UDC 623.764

P.S. SAPATY[*]

ADVANCED DISTRIBUTED MANAGEMENT OF ROADS WITH DRIVERLESS CARS

[*]Institute of Mathematical Machines and Systems Problems, National Academy of Sciences of Ukraine, Kyiv, Ukraine

Анотація. *Очікуване масове використання автономних транспортних засобів може кардинально змінити стратегії і бізнес-моделі для виробників автомобілів. Але автомобілі без водіїв можуть ще довго залишатись просто іграшками, тому що головною та надзвичайно складною проблемою стає ефективне автоматичне управління дорогами з численними автономними автомобілями. Спробуємо пролити світло на фактичне управління складними дорожніми мережами з безпілотними автомобілями, використовуючи для цього розроблену високорівневу мережеву технологію, перевірену на багатьох використаннях як у цивільній, так і військовій сфері. Поруч з оптимізацією на рівні транспортної інфраструктури будуть показані рішення для виникаючих різноманітних місцевих проблем за допомогою розподілених ситуативних сценаріїв, які можуть динамічно охоплювати будь-які території з довільною кількістю взаємодіючих транспортних засобів у них. Кожне рішення буде представлене та пояснене на спеціальній мові високого рівня, яка може ефективно інтерпретуватись у повністю розподіленому режимі, без будь-яких центральних засобів і з абсолютною мобільністю надзвичайно компактного коду операційних сценаріїв. Частина цього матеріалу прийнята до презентації на Всесвітній конференції по майбутньому транспорту в Кельні (Німеччина) 5–6 липня 2017 р.*

Ключові слова: *автомобілі без водіїв, управління транспортною мережею, технологія розподіленого керування, динамічні сценарії, що саморозвиваються, шеренга автомобілів, пошук найкоротшого шляху.*

Аннотация. *Широко ожидаемое использование автономных транспортных средств может кардинально изменить стратегию и бизнес-модели для производителей автомобилей. Но автомобили без водителей могут еще долго оставаться просто игрушками, потому что главной и чрезвычайно сложной проблемой становится эффективное автоматическое управление дорогами с многочисленными автономными автомобилями. Попытаемся пролить свет на реальное управление сложными дорожными сетями с беспилотными автомобилями, используя для этого высокоуровневую сетевую технологию, проверенную на многих приложениях как в гражданской, так и военной сфере. Наряду с оптимизацией на уровне транспортной инфраструктуры будут показаны решения для возникающих локальных проблем с помощью распределенных ситуативных сценариев, динамически охватывающих произвольные регионы с любым количеством взаимодействующих транспортных средств. Каждое решение будет представлено и объяснено на специальном языке высокого уровня, который может эффективно интерпретироваться в полностью распределенном режиме, без каких-либо центральных ресурсов и с высокой мобильностью чрезвычайно компактного кода операционных сценариев. Часть этого материала принята к презентации на Всемирной конференции по будущему транспорта в Кёльне (Германия) 5–6 июля 2017 г.*

Ключевые слова: *автомобили без водителя, управление транспортной сетью, технология распределенного управления, динамические саморазвивающиеся сценарии, шеренга автомобилей, поиск кратчайшего пути.*

Abstract. *The widely expected use of autonomous cars can change the landscape for car manufacturers who can rethink business models. But driverless cars may remain just toys for a long time, because the main, and extremely complex, problem may be the effective automatic management of roads with numerous driverless cars. We will try to shed some light on the real management of complex road networks with autonomous cars using high-level networking technology already tested on many applications, both civil and military. Along with optimisation on the infrastructure level, solutions will be shown for local problems by distributed emergency scenarios dynamically covering arbitrary regions with any number of communicating vehicles. Each solution will be presented and explained in a special high-*

© Sapaty P.S., 2017

level language which can be effectively interpreted in a fully distributed mode, without any central resources, and with high mobility of extremely compact scenario code. Part of this material has been accepted for presentation at The Future of Transportation World Conference in Cologne (Germany) July 5–6, 2017.
Keywords: *driverless cars, autonomous cars, road network management, distributed control technology, dynamic self-evolving scenarios, car platooning, shortest path finding.*

1. Introduction: Driverless Cars and advanced Road Management

Autonomous vehicles represent one of the most prominent technologies since the creation of automobile itself. World most influential companies are making billion-dollar investments in the driverless transport. Autonomous vehicles can fundamentally change transportation by reducing crashes, energy/fuel consumption, pollution and the costs of congestion. Human error is estimated to cause more than 90% of traffic accidents, a percentage that might be drastically reduced by the implementation of self-driving cars featuring smart systems that control most aspects of driving.

An autonomous (driverless, self-driving, or robotic) car is a vehicle that is capable of sensing its environment and navigating without human input, and many such vehicles are being developed. Autonomous cars use a variety of techniques to detect their surroundings, such as radar, laser, GPS, odometry, and computer vision. Advanced control systems interpret sensory information to identify appropriate navigation paths and obstacles. The levels of vehicle’s autonomy are often classified as follows.

Level 0: The human driver controls everything like steering, brakes, throttle, power, etc.

Level 1: Most functions are still controlled by the driver, but steering or accelerating can be automated.

Level 2: Additionally to Level 1, cruise control and lane-centering can be automated. The driver can be disengaged from physically operating the vehicle, but still must always be ready to take control of the vehicle if needed.

Level 3: Drivers are still necessary, but can completely shift safety-critical functions to the vehicle under certain traffic or environmental conditions. The driver is still present but is not required to monitor the situation in the same way as for the previous levels.

Level 4: This means an essentially autonomous level where vehicles can perform all safety-critical driving functions and monitor roadway conditions for the entire trip. This, however, may not cover every driving scenario.

Level 5: This expects vehicle's performance being equal to that of a human driver, including extreme environments like, say, dirt roads.

Another challenging problem, even more complex that the development of autonomous cars itself, is the effective management of road networks with many driverless cars and considerable reduction of human involvement in the management process, on both local and global levels (Fig. 1). This is an extremely difficult and so far unexplored task, whose success or failure will be determining the fate of this promising driverless field for the decades to come.

The rest of this paper briefs the developed Spatial Grasp model and Technology (SGT) allowing us to solve arbitrary complex problems in large distributed networked systems, which has been tested on a variety of civil and military applications and which can be useful for solving numerous regular and emergency tasks of road management with many driverless cars.

Using Spatial Grasp Language (SGL), the core of SGT, some basic emergent situations on roads with their solutions will be described and explained in detail, where vehicles can directly communicate with each other for finding suitable collective solutions using either short distance, direct V2V communication channels, or for longer distances access the infrastructure levels by V2I types of communications. Among the tasks considered in SGL will be finding optimum route in an arbitrary road network, to be used to guide the car which requested this route. The solution is shown on the infrastructure level in a fully distributed mode by navigating the latest, dynamic, and scattered data on roads rather than acquiring this from the static maps. Conclusions and references conclude the paper.

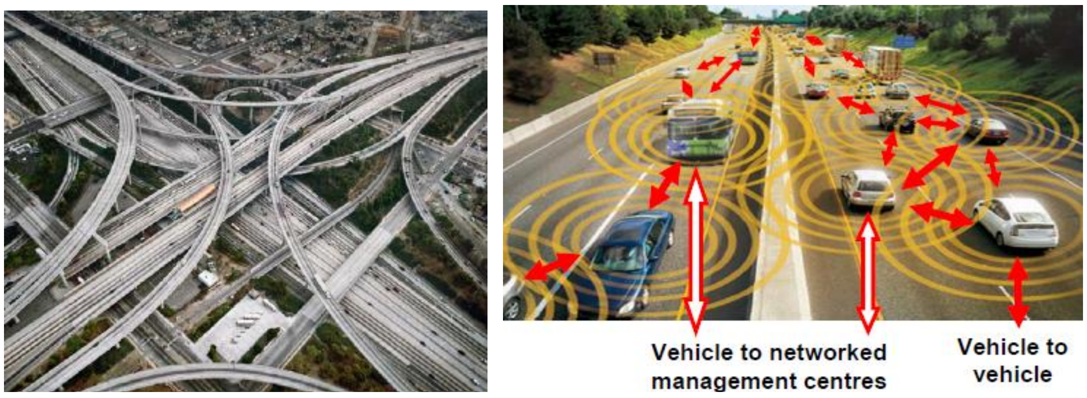

Fig. 1. Road networks with driverless cars

2. Spatial Grasp Technology, SGT

2.1. SGT General Issues

Starting from any point of space, SGT [5–12] allows us to create distributed operational infrastructure in a highly dynamic virus-like mode with unlimited code mobility in computer networks. These infrastructures, covering any regions needed, can solve complex spatial problems in them without any central resources and in parallel. Such emergent infrastructures can effectively withstand different unpredictable, crisis, and asymmetric situations in distributed systems of both civil and defence orientation. The created infrastructures can self-recover and self-repair after indiscriminate damages while always securing mission objectives. After the task completion, the infrastructures can also self-clean and self-remove if not needed any more. The key element of SGT is its Spatial Grasp Language, SGL, in which all mission scenarios are formulated.

2.2. Spatial Grasp Language and its Distributed Interpretation

Pattern-based SGL can provide highly integral, holistic, gestalt-based solutions directly in physical, virtual, and executive worlds. SGL has universal recursive structure capable of representing any parallel and distributed algorithms in distributed environments (Fig. 2 *a*).

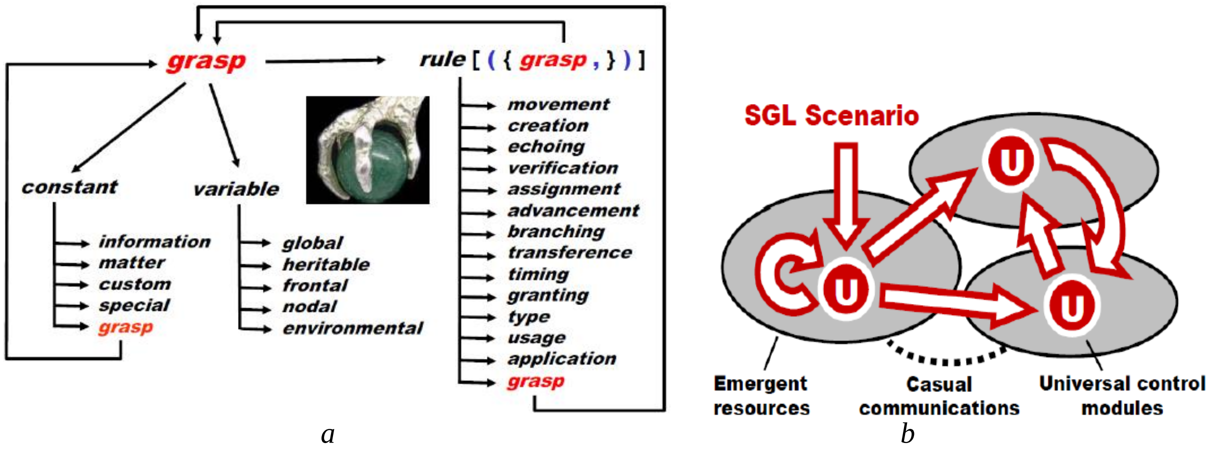

Fig. 2. Spatial Grasp Language (*a*) and its distributed interpretation (*b*)

SGL is collectively interpreted by a network of universal control modules U, as SGL interpreters, embedded into key system points (humans, robots, sensors) with absolute scenario code and data mobility in space (Fig. 2 *b*). SGL scenarios can start from any node, covering at runtime the whole system or its parts needed with operations and control.

Spreading SGL scenarios can create knowledge infrastructures arbitrarily distributed between system components (robots, sensors, humans). Navigated by same or other scenarios, these can effectively support distributed databases, C2, situation awareness, and autonomous decisions. Also simulate any other existing or hypothetic computational and/or control models.

SGL interpreter consists of a number of specialized modules handling & sharing specific data structures. The whole network of the interpreters can be mobile and open, changing the number of nodes and communication structure in between at runtime. A backbone of the distributed interpreter is its spatial track system providing overall integrity, global awareness & automatic C2 over distributed processes.

The dynamically networked SGL interpreters extended by and integrated with other facilities and gadgets, like mobile robots, can form universal spatial machines operating with both information and physical matter. These networked machines, working without any central resources under intelligent scenarios injected at any time and from any nodes, can perform complex computational, knowledge processing and control operations.

2.3. Embedding SGL Interpreters in Distributed Systems

By embedding SGL interpreters into robotic vehicles and traditional electronic devices associated with humans (as in Fig. 3) we can easily organize any needed collective behavior of them.

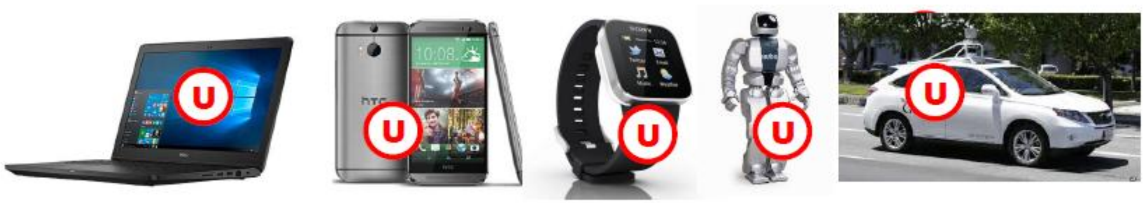

Fig. 3. Embedding SGL interpreter into different devices

SGT allows us to integrate into holistic teams any similar or dissimilar components operating under unified distributed command and control. The collective mission scenario can start from any unit and cover, activate, and control at runtime the whole group (Fig. 4).

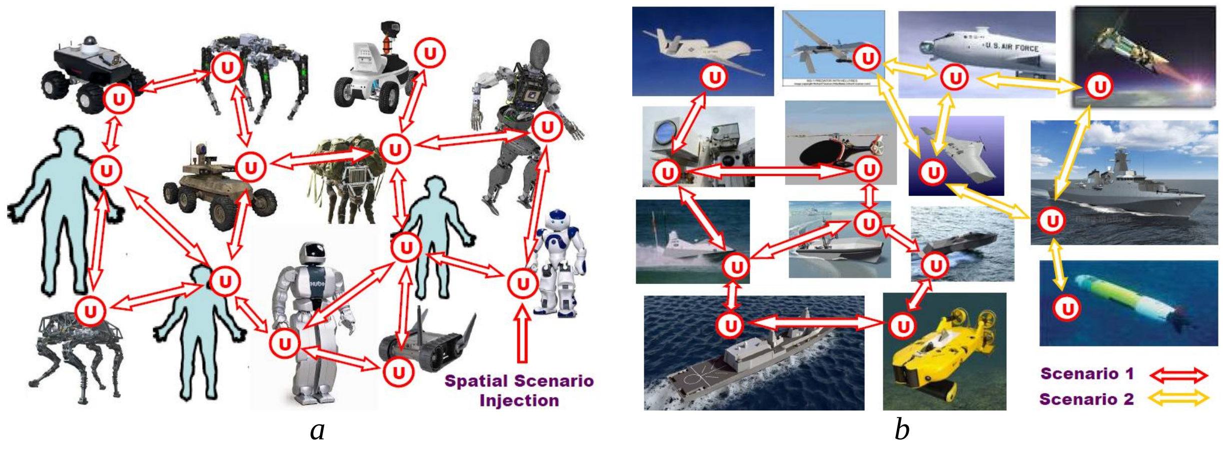

Fig. 4. Distributed teaming under SGT: a) Heterogeneous human-robotic collectives; b) Heterogeneous manned-unmanned defense solutions

Collective road management under SGT with the involvement of multiple driverless cars (Fig. 5) can mark a breakthrough in the creation of advanced transportation infrastructures with considerable reduction of human involvement, where holistic spatial intelligence provided by the technology can challenge if not overcome the collective capabilities of human drivers when solving both regular and critical situations on roads and highways.

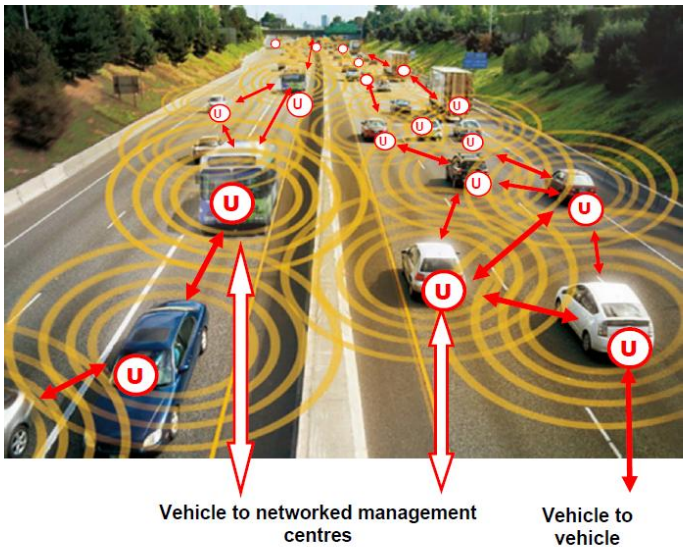

Fig. 5. Cooperative road management capabilities under SGT

3. Examples of Collective Road Management Solutions

Exemplary SGL scenarios of solving some very basic management problems on roads with autonomous cars, where cars can directly communicate and cooperate with each other, will be presented and explained. These solutions can start from any car and dynamically involve as many cars as needed in its neighbourhood, also covering any regions needed within collective intelligent behaviours. The dynamically created operational infrastructures automatically cease to exist if not needed any more or can remain any period if time if continue to support lasting or regular situations.

3.1. Narrowing the Gap before an Individual Vehicle

Any vehicle seeing enough empty space before it (with certain given gap threshold) and also getting information on the speed of the nearest vehicle ahead can update, say increase, its speed if its current speed and potential maximum speed can allow this. The vehicle can constantly make such checking in certain time intervals, with the corresponding solution shown in Fig. 6 (di as current distances between neighbouring vehicles, improvised spatial sensing-communication-processing diagram is used), and the SGL scenario code below.

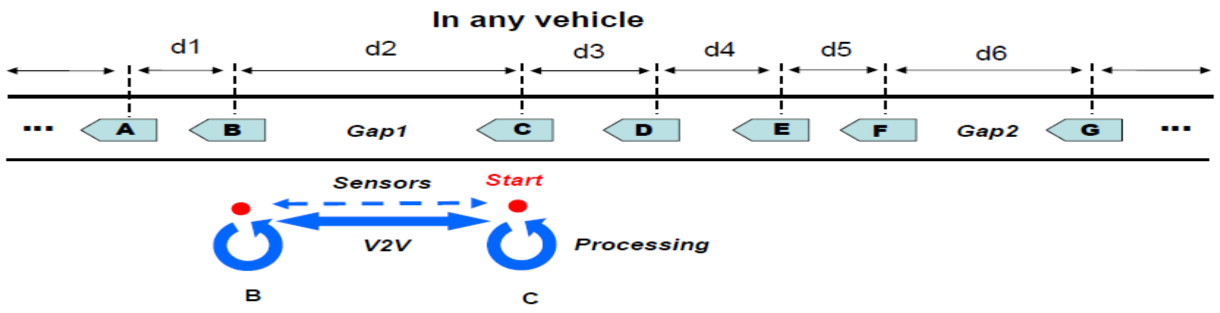

Fig. 6. Narrowing the road gap before a vehicle

```
nodal(Gap = …, Far, Speed_ahead);
sling(
  sleep(Delay);
  Far = distance(ahead) > Gap;
  Speed_ahead = (hop(first_ahead); SPEED);
  update(SPEED, (MAXSPEED, Speed_ahead, Far)))
```

Natural language comments on this compact and formal SGL scenario may be as follows.
• Define types and initial values of spatial variables used.
• Repeat the following regularly with certain time intervals.
• Use sensors to measure nearest distance to the vehicles ahead.
• Do the following if the gap ahead exceeds the threshold given.
• Try to enter the first vehicle ahead via V2V, copy and bring back its current speed, which may be nil if this action fails.
• Update the current vehicle's speed taking into account its current speed, possible maximum speed, current speed of the vehicle ahead, and distance to the first vehicle ahead.

3.2. Collective Simultaneous Narrowing the Gap before a Sequence of Vehicles

If a vehicle, like in the previous case, has a gap before it, and the directly following vehicle, as well as possibly all other following ones in the chain, have distances in between below threshold given, the speed changing operation can be done in all of them almost simultaneously, with the first one as the leader and all others trying to negotiate new speed in a direct contact with the previous vehicle. In a normal situation all vehicles behind the first one may refrain from changing their speed at this moment of time, but the first vehicle behind the gap can convince them this is the proper time, and for common benefit. This situation is shown in Fig. 7 and by SGL scenario that follows.

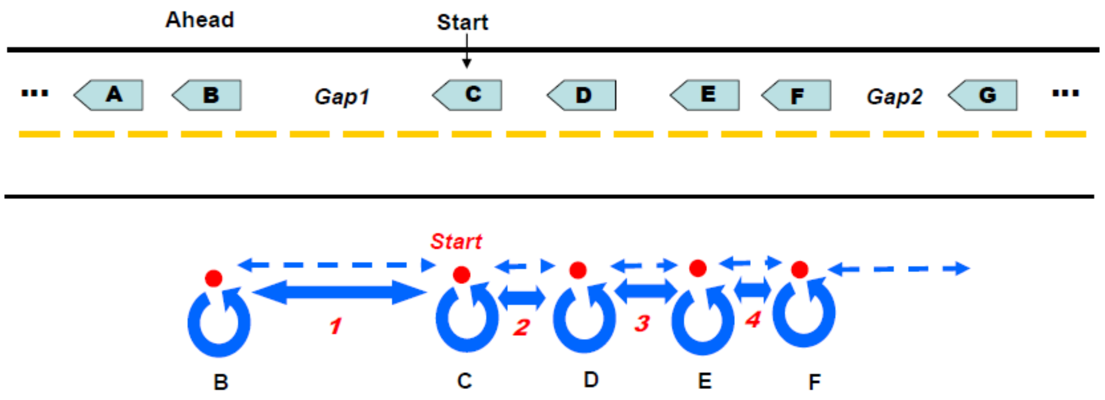

Fig. 7. Collective gap narrowing

```
frontal(Gap = …, Speed_ahead, Far);
sling(
 sleep(Delay);
 Far = distance(ahead) > Gap;
 Speed_ahead = (hop(first_ahead); SPEED);
 repeat(
  update(SPEED, (MAXSPEED, Speed_ahead, Far));
  Far = distance(behind) < Gap; Speed_ahead = SPEED;
  hop(next_behind)))
```

Informal natural language description of this scenario will be as follows.
• Define types and initial values of spatial variables used.
• Repeat the following regularly with certain time intervals.

• Use sensors to measure nearest distance to vehicles ahead.

• Do the following if the gap ahead exceeds threshold given, this vehicle declared to be the first and leading one.

• Try to enter the first vehicle ahead via V2V, copy and bring back its current speed, which may be nil if this action fails.

• Do the following repeatedly until possible.

• Update the current vehicle's speed taking into account its current speed, possible maximum speed, current speed of the vehicle ahead, and distance to the first vehicle ahead.

• Use sensors to measure nearest distance to vehicles behind.

• If it is less than the given gap threshold, declare the current speed as the speed of the vehicle ahead, enter the first vehicle behind via V2V, which becomes the current one now.

3.3. Lane Manoeuvring for the Fastest Vehicle between Two Gaps

A situation may occur that in a chain of vehicles with distances in between below threshold, like in the previous case, there may be vehicles with quite different maximum possible speed, and it could be reasonable to let the fastest one to make a solo lane manoeuvre and appear before the first vehicle behind the gap on the road. This situation is shown in Fig. 8 and by SGL scenario that follows, where, starting from the first vehicle, the whole chain of them is 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.

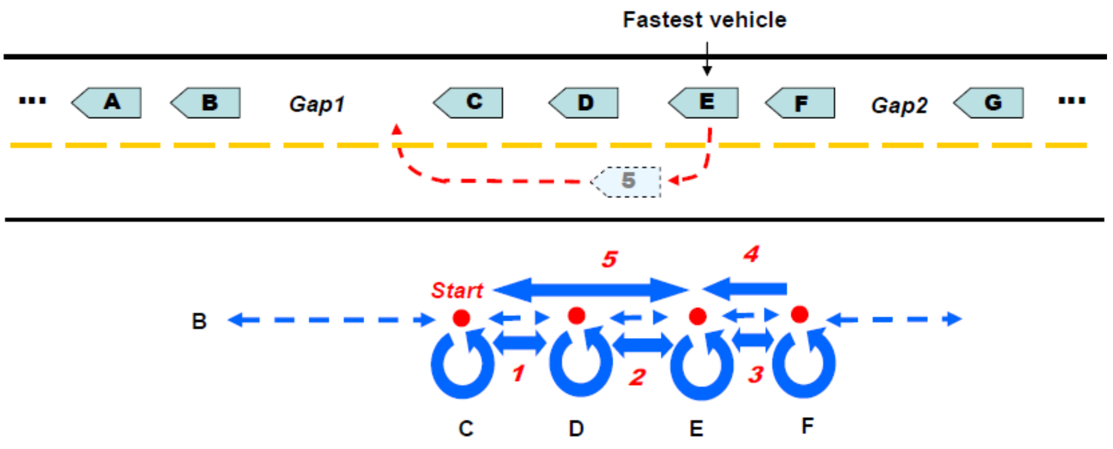

Fig. 8. Lane manoeuvring for the fastest vehicle

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

A natural language explanation of this solution may be as follows.

• Define types and initial values of spatial variables used.

• Repeat the following regularly with certain time intervals.

• 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.

• 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.

• 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.

3.4. Collective Avoiding of a Broken Car, with Activity Starting from the Broken Vehicle

A vehicle may accidentally stop, say, being damaged or somehow malfunctioning. Assume also that this vehicle can still electronically communicate with other vehicles on its initiative to inform them to avoid itself, and its distance to the nearest vehicle ahead is big enough for a collective lane manoeuvring for the whole chain of vehicles behind it. Such collective scenario launched by the initiative of the damaged vehicle is depicted in Fig. 9 with the corresponding SGL code following.

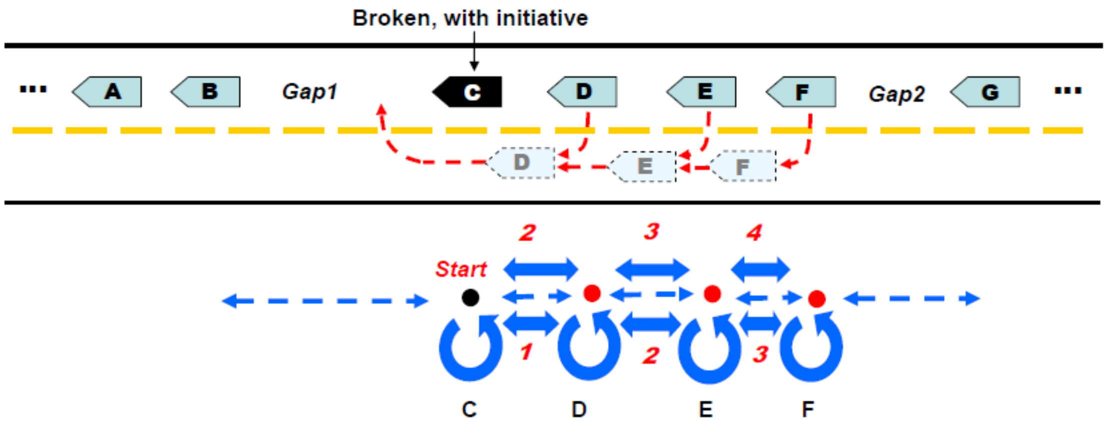

Fig. 9. Collective avoiding of the broken car by its initiative

```
frontal(Gap = …, Status);
sling(
  sleep(Delay);
  STATE == down; distance(ahead) > Gap;
  Status = head;
  repeat(
    distance(behind) < Gap; hop(first_behind);
    free(lane_manouvre(Status, BEFORE));
    Status = follower))
```

Natural language comments on this scenario may be as follows.

• Define types and initial values of spatial variables used.

• Repeat the following regularly and with certain time intervals.

• If the vehicle's state is "down", and measured by sensors nearest distance to vehicles ahead exceeds the given threshold, allow the following.

• Declare a possible next vehicle's status as "head" of a possible platoon-like chain of vehicles.

• Do the following repeatedly until possible.

• Use sensors to measure distance to vehicles behind, and if less than the given gap threshold, enter the first vehicle behind via V2V.

• Activate (in parallel with the rest of the scenario) lane manoeuvre procedure to avoid the damaged vehicle (with the “head” communicating with and orienting on the damaged one while others just interacting with and following the previous vehicles wherever they go and by which lane.

• Setting the state of a possible next vehicle as “follower”.

3.5. Collective Avoidance Activity Starting from the First Car after the Broken One Still Responding

This scenario differs from the previous one by the fact that the damaged vehicle can still respond electronically but is unable to keep the whole initiative of activation and supervision of the following vehicles for a collective manoeuvre to become ahead of it. This initiative is instead delegated to the first vehicle behind the damaged one, which is becoming the leading vehicle for the whole operation, capable, however, of communication with the damaged vehicle to make the avoidance procedure as smooth as possible. This scenario is depicted in Fig. 10 with SGL code just following.

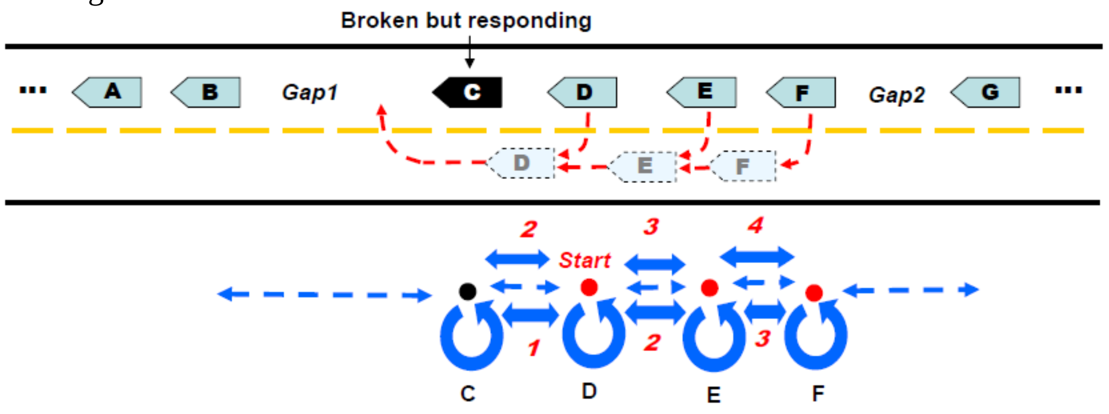

Fig. 10. Collective avoiding of the broken car by the initiative of the next vehicle

```
frontal(Gap = …, Staus);
sling(
  sleep(Delay); distance(ahead) < Gap;
  hop(first_ahead); STATE == down;
  distance(ahead) > Gap; hop(BACK);
  Staus = head;
  repeat(
    free(lane_manouvre(Staus, BEFORE));
    distance(behind) < Gap; hop(first_behind);
    Staus = follower))
```

Natural language comments for this scenario are as follows.

• Define types and initial values of spatial variables used.

• Repeat the following and with certain time intervals.

• If measured by sensors nearest distance to vehicles ahead is less than the given gap threshold, allow the following.

• Using V2V, hop to the nearest vehicle ahead, check its state, and if it is responding with «down» and measured distance to vehicles ahead of it is greater than the gap threshold given, allow the following.

• Hop back to the starting vehicle using V2V and name its position as “head”.

• Do the following repeatedly until possible.

• Activate (in parallel with the rest of the scenario) the lane manoeuvre procedure to avoid the damaged vehicle (with the “head” vehicle communicating with and orienting on the damaged one while others interacting with and following the previous vehicles wherever they go and by which lane.

• Use sensors to measure distance to vehicles behind, and if less than the given gap threshold, enter the first vehicle behind via V2V and set its state as "follower".

3.6. Collective Avoidance Activity Starting from the First Car after the Completely Broken One

This case concludes that the damaged vehicle is completely "dead", actually becoming an obstacle to be fully ignored and avoided by all subsequent vehicles (i.e. not responding to the following vehicle trying to communicate with it electronically). The next to the damaged vehicle, instead of active dialogue with it, tries to determine its physical coordinates using sensors, which should be avoided when leading the chain of vehicles via another lane to appear ahead of the damaged one. This situation is shown in Fig. 11 and by SGL code below.

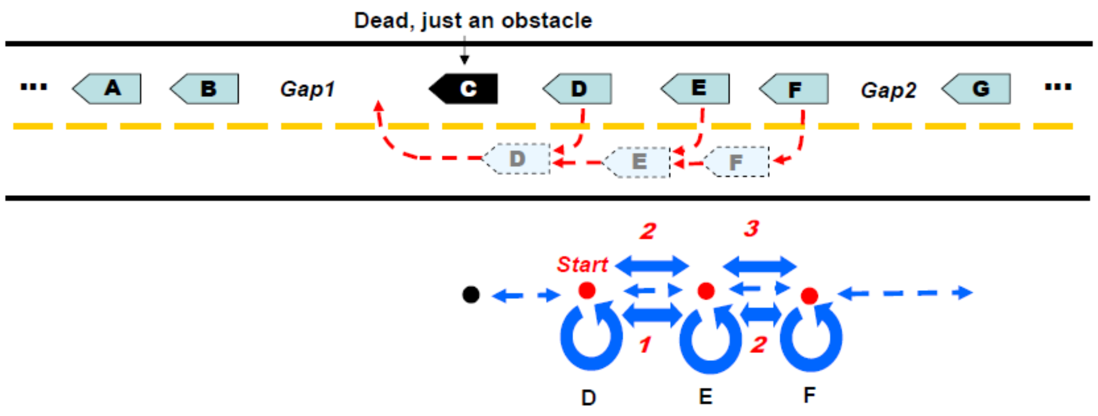

Fig. 11. Collective avoidance of a completely damaged vehicle

```
frontal(Gap = …); nodal(Place);
sling(
  sleep(Delay); distance(ahead) < Gap;
  or((hop(first_ahead); done),
     Place = measure(first_ahead));
  free(go_around(Place));
  repeat(
    distance(behind) < Gap; hop(first_behind);
    free(lane_manouvre(BEFORE))))
```

Natural language comments for this scenario variation will be as follows.

• Define types and initial values of spatial variables used.

• Repeat the following and with certain time intervals.

• If measured by sensors nearest distance to vehicles ahead is less than the given gap threshold, allow the following.

• If using V2V to enter the nearest vehicle or obstacle ahead fails, use sensors to determine physical coordinates of the vehicle or obstacle ahead.

• Organize parallel with the rest of the scenario lane manoeuvre for the current vehicle to go around the obstacle.

• Do the following repeatedly until possible.

• Use sensors to measure distance to vehicles behind, and if less than the given gap threshold, enter the first vehicle behind via V2V.

• Activate in it (in parallel with the rest of the scenario) the manoeuvre procedure to follow the previous vehicle wherever it goes and by which lane.

3.7. Collective Platoon Management Starting from Its Head

Platooning, a closely spaced multiple-vehicle chain on a highway, 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. The solution depicted in Fig. 12 and followed by the related SGL scenario, starting in the head vehicle, regularly and accesses all vehicles in their chain while updating their speed to keep the established standard distance between vehicles and orient the whole platoon on the speed of the head vehicle.

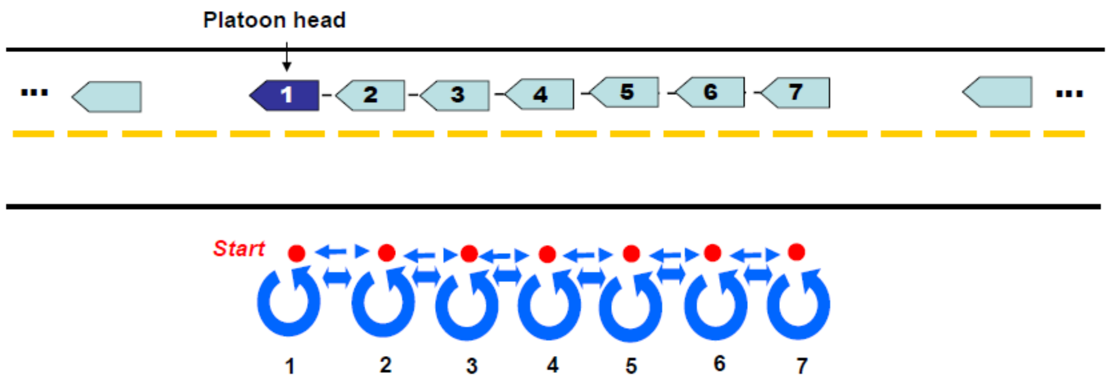

Fig. 12. Collective platoon management

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

Explanation for this scenario will be as follows.

• Define types and initial values of spatial variables used.

• Start in the platoon’s head vehicle and copy its current speed to be used for all other vehicles.

• Repeat the following regularly and with certain time intervals.

• Repeat the following until possible.

• Increment vehicle’s order to point to the next vehicle, not exceeding the total number of platoon vehicles.

• Enter the next vehicle by V2V with its ID to correspond to the current order.

• Update its speed taking into account the speed of the first vehicle and the needed and initially established distance between all vehicles in the platoon.

3.8. Collective 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 from fragmentation. Such a situation is depicted in Fig. 13 where between platoon vehicles 4 and 5 an emergency vehicle happened to appear on the same lane, which divided the platoon in two parts. 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 following.

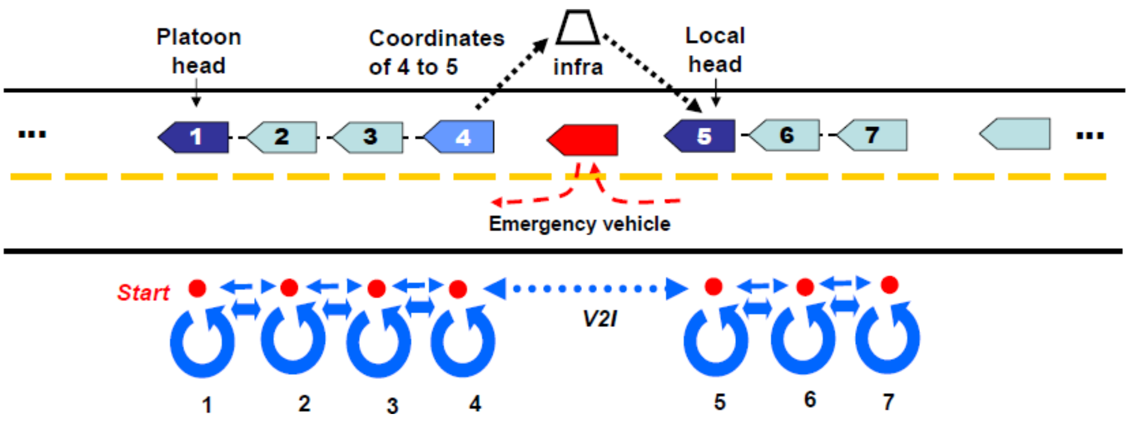

Fig. 13. Collective management of a fragmented platoon

```
frontal(Distance = …, Number = …, Order = 1, Position);
sling(
  sleep(Delay); Speed = SPEED;
  repeat(
    Order += 1 <= Number;
    or(
     (hop(first_behind, Order);
      free(update(SPEED, (Speed, Distance, BEFORE)))),
     (Position = WHERE;
      hop(or(behind, infra), Order);
      free(update(Position))))))
```

Natural language detailed explanation of this extended scenario is as follows.

• Define types and initial values of spatial variables used.

• Start in the platoon's head vehicle and copy its current speed to be used in all other vehicles.

• Repeat the following regularly and with certain time intervals.

• Repeat the following until possible.

Increment vehicle's order to point to the next vehicle, not to exceed the total number of platoon vehicles.

• May be two options:

• Option 1. Enter next vehicle by V2V with its ID to correspond to the current order, and update its speed taking into account the speed of the first platoon vehicle and the needed and initially established distance between all vehicles in the platoon.

• Option 2. In case next vehicle is not accessible by V2V (and this may be when the platoon is broken into pieces being at some distance from each other), the physical coordinates of the current vehicle are copied and the next vehicle is tried to be contacted via more powerful V2I communications, and if successful, it is set to move to the coordinates of the previous vehicle.

3.9. Returning to Normal Platoon Management after Reducing the Gap between Parts

After vehicle 5 comes again into the vicinity of 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 Fig. 14. This, however, will not guarantee the platoon of not being fragmented by certain forces once more, so the previous scenario will be operating again for the platoon integration.

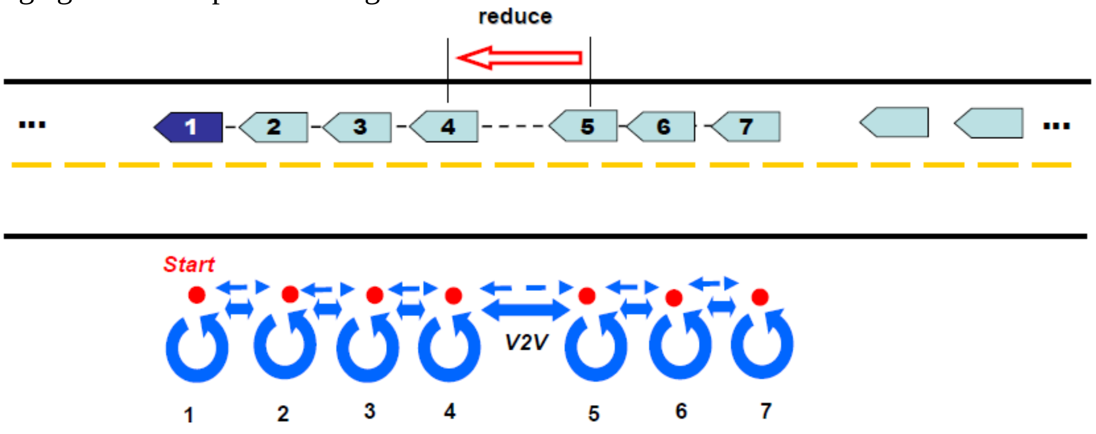

Fig. 14. Recovery of the platoon's structure

4. Distributed Routing in Road Networks

A solution will be shown in SGL for finding shortest path between starting and final locations in distributed road networks of arbitrary complexity. The parallel spatial solution is based on directly navigating road infrastructure holding the latest data on roads availability and their throughput and actual length, rather than using static predetermined maps. The parallel and fully distributed shortest path solution is based on the one described in [11]. A road network example with some start and end points between which an optimum route should be found is shown in Fig. 15.

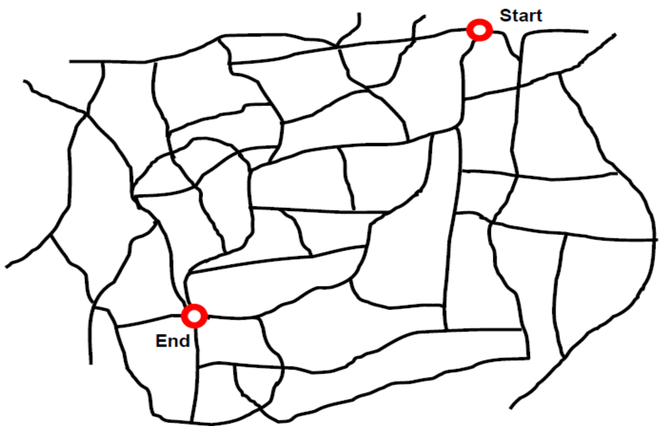

Fig. 15. Road network example

Example of a shortest path for this network within the allowed or restricted search area is shown in Fig. 16.

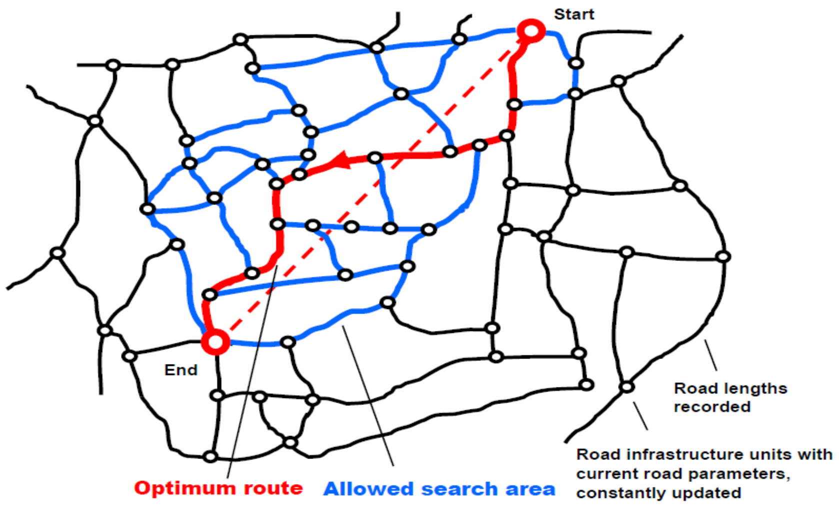
Start
End
Road lengths recorded
Road infrastructure units with current road parameters, constantly updated
Optimum route
Allowed search area

Fig. 16. Finding optimum path in the network within a restricted search area

Possible parameters for defining the allowed or recommended search area are shown in Fig. 17, namely: Start and End nodes physical coordinates; theoretical, or direct, distance between Start and End nodes; already accumulated summary length of the considered path via the existing roads in the current node coordinates; theoretical, direct, distance to the End node from the current node coordinates; and also physical deviation from the current node coordinates from the direct line connecting Start and End nodes. Using these parameters within the shortest path distributed algorithm of [11] will allow us, applying some heuristics, to narrow the absolute shortest path procedure and speedup the solution within a reasonable quality and speed.

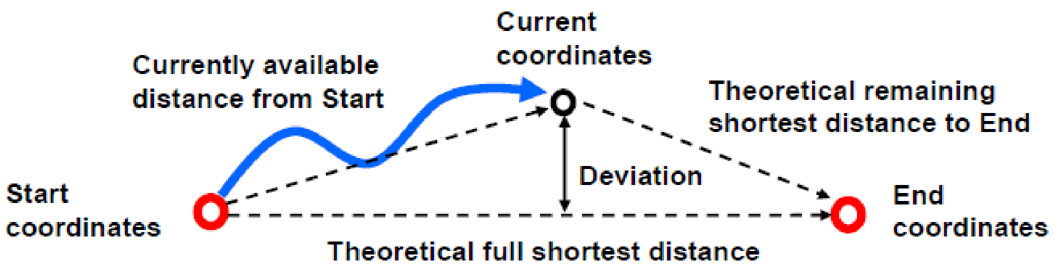
Current coordinates
Currently available distance from Start
Theoretical remaining shortest distance to End
Deviation
Start coordinates
End coordinates
Theoretical full shortest distance

Fig. 17. Setting possible constraints on the shortest path solution

The following is SGL shortest path fully distributed implementation taking into account the above mentioned restrictions narrowing and speeding the distributed solution.

```
frontal(Far, Path, Start = …; End = …);
nodal(Distance, Before);
hop(Start); Distance = 0;
sequence(
  repeat(
    hop(road_infra, all); Far += LENGTH;
    acceptable(Far, WHERE, Start, End);
    or(Distance == nil, Far < Distance);
    Distance = Far; Before = PREVIOUS),
  Path = repeat(
    Path &= NAME;
    if(WHERE == End, stop(Path));
    hop(road_infra, all); PREVIOUS == Before),
  repeat(move(withdraw(Path, 1))));
```

Natural language comments to this scenario will be as follows.

• Define types and initial values of spatial variables used.

• Start in the current road infrastructure (RI) node using V2R connection.

• Proceed in a sequence the following three stages, each from the current RI node.

• Stage 1. Repeat the following until possible by creating shortest path tree (SPT).

• Hop to all RI neighbouring nodes associated with neighbouring junctions.

• Bring into them the growing physical distance from the starting node incremented by the passed road length.

• If the evolving solution is within the permitted boundaries, and the brought distance is smaller than the one recorded in the node before, change the latter by the former and redefine the predecessor node.

• Stage 2. Navigate the obtained distributed SPT in parallel top-down fashion, accumulating paths with visited nodes until reach the destination node, the current path will be the needed solution, which is brought back to the starting node.

• Stage 3. Organize physical car driving guided by the sequence of road junctions recorded in the path found.

5. Conclusions

We have shown how the developed Spatial Grasp Technology, SGT, with its core element Spatial Grasp Language, SGL, can be effectively used for advanced road management with driverless cars – from emergent collective multiple car solutions to global management of large distributed road infrastructures. Only some very elementary and exemplary solutions for advanced road management using the technology invented have been exhibited. Any other cases can be effectively described and implemented too, say, for roads with multiple lanes or optimized management at road crossings, with more complex scenarios being currently successfully programmed, which will be revealed in the following publications, new planned book including. The technology had a number of trial implementations in different countries with its latest version being patented again. It can be implemented on an agreement on any platform needed and within limited period of time.

REFERENCE

1. Autonomous Vehicle Technology [Електронний ресурс] / M. James Anderson, K. Nidhi, S. Karlyn [et al.] // A Guide for Policymakers. – RAND Corporation, RR-443-2-RC. – 2016. – 214 p. – Режим доступу: http://www.rand.org/content/dam/rand/pubs/research_reports/RR400/RR443-2/RAND_RR443-2.pdf.
2. Litman T. Autonomous Vehicle Implementation Predictions Implications for Transport Planning, Victoria Transport Policy Institute [Електронний ресурс] / T. Litman // 1st May, 2017. – Режим доступу: http://www.vtpi.org/avip.pdf.
3. Autonomous Vehicles Revolutionizing Our World [Електронний ресурс]. – Report Borden Ladner Gervais, BLG. – 2016. – Режим доступу: http://blg.com/en/News-And-Publications/Documents/Autonomous-Vehicles_1033.pdf.
4. The Pathway to Driverless Cars: A Code of Practice for Testing [Електронний ресурс]. – UK Department of Transport. – 2015. – Режим доступу: www.gov.uk/dft, file:///C:/Users/USER/Downloads/pathwaydriverlesscars.pdf.
5. Sapaty P. Mobile Processing in Distributed and Open Environments / Sapaty P. – New York: John Wiley & Sons, 1999. – 410 p.
6. Sapaty P. Ruling Distributed Dynamic Worlds / Sapaty P. – New York: John Wiley & Sons, 2005. – 255 p.
7. Sapaty P. Distributed Air & Missile Defense with Spatial Grasp Technology / P. Sapaty // ICA. – 2012. – Vol. 3, N 2. – P. 117 – 131.
8. Sapaty P. Providing Over-operability of Advanced ISR Systems by a High-Level Networking Technology / P. Sapaty // Airborne ISR, London, 2015. – 56 p.
9. Sapaty P. Military Robotics: Latest Trends and Spatial Grasp Solutions / P. Sapaty // IJARAI. – 2015. – Vol. 4, N 4. – P. 9 – 18.
10. Sapaty P. Distributed Missile Defence with Spatial Grasp Technology / Sapaty P. – London: Military Space, 2015. – 65 p.
11. Sapaty P. Managing Distributed Dynamic Systems with Spatial Grasp Technology / Sapaty P. – Springer, 2017. – 284 p.
12. European Patent N 0389655 A distributed processing system / Sapaty P.; Publ. 10.11.93, European Patent Office. – 32 p.

Стаття надійшла до редакції 11.05.2017