METHOD, SYSTEM AND APPARATUS FOR CONTROLLING SELF-DRIVING VEHICLES

(51) Int. Cl. G05D 1/02 (2006.01) G05D 1/00 (2006.01)

(52) U.S. Cl. CPC ........ G05D 1/0212 (2013.01); G05D 1/0088 (2013.01)

(57) ABSTRACT

Systems, methods and apparatus are provided for controlling self-driving vehicles. The system comprises: a processor, a memory storing operational constraints for a self-driving vehicle, and a communications interface. A plurality of path portions are assembled at the system to define an area in a physical space in which the self-driving vehicle is to navigate, each of the plurality of path portions associated with a respective given subset of operational constraints stored in the memory. The system provides, to the self-driving vehicle, respective given subsets of the operational constraints of the plurality of path portions that define the area, and associated positions of each of the plurality of path portions in the physical space.

RELATED ART

100

108 Computing Device

110

113

114-1

114-2

114-3

112 Network

104-1

104-2

104-3

Storage

Work Station

Conveyor

120

116

124

128
405 Receive and store master definitions

410 Receive request to edit operational constraints

415 Edit or create?

Create

420 Present available masters for selection

425 Receive selection, present properties

430 Receive and store properties

435 Validate properties

440 Receive selection of zone, display properties

435 Present existing zones for selection

Edit

Fig. 4
Create New Zone

Select Zone Type

Speed Limit

One-way

Create One-Way Zone

Location

Direction

Fig. 5
Vehicle 104
(Begin Operation)

705
Operational constraints required?

600
No

710
Cache valid?

500
Yes

715
Retrieve from cache

720
Send request with location and type

730
Retrieve and send operational constraints

800

725
Send request with location and type

735
Cache operational constraints

740
Update operation

Computing Device 108

725
Receive request
Assemble a plurality of path portions to define an area in a physical space in which the self-driving vehicle is to navigate, each of the plurality of path portions associated with a respective given subset of operational constraints stored in the memory.

801

Provide, to the self-driving vehicle, respective given subsets of the operational constraints of the plurality of path portions that define the area, and associated positions of each of the plurality of path portions in the physical space.

803
METHOD, SYSTEM AND APPARATUS FOR CONTROLLING SELF-DRIVING VEHICLES

FIELD


BACKGROUND

[0002] Self-driving vehicles (e.g., autonomous mobile robots) operate in a wide variety of environments. Such environments can have various physical characteristics that the vehicles may be required to navigate around, interact with and the like, in order to operate successfully. Physical characteristics such as those mentioned above can generally be represented in maps of the environments stored by the vehicles. The operating environments of the vehicles, however, may also impose other restrictions on the operation of the vehicles that do not arise directly from physical characteristics of the environments, or that arise from physical characteristics that are not readily detectable by the vehicles. Such restrictions are less suitable for representation in maps, and can therefore render the operation of vehicles difficult.

SUMMARY

[0003] In general, this disclosure is directed to a method, system and apparatus for handling operational constraints for the control of self-driving vehicles. In particular, a computing device in communication with one or more self-driving vehicles stores a plurality of operational constraints each associated with a region of the environment in which the one or more self-driving vehicles are to operate and a property defining a constraint on the operation of the one or more self-driving vehicles within the region. The computing device provides operational constraints to a self-driving vehicle to control how the self-driving vehicle will operate in the region of the environment associated with the transmitted operational constraint. For example, operational constraints can be provided upon request by the self-driving vehicle and/or by a computing device configured to assign tasks to be carried out by self-driving vehicles within a physical space.

[0004] Furthermore, in some implementations, a plurality of path portions are assembled to define an area in a physical space (e.g., an environment) in which the self-driving vehicles are to navigate, each of the plurality of path portions associated with a respective given subset of operational constraints stored in a memory. Respective given subsets of the operational constraints of the plurality of path portions that define the area are transmitted to the self-driving vehicles with associated positions (and/or identifiers thereof) of each of the plurality of path portions in the physical space. Such a scheme can obviate defining operational constraints of the self-driving vehicles outside the regions of the associated positions of each of the plurality of path portions in the physical space.

[0005] The present specification provides a system comprising: a plurality of self-driving vehicles for deployment in an environment; a computing device connected to the plurality of self-driving vehicles via a network; the computing device storing, in a memory, a plurality of operational constraints; each operational constraint including (i) a type identifier, (ii) an indication of a region of the environment, and (iii) a property defining a constraint on the operation of the self-driving vehicles within the region; the computing device configured to: receive a request from one of the self-driving vehicles, the request identifying an operational constraint; responsive to receiving the request, retrieve an operational constraint from the memory based on the request; and send the retrieved operational constraint to the one of the self-driving vehicles.

[0006] The region can comprise at least one of an area and a volume within the environment. A set of regions for a plurality of the operational constraints having the same type can be stored as polygons in an image file; and the properties corresponding to the set of regions can be stored in association with the polygons.

[0007] The request can identify a location within the environment; the computing device can be configured to retrieve any operational constraints indicating regions that intersect the location.

[0008] The request can identify a type of operational constraint; the computing device can be configured to retrieve any operational constraints including types that match the requested type.

[0009] The one of the self-driving vehicles can be further configured, prior to sending the request, to execute a navigational process and to determine whether operational requirement data is required for the navigational process. The navigational process can comprise at least one of path generation, path execution, and mapping and localization. The one of the self-driving vehicles can be further configured, when the determination is affirmative, to examine a cache in a vehicle memory to assess whether the required operational constraints are present in the cache. The self-driving vehicle can be further configured, when the required operational constraints are not present in the cache, to send the request. The one of the self-driving vehicles can be further configured to receive the operational constraint from the computing device, to store the operational constraint in the vehicle memory, and to resume execution of the navigational process. The navigational process can be a path execution process; the self-driving vehicle can be configured to set a speed of travel for the path execution process based on the operational constraint.

[0010] The computing device can be further configured, prior to storing the plurality of operational constraints, to store operational constraint templates, each template including (i) a type and (ii) a property definition. The computing device can be further configured to receive a request to create an operational constraint. The computing device can be further configured, responsive to receiving the request to create an operational constraint, to present a list of the templates and to receive a selection of one of the templates. The computing device can be further configured to retrieve and present the property definition for the selected template, to receive a value corresponding to the property definition, and to create and store a new operational constraint containing (i) the type of the template and (ii) a property containing the value. The computing device can be further configured to determine any of the operational constraints having overlapping regions contain conflicting properties. The computing device can be further configured to determine whether any of the operational constraints having adjacent regions contain potentially conflicting properties.
Another aspect of the specification provides a system comprising: a processor, a memory storing operational constraints for a self-driving vehicle, and a communications interface, the processor configured to: assemble a plurality of path portions to define an area in a physical space in which the self-driving vehicle is to navigate, each of the plurality of path portions associated with a respective given subset of operational constraints stored in the memory; and, provide, to the self-driving vehicle, respective given subsets of the operational constraints of the plurality of path portions that define the area, and associated positions of each of the plurality of path portions in the physical space.

The system can further comprise an input device, wherein the plurality of path portions can be assembled based on data received from the input device, the data indicating one or more of selections of each of the plurality of path portions and an order of the plurality of path portions.

The area can define a path between at least two positions in the physical space.

The processor can be further configured to assemble the plurality of path portions by: rendering, at a display device, representations of the plurality of path portions at a representation of the physical space.

Each of the operational constraints stored in the memory can define a rule that the self-driving vehicle is to follow when located in an associated position.

The processor can be further configured to: define one or more values associated with the respective given subset of operational constraints; and provide, to the self-driving vehicle, the one or more values with the respective given subsets of the operational constraints of the plurality of path portions that define the area.

The processor can be is further configured to provide, to the self-driving vehicle, an indication of the area with the respective given subsets of the operational constraints of the plurality of path portions that define the area.

Each of the plurality of path portions can comprise a feature associated with an operational constraint in the respective given subset of operational constraints associated therewith. The feature can comprises a general indication of a rule that the self-driving vehicle is to follow, and the operational constraint associated with the feature can comprise a more precise rule that the self-driving vehicle is to follow, the self-driving vehicle navigating according to the more precise rule and not the general indication of the rule when located in an associated position.

The system can further comprise the self-driving vehicle, and one or more of the processor and the self-driving vehicle can be configured to: plan a path between two positions in the physical space based on the respective given subsets of the operational constraints of the plurality of path portions that define the area.

The system can further comprise the self-driving vehicle, and one or more of the processor and the self-driving vehicle can be configured to: plan a path between two positions in the physical space based on an indication of the area in the physical space defined by the plurality of path portions.

The system can further comprise the self-driving vehicle, and one or more of the processor and the self-driving vehicle can be configured to: restrict a position of the self-driving vehicle only to positions in the physical space defined by the plurality of path portions.

The processor can be a component of the self-driving vehicle.

Yet a further aspect of the specification provides a method comprising: at a system comprising: a processor, a memory storing operational constraints for a self-driving vehicle, and a communications interface, assembling, at the processor, a plurality of path portions to define an area in a physical space in which the self-driving vehicle is to navigate, each of the plurality of path portions associated with a respective given subset of operational constraints stored in the memory; and, providing, using the processor, to the self-driving vehicle, respective given subsets of the operational constraints of the plurality of path portions that define the area, and associated positions of each of the plurality of path portions in the physical space.

The system can further comprise an input device, wherein the plurality of path portions can be assembled based on data received from the input device, the data indicating one or more of selections of each of the plurality of path portions and an order of the plurality of path portions.

The area can define a path between at least two positions in the physical space.

The method can further comprise: assembling, using the processor, the plurality of path portions by: rendering, at a display device, representations of the plurality of path portions at a representation of the physical space.

Each of the operational constraints stored in the memory can define a rule that the self-driving vehicle is to follow when located in an associated position.

The method can further comprise: defining, at the processor, one or more values associated with the respective given subset of operational constraints; and provide, to the self-driving vehicle, the one or more values with the respective given subsets of the operational constraints of the plurality of path portions that define the area.

The method can further comprise: providing, using the processor, to the self-driving vehicle, an indication of the area with the respective given subsets of the operational constraints of the plurality of path portions that define the area.

Each of the plurality of path portions can comprise a feature associated with an operational constraint in the respective given subset of operational constraints associated therewith. The feature can comprises a general indication of a rule that the self-driving vehicle is to follow, and the operational constraint associated with the feature can comprise a more precise rule that the self-driving vehicle is to follow, the self-driving vehicle navigating according to the more precise rule and not the general indication of the rule when located in an associated position.

The system can further comprise the self-driving vehicle, and the method can further comprise: planning, at one or more of the processor and the self-driving vehicle, a path between two positions in the physical space based on the respective given subsets of the operational constraints of the plurality of path portions that define the area.

The system can further comprise the self-driving vehicle, and the method can further comprise: planning, at one or more of the processor and the self-driving vehicle, a path between two positions in the physical space based on an indication of the area in the physical space defined by the plurality of path portions.
[0033] The system can further comprise the self-driving vehicle, and the method can further comprise: restricting, at one or more of the processor and the self-driving vehicle, a position of the self-driving vehicle only to positions in the physical space defined by the plurality of path portions.

[0034] The processor can be a component of the self-driving vehicle.

[0035] Yet a further aspect of the specification provides a non-transitory computer-readable medium storing a computer program, wherein execution of the computer program is for: at a system comprising: a processor, a memory storing operational constraints for a self-driving vehicle, and a communications interface, assembling, at the processor, a plurality of path portions to define an area in a physical space in which the self-driving vehicle is to navigate, each of the plurality of path portions associated with a respective subset of operational constraints stored in the memory; and, providing, using the processor, to the self-driving vehicle, respective given subsets of the operational constraints of the plurality of path portions that define the area, and associated positions of each of the plurality of path portions in the physical space.

**BRIEF DESCRIPTIONS OF THE DRAWINGS**

[0036] Implementations are described with reference to the following figures, in which:

[0037] FIG. 1 depicts a system for controlling self-driving vehicles, according to a non-limiting implementation.

[0038] FIG. 2 depicts certain components of a self-driving vehicle of the system of FIG. 1, according to a non-limiting implementation.

[0039] FIG. 3 depicts certain internal components of the computing device of FIG. 1, according to a non-limiting implementation.

[0040] FIG. 4 depicts a method of receiving and storing operational constraints in the system of FIG. 1, according to a non-limiting implementation.

[0041] FIG. 5 depicts example interfaces presented by the computing device of FIG. 1 during the method of FIG. 4, according to a non-limiting implementation.

[0042] FIG. 6 depicts an example data structure for storing the operational constraints received in the method of FIG. 4, according to a non-limiting implementation.

[0043] FIG. 7 depicts a method of deploying the operational constraints received in the method of FIG. 4, according to a non-limiting implementation.

[0044] FIG. 8 depicts a method of controlling a self-driving vehicle according to operational constraints defined using a plurality of path portions, according to a non-limiting implementation.

[0045] FIG. 9 depicts a rendering of a physical space at a display device, along with renderings of path portions, according to a non-limiting implementation.

[0046] FIG. 10 depicts details of operational constraints associated with the path portions of FIG. 9, according to non-limiting implementations.

[0047] FIG. 11 depicts an initiation of the method of FIG. 8 at the rendering of FIG. 9, according to a non-limiting implementation.

[0048] FIG. 12 depicts a path between two points at the rendering of FIG. 9, according to a non-limiting implementation.

[0049] FIG. 13 depicts paths between three points at the rendering of FIG. 9, according to a non-limiting implementation.

[0050] FIG. 14 depicts paths between seven points at the rendering of FIG. 9, according to a non-limiting implementation.

[0051] FIG. 15 depicts the certain internal components of the computing device of FIG. 1, paths within a rendering of a physical space are defined, according to a non-limiting implementation.

[0052] FIG. 16 depicts the system of FIG. 1 in which a computing device communicates operational constraints associated with a path to a self-driving vehicle, according to a non-limiting implementation.

**DETAILED DESCRIPTION OF THE IMPLEMENTATIONS**

[0053] FIG. 1 depicts a system 100 including a plurality of self-driving vehicles 104-1, 104-2 and 104-3 (collectively referred to as self-driving vehicles 104, and generically referred to as a self-driving vehicle 104, or simply a vehicle 104; similar nomenclature is used for other reference numerals herein) for deployment in a facility, such as a manufacturing factory, warehouse or the like. The facility can be any one of, or any suitable combination of, a single building, a combination of buildings, an outdoor area, and the like. A greater or smaller number of self-driving vehicles 104 may be included in system 100 than the three shown in FIG. 1. Self-driving vehicles 104 can have a wide variety of operational characteristics (e.g., maximum payload, dimensions, weight, maximum speed, battery life, and the like).

[0054] System 100 also includes a computing device 108 for connection to self-driving vehicles 104 via a network 112. Computing device 108 can be connected to network 112 via, for example, a wired link 113, although wired link 113 can be any suitable combination of wired and wireless links in other implementations. Self-driving vehicles 104 can be connected to network 112 via respective wireless links 114-1, 114-2 and 114-3. Links 114 can be any suitable combination of wired and wireless links in other examples, although generally wireless links are preferable to reduce or eliminate obstacles to the free movement of self-driving vehicles 104 about the facility. Network 112 can be any suitable one of, or any suitable combination of, wired and wireless networks, including local area networks (LAN or WLAN), wide area networks (WAN) such as the Internet, and mobile networks (e.g., GSM, LTE and the like).

[0055] Computing device 108 can control self-driving vehicles 104, for example by instructing self-driving vehicles 104 to carry out tasks within the facility. The nature of the tasks performed by self-driving vehicles 104 under the control of computing device 108 is not particularly limited. In general, the tasks assigned to self-driving vehicles 104 require self-driving vehicles 104 to perform various actions at various locations within the facility. Data defining the actions and locations are provided to self-driving vehicles 104 by computing device 108 via network 112.

[0056] The actions, items and locations mentioned above are not particularly limited. For example, a self-driving vehicle 104 can be instructed to simply travel to a specific location. In other examples, a self-driving vehicle 104 can be instructed to travel to a specified location and pick up, drop off, or otherwise manipulate, an item (e.g., a tool, container, and the like), or perform any other suitable action.
Locations include any regions within the facility bounded by coordinates. Such regions can be three-dimensional (i.e., volumes), two-dimensional (i.e., areas), one-dimensional (i.e., lines) or zero-dimensional (i.e., points).

In the present example, a first location 120 is illustrated, which may be employed to store items, such as an item 116 (e.g., a container). Location 120 can be an area defined on a floor of the facility for storage of items. A second location 124 is also illustrated, containing, for example, a work station where materials are to be removed from or placed in item 116, or where item 116 is to be labelled or otherwise modified. A wide variety of other work station activities will occur to those skilled in the art (e.g., welding stations, paint spray booths, and so on). A third location 128 is also illustrated in FIG. 1. In the present example, third location 128 contains a conveyor apparatus, which may carry item 116 to another part of the facility.

When a vehicle 104 is assigned a task by computing device 108, that vehicle 104 is configured to generate a path for completing the task (e.g., a path leading from the vehicle’s current location to the end location of the task; the path may include one or more intermediate locations between the start location and the end location). In some implementations, computing device 108 can assist the vehicle 104 in path generation (also referred to as path planning), or can generate the path without the involvement of the vehicle 104 and send the completed path to the vehicle 104 for execution.

Generation of the above-mentioned paths can be based on, for example, a map of the facility stored at one or both of computing device 108 and vehicles 104. Path generation may also depend on attributes of the relevant vehicle 104. For example, the map may indicate that a certain area of the facility contains constrained areas unsuitable for vehicles 104 greater than certain dimensions; if a vehicle 104 has dimensions greater than those of the constrained areas, a path may therefore be generated for that vehicle 104 that avoids the constrained areas.

Various other information can also impact not only the generation of paths for vehicles 104, but also the execution of those paths by vehicles 104 and the performance of actions by vehicles 104 in connection with the paths. Such other information may not be amenable to storage in the above-mentioned map (e.g., because the information may not relate directly to physical features of the facility). As will be discussed in greater detail below, system 100 is configured to receive, store and deploy to vehicles 104 a wide variety of such other information, referred to broadly herein as operational constraints for vehicles 104.

Before describing the handling of operational constraints by system 100 in greater detail, an example vehicle 104 and certain internal components of computing device 108 will be described.

Referring now to FIG. 2, an example self-driving vehicle 104 is shown. In particular, self-driving vehicle 104-3 is depicted according to a non-limiting implementation. Other vehicles 104 need not be identical to vehicle 104-3 as depicted, but are generally as described below. Self-driving vehicle 104-3 is depicted as a terrestrial vehicle, although it is contemplated that self-driving vehicles 104 can also include aerial vehicles and watercraft. It is also contemplated that self-driving vehicle 104 can comprise an unmanned vehicle or a manned vehicle which carries passengers, but is otherwise at least temporarily self-driving.

Self-driving vehicle 104-3 includes a chassis 200 containing or otherwise supporting various other components, including one or more locomotive devices 204. Devices 204 in the present example are wheels, although in other implementations any suitable locomotive device, or combination thereof, may be employed (e.g., tracks, propellers, and the like).

Locomotive devices 204 are powered by one or more motors (not shown) contained within chassis 200. The motors of self-driving vehicle 104-3 can be electric motors, internal combustion engines, or any other suitable motor or combination of motors. In general, the motors drive the locomotive devices 204 by drawing power from an energy storage device (not shown) supported on or within chassis 200. The nature of the energy storage device can vary based on the nature of the motors. For example, the energy storage can include batteries, combustible fuel tanks, or any suitable combination thereof.

Self-driving vehicle 104-3 also includes a load-bearing surface 208 (also referred to as a payload surface), for carrying an item such as item 116 thereon. In some examples, payload surface 208 can be replaced or supplemented with other payload-bearing equipment, such as a cradle, a manipulator arm, or the like.

Self-driving vehicle 104-3 can also include a variety of sensors. In the present example, such sensors include at least one load cell 212 coupled to payload surface 208, for measuring a force exerted on payload surface 208 (e.g., by an item being carried by self-driving vehicle 104-3). The sensors of self-driving vehicle 104-3 can also include machine vision sensors 216, such as any suitable one of, or any suitable combination of, barcode scanners, laser-based sensing devices (e.g., a LIDAR sensor), cameras and the like. Self-driving vehicle 104-3 can also include a location sensor (not shown) such as a GPS sensor, for detecting the location of self-driving vehicle 104-3 with respect to a frame of reference. The frame of reference is not particularly limited, and may be, for example, a global frame of reference (e.g., GPS coordinates), or a facility-specific frame of reference. Other sensors that can be provided with self-driving vehicle 104-3 include accelerometers, fuel-level or battery-level sensors, and the like.

Self-driving vehicle 104-3 can also include a control panel 220, as well as anchors 222 for securing items or other equipment to chassis 200, or for lading chassis 200 (e.g., for maintenance). Self-driving vehicle 104-3 can also include any of a variety of other features, such as indicator lights 228.

In addition, self-driving vehicle 104-3 includes a processor 250 (which can include, but is not limited to, one or more central processing units (CPU), and the like), interconnected with a non-transitory computer-readable medium such as a memory 254. Processor 250 and memory 254 are generally comprised of one or more integrated circuits (ICs), and can have a variety of structures, as well now occur to those skilled in the art (for example, more than one CPU can be provided). Memory 254 can be any suitable combination of volatile (e.g., Random Access Memory (“RAM”)) and non-volatile (e.g., read only memory (“ROM”), Electrically Erasable Programmable Read Only Memory (“EEPROM”), flash memory, magnetic computer storage device, or optical disc) memory.
Self-driving vehicle 104-3 also includes a communications interface 258 (e.g. a network interface controller or NIC) interconnected with processor 250. Via communications interface 258, link 114-3 and network 112, processor 250 can send and receive data to and from computing device 108. For example, self-driving vehicle 104-3 can send updated location data to computing device 108, and receive operational constraints from computing device 108.

Additionally, processor 250 is interconnected with the other components of self-driving vehicle 104-3 mentioned above, such as sensors 212 and 216 and control panel 220.

In some optional implementations, self-driving vehicle 104-3 can further comprise at least one input device 268 and/or a display device 269, each interconnected with processor 250. Each of input device 268 and display device 269 is drawn in broken lines to indicate that they are optional. When present, input device 268 can comprise a keyboard, a mouse, a pointing device, a touchscreen integrated with display device 269 (when present), and the like. Display device 269, when present, can comprise a cathode ray tube (CRT), a flat panel display, a touchscreen display and the like. In general, input device 268 and display device 269 can function as a human-machine-interface (HMI) to self-driving vehicle 104-3 such that self-driving vehicle 104-3 can be programmed and/or interfaced with using input device 268 and a display device 269. Alternatively, input device 268 and/or display device 269 may not be present at self-driving vehicle 104-3, however self-driving vehicle 104-3 and/or processor 250 can be connectable to an external input device and/or an external display, for example in a provisioning mode.

Memory 254 stores a plurality of computer-readable programming instructions, executable by processor 300, in the form of various applications, including a vehicle control application 262. As will be understood by those skilled in the art, processor 250 can execute the instructions of application 262 (and any other suitable applications stored in memory 254) in order to perform various actions defined within the instructions. In the description below processor 250, and more generally any vehicle 104, is said to be “configured to” perform certain actions. It will be understood that vehicles 104 are so configured via the execution of the instructions of the applications stored in memory 254. Memory 254 also stores a cache 264, to be discussed in greater detail below.

Turning now to FIG. 3, certain internal components of computing device 108 are illustrated. Computing device 108 can be any one of, or any combination of, a variety of computing devices. Such devices include desktop computers, servers, mobile computers such as laptops and tablet computers, and the like. Computing device 108 therefore includes at least one central processing unit (CPU), also referred to herein as a processor, 300. Processor 300 is interconnected with a non-transitory computer-readable medium such as a memory 304. Processor 300 is also interconnected with a communications interface 308.

Processor 300 and memory 304 are generally comprised of one or more integrated circuits (ICs), and can have a variety of structures, as will now occur to those skilled in the art (for example, more than one CPU can be provided). Memory 304 can be any suitable combination of volatile (e.g. Random Access Memory ("RAM")). Programmable Read Only Memory ("EPROM"), flash memory, magnetic computer storage device, or optical disc) memory.

Communications interface 308 allows computing device 108 to connect with other computing devices (e.g. self-driving vehicles 104) via network 112. Communications interface 308 therefore includes any necessary hardware (e.g. network interface controllers (NICs), radio units, and the like) to communicate with network 112 over link 113. Computing device 108 can also include input and output devices, such as keyboards, mice, displays, and the like.

For example, as depicted, in some implementations, computing device 108 can further comprise at least one input device 368 and/or a display device 369 each interconnected with processor 300. Each of input device 368 and display device 369 is drawn in broken lines to indicate that they are optional. When present, input device 368 can comprise a keyboard, a mouse, a pointing device, a touchscreen integrated with display device 369 (when present), and the like. Display device 369, when present, can comprise a cathode ray tube (CRT), a flat panel display, a touchscreen display and the like. In general, input device 368 and display device 369 can function as a human-machine-interface (HMI) to computing device 108 such that computing device 108 can be programmed and/or interfaced with using input device 368 and a display device 369. Alternatively, input device 368 and/or display device 369 may not be present at computing device 108, however computing device 108 and/or processor 300 can be connectable to an external input device and/or an external display, for example in a provisioning mode.

Memory 304 stores a plurality of computer-readable programming instructions, executable by processor 300, in the form of various applications, including an operational constraints handling application 312. As will be understood by those skilled in the art, processor 300 can execute the instructions of application 312 (and any other suitable applications) in order to perform various actions defined within the instructions. In the description below processor 300, and more generally computing device 108, are said to be “configured to” perform those actions. It will be understood that they are so configured via the execution of the instructions of the applications stored in memory 304.

Memory 304, in the present example, also stores various types of data for retrieval, processing and updating during the execution of application 312. In particular, memory 304 stores an operational constraints database 316. Memory 304 may also store other data (not shown), such as a map of the facility of FIG. 1, as well as vehicle attributes, location-related data, and the like.

Turning now to FIG. 4, a method 400 for generating and storing operational constraints is illustrated. The performance of method 400 will be described in connection with its performance in system 100, although it is contemplated that method 400 can also be performed in other suitable systems. The blocks of method 400 as described below are performed by computing device 108, via the execution of application 312 by processor 300. In other implementations, however, method 400 can also be performed by any of self-driving vehicles 104 (that is, by processor 250 of a given vehicle 104).

Beginning at block 405, computing device 108 is configured to receive and store operational constraint type definitions, also referred to as operational constraint tem-
plates or masters. Each operational constraint master defines a type of operational constraint, as well as the properties that can be assigned to that type of operational constraint. The nature of the receipt of operational constraint masters at block 405 is not particularly limited. For example, the masters may be received at processor 300 via at least one input device 368 such as a keyboard and a mouse, and the like. Upon receipt the operational constraint masters are stored in database 316.

[0081] Table 1 illustrates examples of operational constraint masters.

<table>
<thead>
<tr>
<th>Operational Constraint Masters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Speed Limit</td>
</tr>
<tr>
<td>Upper Limit</td>
</tr>
<tr>
<td>Lower limit</td>
</tr>
<tr>
<td>Time limit</td>
</tr>
<tr>
<td>One-Way</td>
</tr>
</tbody>
</table>

[0082] In particular, two types of operational constraints are illustrated above. A speed limit operational constraint type provides a definition for creating speed limit operational constraints that apply to the facility of FIG. 1. As required by the master above, each speed limit operational constraint (also referred to herein as zone) defines a space on the map of the facility where that constraint applies. The space may be a volume, an area, a line, or a point, and can be defined in a variety of ways (e.g. by coordinates). Thus, the location properties for zones correspond to “real” physical spaces in the facility. Each speed limit zone also defines an upper speed limit for self-driving vehicles 104 within the zone, a lower speed limit, and a time period during which the operational constraint applies. It will be appreciated that a wide variety of properties may be defined for each operational constraint type. Further, some properties may be indicated as mandatory, while others may be optional (e.g. a lower speed limit and the time property may be optional).

[0083] Another example of an operational constraint type is a one-way operational constraint. Based on the master above, each one-way zone includes a location, as well as a direction, with or without an associated tolerance. For example, a one-way zone may state that self-driving vehicles in a certain area of the facility must travel in a direction ten degrees east of north, plus or minus five degrees. As will now be apparent to those skilled in the art, a variety of other ways may also be employed to represent directions and tolerances.

[0084] Having stored operational constraint templates, at block 410 computing device 108 is configured to receive a request to edit operational constraints. For example, computing device 108 can received such a request in the form of input data from a keyboard or mouse, or via network 112 via another computing device. In some examples, the request can be received from a vehicle 104.

[0085] At block 415, computing device 108 is configured to determine whether the request received at block 410 is a request to edit an existing zone, or to create a new zone. For example, the request received at block 410 can be a selection of one of several user-selectable elements of a graphical user interface (GUI) presented on a display connected to processor 300. The determination at block 415 can therefore include a determination of which selectable element was selected (e.g. an element corresponding to zone creation, or an element corresponding to zone editing).

[0086] When the request received at block 410 is a request to create a new zone, performance of method 400 advances to block 420, at which computing device 108 retrieves the zone types defined by the masters received and stored at block 405. Thus, if Table 1 represents the currently stored zone masters, at block 420 computing device 108 retrieves the zone types “speed limit” and “one-way” and presents those zone types for selection. Presenting zone types for selection can involve controlling a display to render an interface 500, shown in FIG. 5, including selectable elements 504 and 508 corresponding to each retrieved zone type.

[0087] In other implementations, rather than rendering the available zone types for selection at block 420, computing device 108 can present the available zone types for selection via other means, such as by transmitting the zone types via network 112 to another computing device.

[0088] Returning to FIG. 4, following the performance of block 420, computing device 108 is configured to receive a selection of one of the zone types presented at block 420, and in response to retrieve and present the properties defined by the master corresponding to the selected zone type. Thus, referring again to FIG. 5, an updated interface 512 can be rendered on a display connected to processor 300, in which selectable elements are provided for defining the location and direction properties defined in Table 1. For example, the location property can be defined by drawing (e.g. via input data received from a mouse, touch screen or the like) an area, volume or the like 516 on a rendering of the map of the facility. In other implementations, the location can be specified by received input data in the form of coordinates on the map.

[0089] The direction property of the one-way zone can be specified, for example, by setting an angle 520 relative to an indicator of cardinal north 524 (or any other known direction. Angle 520 can be specified directly on the map shown in interface 512 in some implementations, rather than as a separate interface element from the map (as shown in FIG. 5). In other implementations, the direction can be specified by way of input data identifying an angle and a tolerance. Returning to FIG. 4, at block 430 computing device 108 is configured to store the newly received zone in database 316.

[0090] If, on the other hand, the request received at block 410 is determined (at block 415) to be a request to edit an existing zone, then at block 435 computing device 108 can be configured to retrieve the existing zones in database 316 and present the zones for selection. In some implementations, the existing zones can be retrieved and presented based on a zone type; for example, between blocks 415 and 435 computing device 108 can present an interface similar to interface 500 for receiving a selection of which zone type is to be edited.

[0091] Upon receipt of a selection of a specific zone to edit at block 440, computing device 108 retrieves the properties of that zone and presents the retrieved properties (e.g. on a display). The display of zone properties at block 440 can be implemented via the presentation of an interface such as Interface 512 shown in FIG. 5. Performance of method 400 then proceeds to block 430, at which properties for the
relevant zone are received and stored as described above. Editing properties presented at block 440 and 430 can include deleting properties. When the editing inputs received via an interface such as interface 512 include the removal of all properties for a zone (or selection of a “delete zone” element), the receipt and storage of properties at block 430 involves removing the zone from memory.

[0992] As a result of repeated performances of method 400 (or, at least, repeated performances of blocks 420-430), a plurality of operational constraints, or zones, can be maintained in database 316, each having a type, a location, and one or more properties.

[0993] In some implementations, computing device 108 is configured to perform a validation or simulation at block 435, after receiving updates to operational constraints (e.g. new operational constraints or edited operational constraints). In particular, computing device 108 can be configured to detect conflicts in the operational constraints, such as one-way zones with incompatible direction properties (e.g. opposite directions). Computing device 108 can also be configured to detect potential conflicts between zones that are not overlapping but adjacent to each other. For example, when two speed limit zones are in close proximity, and one zone has a greater minimum speed than the maximum speed of the other zone, computing device 108 can compare the difference between the required speeds of the zones to the known accelerations and decelerations of self-driving vehicles 104. Computing device 108 can thus determine whether the proximity of the zones, in combination with their differing requirements, would result in self-driving vehicles 104 being unable to comply with the operational constraints when traversing both zones (e.g. because the vehicles 104 cannot accelerate or decelerate quickly enough). When conflicts or potential conflicts are detected, computing device 108 can generate a warning message, for example on the display mentioned in connection with FIG. 5.

[0994] The zones and their properties can be stored in a wide variety of ways. For example, turning to FIG. 6, three layers 600, 604 and 608 of zones are depicted as stored in database 316. The layers can be depicted, for example, in a plurality of image files (e.g. vector or raster-based images). In other implementations, the layers can be stored in a variety of other formats, such as tables of coordinates or the like. In the present example, each layer can store the locations of zones of a particular type as polygons (or volumes, points, lines or the like). Thus, layer 600 can contain one-way zone locations (e.g. locations for two zones, 612 and 616, are visible), layer 604 can contain speed limit locations, and layer 608 can contain locations for another type of zone (e.g. emergency aisles in the facility). Each layer (e.g. image file) can contain the remaining properties 620 corresponding to the zones, or can contain references to those properties stored separately in database 316. In other implementations, database 316 can contain an index linking layers 600, 604, 608 and properties 620. Thus, in some implementations, location and zone type can be considered primary properties of the zones, as those are the properties defining which layer the zone is depicted in, while the remaining properties can be considered secondary properties.

[0995] A wide variety of zone types and properties are contemplated, in addition to those discussed above. Other examples of zone types and corresponding properties (beyond those already discussed) include: stop before proceeding (e.g. with properties specifying a length of time to stop); restricted areas, such as the above-mentioned emergency aisles (e.g. with properties specifying that such zones are active only when an alarm is sounding in the facility); parking zones indicating areas where vehicles 104 may dock for periods of inactivity; height-restricted zones (e.g. with maximum permitted vehicle heights); weight-restricted zones (e.g. with maximum permitted vehicle weights); map quality zones (e.g. with properties indicating a level of confidence in the map in each zone). Other zone types and properties will also occur to those skilled in the art. As a further example, restricted area zones may include properties identifying classes or vehicles 104 or individual vehicles 104 that are prohibited or permitted from entering such areas.

[0996] Still other examples of zone types include undetectable (by self-driving vehicles 104) physical features of the facility that are therefore less well-suited for representation in the map. For example, ramps or other such features of the facility may be represented as operational constraints. Further examples of properties include transitional properties associated with the boundaries of zones. For example, a speed limit zone may include a secondary speed limit property that is applied when a self-driving vehicle is within a predetermined distance of the edge of the zone.

[0997] Computing device 108 can also be configured to allocate tasks to self-driving vehicles 104 based on operational constraints. For example, computing device 108 can receive a task for assignment, and retrieve operational constraints associated with a location identified in the task. In combination with self-driving vehicle characteristics, computing device 108 can then select a self-driving vehicle 104 to perform the task (e.g. picking up an item in a specific location). For example, computing device may exclude certain self-driving vehicles 104 from the above-mentioned selection when the task to be assigned lies within a height-restricted zone with a maximum height that is smaller than the known height of those self-driving vehicles 104.

[0998] Turning now to FIG. 7, a method 700 for deploying operational constraints is illustrated. Method 700 will be described in connection with its performance in system 100, although it is contemplated that method 700 can also be performed in other suitable systems.

[0999] Beginning at block 705, a vehicle 104 is configured to determine whether operational constraint data is required. The determination at block 705 can take various forms, and generally involves determining whether a navigational process executed by the processor 250 of the vehicle 104 requires operational constraint data. For example, the vehicle 104 can begin executing a path planning (i.e. path generation) process that incorporates operational constraint data, following receipt of a task assignment from computing device 108. In another example, the vehicle 104 can perform a path execution process to travel a previously generated path. The path execution process can incorporate operational constraint data. In some examples, the path generation and path execution processes can incorporate different operational constraints. For example, path generation may require the use of one-way zones, whereas path execution data may require the use of speed limit zones (which can be ignored during path generation in some implementations).

[1000] In other examples, the determination at block 705 can be a determination of whether a current location of the
vehicle 104 is acceptable (i.e. complies with operational constraints). In still other examples, the vehicle 104 can initiate a mapping or localization process and determine that operational constraint data is required to complete the process.

[0101] When the determination at block 705 is negative, the self-driving vehicle 104 continues operating as before. When the determination at block 705 is affirmative, however, the vehicle 104 determines, at block 710, whether the required operational constraint data is present in cache 264. It is therefore contemplated that when the determination at block 705 is affirmative, the vehicle 104 is configured to identify required operational constraint data, such as a zone type, a location, or the like. For example, when block 705 is performed in connection with a path execution process, the required location may be a portion of the path (or the entire path), and the zone types may include any zone types relevant to path execution (e.g. speed limits).

[0102] The performance of block 710 thus includes examining the contents of cache 264 for operational constraints corresponding to any requirements identified at block 705. In some implementations, the determination at block 710 can include simply determining whether operational constraints corresponding to those identified at block 705 are present in cache 264. In other implementations, when the required operational constraints are present in cache 264, the vehicle 104 can also determine whether the required constraints are valid, for example by determining the age of those constraints in cache 264. The vehicle 104 may also send a request (not shown) to computing device 108 to retrieve a timestamp indicating the last time database 316 was modified. If the timestamp is more recent than the age of cache 264, the determination at block 710 can be negative (even if the required constraints are present in cache 264).

[0103] When the determination at block 710 is affirmative, the vehicle 104 proceeds to block 715 and retrieves, from cache 264, the operational constraint data identified as being required at block 705. When the determination at block 710 is negative, however, the vehicle 104 proceeds to block 720.

[0104] At block 720, the vehicle 104 is configured to transmit a request for operational constraint data to computing device 108. The request can contain one or more of a location (e.g. the current location of vehicle 104, another specified location in the facility, including a set of locations such as a path, and the like), and a zone type. As will now be apparent to those skilled in the art, the request can include identifiers (e.g. of a zone type and location) corresponding to the required data identified at block 705. The location can also be omitted in order to request all available data for the facility. Likewise, zone types can also be omitted from the request to request data for all available types. Thus, vehicle 104 can send a request without a location or zone type in order to request all available zone data. In further implementations, computing device 108 can store, in database 316, identifiers of zone types in association with identifiers of vehicles 104. Thus, vehicles 104 can omit zone types from requests and receive zone data of a specific type (or set of types) based on the above associations.

[0105] At block 725, computing device 108 is configured to receive the request via network 112. At block 730, responsive to receiving the request at block 725, computing device 108 is configured to retrieve the data identified in the request and send the requested data to the vehicle 104. Computing device 108 can be configured to retrieve the data in a variety of ways. For example, computing device 108 can be configured to select any zone having the type specified in the request and intersecting the location specified in the request. It will now be apparent that if no type was specified in the request, zones of all types may be retrieved at block 730.

[0106] At block 735, the vehicle 104 is configured to receive the operational constraints sent by computing device 108 and store the operational constraints in cache 264. Proceeding then to block 740, the vehicle 104 is configured to update its operation based on the operational constraint data received from computing device 108 or retrieved from cache 264 at block 715. Updating the operation of the vehicle 104 can include updating the vehicle’s trajectory, performing a preconfigured action, sending a signal to another device, or any of a wide variety of other operational behaviours. In general, the vehicle 104 is configured to complete the process that gave rise to the requirement identified at block 705.

[0107] The vehicle 104 is therefore configured, at block 740, to resume the process that lead to the affirmative determination at block 705. When the process was a path execution process, the received (or retrieved) operational constraint data, such as a speed limit, can be incorporated into the path execution process to set a speed of the vehicle 104 during execution of the path. When the process was a path generation process, the operational constraint data can be incorporated into the process, for example by rerouting the path to avoid travelling against the direction mandated by one-way zones.

[0108] In further implementations, the vehicle 104 can be configured to initiate a predetermined behaviour based on the operational constraint data received from computing device 108 or retrieved from cache 264. For example, each vehicle 104 can maintain, in memory 254, sets of instructions defining specific routines, such as a sequence of movements for parking or docking the vehicle 104. In some implementations, a type of operational constraint can define parking zones within the facility, and thus at block 740 a vehicle 104 can be configured, having requested parking zone data, to initiate a parking routine upon determining that its current location is within a parking zone.

[0109] Variations to the above systems and methods are contemplated. For example, in some implementations, computing device 108 itself can perform the determination at block 705. For example, in some systems computing device 108, rather than vehicles 104, is responsible for generating paths for vehicles 104. Thus, the receipt of a request to generate a path can result in an affirmative determination at block 705 (at computing device 108 rather than a vehicle 104). Responsive to such a determination, computing device 108 can be configured to perform blocks 720, 725, 730 (which would be performed internally within computing device 108) and 740 by retrieving operational constraints required for path generation, generating a path and sending the path to the relevant vehicle 104. Blocks 710, 715 and 735 would be omitted from such implementations.

[0110] In a further variation, vehicles 104 may request, at block 720, a partial operational constraint or a binary decision, rather than complete operational constraint data. For example, a vehicle 104 can request a speed limit for the vehicle’s current location. In response, computing device 108 can transmit only a speed limit to vehicle 104 (or indeed, any other requested property), rather than the zone to which
the speed limit applies). In another example, the vehicle 104 can request confirmation that a current location of the vehicle is acceptable (i.e. complies with operational constraints). Rather than providing the operational constraints to the vehicle 104, computing device 108 can perform the determination locally and send an indication of whether or not the vehicle’s current location is acceptable or not (that is, without sending any operational constraints).

In a further variation, vehicles 104 can update operational constraints. As noted earlier, vehicles 104 are equipped with sensors and can thus gather data about their environments. Vehicles can thus be configured to compare data gathered via their sensors to operational constraint data in cache 264. When the comparison reveals that the operational constraint data does not match the sensed data, a vehicle 104 can send a request to computing device 108 to update an operational constraint (e.g. a request received by computing device 108 at block 410 of method 400). As an example of such a vehicle-driven update, a vehicle 104 may have a sensor capable of measuring the height of surrounding objects in the facility. The vehicle may therefore measure the height clearance in a region of the facility and determine that the height clearance specified in a corresponding operational constraint does not match the measurement.

In some implementations, providing operational constraints for all areas of an environment can be onerous. For example, in order to indicate that a vehicle is not to go leave a particular path, in some implementations restricted areas of movement can be associated with operational constraints indicating that a self-driving vehicle is not to enter the restricted areas; in instances where not all restricted areas are specified, the self-driving vehicle could enter such restricted areas. Similarly, when a self-driving vehicle is to travel between two positions in an environment, which can also be referred to as a physical space, and not all operational constraints associated with a path between the two positions, the self-driving vehicle may not be able to navigate there between.

Hence, attention is next directed to FIG. 8 which depicts a block diagram of a flowchart of a method 800 of e, according to non-limiting implementations. In order to assist in the explanation of method 800, it will be assumed that method 800 is performed using system 100, and specifically by computing device 108, for example when processor 300 processes instructions stored at memory 304. Indeed, method 800 is one way in which system 100 can be configured. However, in other implementations, at least a portion of method 800 can be implemented by a self-driving vehicle, for example when processor 250 processes instructions stored at memory 304.

Furthermore, the following discussion of method 800 will lead to a further understanding of system 100, and its various components. However, it is to be understood that system 100 and/or method 800 can be varied, and need not work exactly as discussed herein in conjunction with each other, and that such variations are within the scope of present implementations.

Regardless, it is to be emphasized, that method 800 need not be performed in the exact sequence as shown, unless otherwise indicated; and likewise various blocks may be performed in parallel rather than in sequence; hence the elements of method 800 are referred to herein as “blocks” rather than “steps”. It is also to be understood, however, that method 800 can be implemented on variations of system 100 as well.

It is further assumed in FIG. 8 that memory 304 (and/or memory 254) stores operational constraints for a self-driving vehicle, for example in database 316, however a format of such operational constraints can differ from that of the format of zones depicted in FIG. 6, as will be described in detail below.

At block 801, a plurality of path portions is assembled to define an area in a physical space in which a self-driving vehicle is to navigate, each of the plurality of path portions associated with a respective given subset of operational constraints stored in a memory.

At block 803, provided, to the self-driving vehicle, are respective given subsets of the operational constraints of the plurality of path portions that define the area and are associated positions of each of the plurality of path portions in the physical space.

Method 800 will now be described in more detail with respect to FIG. 8 to 16.

Attention is next directed to FIG. 9, which depicts a rendering of a physical space 901 at display device 369. In particular, the rendering of physical space 901 represents a map of physical space 901 (e.g. an environment) in which a self-driving vehicle is to navigate, including, but not limited to, vehicles 104. However, the following discussion will refer to vehicles 104 by way of example only. In particular, physical space 901 comprises various features including zones and/or regions 903 where vehicles 104 are not to navigate, as well as docking stations 905-1, 905-2, 905-3, 905-4, 905-5, 905-6, interchangeably referred to, collectively, as docking stations 905 and/or docks 905, and generically as a docking station 905 and/or a dock 905. Hence, for example, physical space 901 can comprise a manufacturing facility, warehouse or the like, and the like, regions 903 can comprise areas where goods, and the like, are manufactured and/or stored, and docks 905 can comprise docking stations where vehicles are to load and/or unload goods. In other words, it is assumed in FIG. 9 that vehicles 104 are configured to transport goods between docks 905; hence, vehicles 104 used in physical space 901 can be similar to vehicle 104-3 depicted in FIG. 2. While not depicted in FIG. 9, it is assumed that physical space 901 can comprise equipment for moving goods between regions 903 and docks 905, and equipment for moving goods between docks 905 and vehicles 104.

Not shown in FIG. 9 are paths between docks 905. According to implementations described with reference to FIG. 1 to 8, each zone of physical space would be specified using operational constraints as being an area of physical space 901 where vehicles 104 are restricted from entering (such as regions 903, docks 905) using operational constraints, or area of physical space 901 where vehicles 104 can navigate, also using operational constraints. However, an inherent risk in such a scheme is that if some zones are not specified as being areas where a vehicle 104 can navigate and/or are left blank with regards to associated operational constraints, a vehicle 104 may not be able to access such zones. For example, if a zone and associated operational constraints around dock 905-2 are not specified, then a vehicle 104 will not be able to access dock 905-2.

Hence, also depicted in FIG. 9 are representations of path portions 911, 912, as well as indicators 913-1, 913-2,
913-3 (interchangeably referred to, collectively, as indicators 913, and generically as an indicator 913) which can be used to modify path portions 911, 912. Path portions 911, 912 and indicators 913 can be rendered at display device 369 adjacent the rendering of physical space 901.

[0123] As described in more detail below, one or more of path portions 911, 912 can be selected and assembled in the rendering of physical space 901 using, for example, input device 368 and/or modified using indicators 913, also using input device 368. For example, a path portion 911 (and/or path portion 912) can be selected at display device 369 using input device 368, and placed within the rendering of physical space 901, as well as duplicated such that a plurality of path portions 911 (and/or a plurality of path portions 912) are assembled to define an area in a physical space in which the self-driving vehicle is to navigate.

[0124] Each path portion 911, 912 is associated with a respective given subset of operational constraints stored in memory 304, for example in database 316, and can be similar or different to a format of the operational constraints of zones depicted in FIG. 6. In particular, attention is directed to FIG. 10, which depicts layers associated with each path portion 911, 912, as well as indicators 913. For example, path portion 911 is associated with a layer 1011, also stored in memory 304, layer 1011 defining operational constraints associated with path portion 911. In particular, path portion 911 can represent a visual layer to be used when assembling path portions to define an area in physical space 901 in which a self-driving vehicle 104 is to navigate, and layer 1011 represents operational constraints associated with the visual layer.

[0125] In particular, each of the operational constraints stored in memory 304 define a rule that a self-driving vehicle 104 is to follow when located in an associated position (i.e. a position in physical space 901 associated with a given path portion, as described in more detail below). For example, with further reference to FIG. 10, path portion 912 comprises features associated with an operational constraint in the respective given subset of operational constraints associated therewith. As depicted, the features comprise two solid parallel lines 1020, each of which represent a boundary within which a vehicle 104 in a region of physical space 901 associated with path portion 912 are to stay, and which can be distinguished from other path portions 911, 912, which can be provided merely as a visual guide to path portion 912. In other implementations, two solid parallel lines 1020 and outer boundaries of path portion 912 can be coincident.

[0126] Alternatively, solid parallel lines 1020 also represent a boundary that vehicles 104 navigating external to the region of physical space 901 associated with path portion 912 are to avoid. In other words, vehicles 104 located within the two solid parallel lines are to stay between them, and vehicles 104 located outside two solid parallel lines 1020 are to stay outside the two solid parallel lines.

[0127] It is appreciated, however, that two solid parallel lines 1020 (e.g. boundary) are not necessarily physically represented in physical space 901 but can be rendered at the rendering of physical space 901 as described in further detail below.

[0128] Similarly, path portion 911 comprises a stippled line 1021 about midway between, and parallel to, two solid parallel lines 1020, stippled line 1021 defining lanes in path portion 911. In other words, a first area between the stippled line 1021 and a first one of the solid lines 1020 represents a first lane, and a second area between the stippled line 1021 and a second one of the solid lines 1020 represents a second lane. In some implementations, a convention can be used which dictates a direction of each lane (e.g. a right hand convention, as in North America, or a left hand convention, as in the United Kingdom and Japan), however in other implementations a direction of each lane can be specified in the associated operational constraints, for example as shown in Table 1. In general, however, a direction of travel in each lane can be in opposite directions, though in other implementations, a direction of travel can be the same.

[0129] Furthermore, the middle line 1021 being stippled can be indicative of a rule that a vehicle 104 in either of the lanes can cross an area of physical space 901 corresponding to the stippled line 1021, for example to pass other vehicles 104 in the lane (and presuming there is an avoidance system for not interfering with vehicles 104 travelling in the other direction).

[0130] In other words, each feature of path portion 911 can be associated with a general indication of a rule and/or operational constraint that a self-driving vehicle 104 is to follow, including, but not limited to, operational constraints provided in Table 1. However, with reference to layer 1011, an operational constraint associated with the feature can comprises a more precise rule that a self-driving vehicle 104 is to follow, the self-driving vehicle 104 navigating according to the more precise rule and not the general indication of the rule when located in an associated position of physical space 901 (i.e. associated with path portion 911).

[0131] For example, layer 1011 represents operational constraints associated with the visual layer represented by path portion 911; in particular, the two solid parallel lines 1020 of path portion 911 are associated with a narrower boundary indicated by respective two solid parallel lines represented in layer 1011 (which are mapped to the two solid parallel lines 1020 of path portion 911 as indicated by the arrows). In other words, the respective two solid parallel lines represented in layer 1011 define narrower lanes that is indicated by the two solid parallel lines 1020 in path portion 911. For example, the respective two solid parallel lines represented in layer 1011 can represent a more precise rule that, not only are vehicles 104 to stay within the boundary represented in path portion 911, the vehicles are to stay within the boundary represented in layer 1011, for example for safety reasons. Vehicles 104 external to the boundary represented in path portion 911 are still to stay outside that boundary.

[0132] Path portion 912 and layer 1012 are respectively similar to path portion 911 and layer 1011, however the lanes defined by the features are different widths, with one lane being larger and another lane being narrower. Lanes of different widths can represent another operational constraint that limits types of vehicles 104 in each respective lane. For example, vehicles 104 of a first given size can navigate in the larger of the two lanes, and vehicles 104 of a second given size, smaller than the first given size, can navigate in the smaller of the two lanes (and/or the larger of the two lanes). In other words, a width of a lane can represent an operational constraint on a minimum and/or maximum size of a vehicle 104 that can navigate in a lane.

[0133] As in path portion 911, lanes in path portion 912 can be directional.
It is appreciated that while one layer 1011, 1012 is respectively associated with each of path portions 911, 912, more than layer with other operational constraints can be associated with each of path portions 911, 912. Indeed, each of path portions 911, 912 can themselves comprise a layer with associated operational constraints; hence, put another way, as depicted, each path portion 911, 912 comprises two layers. However, each path portion 911, 912 can comprise more than two layers, similar to the format of zones depicted in FIG. 6. In addition any number and type of suitable operational constraints can be represented by features in each path portion 911, 912, including, but not limited to, operational constraints provided in Table 1.

Indeed, it is appreciated that the format of path portions 911, 912 depicted in FIG. 10 can be similar to the format of zones depicted in FIG. 6, but that each path portion 911, 912 stored in memory 304 is not initially associated with a specific position in physical space 901, and further includes a visual layer indicating features associated with operational constraints, such that path portions 911, 912 and/or copies thereof can be assembled into paths as described in more detail below.

Also depicted in FIG. 10 are layers 1013-1, 1013-2, 1013-3, respectively associated with indicators 913-1, 913-2, 913-3. For example, indicator 913-1 indicates that a vehicle 913-1 should stop at a position of physical space 901 corresponding to a position of indicator 913-1 in the rendering of physical space 901; hence, indicator 913-1 represents an operational constraint of where vehicles 104 should stop. Associated layer 1013-1 provides a more precise rule/operational constraint that a vehicle 104 should begin slowing about 10 m prior to the location associated with indicator 913-1. While not depicted, layer 1013-1 could include another rule/operational constraint indicating a rate of decrease of a speed of a vehicle 104 in a region prior to the location.

Similarly, indicator 913-2 indicates that a vehicle 104 located in a position of physical space 901 corresponding to a position of indicator 913-2, in the rendering of physical space 901 should move at a given speed of "X" m/s, where X can be specified by editing indicator 913-2. Associated layer 1013-2 provides a more precise rule/operational constraint that a vehicle 104 should move at X m/s and put limits on variations in such a speed (e.g. +/-10%). Thus such limits can also be specified by editing layer 1013-2.

Indicator 913-3 indicates that a vehicle 104 located in a position of physical space 901 corresponding to a position of indicator 913-3 should follow rules/operational constraints associated with stoplights, while layer 1013-3 indicates a more precise rule/operational constraint on vehicle behavior if the stoplight represented by indicator 913-3 is yellow; again, such behavior can be specified by editing layer 1013-3. It is further more assumed that indicator 913-3 is dynamic, and that a state of indicator 913-3 can be controlled by computing device 108 and/or any other device which controls behavior and paths of vehicles 104 navigating physical space 901.

Furthermore, each of indicators 913 can be used to modify one or more of path portions 911, 912 and/or copies thereof. For example, in an editing mode, one or more of indicators 913 can be moved and/or dragged to a path portion 911, 912, and/or a copy thereof, to indicate further behavior of a vehicle 104 in that path portion 911, 912.

Attention is hence next directed to FIG. 11 which is substantially similar to FIG. 9, with like elements having like numbers. However, in FIG. 11, path portion 911 has been selected, for example using input device 368, to initiate method 800, and a copy of path portion 911 has been placed adjacent dock 901-1 in rendering of physical space 901 at display device 369. Furthermore, the copy of path portion 911 has been rendered with a handle 1101, on a side opposite dock 905-1, indicating that further copies of path portion 911 can be rendered when handle 1101 is selected and “dragged”. For example, processor 300 can be generally further configured to provide such handles on path portions copied to rendering of physical space 901, for example, in an editing mode, and on sides of copies of path portions that can be extended, and/or dragged, to affect copying and/or assembling of the copies of path portions. In particular, as the copy of path portion 911 should not be extended into dock 905-1, as vehicles 104 can navigate to dock 905-1, but not onto dock 905-1, handle 1101 is provide at the copy of path portion 911 on a side opposite dock 905-1, and/or on a side of the copy of path portion 911 adjacent an otherwise empty portion of the rendering of physical space 901. Furthermore, placement of such handles 1101 can be restricted to ends of copies of path portions that receive lanes and/or from which lanes can be extended.

In any event, handle 1101 can be selected and dragged to create further copies of path portion 911. For example, attention is next directed to FIG. 12 which is substantially similar to FIG. 11, with like elements having like numbers, however, in FIG. 12, handle 1101 has been dragged to dock 905-2, thereby assembling path portions (and in particular copies of path portions 911) to define an area in physical space 901 in which a self-driving vehicle 104 is to navigate (i.e. at block 801 of method 800). In particular, the assembled copies of path portion 911 form a path between docks 905-1, 905-2.

Alternatively, input device 368 could be used to again select rendering of path portion 911 and place another copy thereof adjacent the copy depicted in FIG. 11; this process could be repeated until an area that defines a path between docks 905-1, 905-2 is provided.

While in FIG. 12, the copy of path portions 911 adjacent dock 905-1 is a given size, relative to dock 905-1, it is appreciated that a smaller and/or shorter copy of path portion 911 can be used, to provide further granularity in the operational constraints along a path of a vehicle 104. For example, handle 1101 can be dragged towards dock 905-1 to shorten the copy of the path portion 911, though other processes can be used to shorten and/or otherwise change a size and/or a length of the copy of path portion 911.

Also depicted in FIG. 12 is a copy of indicator 913-1 along the path between docks 905-1, 905-2, indicating that, when a vehicle 104 approaches a physical location in physical space 901 corresponding to a position of indicator 913-1, the vehicle should stop according to the operational constraints associated with indicator 913-1. In some implementations, copy of indicator 913-1 can be provided by way of input device 368 being used to select indicator 913-1 and drag a copy onto the path between docks 905-1, 905-2.

In other implementations, input device 368 can be used to edit and/or create a new path portion that is a combination of path portion 911 and indicator 913-1, and which can be provided adjacent path portions 911, 912; in these implementations, handle 1101 can be used to drag the
copy of path portion 911 to a first position on rendering of physical space 901, and a copy of the new path portion (that includes indicator 913-1, but is otherwise similar to path portion 911) can be placed adjacent an end of the copies of path portion 911. Once the copy of the new path portion is in place, another copy of path portion 911 can be placed adjacent the new path portion and further copies of same can be assembled to complete the path to dock 905-2.

Similarly, also depicted in FIG. 12 are copies of indicators 913-2, which have been edited to indicate a speed limit of vehicles 104 in regions of copies of portion 911 associated with the copies of indicators 913-2. For example between dock 905-1 and indicator 913-1, a speed of vehicles should be 10 m/s, while closer to dock 905-2, a speed of vehicles should be 5 m/s. Such speeds are indicated in operational constraints associated with each of the copies of indicators 913-2.

Hence, according to present implementations, a plurality of path portions can be assembled based on data received from an input device, such as input device 368, the data indicating one or more of selections of each of the plurality of path portions and an order of the plurality of path portions. For example, such data from input device 368 indicates that a selection of a plurality of path portions 911 by way of selecting and dragging handle 1101 to assemble copies of path portions 911 between docks 905-1, 905-2.

In other words, a wide variety of processes can be used to assemble path portions at block 801 of method 800. In particular, processor 300 can be further configured to assemble a plurality of path portions by: rendering, at display device 369, representations of the plurality of path portions at rendering of physical space 901, from which the representations can be selected and copies thereof rendered in the rendering of physical space 901.

Furthermore, as depicted in FIG. 12, the area in physical space 901 in which the self-driving vehicle is to navigate, as defined by the assembled path portions, can define a path between two positions in physical space 901. While in FIG. 12, a path between docks 905-1, 905-2 is depicted, other path portions 911, 912 can be assembled into other paths between at least two positions in physical space 901.

For example, attention is next directed to FIG. 13 which is substantially similar to FIG. 12, with like elements having like numbers. However, in FIG. 13 additional copies of path portion 911 have been assembled into a path between docks 905-1, 905-3 (and/or between docks 905-2, 905-3), which includes all one curve and/or corner and/or bend in the path between docks 905-1, 905-3 (and/or between docks 905-2, 905-3); hence, lanes in the path between docks 905-1, 905-3 (and/or between docks 905-2, 905-3) also include curves/edges/bends and the like, as indicated at least by the depicted curve in the stippled line. Such curves/edges/bends can also be associated with operational constraints for example to a radius of curvature of such curves/edges/bends and/or to control a speed of vehicles 104 as they navigate such curves/edges/bends. In some implementations, operational constraints associated with curves/edges/bends can be particular to given vehicle 104 and/or a given vehicle type and/or a given vehicle size.

It is further assumed in FIG. 13 that the path between docks 905-1, 905-3 (and/or between docks 905-2, 905-3) can be assembled at block 801 of method 800 in a manner similar to the path between docks 905-1, 905-2 described above with respect to FIG. 12. Similarly, a copy of indicator 913-2 is also provided adjacent dock 905-3, edited to include an operational constraint of 5 m/s.

Attention is next directed to FIG. 14, which is substantially similar to FIG. 13, with like elements having like numbers. However, in FIG. 14 additional copies of path portion 911 have been assembled into a path between docks 905-5, 905-7, and copies of path portion 912 have been assembled into a path between docks 905-4, 905-6. Each of the paths between docks 905-5, 905-7, and docks 905-5, 905-7 intersect the path between docks 905-1, 905-2, while not depicted, vehicles 104 could use any path to navigate between any of docks 905 using the depicted paths, assuming that vehicles 104 operate according to associated operational constraints. While curves between all the depicted paths are not depicted, such curves can be assumed to be present. Indeed, processor 300 can be configured to provide and/or generate operational constraints for such intersections of paths and/or specialized path portions can be provided for selection and assembly that define such intersections, for example adjacent path portions 911, 912 depicted in FIG. 9. Such specialized intersection path portions can be associated with operational constraints that define vehicle behaviour at intersections.

Furthermore, when such intersections occur, a rendering of same at display device 369 can be particular to such intersections, such that outer boundaries of paths are depicted or removed for clarity.

Furthermore, as indicator 913-1 is present at an intersection between path between docks 905-1, 905-2, and a path between docks 905-4, 905-6, it is assumed that all vehicles 104 that approach the intersection will operate according to operational constraints associated with indicator 913-1.

Similarly, indicator 913-3 has been assembled at an intersection between path between docks 905-1, 905-2, and a path between docks 905-5, 905-7, and it is assumed that all vehicles 104 that approach the intersection will operate according to operational constraints associated with indicator 913-3 as described above.

In any event, paths between all docks 905 in the rendering of physical space 901 can be provided using block 801 of method 800 to assemble a plurality of path portions there between without having to further specify regions and/or zone that vehicles 104 are restricted from entering. Specifying such paths using block 801 of method 800 can obviate a need to specify, for example that vehicles 104 are not to navigate in regions and/or zone associated with regions 903 and/or in docks 905, and the like. Furthermore, specifying such paths using block 801 of method 800 can ensure that paths to all docks 905 are provided in physical space by providing visual indicators thereof.

Attention is next directed to FIG. 15, which is substantially similar to FIG. 3, with like elements having like numbers. However, in FIG. 15, database 316 has been updated to store respective given subsets 1501-1, 1501-2, 1501-3...1501-n of operational constraints of the plurality of path portions that define areas in physical space 901 in which a self-driving vehicle 104 is to navigate, and associated positions 1502-1, 1502-2, 1502-3...1502-n of each of the plurality of path portions in the physical space 901. For convenience, subsets 1501-1, 1501-2, 1501-3...1501-n will interchangeably referred to hereafter, collectively, as subsets 1501, and generically as a subset 1501; similarly,
positions 1502-1, 1502-2, 1502-3 . . . 1502-n will interchangeably referred to hereafter, collectively, as positions 1502, and generically as a position 1502.

[0158] In particular, subsets 1501 and associated positions 1502 define operational constraints along the paths defined between positions in physical space 901 as described above with respect to FIG. 11-14. In other words, for each position 1502 in physical space 901 associated with the paths, a subset 1501 of operational constraints is associated therewith. As described above, for example, a size and/or length of copies of path portions 911 can be changed when assembling path portions 911; a size and/or length of copies of path portions 911 can hence determine a granularity of subsets 1501 and associated positions 1502, with larger copies of path portions 911 leading to larger granularity, and smaller copies of path portions 911 leading to smaller granularity. In other words, positions 1502 can define a range of positions in physical space 901 over which an associated subset 1501 of operational constraints are to be implemented a vehicle 104 navigating therein. When a given indicator 913 is located at a given point in physical space, however, a position 1502 of such an indicator 913 can be indicated along with the associated subset 1501 of operational constraints (i.e. rather than a range of positions).

[0159] In any event, database 316 can be populated with any number of subsets 1501 and associated positions 1502, which can depend, for example, on the aforementioned granularity, a number and/or length of paths and the like. Furthermore, subsets 1501 and associated positions 1502 can be stored in any suitable format, including, but not limited to, database formats. Formats of each of subsets 1501 and associated positions 1502 can also be any suitable respective format for example positions 1502 can be specified using GPS (Global Positioning System) formats and/or with respect to a given origin position in physical space 901 (e.g. a given point in physical space 901, and hence in the representation thereof in FIG. 11-14, can be designated as a “0” and/or “0,0” position); similarly, associated subsets 1501 can be provided in any format compatible with vehicles 104.

[0160] Hence, processor 300 can plan a path for a vehicle 104 in physical space 901 by determining a starting point and an ending point in physical space 901, determining a path there between, and determining associated operational constraints along that path. In some implementations, such path planning can occur upon request from a self-driving vehicle 104 and/or when a task to be implemented is received from another computing device. As such, at block 803 of method 800, processor 300 provides, to a self-driving vehicle 104, respective given subsets 1502 of the operational constraints of a plurality of path portions that define an area in physical space 901 in which the self-driving vehicle 104 is to navigate, and further provides to the self-driving vehicle 104 associated positions of each of the plurality of path portions in the physical space 901. In other words, a self-driving vehicle 104 is provided with instructions and/or rules on navigating an area in physical space 901 and/or between two points in physical space 901. Furthermore, block 803 can be implemented when a request is received from a self-driving vehicle and/or from a computing device that assigns tasks to be implemented in physical space 901.

[0161] For example, attention is next directed to FIG. 16 which is substantially similar to FIG. 1, with like elements having like numbers. It is assumed in FIG. 16 that method 800 is being implemented at computing device 108 and in particular that computing device 108 has already performed block 801, and is presently implementing block 803. As such, computing device 108 is depicted providing, to vehicle 104-3, a respective given subset 1501 of operational constraints of the plurality of path portions that define an area in physical space 901 in which self-driving vehicle 104-3 is to navigate, with associated positions 1502 of each of the plurality of path portions in physical space 901. Thereafter, vehicle 104-3 implements the method associated with the received data, for example to navigate between two docks 905 to transport goods in a warehouse.

[0162] In particular, to implement method 800, processor can be further configured to: define one or more values associated with the respective given subset 1501 of operational constraints, or example to define a speed in an indicator 913-2; and provide, to a self-driving vehicle 104, the one or more values with the respective given subsets 1501 of the operational constraints of the plurality of path portions that define an area in physical space 901 in which the self-driving vehicle 104 is to navigate. In other words, values particular to given positions 1502 can be saved as part of a subset 1501 operational constraints in association with the positions 1502, and provided to a self-driving vehicle 104 to indicate, for example, how fast the self-driving vehicle 104 is to move in the associated position 1502 and/or positions 1502.

[0163] Similarly, processor 300 is further configured to provide, to the self-driving vehicle, an indication of the area with the respective given subsets 1501 of the operational constraints of the plurality of path portions that define the area, for example as positions 1502 as depicted in FIG. 16.

[0164] In other words, processor 300 can implement a path planning algorithm, for example to implement warehouse functionality in physical space 901. In particular, once block 801 of method 800 is implemented, processor 300 can plan a path between two docks 905 and then implement block 803 to transmit operational constraints of the path to a vehicle 104.

[0165] Persons skilled in the art will appreciate that there are yet more alternative implementations and modifications possible. For example, method 800 has been described as being implemented at computing device 108. However, in other implementations, one or more of blocks 801, 803 can be implemented at a self-driving vehicle 104 using, for example, an associated processor, memory, display device and input device such as, respectively, processor 205, memory 254, input device 268 and display device 269. In some of such implementations, at block 803, when a processor provides given subsets of the operational constraints and associated positions to a self-driving vehicle 104, such a processor can provide such data to itself. In other words, the processor referred to in method 800 can be a component of a self-driving vehicle 104.

[0166] Furthermore, path planning can occur at one or more of processor 300 of computing device 108 and a self-driving vehicle 104. In other words, one or more of processor 300 of computing device 108 and a self-driving vehicle 104 can be configured to: plan a path between two positions in physical space 901 based on the respective given subsets 1501 of the operational constraints of the plurality of path portions that define an area in which self-driving vehicle 104 is to navigate.
Furthermore, one or more of processor 300 of computing device 108 and a self-driving vehicle 104 can be configured to: plan a path between two positions in the physical space based on an indication of the area in the physical space defined by the plurality of path portions, and specifically the paths between physical points defined in the rendering of physical space 901 as described with reference to FIG. 12 to 14.

Furthermore, as described above, by providing areas and/or paths that define where a vehicle 104 can navigate in physical space 901, at least inherently vehicles 104 are prevented from entering zones and/or regions outside of such areas and/or paths. Hence, one or more of processor 300 and a self-driving vehicle 104 can be configured to: restrict a position of self-driving vehicle 104 only to positions in physical space 901 defined by the plurality of path portions that in turn define an area in physical space 901 in which self-driving vehicle 104 is to navigate.

While present implementations are described with respect to warehousing applications, methods, systems and devices described herein could be applied to other applications, including, but not limited to, systems in which self-driving vehicles are deployed to carry out tasks on a public and/or private highway system.

Furthermore, systems described herein can also be referred to as content distribution systems as content is distributed to self-driving vehicles 104, for example as described with reference to FIG. 16 in which content (e.g. subsets 1501 and positions 1502) are distributed by a computing device 108.

Those skilled in the art will appreciate that in some implementations, the functionality of computing device 108 and/or vehicles 104 can be implemented using pre-programmed hardware or firmware elements (e.g., application specific integrated circuits (ASICs), electrically erasable programmable read-only memories (EEPROMs), etc.), or other related components. In other implementations, the functionality of computing device 108 and/or vehicles 104 can be achieved using a computing apparatus that has access to a code memory (not depicted) which stores computer-readable program code for operation of the computing apparatus. The computer-readable program code could be stored on a computer readable storage medium which is fixed, tangible and readable directly by these components, (e.g., removable diskette, CD-ROM, ROM, fixed disk, USB drive, flash memory, and the like). Furthermore, the computer-readable program code can be stored as a computer program product comprising a computer usable medium. Further, a persistent storage device can comprise the computer readable program code. The computer-readable program code and/or computer usable medium can comprise a non-transitory computer-readable program code and/or non-transitory computer usable medium. Alternatively, the computer-readable program code could be stored remotely but transmittable to these components via a modem, network interface card, or other interface device connected to a network (including, without limitation, the Internet) over a transmission medium. The transmission medium can be either a non-mobile medium (e.g., optical and/or digital and/or analog communications lines) or a mobile medium (e.g., microwave, infrared, free-space optical or other transmission schemes) or a combination thereof.

Persons skilled in the art will appreciate that there are yet more alternative implementations and modifications possible, and that the above examples are only illustrations of one or more implementations. The scope, therefore, is only to be limited by the claims appended hereto.

We claim:

1. A system comprising:
   - a processor, a memory storing operational constraints for a self-driving vehicle, and a communications interface,
   - the processor configured to:
     - assemble a plurality of path portions to define an area in a physical space in which the self-driving vehicle is to navigate, each of the plurality of path portions associated with a respective given subset of operational constraints stored in the memory; and
     - provide, to the self-driving vehicle, respective given subsets of the operational constraints of the plurality of path portions that define the area, and associated positions of each of the plurality of path portions in the physical space.
   - The system of claim 1, further comprising an input device, wherein the plurality of path portions are assembled based on data received from the input device, the data indicating one or more of selections of each of the plurality of path portions and an order of the plurality of path portions.
   - The system of claim 1, wherein the area defined a path between at least two positions in the physical space.
   - The system of claim 1, wherein the processor is further configured to assemble the plurality of path portions by: rendering, at a display device, representations of the plurality of path portions at a representation of the physical space.
   - The system of claim 1, wherein each of the operational constraints stored in the memory define a rule that the self-driving vehicle is to follow when located in an associated position.

6. The system of claim 1, wherein the processor is further configured to: define one or more values associated with the respective given subset of operational constraints; and provide, to the self-driving vehicle, the one or more values with the respective given subsets of the operational constraints of the plurality of path portions that define the area.

7. The system of claim 1, wherein the processor is further configured to provide, to the self-driving vehicle, an indication of the area with the respective given subsets of the operational constraints of the plurality of path portions that define the area.

8. The system of claim 1, wherein each of the plurality of path portions comprises a feature associated with an operational constraint in the respective given subset of operational constraints associated therewith.

9. The system of claim 8, wherein the feature comprises a general indication of a rule that the self-driving vehicle is to follow, and the operational constraint associated with the feature comprises a more precise rule that the self-driving vehicle is to follow, the self-driving vehicle navigating according to the more precise rule and not the general indication of the rule when located in an associated position.

10. The system of claim 1, further comprising the self-driving vehicle, one or more of the processor and the self-driving vehicle configured to: plan a path between two positions in the physical space based on the respective given subsets of the operational constraints of the plurality of path portions that define the area.

11. The system of claim 1, further comprising the self-driving vehicle, one or more of the processor and the
self-driving vehicle configured to: plan a path between two positions in the physical space based on an indication of the area in the physical space defined by the plurality of path portions.

12. The system of claim 1, further comprising the self-driving vehicle, one or more of the processor and the self-driving vehicle configured to: restrict a position of the self-driving vehicle only to positions in the physical space defined by the plurality of path portions.

13. The system of claim 1, wherein the processor is a component of the self-driving vehicle.

14. A method comprising:
   at a system comprising: a processor, a memory storing operational constraints for a self-driving vehicle, and a communications interface,
   assembling, at the processor, a plurality of path portions to define an area in a physical space in which the self-driving vehicle is to navigate, each of the plurality of path portions associated with a respective given subset of operational constraints stored in the memory; and,
   providing, using the processor, to the self-driving vehicle, respective given subsets of the operational constraints of the plurality of path portions that define the area, and associated positions of each of the plurality of path portions in the physical space.

15. The method of claim 14, wherein the system further comprises an input device, and wherein the plurality of path portions are assembled based on data received from the input device, the data indicating one or more of selections of each of the plurality of path portions and an order of the plurality of path portions.

16. The method of claim 14, further comprising: assembling, using the processor, the plurality of path portions by: rendering, at a display device, representations of the plurality of path portions at a representation of the physical space.

17. The method of claim 14, further comprising one or more of:
   defining, at the processor, one or more values associated with the respective given subset of operational constraints; and provide, to the self-driving vehicle, the one or more values with the respective given subsets of the operational constraints of the plurality of path portions that define the area; and,

providing, using the processor, to the self-driving vehicle, an indication of the area with the respective given subsets of the operational constraints of the plurality of path portions that define the area.

18. The method of claim 14, wherein each of the plurality of path portions comprises a feature associated with an operational constraint in the respective given subset of operational constraints associated therewith.

19. The method of claim 14, wherein the system further comprises the self-driving vehicle, and the method further comprises: one or more of:
   planning, at one or more of the processor and the self-driving vehicle, a path between two positions in the physical space based on the respective given subsets of the operational constraints of the plurality of path portions that define the area;
   planning, at one or more of the processor and the self-driving vehicle, a path between two positions in the physical space based on an indication of the area in the physical space defined by the plurality of path portions; and,
   restricting, at one or more of the processor and the self-driving vehicle, a position of the self-driving vehicle only to positions in the physical space defined by the plurality of path portions.

20. A non-transitory computer-readable medium storing a computer program, wherein execution of the computer program is for:
   at a system comprising: a processor, a memory storing operational constraints for a self-driving vehicle, and a communications interface,
   assembling, at the processor, a plurality of path portions to define an area in a physical space in which the self-driving vehicle is to navigate, each of the plurality of path portions associated with a respective given subset of operational constraints stored in the memory; and,
   providing, using the processor, to the self-driving vehicle, respective given subsets of the operational constraints of the plurality of path portions that define the area, and associated positions of each of the plurality of path portions in the physical space.

* * * * *