

Mosaic warfare: from philosophy to model to solutions

Abstract

The paper relates to advanced management of large distributed dynamic systems in unpredictable and crisis situations. It briefs the newest DARPA Mosaic Warfare concept oriented on rapidly composable networks of low-cost sensors, multi-domain command and control nodes, and cooperative manned and unmanned systems, with runtime integration of scattered resources which should operate together as one holistic system. It may have the highest value for solving complex national and international defense and security problems for which scattered throughout the world “mosaic” resources should be quickly integrated to collectively fight disaster and crisis situations, with separate nations unable of doing this individually. The paper shows how distributed mosaic systems can be modeled under the developed Spatial Grasp Technology (SGT) using active distributed knowledge networks and self-spreading and self-matching holistic spatial scenarios, with solving such exemplary problems on them as runtime collection of scattered resources into integral forces operating under unified control, and grouping of distributed facilities for surrounding and collective elimination of unwanted phenomena. Of practical importance may be mosaics-related approaches using massive robotics with unmanned units behaving as tiles in distributed environments. Explained is how to assemble teams of UCAVs intelligently swarming, self-restructuring and collectively observing territory with collecting and impacting the targets discovered. Another SGT scenario is organizing automatic fight of an aerial swarm with another group/swarm without centralized or external control. Also shown how broken into pieces the moving platoon of unmanned vehicles, due to road situations, is self-recomposing into a regular platoon chain, with vehicles symbolically considered as mosaic tiles too. The paper also summarizes the investigated advantages of using SGT for implementation of different Mosaic Warfare features.

Keywords: mosaic warfare, spatial grasp technology, distributed networking, multi-domain control, robotic swarming

Volume 5 Issue 5 - 2019

Peter Simon Sapaty

Institute of Mathematical Machines and Systems, National Academy of Sciences, Ukraine

Correspondence: Peter Simon Sapaty, Institute of Mathematical Machines and Systems, National Academy of Sciences, Glushkova Ave 42, 03187 Kiev, Ukraine, Email peter.sapaty@gmail.com

Received: August 18, 2019 | **Published:** September 19, 2019

Introduction

The paper relates to advanced management of distributed dynamic systems in unpredictable and crisis situations, which may appear any time and in any points/places throughout the world while covering large territories and often ignoring borders between different countries. It is organized as follows.

Section 2 first describes the newest DARPA Mosaic Warfare concept oriented on rapidly composable networks of low-cost sensors, multi-domain command and control nodes, and cooperative manned and unmanned systems, with runtime integration and goal orientation of scattered resources which should operate together as one holistic system. Section 3 briefs the Spatial Grasp model and Technology (SGT) aiming at solving distributed supervision, integration and management problems in a way similar to the announced Mosaic Warfare goals, which has been developed for the last decades and in different countries, with numerous applications in both civil and defense areas.

Section 4 shows how distributed mosaic systems can be effectively modeled under SGT using active distributed knowledge networks with nodes behaving as mosaic tiles, some of which expected missing.

Solutions of exemplary problems are demonstrated on this model, such as runtime collection of scattered particular type resources throughout large territories into more powerful forces operating under unified control, and grouping of certain distributed tiles-facilities into spatial fences for cooperating surrounding, supervision, and elimination of dangerous elements or phenomena.

Of practical importance may be mosaics-related approaches using massive robotics with unmanned units behaving as tiles in distributed environments. In Section 5, it is demonstrated how easy is to assemble teams of UCAVs intelligently swarming, self-optimizing, and collectively observing the covered territory with collecting, disseminating, and impacting the targets discovered. Another SGT scenario (Section 6) is organizing automatic fight of one aerial swarm against another unmanned or manned group/swarm without any external control. One more example (Section 7) shows how broken into pieces the moving platoon of manned or unmanned vehicles, due to the situations on roads, is self-recomposing into the regular platoon chain again, with vehicles symbolically considered as mosaic tiles too.

In Conclusions (Section 8), the paper highlights and explains advantages of using SGT for simulation and implementation of different Mosaic Warfare features, also revealing future plans

for continuation and extension of this research, new publications including.

The DARPA mosaic concept

The new DARPA's Strategic Technology Office (STO) strategy, called Mosaic Warfare,^{1–8} seeks of getting a new asymmetric advantage, one that imposes complexity on adversaries by harnessing the power of dynamic, coordinated, and highly autonomous systems. This is by turning complexity into a powerful weapon via rapidly composable networks of low-cost sensors, multi-domain command and control nodes, and cooperative manned and unmanned systems, as symbolically in Figure 1.¹



Figure 1 DARPA's concept of "mosaic warfare."

Under the mosaic approach, air, cyber, land, sea and space domains will focus on operating in a more integrated framework. The new approach also recognizes the reality that it will be impossible for the U.S. military to maintain an asymmetric technological advantage in the form of advanced satellites, stealth aircraft and other systems in the future. DARPA's research and development efforts are also focusing on humanoid robots, artificial intelligence, synthetic biology, distributed space architectures, hypersonics and quantum sensing. Advances in microelectronics and communications are making possible a degree of networked coordination and collaboration between different systems almost unimaginable just a few years ago. This, in turn, is enabling distributed system-of-systems architectures that will be more resilient to attack, less costly to develop and faster to upgrade when compared to today's centralized expensive monolithic system.

The mosaic approach will require a shift in military policy from dominance to "lethality". It is believed that various domains currently spend too much time; money and effort in trying to ensure its weapons systems are more advanced than adversaries. It's imperative to focus on lethality and the ability to win regardless of whose individual weapons system is the best. A fundamental way to achieve this lethality is by distributing and disaggregating the sensors and weapons that today are tightly bound together and integrated on monolithic platforms.

Efforts of various domains to engage in the concept of joint multi-domain battle in recent years have been limited by the degree of machine-to-machine connectivity available to them today. Instead, a future is considered where computers will be distributed across the battle space, and can all communicate and coordinate with each other. Traditional asymmetric technology advantage is now lower than it once was, due to increased global access to comparable high-tech systems and components, many of which are now commercially available. The military has found that high cost and sometimes decades-long development of new systems are unable to compete with the fast refresh rate of commercial electronics technology, which can make

new military systems obsolete before they're even delivered. Within mosaic concept to warfare, lower-cost, less complex systems may be linked together in a vast number of ways to create desired, interwoven effects tailored to any scenario.

The individual parts of a mosaic are attritable but together are invaluable for how they contribute to the whole. This means that even if an adversary can neutralize a number of pieces of the mosaic, the collective can instantly respond as needed to still achieve the desired, overall effect. A tile in a mosaic is one small part of a bigger picture, as in Figure 2.² if you lose one tile, this should not be a big deal, as the whole picture remains understandable and functional.



Figure 2 Mosaic picture with missing tiles but still understandable content.

The mosaic concept also fundamentally differs from the traditional "system of systems" model, where each part is uniquely designed and integrated to fill a specific role. Mosaic warfare envisions a bottom-up composition capability, where individual elements, like individual tiles in a mosaic, are combined to create an effect in ways not previously contemplated, and potentially dynamically.

Unlike today's monolithic systems and rigid architectures that take decades to develop, Mosaic Warfare is expected to utilize rapid machine-to-machine interoperability and AI to network manned and unmanned systems together, creating resilient and distributed architectures at campaign, and eventually, mission speeds. It focuses on speed and adaptation, networking sensors, command and control, and effects together across domains to form solutions that adapt to dynamic threats and environments, orienting on creation of resilient systems that retain legacy capabilities, but mitigating the vulnerabilities of monolithic systems.

The new DARPA approach, in our opinion, may have much greater, including philosophical and psychological, sense and dimension rather than just uniting distributed casual forces. This may inevitably connect it with the Gestalt psychology and theory^{9–13} proclaiming unique capability of human mind and brain to directly grasp the whole of phenomena while treating parts, which may be incomplete, in the context of this whole rather than vice versa. The Spatial Grasp model and Technology, briefed in the next section, is just attempting to conceptually and practically catch this wholeness, of distributed dynamic systems too, which can be particularly useful for this DARPA mosaics concept.

Spatial grasp technology (SGT)

We are providing here only main ideas of SGT on which there are many available sources including European patent,¹⁴ John Wiley, Springer and Emerald books,^{15–19} as well as numerous publications

of different technology versions along with their networked implementations and diverse applications.^{20–68}

Within SGT, a high-level scenario for any task to be performed in a distributed world is represented as an *active self-evolving pattern* rather than traditional program, sequential or parallel. This pattern, written in a high-level Spatial Grasp Language (SGL) and expressing top semantics of the problem to be solved, can start from *any world point* as a source of pattern's activity. It then spatially *propagates*, *replicates*, *modifies*, *covers* and *matches* the distributed world in a parallel wavelike mode, while echoing the reached control states and data found or obtained by given operations (which may be arbitrarily remote) for making decisions and processing at higher levels, with further space navigation from new or previously reached points, and so on. This inherently parallel and fully distributed spatial *spreading-covering-conquering-matching-grasping-echoing* process is symbolically shown in Figure 3 together with its simplified physical analogy.



Figure 3 The basic idea of SGT: a) controlled echoed space navigation & matching; b) symbolic physical analogy.

Many spatial processes in SGL can start any time and from any world places and develop simultaneously worldwide while cooperating or competing with each other, depending on applications, as in Figure 4.

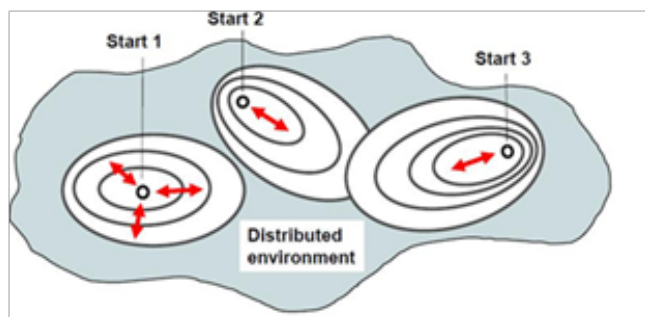


Figure 4 Multi-source space coverage & matching.

The self-spreading and self-matching SGL patterns can create *knowledge infrastructures* arbitrarily distributed between system components having embedded SGL interpreters, which may cover any regions, the whole world including, as in Figure 5.

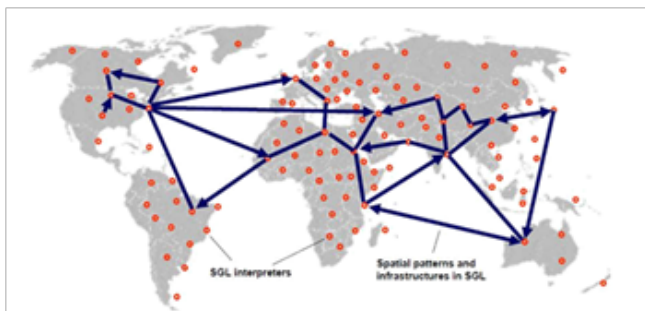


Figure 5 Spreading spatial patterns and creation of distributed infrastructures.

The created infrastructures, which may be left active, can effectively support or express distributed databases, command and control, situation awareness, autonomous and collective decisions, as well as any other existing or hypothetical computational and/or control models (like, say, neural networks or Petri nets). They can also be the result of spatial matching of arbitrary graph-based patterns in SGL with already existing knowledge networks previously organized by SGT or any other models and technologies. SGL has a deep recursive structure with its top level shown below, also in Figure 6, which reflects the space-covering-grasping-creation-matching-echoing nature of SGL scenarios (where words in italics represent syntactic categories, square brackets show optional constructs, braces indicate repetitive parts, and parentheses and comma being the language symbols).

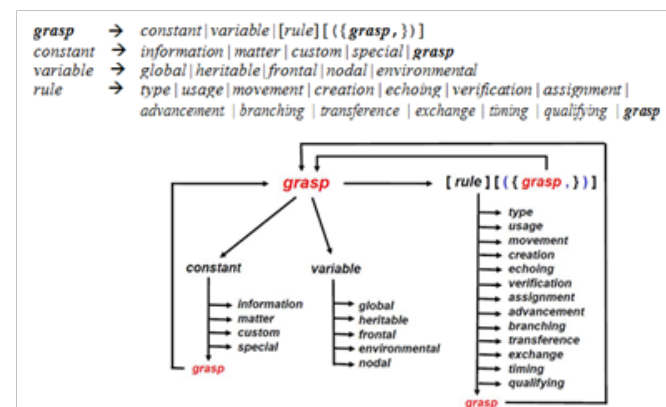


Figure 6 SGL recursive syntax.

The SGL scenario can dynamically *spread & process & match* the world or its parts needed, with scenario code capable of virtually or physically splitting, replicating, modifying and moving in the distributed spaces being accompanied with transitional data (including in special *frontal variables*). This movement can take place in single or multiple scenario parts dynamically and spatially interlinked under the overall control, which is spreading, covering, and matching the navigated world too.

Numerous communicating SGL interpreter copies can be installed worldwide in thousands to millions to billions of copies and integrated with other systems or operate autonomously in critical situations. They are forming altogether a sort of global collective spatial brain governed on top level by multiple and freely moving soul-like recursive scenarios in SGL.

SGT-based distributed mosaic simulation

We will be symbolically representing the mosaic space here by a regular distributed network model expressed in SGT with possibility of some nodes missing, also showing exemplary nonlocal operations on it which may need a sort of holistic vision of this distributed space to be successful.

Networked representation of mosaic space

Symbolic mosaic space consisting of differently colored tiles—units is shown in Figure 7, assuming the tiles representing nodes of a regular distributed network, with colors reflecting different orientation (or content) of the nodes.

The tiles—nodes may have different links with surrounding nodes. We are assuming here that each node has links with four direct

neighbors, if all exist, having common edges with them, as in Figure 8a (on its up, down, left, and right). It may also have additional links with the four corner neighbors (if all exist too) as in Figure 8b, thus generally having eight all neighbors in total, as in Figure 8c, if all such neighbors are present in the mosaic picture.

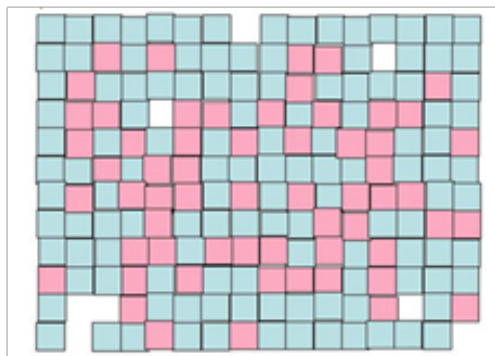


Figure 7 Mosaic space simplified model.

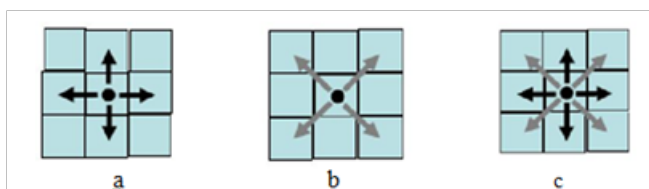


Figure 8 Links to neighboring tiles: a: direct_neighbors; b: corner_neighbors; c: all_neighbors.

We will be showing below two exemplary distributed mosaics-related tasks and their solutions on this simplified networked model.

Grouping particular type neighboring elements

The task is to find all fully interconnected groupings of units (as direct_neighbors to each other) of particular type (let it be pink) with given threshold number of components in such groups (say, not less than 4), and fix their entry or head units. This task may, for example, relate to runtime grouping of certain distributed military or security units into more powerful forces with each under united control. The related SGL scenario may be as follows (starting in parallel in all networked tiles-nodes, with the obtained resultant nodes-entries in each group having their names-values like X-Y coordinates as maximum in relation to the other group nodes):

```
hop(Headquarters);
```

```
nodal(Heads, Threshold = 4, Subordinates); Heads =(hop(all);  
TYPE == pink; IDENTITY = NAME; hopfirst(current);
```

```
Subordinates = contain(repeat(hopfirst(direct_neighbors); COLOR  
== pink; if(NAME < IDENTITY, abort); blind(NAME), stay);
```

```
count(Subordinates) + 1 >= Threshold; NAME); output(Heads)
```

The issued result will be: *Name1, Name2, Name3* (with these head names corresponding to the related X-Y node coordinates) as shown in Fig. 9, where the groped pink elements highlighted in red.

Example of a global order delivering some *action* command to all heads of the obtained integrated forces and then to their all subordinates (with both heads and subordinates, registered at heads, obliged of executing it) may be as follows:

```
hop(Heads); hop(Subordinates), stay; perform(action)
```

With a slight scenario complication, we could also appoint heads of the obtained groups as being topologically most central to the other group members, say, for more effective command and control of the whole groups, like shown in Figure 10.

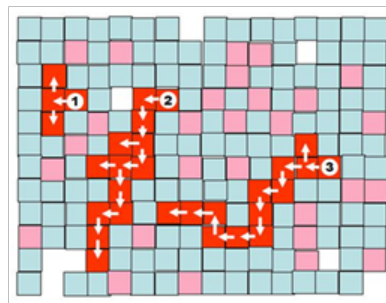


Figure 9 Grouping of direct neighbors under threshold given.

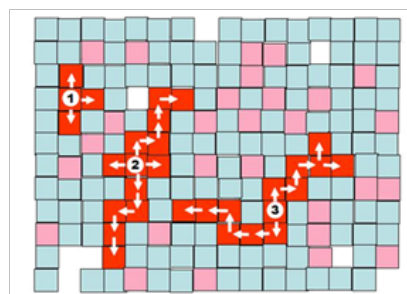


Figure 10 A better solutions for entry nodes of the groups found.

Collective surrounding and impacting of a danger element

Having discovered some alien or dangerous element in the distributed mosaic (like the tile in black in Figure 11), let us try to find the nearest full chain or “fence” around it consisting of certain (let them be pink again) elements (generally linked by all_neighbors to each other). Such full chain, if exists, will not let the danger element escape (assumed capable of moving only via non-pink tiles and as direct_neighbors to each other).

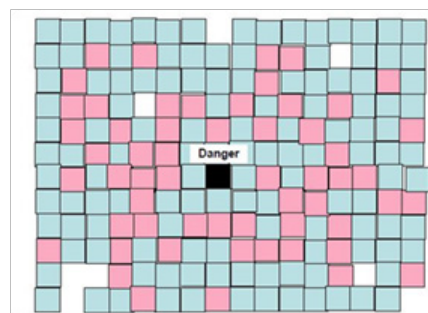


Figure 11 Discovering a danger element on distributed mosaics.

The following scenario is supposed to be applied from some mission Headquarters, which may be inside or outside the considered networked model. Starting from the black tile with known *Danger_coordinates*, it spreads in parallel throughout the tiled space via non-pink tiles as direct_neighbors, and as far as possible, with blocking

cycling to the already visited tiles. This is being done unless meets the first pink tiles, where it stops, as shown in Figure 12 (these finally reached pink tiles are highlighted now in red too).

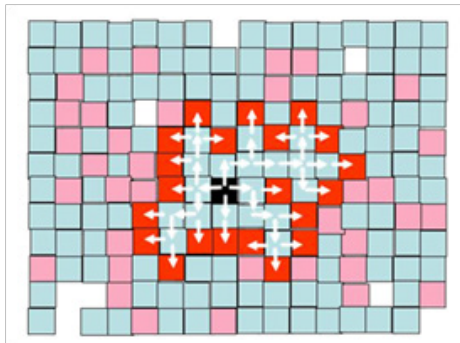


Figure 12 Finding a full chain of certain elements surrounding the danger tile.

The found names—addresses of nodes of this nearest fence are echoed to the Headquarters which then issues general command to them all to trace and destroy this black tile which may be moving, at any time they could be able to, which is accomplished by the following scenario.

```
hop(Headquarters);
nodal(Names)=(hopfirst(Danger_coordinates);
repeat(COLOR != pink; hopfirst(direct_neighbors)); NAME);
hop(Names);
trace_destroy(Danger_coordinates)
```

Let us make this scenario a bit more advanced. In the current solution, if no full chain of pink tiles exists around the Danger element, the spreading parallel spanning tree algorithm may not stop at all, or can just reach the tiled space boundaries if they are limited in distance. We may introduce certain allowed *max_distance* using variable Depth to forcefully stop this tree-spreading process if it exceeds the allowed depth, with indication and warning that the closed fence solution around the danger cannot be found. Also, in case of existence of the closed fence, we may involve in the tracing–destroying operation not only each finally reached pink (now red) tile separately, but also with their direct all_neighbors also belonging to the fence, say, by asking simultaneously the latter to assist in this operation. This setting cooperation between all perimeter neighbors, like of integral system, as shown in Figure 13 and by the extended SGL scenario that follows.

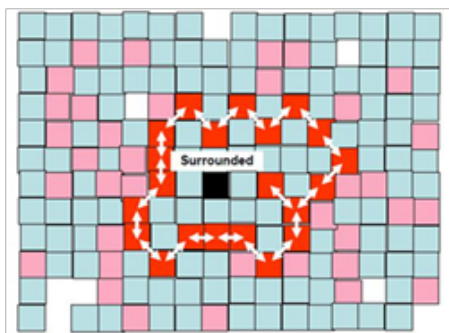


Figure 13 Setting cooperation between all neighboring elements of the full chain found.

```
hop(Headquarters);
```

```
frontal(Depth) = max_distance; nodal(Mark, Names); Names = contain(
```

```
hopfirst(Danger_coordinates);
```

```
repeat(COLOR != pink; hopfirst(direct_neighbors);
decrement(Depth, 1); if(Depth == 0, abort));
```

```
Mark = 1; NAME);
```

```
if(count(Names) == 0,
```

```
quit(output("failure")), hop(Names)); parallel(
```

```
trace_destroy(Danger_coordinates),
```

```
(hop(all_neighbors); Mark == 1;
```

```
assist_destroy(Danger_coordinates)))
```

Being fully surrounded by the red perimeter, the danger object (which may be trying to move and escape) will be constantly analyzed by all the perimeter units having received its coordinates and the destruction or assistance order, and finally caught or destroyed by some, say, closest to it red unit in cooperation with its neighbors, as shown in Figure 14.

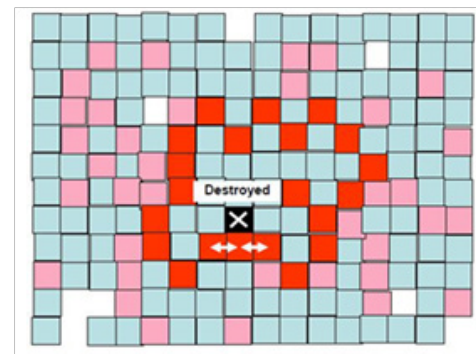


Figure 14 The final elimination of the danger object.

An extended surrounding scenario

The scenario above presumed that coordinates of the danger object were given in advance. But we may complicate and extend this scenario by adding initial analysis of the whole network on the presence of such dangerous, or black, tiles there (which may be more than one) with the return of their exact coordinates to the Headquarters. We may further organize in SGT parallel finding of individual fences for all discovered danger objects (such fences may happen to be common for more than one danger object) with issuing personal commands to trace and destroy them, as follows.

```
hop(Headquarters);
```

```
nodal(Targets)=(hop(all_nodes); COLOR == black; ADDRESS);
nonempty(Targets); split(Targets); frontal(Target) = VALUE;
IDENTITY = Target; nodal(Mark, Names); frontal(Depth) = max_
distance; Names = contain(hopfirst(Target);
```

```
repeat(COLOR != pink; hopfirst(direct_neighbors);
decrement(Depth, 1); if(Depth == 0, abort));
```

```
Mark = 1; NAME);
```

```

if(count(Names) == 0, stop(output("failure", Target)),
hop(Names));
parallel(trace_destroy(Target),
(hop(all_neighbors); Mark == 1;
assist_destroy(Target)))

```

More realistic links between mosaic tiles

In the examples above, we used some hypothetical mosaics model in the form of regular two-dimensional network structure (with possibility of some nodes missing), just to show how some SGT and SGL mechanisms could work in principle in fully distributed environments. In reality for this DARPA mosaics concept, links between mosaic elements may be more complex, irregular, not fixed in advance, but rather appearing dynamically, at runtime. The examples in the following sections are just describing such situations where mosaic elements may be represented as separate manned or unmanned units trying to solve different problems cooperatively, with the help of SGT.

Combining swarming with hierarchical command and control

Imagine that a distributed area needs to be investigated by multiple unmanned aerial vehicles that should randomly search the space, collect information on unwanted objects classifying them as targets, and organize collective reaction on the emerging threats. Different group functionalities for this can be expressed in SGL and effectively integrated into the resultant holistic group scenario, as follows, with more details on the related scenarios can be found in existing publications, including.¹⁵⁻¹⁷

Initial distribution of networked units

These units, like UCAVs, are presumed having embedded SGL interpreters (U), which can communicate, as shown in Figure 15. Their collective behavior (any imaginable or even so far unimaginable) can be organized by self-navigating spatial scenarios in SGL which can start from any unit and operate in parallel and fully distributed mode, without dedicated in advance centralized or external control facilities.

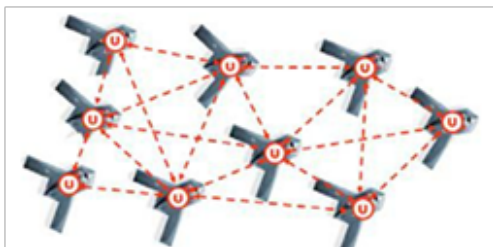


Figure 15 Networked UCAVs.

Swarm movement scenario

Randomized (within certain Limits) swarm movement of individual UCAVs, starting in any node and with minimum allowed threshold or Range distance between them can be organized by the following scenario (let us call this as *swarm_move*):

```

hop(all_nodes);
nodal(Limits = (dx(0,8), dy(-2,5)), Range = 200, Shift);

```

```

repeat(Shift = random(Limits);
if(empty(Shift, Range), WHERE += Shift); sleep(delay))

```

Finding topologically central unit

After finding such central unit, as in Figure 16, hopping into it, while starting this center finding from any unit, calling this as *find_hop_center*.

```

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

```

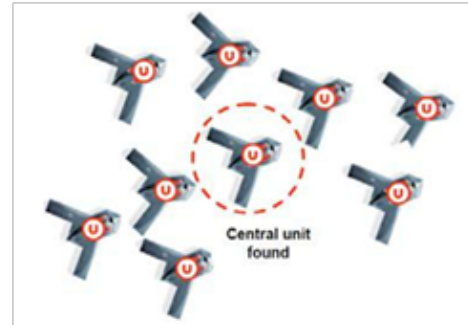


Figure 16 Runtime finding of central unit.

Creating runtime operational infrastructure

The infrastructure creation is starting from the found central unit with the use of oriented semantic links *infra* and maximum physical distance or Depth between nodes allowed to form direct control and operational links, let this be named *infra_build*, see also Figure 17.

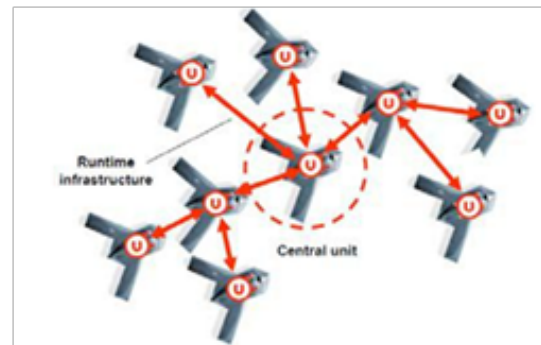


Figure 17 Creating runtime distributed operational infrastructure.

```

frontal(Depth = 300); firstcome(current); repeat(linkup(+infra,
firstcome, Depth))

```

Providing more details for this infrastructure creation, with first removing previous infrastructure links, if any:

```

frontal(Depth = 300); firstcome(current); repeat(hop(firstcome,
Depth);
stay_unlink(all); linkup(-infra, BACK))

```

Targets collection & distribution & impact

This distributed operation of discovering targets by individual units throughout the area covered, their collection in one point and further distribution to all units engaged, will be starting from the central unit found (naming this stage *collect_distribute_impact*), with

the following scenario:

```
frontal(Seen) = repeat(free_detect(targets), hop_links(+infra));
nonempty(Seen);
```

```
repeat(free_select_shoot(Seen), hop_links(+infra))
```

Full scenario integration

Using the accumulated SGL scenarios for different behavioral stages, as above, we can easily integrate them within the global behavioral & operational scenario. The resultant combined scenario will be uniting the previous SGL-based solutions named in bold while starting from any unit, and will be as follows (where *threshold_time* means certain allowed time interval):

```
hop(random, all_nodes);
parallel(swarm_move,
repeat(find_hop_center; stay(infra_build);
allowed(threshold_time,
sling(collect_distribute_impact))))
```

The obtained resultant scenario combines loose and randomly semi-oriented swarm movement in a distributed space with periodic updating of topologically central unit (as units are changing distances and relative positions in space), and regularly forming updated hierarchical infrastructure between them starting from the new central units. This infrastructure controls observation of distributed territory with collection of potential targets at the current central units, distributing them back to the vehicles for local assessment, selection and impact of most available or suitable targets.

Swarm against swarm aerial scenario

As a more complex scenario example in SGL we will consider here the case where an unmanned aerial swarm is fighting another manned or unmanned group/swarm, with symbolic swarms shown in Figure 18. More details on similar solutions can be found in the mentioned existing publications on SGT,^{15–17} including.



Figure 18 Different UCAVs swarming with potential conflict between swarms.

Basic ideas of a possible swarm against swarm fight scenario may be as follows, some also shown in Figure 19.

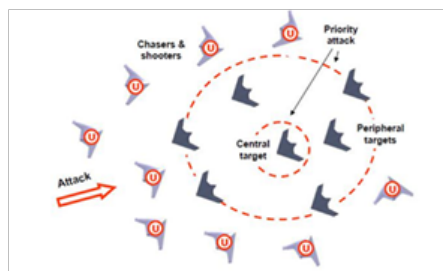


Figure 19 Swarm against swarm fight scenario ideas.

- Initial launch of the swarmed units (let us call them “chasers”), shown in Figure 17 with embedded SGL interpreters U which can communicate with each other, into the expected conflict area.
- Starting from randomly chosen unit, connecting with other chasers which collect and send back data on the hostile objects, as “targets”, seen in their vicinity.
- Forming the targets priority list by their locations in physical space, with maximum priority assigned to topologically central targets as potential control units of the enemy’s group.
- Other targets are sorted by their distances from the topological center of their group.
- Most peripheral targets, i.e. those in maximum distance from the topological center, are considered with higher (even highest) priority too, as potentially having more chances to escape.
- Assigning available chasers to most appropriate targets, classifying these chasers as *engaged*, with chasing & neutralizing the targets and subsequently returning into the status *vacant* after successful performance i.e. if not destroyed themselves. The vacant chasers are again engaged in the targets selection & impact procedure, and so on.
- If all collected targets processed, repeating the whole scenario from another randomly chosen chaser, if still exist, i.e. from step 2 for the search and impact of new targets, and so on.

This potentially parallel and distributed swarm-against-swarm scenario can be expressed in SGL in a compact form, as follows:

```
frontal(Chasers) = ..., Targets, Next, List, Center);
repeat(hop(random, Chasers, vacant);
```

```
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);
or(loop(nonempty(List); Next = withdraw(List, 1) : 2; free(min_
desination(hop(Chasers); STATUS == vacant; distance(WHERE,
Next)); STATUS = engaged;
```

```
If(pursue_neutralize(Next), STATUS = vacant))), stay))
```

It is worth mentioning that this chaser swarm management scenario executes exclusively within the swarm itself, without any central or external management, which can dramatically simplify organization of this and other similar multiple-drone operations.

Distributed driverless platoon management

Platooning, a closely spaced multiple-vehicle chain on a highway (as in Figure 20), has multiple benefits such as fuel saving, accident prevention, and so on. But it requires close cooperation among participating vehicles to maintain the platoon structure in case of different road situations. We will be showing below distributed management of different platoon’s states in SGL, with vehicles as symbolic mosaic “tiles” in DARPA’s recent terminology, where more examples and details on the related driverless management scenarios can be found in.^{15,17,21}

Regular management starting from platoon's head

The normal platoon management starts in the head vehicle, see Figure 21. It regularly accesses all vehicles in their chain while updating their speed to keep the needed standard distance between vehicles and orient the whole platoon on the speed of the head vehicle. This can be done by the following SGL scenario, where Number provides the number of vehicles in the platoon, with each vehicle identified by its numerical order in their chain.

```
frontal(Distance = ..., Number = ..., Order = 1, Speed); hop(Order);
sling(sleep(Delay); Speed = SPEED;
repeat(Order += 1 <= Number;
hop(behind, Order);
update(SPEED, (Speed, Distance, BEFORE))))
```

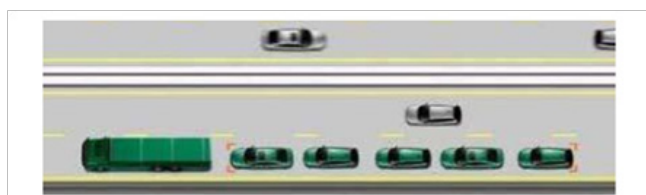


Figure 20 Cars platooning on roads.

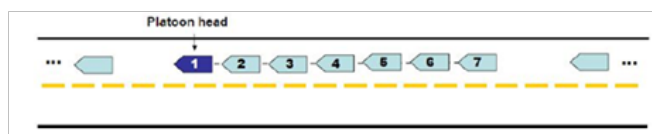


Figure 21 Regular collective platoon management.

Management of a fragmented platoon

Due to dynamic road conditions, traffic signals, road speed limits, and other factors like, for example, providing highest priority to emergency or police vehicles, a car platoon may suffer fragmentation. Such a situation is depicted in Figure 22, where between platoon vehicles 4 and 5 an emergency vehicle happened to appear on the same lane, which divided the platoon in two parts.

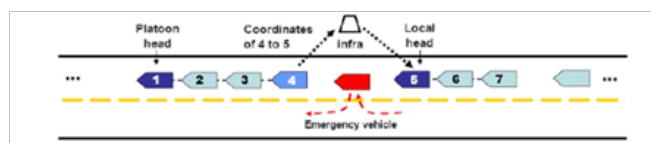


Figure 22 Collective management of a fragmented platoon.

For inclusion of such cases into platoon management, the previous scenario can be extended where in case of impossibility to contact next in line vehicle by V2V direct links, the current vehicle uses more powerful V2I links with the road infrastructure. This is to find and contact the next vehicle which may happen to be at some distance or even far away, and transfer to it the current physical coordinates for the subsequent possibility of coming into the needed vicinity again, which will also be influencing all remaining platoon vehicles (i.e. vehicles 6 and 7). The corresponding SGL scenario code will be as follows.

```
frontal(Distance = ..., Number = ..., Order = 1, Speed, Position);
```

```
hop(Order);
```

```
sling(sleep(Delay); Speed = SPEED;
```

```
repeat(Order += 1 <= Number;
```

```
or((hop(first_behind, Order);
```

```
update(SPEED, (Speed, Distance, BEFORE))),
```

```
(Position = WHERE; hop(or(behind, infra), Order);
update(Position))))
```

Returning to normal platoon operation

After vehicle 5 comes into the normal distance after vehicle 4 by appearing as the next nearest vehicle to it, and the distance between them is covered by V2V communications, the whole platoon will be operating in one piece again, as shown in Figure 23. This, however, will not guarantee the platoon for not being fragmented by certain forces and situations once more, so the previous scenario may be in action again.



Figure 23 Recovery of the platoon's structure.

Conclusion

We have discussed the newest DARPA Mosaic Warfare concept oriented on dynamic management of large distributed systems and possibility of its expression by Spatial Grasp model and Technology (SGT), with the latter potentially serving as a basic philosophy, ideology, methodology and technology for different mosaic-like organizations. Some investigated and already tested advantages of applicability of SGT for Mosaic Warfare may be as follows.

- Very compact, semantic level, spatial mission scenarios naming only main operations and decisions to be taken, with traditional system management and command and control routines effectively hidden inside automatic high-level language implementation.
- The spatially evolving and spreading scenarios can engage at different stages and places any available operational resources at runtime, during their development rather than ahead of it, and these resources may not be known in advance.
- Any explicit command and control can be expressed in SGL too, taking any centralized or distributed forms, which can be automatically updated and even completely changed at runtime in case of unexpected situations or damages.
- SGT can effectively provide distributed situational awareness by parallel worldwide coverage while collecting data not only from multiple separate points but also as a result of matching of complex patterns with large physical and virtual spaces.
- SGT can cover different networking layers from top semantic tasking to most basic communication protocols and routines, which can be especially vital during nonlocal crises where traditional communications, internet including, may fail to operate.

- f. The operational scenarios in SGL are well understood by both manned and unmanned components, allowing us to assemble mixed teams with any ratio between humans and robots which may be changing at runtime, while always preserving mission objectives and goal orientation.
- g. Already investigated SGT capabilities for extrapolating gestalt theory laws to holistic grasping of complex situations in distributed environments may be particularly useful for Mosaic Warfare in order to dominate in rapidly changing and unpredictable situations.

In the further engagement and research in this DARPA–launched mosaic concept, we plan to use the gained experience from previous versions of SGT (called WAVE) for distributed interactive simulation of large military systems.^{40,48–52} More conference, journal and book publications on integration of Mosaic Warfare with SGT are planned too, truly believing in importance of this mosaic concept which can moving toward global military and security doctrine in the 21st century.

Acknowledgments

None.

Conflicts of interest

The author declares there are no conflicts of interest.

Funding

None.

References

1. Bellamy W. DARPA Seeks ‘Mosaic Warfare’ Approach to Future Conflicts. *Rotor&Wing International*. 2018.
2. Magnuson S. DARPA Pushes ‘Mosaic Warfare’ Concept. *National DEFENCE*. 2018.
3. Daigle L. DARPA office unveils “mosaic warfare” approach to future conflicts. *Military EMBEDDED Systems*. 2019.
4. Williams BD. DARPA’s ‘mosaic warfare’ concept turns complexity into asymmetric advantage. *FIFTH DOMAIN*. 2017.
5. Strategic Technology Office Outlines Vision for “Mosaic Warfare”. *CHIPS*. 2017.
6. Strategic Technology Office Outlines Vision for “Mosaic Warfare”. *DARPA*. 2017.
7. Piecing Together the Future Battlefield with Mosaic Warfare at DCS. *PHOTONICS MEDIA*. 2019.
8. Strategic Technologies. FedBizOpps.gov. 2019.
9. Koffka K. Principles of Gestalt psychology. London: Lund Humphries; 1935.
10. Wagemans J, Elder JH, Kubovy M, et al. A Century of Gestalt psychology in Visual Perception: I. Perceptual Grouping and Figure-Ground Organization. *Psychol Bull*. 2012;138(6):1172–1217.
11. Bradley S, Design Principles: Visual Perception And The Principles Of Gestalt. *Smashing Magazine*. 2014.
12. Sapaty P. Gestalt-Based Ideology and Technology for Spatial Control of Distributed Dynamic Systems, International Gestalt Theory Congress. 16th Scientific Convention of the GTA; 2009 March 26–29; Germany: University of Osnabrück; 2009.
13. Sapaty P. Gestalt-Based Integrity of Distributed Networked Systems. bcc Berliner Congress Centre; 2009; Germany. Berlin: SPIE Europe Security + Defence; 2009.
14. Sapaty P. A Distributed Processing System, European Patent No. 0389655, Publ. 10.11.93, European Patent Office, Munich; 1993.
15. Sapaty P. Complexity in International Security: A Holistic Spatial Approach. Emerald Publishing, 2019.
16. Sapaty P. Holistic Analysis and Management of Distributed Social Systems. *Springer*. 2019.
17. Sapaty P. Managing Distributed Dynamic Systems with Spatial Grasp Technology. *Springer*. 2017.
18. Sapaty P. Ruling Distributed Dynamic Worlds. New York: John Wiley & Sons; 2005.
19. Sapaty P. Mobile Processing in Distributed and Open Environments. New York: John Wiley & Sons; 1999.
20. Sapaty P. Conflict and Emergency Management in a Post-Liberal World. *International Relations and Diplomacy*. 2019;7(1):14–36.
21. Sapaty P. Distributed Control Technology for Management of Roads with Autonomous Cars. *International Journal of Intelligent Unmanned Systems*. 2017;5(2/3):83–98.
22. Sapaty P. Towards Global Goal Orientation, Robustness and Integrity of Distributed Dynamic Systems. *Journal of International Relations and Diplomacy*. 2016;4(6):418–425.
23. Sapaty P. Wholeness and Integrity of Distributed Dynamic Systems. *Journal of Computer Science & Systems Biology*. 2016;9(3):1–3.
24. Sapaty P. Towards Massively Robotized Systems under Spatial Grasp Technology. *Journal of Computer Science & Systems Biology*. 2016;9(1).
25. Sapaty P. A Brief Introduction to the Spatial Grasp Language (SGL). *Journal of Computer Science & Systems Biology*. 2016;9(2).
26. Sapaty P, Distributed Human Terrain Operations for Solving National and International Problems. *International Relations and Diplomacy*. 2014;2(9).
27. Sapaty P. *The World as an Integral Distributed Brain under Spatial Grasp Paradigm*. London. UK: Intelligent Systems for Science and Information; 2013.
28. Sapaty P. Providing Global Awareness in Distributed Dynamic Environments. London: International summit ISR. 2013.
29. Sapaty P. Withstanding Asymmetric Situations and Threats in Distributed Dynamic Worlds. *Journal of Mathematical Machines and Systems*. 2012.
30. Sapaty P. Global Electronic Dominance with Spatial Grasp. *IJCNS*. 2012;5(11):724–735.
31. Sapaty P. Meeting the World Challenges with Advanced System Organizations. Informatics in Control Automation and Robotics. 2011;85:29–46.
32. Sapaty P. High-Level Technology to Manage Distributed Robotized Systems. Proceedings of Military Robotics; 2010 May 25–27; UK: Jolly St Ermins; 2010.
33. Sapaty P, M. Sugisaka, Countering Asymmetric Situations with Distributed Artificial Life and Robotics Approach. Proc. Fifteenth International Symposium on Artificial Life and Robotics; 2010 February 5–7. Japan; 2010.
34. Sapaty P. Distributed Technology for Global Control. *Informatics in Control, Automation and Robotics*. 2009;37(1):3–24.
35. Sapaty P. Gestalt-Based Integrity of Distributed Networked Systems. Berlin. Germany: SPIE Digital Library; 2009.

36. Sapaty PS. A New Technology for Integration, Simulation, and Testing of Distributed Dynamic Systems, NATO Proceedings Integration of Simulation with System Testing, RTO-MP-083, AC/323(SCI-083)TP/43; 2002. 12 p.
37. Sapaty PS. Cooperative exploration of distributed worlds in WAVE. *Artificial Life and Robotics*. 2000;4(2):109–118.
38. Sapaty PS. High-level spatial scenarios in WAVE. Proceedings of the International Symposium AROB 5th Conference, Japan; 2000. p. 301–304.
39. Sapaty PS. Cooperative conquest of distributed worlds in WAVE. Proceedings of the Symposium and Exhibition of the Unmanned Systems of the New Millennium, AUUSI'99, Baltimore: MD; 1999.
40. Sapaty PS. Mobile Programming in WAVE. *Mathematical Machines and Systems*. 1998;1:3–31.
41. Sapaty PS. *Live demonstration of the WAVE system and applications at the Workshop on Mobile Agents and Security 97, Maryland Center for Telecommunications Research*. Maryland: UMBC; 1997. p. 27–28.
42. Sapaty PS. *WAVE: creating dynamic worlds based on mobile cooperative agents, Dartmouth Workshop on Transportable Agents*. New Hampshire: Dartmouth College; 1996.
43. Sapaty PS. Mobile wave technology for distributed knowledge processing in open networks, Proc. Workshop on New Paradigms in Information Visualization and Manipulation, in conjunction with the Fourth International Conference on Information and Knowledge Management (CIKM'95). Baltimore: Maryland; 1995.
44. Sapaty PS, Corbin MJ, Borst PM. Towards the development of large-scale distributed simulations, Proc. 12th Workshop on Standards for the Interoperability of Distributed Simulations, IST UCF, and Orlando: Florida; 1995. p. 199–212.
45. Sapaty PS, Corbin MJ, Borst PM. Mobile WAVE programming as a basis for distributed simulation and control of dynamic open systems. A special session on the WAVE Technology at the 15th Intl. Conf. on Distributed Computing Systems. Vancouver, BC: Canada; 1995.
46. Sapaty PS, Corbin MJ, Seidensticker S. Mobile Intelligence in Distributed Simulations, Proc. 14th Workshop on Standards for the Interoperability of Distributed Simulations, IST UCF. Orlando: Florida; 1995.
47. Sapaty PS, Borst PM, Corbin MJ, et al. Towards the intelligent infrastructures for distributed federations, Proc. 13th Workshop on Standards for the Interoperability of Distributed Simulations. IST UCF. Orlando: Florida; 1995. p. 351–366.
48. Sapaty PS, Corbin MJ, Borst PM. Mobile WAVE programming as a basis for distributed simulation and control of dynamic open systems. Report at the 4th UK SIWG National Meeting, SGI Reality Centre, Theale, Reading; 1994.
49. Sapaty PS. The WAVE paradigm, Proc. JICSLP'92 Post-Conference Joint Workshop on Distributed and Parallel Implementations of Logic Programming Systems. Washington, D.C. USA; 1992. p. 13–14.
50. Sapaty PS. *Logic flow in active data, in VLSI for Artificial Intelligence and Neural Networks*. Moore WR, Delgado-Frias J, editors. UK: Plenum Press; 1991.
51. Sapaty PS, Zorn W. The WAVE model for parallel processing and its application to computer network management. Intl. Networking Conference INET'91. Copenhagen: Denmark; 1991.
52. Sapaty P, Corbin MJ, Borst PM. Mobile WAVE programming as a basis for distributed simulation and control of dynamic open systems". Report at the 4th UK SIWG National Meeting, Theale: SGI Reality Centre; 1994.
53. Sapaty P. The WAVE paradigm. Proceedings of. JICSLP'92 Post-Conference Joint Workshop on Distributed and Parallel Implementations of Logic Programming Systems: Washington DC; 1992.
54. Sapaty P. *Logic flow in active data, in VLSI for Artificial Intelligence and Neural Networks*. Moore WR, Delgado-Frias J, editors. New York: Plenum Press; 1991.
55. Sapaty P, Zorn W. The WAVE model for parallel processing and its application to computer network management. Intl. Networking Conference INET'91; Copenhagen; 1991.
56. Sapaty PS. The WAVE machine project, Proc. IFIP Workshop on Silicon Architectures for Neural Nets. France: Springer. 1990;28–30.
57. Sapaty PS. The WAVE Model for advanced knowledge processing, in CAD Accelerators. In: Ambler P Agrawal, WR Moore (Editors). *Elsevier Science Pub*. 1990.
58. Sapaty PS. The WAVE model for advanced knowledge processing. England: Springer. 1989.
59. Sapaty PS. *WAVE-1: A new ideology of parallel processing on graphs and networks, Future Generations Computer Systems*. North-Holland: Elsevier. 1988:1–14.
60. Sapaty PS. WAVE-1: A new ideology of parallel processing on graphs and networks, Proc. Intl. Conf. Frontiers in Computing. Netherlands; 1987.
61. Sapaty PS. *The WAVE-1: A new ideology and language of distributed processing on graphs and networks, computers*. Netherlands: Elsevier Science Publishers; 1988.
62. Sapaty PS, Varbanov A, Ilijenko. The WAVE Model and Architecture For Knowledge Processing, Proc. Fourth Int. Conf. Artificial Intelligence And Information-Control Systems Of Robots, Smolenice: Springer; 1987.
63. Sapaty PS. The wave approach to distributed processing of graphs and networks, Proc. Int. Working Conf. Knowledge and Vision Processing Systems, Smolenice: Springer; 1986.
64. Sapaty PS. A wave language for parallel processing of semantic networks. *Computers and Artificial Intelligence*. 1986;5(4):289–314.
65. Sapaty PS, Varbanov S, Dimitrova M. Information systems based on the wave navigation techniques and their implementation on parallel computers, Proc. Intl. Working Conf. Knowledge and Vision Processing Systems, Smolenice. 1986
66. Sapaty PS, Kocis I. A parallel network wave machine, Proc. 3rd Int. Workshop PARCELLA'86. Berlin: Akademie-Verlag; 1986.
67. Sapaty PS. The WAVE-0 language as a framework of navigational structures for knowledge bases using semantic networks. Proc. USSR Academy of Sciences. Technical Cybernetics. 1986.
68. Sapaty PS. A wave approach to the languages for semantic networks processing, Proc. Int. Workshop on Knowledge Representation. Section 1: Artificial Intelligence. Ukraine: Springer; 1984.