The interpreter (its architecture stemming from [13], more in [14-16]) consists of a number of specialized modules handling and sharing specific data structures, as in Figure 4.

SGL interpreters can communicate with each other, and a distributed network of the interpreters can be mobile and open, changing the number of nodes and communication structure in between at runtime.

The backbone and nerve system of the distributed interpreter is its dynamic spatial track system with its parts kept in the Track Forest memory of local interpreters. These are logically interlinked with similar parts in other interpreter copies forming altogether the

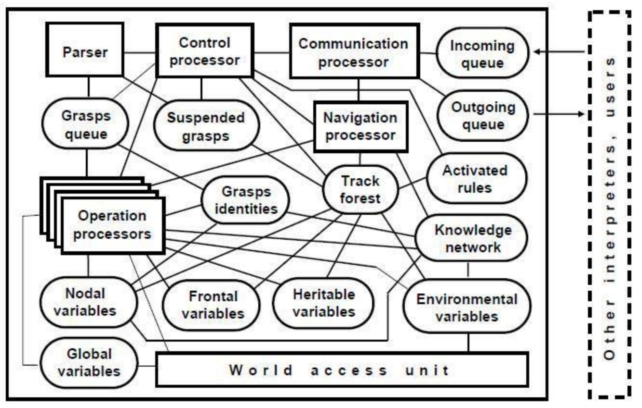

Figure 4: SGL interpreter organization and main components.

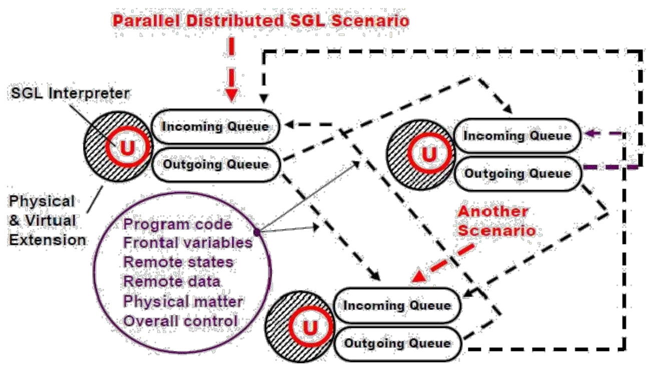

Figure 5: Universal spatial machines based on networked SGL interpretation.

global control coverage. This forest-like distributed track structure enables for both hierarchical and horizontal control as well as remote data and code access, with high integrity of emerging parallel and distributed solutions achieved without any centralized resources.

Dynamically crated track trees (forests) spanning the systems in which SGL scenarios evolve are also used for supporting spatial variables and echoing & merging control states and remote data. They are self-optimizing in parallel echo processes while providing automatically of what is usually called (adaptive) command and control, or C2. They also route further grasps to the positions in physical, virtual, execution or combined spaces reached by the previous grasps, uniting them with frontal variables left there by preceding grasps.

The distributed SGL interpreter may have any number of nodes, up to millions even billions, spread worldwide. Copies of the interpreter can be concealed if operate in hostile environments, allowing the latter to be analyzed and impacted in a stealth manner. Dynamically networked SGL interpreters extended by and integrated with other facilities and gadgets (like, for example, mobile robots or existing C2 systems) can form universal spatial machines operating with both information and physical matter, as in Figure 5.

These networked machines working under intelligent scenarios injected at any time and from any nodes, can perform any knowledge & matter processing and control operations throughout any areas, the whole world including. By embedding SGL interpreters into robotic vehicles and electronic devices (including those associated with humans like smart phones, laptops or smart watches) we can organize any collective behaviour needed, integrating them into holistic teams under unified and distributed command and control. The collective mission scenario can start from any unit and cover, activate, and control the whole group at run time.

Elementary Examples

Assignment of the sum of three values 27, 33 and 55.6 to a variable named Result, as in Figure 6.

assign(Result, add(27, 33, 55.6))

The variable Result will be created, if not existing yet, and will be associated together with the obtained value with the world position where the scenario started. Simplified and shortened version in traditional style may be as follows:

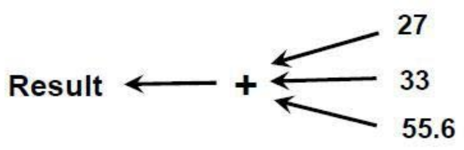

Figure 6: Assigning the sum of values to a variable.

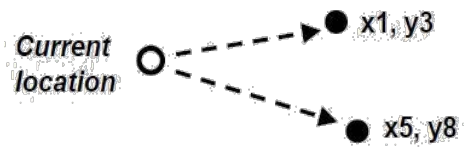

Figure 7: Moving independently to two physical locations.

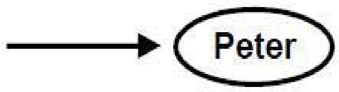

Figure 8: Creating a virtual node.

Result = 27 + 33 + 55.6

Move physically from the current location independently and simultaneously to locations (x1, y3) and (x5, y8), see Figure 7:

```
branch(move(location(x1, y3),
        move(location(x5, y8)))
```

This will cause movement from the current physical position to the two new physical positions by given coordinates independently and possibly in parallel (if the latter supported by implementation). A shortened version may be as follows:

```
move(x1, y3), move(x5, y8)
```

Creation of a virtual node Peter, see Figure 8:

Starting from the current world location, a new, isolated, virtual node with the given name will be created with the resultant control moving into it. Shortened version:

```
create(Peter)
```

- Extending the virtual network (already having node Peter) with a new link-node pair stating that "Peter is father of Alex", see Figure 9:

```
advance(hop(node(Peter)),
         create(link(+fatherof), node(Alex)))
```

The scenario first directly hops into the already existing node Peter and from it creates new link-node pair with both link and node properly named, where the succession in virtual space is provided by the rule advance. Simplified version:

```
hop(Peter); create(+fatherof, Alex)
```

- Giving a command to soldier John to use robot Shooter to fire by coordinates (x,y) with confirmation of the robot's success or failure, see Figure 10.

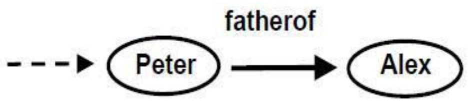

Figure 9: Semantic network extension with link-node pair.

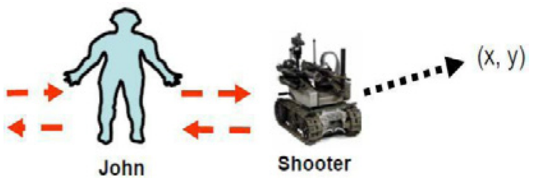

Figure 10: Ordering soldier to use robot to shoot by coordinates.

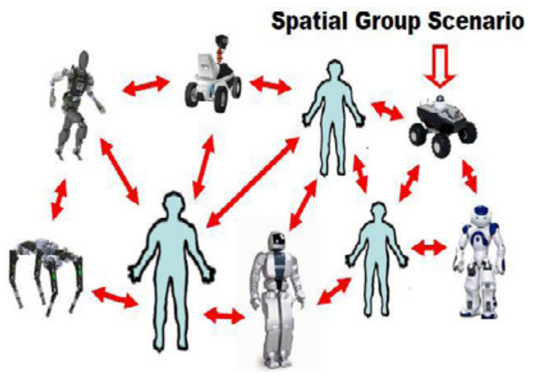

Figure 11: Mixed team with same status of units.

```
hop(John);
report_if((hop(Shooter); fire(x, y)),
           success, failure)
```

Integral Human-Robotic Teams

In the previous example we showed selective tasking of a human and a robot, whereas in this section will consider simple scenarios for mixed teams with humans and robots having equal status, as symbolically shown in Figure 11.

The highlighted below (in bold) scenario parts may be executed by biological brain & sensors (say, in a dialog mode via a screen) if appear in SGL interpreters associated with humans, otherwise entirely handled by robots, with overall group control fully automatic too.

- Randomized collective group movement, starting in any node, with Range distance allowed between units when moving; units reporting individually if "aliens" seen.

```
hop(all);
nodal(Limits = (dx(0,8), dy(-2,5)),
        Range = 200, Shift);
repeat(
         if(seen(unknown), report('alien'));
         Shift = random(Limits);
```

```
            if(empty(Shift, Range), WHERE += Shift);
            sleep(delay))
```

- Starting from any node, finding topologically central unit of the moving group and hopping into it.

```
frontal(Aver) = average(hop(all); WHERE);
min_destination(
   hop(all); distance(Aver, WHERE))
```

- Creating hierarchical infrastructure from the center found using oriented links infra and depth as a certain maximum allowed linking distance:

```
repeat_linkup(+infra, firstcome, depth))
```

- Using the created infrastructure, collect at its top and analyze all objects (symbolically: targets) discovered throughout the whole territory covered by the group, issuing OK or alarm if danger.

```
frontal(Seen) = repeat(free_detect(targets), hop(+infra));
if(analyze(Seen), out(OK), out(alarm))
```

Integration of the above four cases within a single united scenario is trivial, allowing the whole group randomly move while keeping threshold distance between units, regularly redefining its changing center and hierarchical infrastructure stemming from it, and collection and analysis of targets. This can be done starting from any human or robotic unit (a related case shown in [17]). Any other collective scenarios can be generated too, often on the fly due to their transparency and compactness.

Fully Semantic Scenario in SGL

At this highest level, it may become possible to describe in SGL only what should be done in a distributed space and which top operations and decisions to make, like the following:

Evaluate damage after disaster in the points with physical coordinates X1_Y1, X2_Y2, and X3_Y3, and report the maximum one.

The SGL expression will be:

```
report_max_assess(X1_Y1, X2_Y2, X3_Y3)
```

This semantic description is fully formal, and can be automatically implemented in physical space by available manned, unmanned or mixed units. The solution by robotic units R1 and R2 and manned M1, scattered somewhere in the region (presumably all having communicating SGL interpreters installed) is shown in Figure 12.

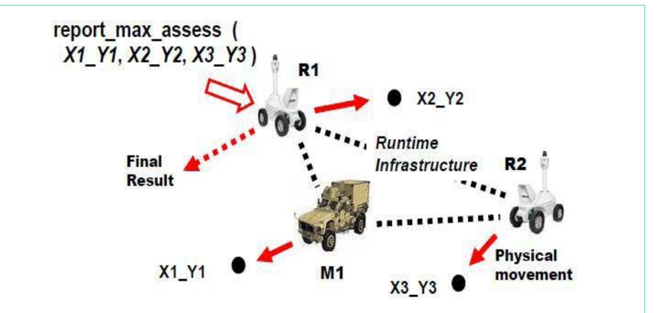

Figure 12: Automatic solution of the semantically defined problem.

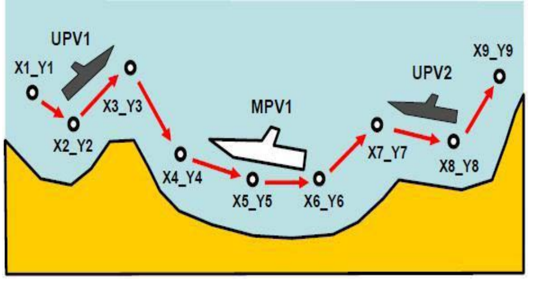

Figure 13: Simultaneous coastal patrol.

Coastal Waters Cooperative Patrol

This is another scenario example where manned and unmanned units can work cooperatively and substitute each other at any time, as in Fig. 13, with new units, if needed, to be involved at runtime too. All of them are following the coastline in changing directions and reporting if discover (which is sensors dependent) “aliens”.

At the beginning we will create a discrete coastal map as a semantic network consisting of coordinates of key points linked with each other by oriented links (all named r). Vehicles will follow this chain along or opposite orientation of the links, changing direction at the end or when see a “colleague” ahead, with the scenario oriented on starting simultaneously in points x1_y1, x5_y5, x9_y9.

```
stay_create(
x1_y1; (+r, x2_y2); ...;(+r, x9_y9));
hop(x1_y1, x5_y5, x9_y9);
frontal(R) = random(+r, -r)
WHERE = CONTENT;
repeat(
 repeat(check_report(vision_depth);
       WHERE = hop(R); none(distance));
 invert(R))
```

This semantic level scenario can, for example, be executed by unmanned UPV1 and UPV2 vehicles and manned MPV1, as in Figure 13. In case of a manned vehicle engaged, the boldfaced operations can be performed manually, whereas in robotic cases – all automatically.

Swarm against Swarm Aerial Scenario

We will consider here the case where a manned, unmanned or mixed aerial swarm is opposing another group of aerial vehicles, which may be manned or unmanned too. This, for example, can relate to fighting criminal and spying drones which are currently spreading worldwide [18,19] and may potentially operate in swarms too.

Main roles of the swarm against swarm scenario, with alien drones as Targets and friendly units as Chasers are shown in Figure 13, with SGL scenario description and explanation of its main steps following.

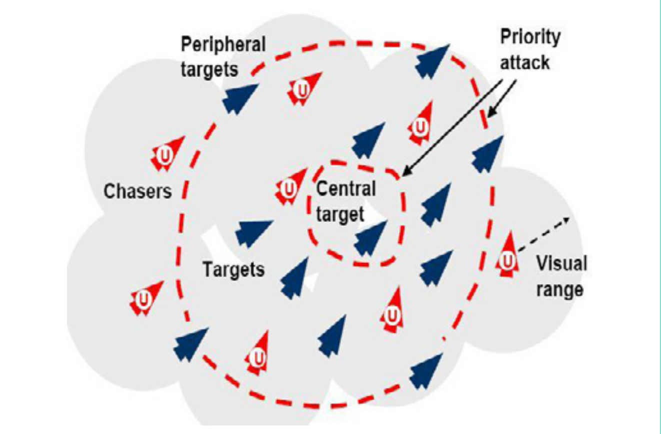

Figure 14: Fighting group targets with manned/unmanned swarms.

```
frontal(Chasers = …, Targets, Next, List, Center);
repeat(
 hop(random, Chasers);
 Targets = merge(hop(Chasers); coordinates(visible));
 nonempty(Targets); Center = average(Targets);
 List = sortdown(
  split(Targets); distance(VALUE, Center) & VALUE);
 List = append(withdraw(List, last), List);
 sling(
  nonempty(List); Next = withdraw(List, 1) : 2;
  min_destination(
    hop(Chasers); STATUS == vacant;
    distance(WHERE, Next));
  STATUS = engaged;
  free(pursue_neutralize(Next);
       STATUS = vacant)))
```

This will be working in the following (including parallel) steps, where each time after distribution of all collected targets the fully mobile scenario is starting from another, randomly chosen chaser.

- Initial launch of the swarmed chasers (in Figure 14 with SGL interpreters U embedded, which can communicate with each other) into the expected operational area.
- Discovering targets, finding their topological center, and forming priority list by their physical positions in relation to this center, where highest priority is assigned to topologically central targets as potential control units of the suspected intruders.
- Other targets are sorted by their growing distance from the topological center of the group.
- The most peripheral targets (those in maximum distance from the group's center), may be assigned higher priority too as potentially having more chances to escape, and being prevented from this.
- Stepwise assigning of available chasers to highest priority targets (for each target the physically nearest chaser is chosen) classifying the chosen chasers as engaged with subsequent individual chasing and neutralizing the targets.
- Restoring status vacant after performing the task if chasers survive themselves.
- The vacant chasers are again engaged in the targets selection & impact.

It is worth noting that all the chaser swarm management has been done exclusively within the swarm itself, by human or artificial intelligence and without external intervention, which can dramatically simplify the outside group tasking, with potentially involving any number of collectively behaving manned or unmanned units.

Manned and Driverless Vehicles Collective Behavior

Autonomous vehicles represent one of the most prominent technologies since the creation of automobile itself [21,28]. They can fundamentally change transportation by reducing crashes, energy/fuel consumption, pollution and the costs of congestion. Such vehicles can directly communicate with each other and with manned ones for finding suitable collective solutions using either short distance direct V2V communication channels, or infrastructure level V2I types of communications for longer distances.

We will consider here an exemplary situation on a road like lane manoeuvring for the fastest vehicle between two gaps. This situation is shown in Figure 15 and by SGL scenario that follows, where starting from the first vehicle after the gap the whole chain of them is autonomously analyzed and the fastest vehicle found. The latter then making this manoeuvre individually, directly cooperating with the first vehicle it wants to be ahead of, and this can be done manually by a human driver or fully automatically by a robotic vehicle.

```
frontal(Gap = …, Max, Before, First, Chosen);
sling(
 sleep(Delay);
 distance(ahead) > Gap; First = ADDRESS;
 repeat(
     if(MAXSPEED > Max, (Max = MAXSPEED; Chosen =
```

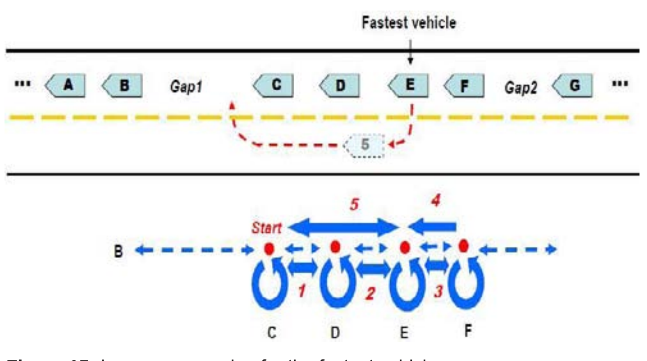

Figure 15: Lane manoeuvring for the fastest vehicle.

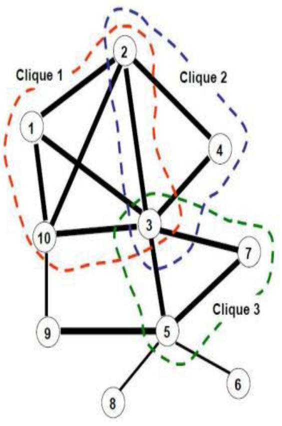

Figure 16: Examples of cliques in the network.

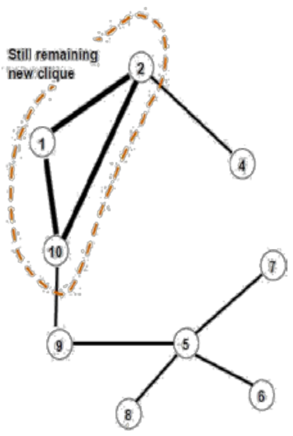

Figure 17: Destroying cliques by removing node 3 common for all cliques.

```
ADDRESS));
      distance(behind) < Gap; hop(next_behind));
    hop(Chosen); lane_manoeuvre(ahead, First))
```

A natural language explanation of this very compact scenario may be as follows.

1) Define types and initial values of spatial variables used.

2) Repeat the following regularly with certain time intervals.

a) Use sensors to measure nearest distance to vehicles ahead and allow the following if the gap ahead exceeds threshold given. Remember network address of the current vehicle, declaring it as the "first" one.

b) Do the following repeatedly until possible.

- If the current vehicle's maximum possible speed exceeds the already accumulated maximum speed among the considered vehicles, the latter changes to the former, and the current vehicle's network address is named as "chosen".

- Use sensors to measure nearest distance to vehicles behind, and if it is less than the given gap threshold, enter the first vehicle behind via V2V, which becomes current now.

c) Directly enter the vehicle finally resulted as "chosen" and activate its lane manoeuvring, to appear ahead of the vehicle declared as "first" and registered by its network address.

Operations on Social Networks

SGT has been extensively investigated for suitability of solving different problems in large social systems reflected by distributed social networks [22], which in integrated human-robotic societies may have both human and robotic nodes interconnected by different kinds of links. Finding different types of clustering of networked nodes by analysing links between them is usually a very important task, with examples of strongest clusters, or cliques, shown for in the network of Figure 16 (having one four-node and two triangular cliques).

Below is parallel and fully distributed solution for finding cliques in a network, where each node has relations with any other node, and these fully interconnected parts are maximum possible by the number of their nodes.

```
hop_nodes(all); frontal(Clique) = NAME;
repeat(hop_links(all); not_belong(NAME, Clique);
     if(andparallel(hop_link(any), Clique),
       if(BACK < NAME, append(Clique, NAME), done),
       fail));
if(length(Clique) >= 3, output(Clique))
```

The final result will be issued in the last nodes of the cliques found, being as follows:

(1,2,3,10),(2,3,4),(3,5,7).

Different operations can be planned on the basis of such analysis of social systems. For example, if this is an adversary network, then removing a single node 3 belonging to all cliques, will destroy all of them, thus representing the cheapest but at the same time the strongest network impact (with the resultant network shown in Figure 17).

The virtual node removal can be done by:

```
delete_hop_direct (3)
```

Destruction of the related physical point, if it is officially associated with this virtual node, by a human or robot physically moving to it using the returned location coordinates registered, say, in CONTENT of virtual node 3, may be organized as follows.

```
destroy_move(hop_direct(3); CONTENT)
```

Conclusions

The current paper pioneered on formalization of semantic level operations and top intelligence as regards large distributed systems, which can be implemented by any available resources regardless of being human or robotic, thus paving a real way to integration of multiple robotics into human societies.

Some remotely related works in this direction have been conducted in military on formalization of Command and Control (C2) to simplify multilingual international cooperation and also improve chances of formal engagement of robotic facilities in advanced operations. But the developed specialized Battle Management Languages (BML) [20] for unambiguous expression of C2 are not programming languages themselves, therefore needing integration with other linguistic facilities and organizational levels. On the contrary, SGL, being fully formal and universal system language, allows for effective and compact semantic expression of any battlefield scenarios and orders, also directly supporting robotized up to fully robotic systems [17]. More on SGL, its history, applications, and international cooperation can be found elsewhere, [24-27] including.

The most general even "nonscientific" explanation and summary of the approach offered may be as follows. Imagine you have a distributed networked world, which may be arbitrarily large, and want to do something (very) good in it. You can describe your desire in a special high level language mentioning only main operations and decisions to be taken. Then you inject this (very short but extremely powerful) scenario from any world point (your personal computer including) which begins self-spreading throughout the world via existing systems and channels in a super-virus mode and doing autonomously and in parallel of what you need. The (interlinked) results obtained may be left in the reached locations throughout this world (possibly, itself created and modified by the same scenario) or returned to you as extracted high-level knowledge, or in both ways. Psychologically, you may feel yourself as having the whole world in your hands! The author has been practicing and cultivating such style of global world coverage, vision and management for almost half a century. It started from the creation of citywide heterogeneous computer networks from the end of sixties [23,29], well before the internet, with subsequent use of self-spreading and self-replicating high-level program code in many civil and defense applications.

The latest and most advanced version of the technology can be put on any platform in a short time and by a small group of system programmers, within existing university environments too. The author would be glad to communicate with organizations and individuals who may get interested in this area of research and cooperation.

References

1. Interoperability.
2. Interoperability for Joint Operations. NATO Backgrounder. 2006.
3. P Sapaty. "Providing Over-operability of Advanced ISR Systems by a High-Level Networking Technology". SMI's Airborne ISR, 26th to 27th October 2015. Holiday Inn Kensington Forum, London, United Kingdom.
4. Sapaty PS. "Over-Operability in Distributed Simulation and Control". The MSIAC's M&S Journal Online. 2002; 4: 8.
5. M Wertheimer. Gestalt theory, Erlangen, Berlin. 1924.
6. P Sapaty. "Gestalt-Based Ideology and Technology for Spatial Control of Distributed Dynamic Systems". International Gestalt Theory Congress. 16th Scientific Convention of the GTA, University of Osnabrück, Germany. 2009.
7. M Minsky. The Society of Mind. Simon & Schuster, New York. 1988.
8. K Wilber. "Waves, Streams, States, and Self: An Outline of an Integral Psychology". The Humanistic Psychologist. 2003; 31.
9. L Gabriel. "Brain Wave Basics – What You Need to Know about States of Consciousness". Thought Medicine. Exploring the Power of Mind from Science to Spirituality. 2011.
10. PS Sapaty. "Spatial Grasp Language (SGL) for Distributed Management and Control". Journal of Robotics, Networking and Artificial Life. 2016; 4: 174-181.
11. P Sapaty. "A Brief Introduction to the Spatial Grasp Language (SGL)". Journal of Computer Science & Systems Biology. 2016; 9.
12. P Sapaty. "Spatial Grasp Language (SGL)". Advances in Image and Video Processing. 2016; 4.
13. PS Sapaty. A Distributed Processing System. European Patent No. 0389655. European Patent Office. 1993.
14. P Sapaty. Mobile Processing in Distributed and Open Environments. John Wiley & Sons, New York. 1999.
15. P Sapaty. Ruling Distributed Dynamic Worlds. John Wiley & Sons, New York. 2005.
16. P Sapaty. Managing Distributed Dynamic Systems with Spatial Grasp Technology. Springer. 2017.
17. PS Sapaty. "Military Robotics: Latest Trends and Spatial Grasp Solutions". International Journal of Advanced Research in Artificial Intelligence. 2015; 4.
18. J Stanley & C Crump. "Protecting Privacy from Aerial Surveillance: Recommendations for Government Use of Drone Aircraft. American Civil Liberties Union. 2011.
19. Sanchez C McKibben. "Worst Case Scenario: The Criminal Use of Drones". Council on Hemispheric Affairs. 2016.
20. Coalition Battle Management Language (C-BML). RTO Technical Report TR-MSG-048. 2012.
21. T Litman. Autonomous Vehicle Implementation Predictions Implications for Transport Planning. Victoria Transport Policy Institute. 2017.
22. P Sapaty. Distributed Human Terrain Operations for Solving National and International Problems. International Relations and Diplomacy. 2014; 2: 597-622.
23. Bondarenko AT, Mikhalevich SB, Nikitin AI, Sapaty PS. "Software of BESM-6 computer for communication with peripheral computers via telephone channels". In Computer Software. 1970; 5.
24. P Sapaty, M Sugisaka. "Optimized Space Search by Distributed Robotic Teams". Proc. World Symposium Unmanned Systems. Baltimore Convention Center, USA. 200315-17.
25. P Sapaty, M Sugisaka. "Countering Asymmetric Situations with Distributed Artificial Life and Robotics Approach". Proc. Fifteenth International Symposium on Artificial Life and Robotics (AROB 15th'10). B-Con Plaza, Beppu, Oita, Japan. 2010.
26. P Sapaty, M Sugisaka. "Advanced Networking and Robotics for Societal Engagement and Support of Elders". Proc. 16th International Symposium on Artificial Life and Robotics (AROB 16th '11). B-Con Plaza. Beppu, Oita, Japan. 2011.
27. PS Sapaty. "Advanced Maritime Operations Under Network-Centric Organizations". Proceedings of the International Conference Warship 2016: Advanced Technologies. In Naval Design, Construction & Operation. 2016.
28. JM Anderson, N Kalra, K Stanley, et al, Autonomous Vehicle Technology. A Guide for Policymakers. RAND Corporation. 2016; 214.
29. Sapaty PS. "A Method of organization of an intercomputer dialogue in the radial computer systems". In The Design of Software and Hardware for Automatic Control Systems. Inst. of Cybernetics Press. Kiev. 1973.