(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
23 May 2024 (23.05.2024)

WIPO | PCT

(10) International Publication Number
WO 2024/108123 A1

(51) International Patent Classification:
B60W 60/00 (2020.01) *G06V 20/58* (2022.01)
G05D 1/00 (2024.01)

(21) International Application Number:
PCT/US2023/080288

(22) International Filing Date:
17 November 2023 (17.11.2023)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
63/426,446 18 November 2022 (18.11.2022) US
63/544,487 17 October 2023 (17.10.2023) US

(71) Applicant: **CORNELL UNIVERSITY** [US/US]; Center for Technology Licensing (CTL), 395 Pine Tree Road, Suite 310, Ithaca, NY 14850 (US).

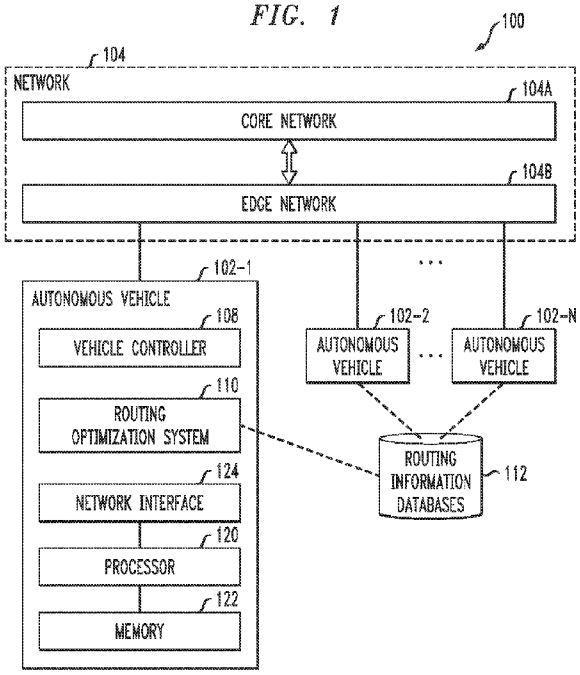
(72) Inventors: **VLADIMIRSKY, Alexander**; 26 Highgate Circle, Ithaca, NY 14850 (US). **GASPARD, Mallory**; 1101 Hasbrouck Apts., Ithaca, NY 14850 (US).

(74) Agent: **RYAN, Joseph, B.**; Ryan, Mason & Lewis, LLP, 48 South Service Road, Suite 100, Melville, NY 11747 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, MG, MK, MN, MU, MW, MX, MY, MZ, NA,

(54) Title: LANE-CHANGE TIMING DETERMINATIONS IN AUTONOMOUS VEHICLE ROUTING

FIG. 1



(57) Abstract: A method comprises obtaining information characterizing at least a portion of a routing environment of an autonomous vehicle, and formulating an optimization problem based at least in part on the obtained information, the optimization problem incorporating multiple levels of urgency for respective possible lane-change actions of the autonomous vehicle. The method further comprises solving the optimization problem utilizing a shortest-path algorithm, and generating at least one control signal for utilization in the autonomous vehicle based at least in part on results of solving the optimization problem and a current state of the autonomous vehicle. In some embodiments, the optimization problem is formulated as an opportunistically stochastic shortest-path (OSSP) problem configured to support selection between a plurality of actions including one or more stochastic transitions and one or more deterministic transitions at each of a plurality of states. The optimization problem may incorporate a transition cost function that guarantees the applicability of

WO 2024/108123 A1

[Continued on next page]

NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, ME, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

non-iterative label-setting methods to solve the OSSP problem.

LANE-CHANGE TIMING DETERMINATIONS IN AUTONOMOUS VEHICLE ROUTING

Related Applications

The present application claims priority to U.S. Provisional Patent Application Serial No. 63/544,487, filed October 17, 2023 and entitled "Lane-Change Timing Determinations in Autonomous Vehicle Routing," and further claims priority to U.S. Provisional Patent Application Serial No. 63/426,446, filed November 18, 2022 and also entitled "Lane-Change Timing Determinations in Autonomous Vehicle Routing," each incorporated by reference herein in its entirety.

Statement of Government Support

This invention was made with U.S. government support under Award Nos. 1645643 and 2111522 of the Division of Mathematical Sciences (DMS) of the National Science Foundation (NSF), Award No. FA9550-22-1-0528 of the Air Force Office of Scientific Research (AFOSR), and a National Defense Science and Engineering Graduate (NDSEG) Fellowship of inventor Mallory E. Gaspard. The U.S. government has certain rights in the invention.

Field

The field relates generally to information processing, and more particularly to routing techniques for autonomous vehicles.

Background

Autonomous vehicle technology has become more widespread over the last decade, but fully integrating these vehicles into everyday traffic scenarios requires onboard routing systems which successfully guide the autonomous vehicle to its destination while paying careful attention to the surrounding environment and giving significant regard for passenger safety. Conventional

routing techniques for autonomous vehicles exhibit various significant deficiencies, and improved routing techniques are therefore needed.

Summary

Illustrative embodiments provide techniques for lane-change timing determinations in autonomous vehicle routing.

It should be noted that the term "autonomous vehicle" as used herein is intended to be broadly construed so as to encompass a wide variety of different vehicle types that are at least partially autonomous. Accordingly, a semi-autonomous vehicle is considered a type of autonomous vehicle as that latter term is broadly used herein, as are other types of driver-assist arrangements in vehicles that have a human operator. It is therefore to be appreciated that an autonomous vehicle as the term is broadly used herein may be at least partially controlled by a human driver within the vehicle.

In one illustrative embodiment, a method illustratively comprises obtaining information characterizing at least a portion of a routing environment of an autonomous vehicle, and formulating an optimization problem based at least in part on the obtained information, the optimization problem incorporating multiple levels of urgency for respective possible lane-change actions of the autonomous vehicle. The method further comprises solving the optimization problem utilizing a shortest-path algorithm, and generating at least one control signal for utilization in the autonomous vehicle based at least in part on results of solving the optimization problem and a current state of the autonomous vehicle. The "current state" of the autonomous vehicle can comprise, for example, a current position of the autonomous vehicle, as well as additional or alternative state-related information.

In some embodiments, the optimization problem is formulated as an opportunistically stochastic shortest-path (OSSP) problem configured to support selection between a plurality of actions including one or more stochastic transitions and one or more deterministic transitions at each of a plurality of states.

Additionally or alternatively, the optimization problem may incorporate a transition cost function that guarantees monotone causality.

The term "shortest-path algorithm" as used herein is also intended to be broadly construed, so as to encompass, for example, any of a plurality of computational methods for improving one or more cumulative performance objectives in planning a graph traversal terminated upon reaching a predetermined subset of one or more nodes in that graph, where the nodes in the graph illustratively correspond to respective states, such as positions along a road-network. In some embodiments, a given shortest-path algorithm can include, for example, stochasticity in transitions and incurred costs as well as a plurality of different performance metrics. Examples of such performance metrics illustratively include time to target, overall fuel usage, safety of travel, passenger comfort, and/or impact on traffic conditions on the road.

It is to be appreciated that the foregoing arrangements are only examples, including examples of potential applications of the disclosed techniques, and numerous alternative arrangements are possible.

These and other illustrative embodiments include but are not limited to systems, methods, apparatus, processing devices, integrated circuits, and computer program products comprising processor-readable storage media having software program code embodied therein.

Brief Description of the Figures

FIG. 1 shows an information processing system comprising autonomous vehicles implementing functionality for lane-change timing determinations in autonomous vehicle routing in an illustrative embodiment.

FIGS. 2, 3 and 4 show examples of lane-change timing determinations in autonomous vehicle routing in illustrative embodiments.

FIG. 5 shows an example of lane-change planning for a three-lane highway with finite available actions in an illustrative embodiment.

FIG. 6 shows an example of lane-change planning for a three-lane highway with infinitely many available actions in an illustrative embodiment.

FIGS. 7 and 8 show aspects of an example of lane-change planning for multiple roundabouts with infinitely many available actions in an illustrative embodiment.

FIG. 9 shows an example of deterministic and probabilistic pruning of respective transition-equivalent and randomized actions in an illustrative embodiment.

FIGS. 10 and 11 show examples of geometric interpretations of monotone causality conditions in OSSP problems in illustrative embodiments.

FIG. 12 shows example computational stencils on a uniform Cartesian grid in illustrative embodiments.

FIGS. 13 and 14 illustrate example bounding arrangements for different speed profiles on one of the computational stencils of FIG. 12.

FIG. 15 shows example simplex and causality conditions for different convex speed profiles in illustrative embodiments.

FIG. 16 shows an example lane-level road network representation of a three-lane highway and associated available actions in an illustrative embodiment.

FIG. 17 shows an example of lane-change planning for a three-lane highway with four urgency levels in an illustrative embodiment.

FIG. 18 shows an example of lane-change planning for a three-lane highway with a continuous urgency spectrum in an illustrative embodiment.

FIG. 19 shows an example of lane-change planning for multiple roundabouts with a continuous urgency spectrum in an illustrative embodiment.

Detailed Description

Illustrative embodiments can be implemented, for example, in the form of information processing systems comprising one or more processing platforms each having at least one computer or other processing device. An autonomous vehicle or a portion thereof is considered

an example of such a processing platform as that term is broadly used herein. A number of examples of such information processing systems will be described in detail herein. It should be understood, however, that embodiments of the present disclosure are more generally applicable to a wide variety of other types of information processing systems and associated autonomous vehicles or other processing devices or other components. Accordingly, the term "information processing system" as used herein is intended to be broadly construed so as to encompass these and other arrangements.

Planning a long-distance trip for an autonomous vehicle typically requires a robust approach for dealing with uncertainties and random changes in the global environment (e.g., weather, traffic patterns, etc.). Robustness in some embodiments is attained by developing contingency plans with many decisions on different scales embedded in a stochastic routing policy, which illustratively describes not only where to turn, but also where to attempt lane-changes, how persistent one needs to be in those attempts, and how the directions for the rest of that trip should change if an attempted lane-change is unsuccessful. Some embodiments disclosed herein provide an approach for determining such routing policies, for example, by minimizing the expected cumulative cost up to the target, where the cost might incorporate time, fuel, and safety considerations. One or more such embodiments advantageously provide efficient computational techniques which are illustratively configured to handle a broad spectrum of lane-changing behaviors.

In some embodiments, vehicle routing problems may be approached by discretizing the roads used to reach the destination into a network of "nodes" connected by "edges" which represent the vehicle's ability to directly transition from one position to another. These nodes illustratively represent physical positions on the road that the vehicle may assume at any given time, and there is a cost associated with traveling along an edge between two connected nodes. This transition cost may represent distance, fuel usage, travel time, or another appropriate combination which reflects the vehicle's optimization criteria on the route to the destination. As the vehicle moves through the network from node to node, it accumulates a cost throughout its

trip. Accordingly, vehicle-routing algorithms are often configured to determine the cheapest route through the road-network to the target destination.

These types of deterministic routing problems may be solved quickly using Dijkstra's algorithm or another suitable shortest-path algorithm. However, in reality, the driving process is inherently stochastic and uncertain; e.g., a driver may attempt to switch lanes and fail due to approaching traffic in the adjacent lane, making it necessary to switch lanes later or to change to an altogether different path to the target. Failure to account for such uncertainties and stochastic perturbations may lead to unnecessarily long travel times, additional fuel usage, or even collisions with nearby traffic. Models which incorporate such uncertainties are based on selecting actions (rather than edges) to take as a next step, with each action described by its immediate cost and a probability distribution over the immediate consequences (i.e., possible transition from the current node of the graph). The resulting optimality equations belong to a broader class of Stochastic Shortest Path (SSP) problems and are generally more expensive computationally. However, a few important SSP-subclasses were shown in prior work to be treatable by faster label-setting (e.g., Dijkstra-like) algorithms.

Illustrative embodiments disclosed herein overcome numerous limitations of conventional approaches. For example, unlike conventional approaches, some embodiments disclosed herein provide an ability to model multiple levels of urgency in "tentative" lane-changes. Additionally or alternatively, some embodiments disclosed herein recognize that an escalation of urgency is only possible on a longer stretch of the road; modeling it as instantaneous is not particularly realistic and might result in suboptimal (and possibly even more dangerous) performance of autonomous vehicles.

In some embodiments disclosed herein, applicability criteria for Dijkstra's algorithm are derived on a broader class of SSPs and illustrated using lane-change planning applications. Example models disclosed herein allow for infinitely many available actions representing a continuous spectrum of urgency levels in lane-changing. Moreover, illustrative embodiments provide a fast algorithm for determining the optimal level of urgency that should be used based on

the current position relative to the target and the traffic conditions on the road-network, although it is to be appreciated that these and other embodiments can additionally or alternatively be configured to handle a setting with finitely many levels of urgency. In an example model disclosed herein, the level of urgency to use (along with the cost and the probability of successful lane-change) are selected for each position along the path. If the lane-change does not happen, the next attempt is only performed further along the road. This example model provides enhanced realism with virtually no increase in computational cost, thereby providing a significantly improved approach relative to conventional practice. Similar improvements are provided in other embodiments.

In some embodiments, there are multiple/interacting levels of trip planning and vehicle control for autonomous vehicles. At an example upper level, a Strategic Plan (SP) is basically a sequence of deterministic turn-by-turn directions from the starting position S to a specified destination D, similar to what is currently produced by Google Maps and similar services. SP is usually precomputed based on the prior/predicted traffic patterns and weather conditions. The technology is implemented in such a way that "close to optimal" directions to D are produced not just from the original starting S, but also from all nodes/positions falling into a small "neighborhood" of the entire planned S->D path. If, for whatever reason, the vehicle slightly deviates from that chosen path, it ends up at one of those neighboring nodes, making it possible to use the original SP from there. If it deviates more, recomputing the SP (starting from the new/current S) becomes necessary.

At a much more basic level, there is a number of Operational Control (OC) algorithms (e.g., continuous trajectory planning, actuator control sequences), which dynamically accelerate/decelerate and steer the vehicle, attempting to follow the SP as much as possible given the real-time-updated data from the sensors and safety considerations. If an SP-prescribed maneuver is not executed in time by OC (e.g., missing the planned offramp from the highway), the vehicle deviates from the planned path, at which point it either follows a somewhat sub-optimal path or a full SP-recomputation is performed from the new "starting" location. In general, OC is

real-time, while SP is generally an offline algorithm, but usually fast enough to recompute in several seconds. Occasional SP-recomputation is unavoidable when caused by global changes in the environment (e.g., a sudden rainstorm or a major accident ahead). But if instead it is frequently caused by inability of OC to follow the plan, this starts to impact the performance metrics such as the fuel efficiency, safety, passenger comfort and time-to-destination.

In addition to SP and OC, there are also intermediate-level tactical decisions such as, for example, planning and performing lane-change operations, also referred to in some embodiments herein as lane-switching maneuvers or LSMs, which are utilized to execute the SP and improve performance metrics. Availability of lane-level details in modern road-network models makes it possible to build a hybrid Strategic/Tactical Plan (STP), which includes detailed instructions for turns as well as LSMs, based on the data of traffic loads expected/statistically observed in different lanes. However, OC may not be able execute an LSM at a precisely pre-planned optimal time (e.g., safety considerations and local traffic patterns in the current or desired new lane might make this difficult – perhaps the gaps between vehicles in that adjacent lane are currently too narrow). So, if the STP instructions are deterministic (as is the case in SP), this will either lead to frequent replanning (whenever LSM execution fails) or would require building in additional for-sure-execution time delays (e.g., planning a switch to an exit lane two miles in advance before the offramp). Either option leads to performance degradation and is undesirable.

Some embodiments herein provide an efficient method to determine the optimal routing policy for a fully-autonomous or semi-autonomous vehicle to determine its STP, which tells the vehicle when to attempt a lane-change, where to attempt a lane-change, how much urgency the vehicle should have in the attempted lane-change (i.e., how willing the vehicle should be to adjust its velocity to ensure a successful lane-change), and how the plan should be adjusted if the attempted lane-change fails.

These and other embodiments disclosed herein can provide a number of significant advantages relative to conventional approaches, including one or more of the following:

(a) allows a spectrum of lane-change urgency levels and provides a framework for modeling the costs expected when using those urgency levels;

(b) is more usable as an assistive technology for human drivers;

(c) develops criteria for applicability of an additional label-setting method (e.g., can use Dijkstra's algorithm and Dial's algorithm); and

(d) avoids modeling LSMs as escalating behavior on a short road segment.

The features (a) and (d) result in more nuanced and realistic models and allow for significant improvements in performance metrics; feature (b) broadens potential applications to the semi-autonomous setting, while feature (c) in some cases makes the STP easier to compute since Dial's algorithm is parallelizable and known to be faster than Dijkstra's algorithm on some problems.

FIG. 1 shows an information processing system 100 comprising a plurality of autonomous vehicles 102-1, 102-2, . . . 102-N that communicate with a network 104. The network 104 illustratively comprises a core network 104A and an edge network 104B, which are assumed to comprise respective sets of core devices and edge devices, with each of the autonomous vehicles 102 being in communication with at least one or more edge devices of the edge network 104B. The core devices may comprise, for example, one or more cloud-based devices, while the edge devices are deployed in relatively closer proximity to the autonomous vehicles 102. The autonomous vehicles 102 are configured to implement functionality for lane-change timing determination in autonomous vehicle routing as disclosed herein, possibly with involvement of one or more of the core devices and and/or one or more of the edge devices. As indicated previously, the term "autonomous vehicle" as used herein is intended to be broadly construed, so as to encompass, for example, semi-autonomous vehicles and other driver-assist arrangements in vehicles that have a human operator.

Additionally or alternatively, a given autonomous vehicle can have multiple distinct modes of operation, such as a fully-autonomous mode and a semi-autonomous mode, the latter involving

at least some control by a human driver, with the autonomous vehicle operating in different ones of the modes at different times.

The autonomous vehicle 102-1 illustratively comprises a vehicle controller 108 and a routing optimization system 110. The vehicle controller 108 may comprise a combination of multiple distinct vehicle controllers for respective different subsystems of the autonomous vehicle 102-1. Each such subsystem may comprise its own set of one or more instances of the vehicle controller 108. Such subsystems are omitted from the figure for clarity and simplicity of illustration, but can include any of a wide variety of different autonomous vehicle subsystems that are utilized to control various aspects of the operation of the autonomous vehicle 102-1, such as lane-change operations, steering operations and other routing-related operations. The routing optimization system 110 interacts with vehicle controller 108, the autonomous vehicle subsystems and their associated sensors, as well as one or more routing information databases 112, to obtain information that is utilized in implementing routing-related functionality of the type disclosed herein within the autonomous vehicle 102-1. It is assumed that each of one or more of the other autonomous vehicles 102 is configured in a manner similar to that illustrated for autonomous vehicle 102-1 in the figure.

In operation, the routing optimization system 110 of the autonomous vehicle 102-1 is illustratively configured to obtain information characterizing at least a portion of a routing environment of the autonomous vehicle 102-1, and to formulate an optimization problem based at least in part on the obtained information, with the optimization problem incorporating multiple levels of urgency for respective possible lane-change actions of the autonomous vehicle 102-1. The routing optimization system 110 is further configured to solve the optimization problem utilizing a shortest-path algorithm. At least one control signal is generated for utilization in the autonomous vehicle 102-1, illustratively by at least one of the vehicle controller 108 and the routing optimization system 110, based at least in part on results of solving the optimization problem and a current state of the autonomous vehicle.

The term "current state" as used herein is intended to be broadly construed, and as indicated previously, can comprise, for example, a current position of the autonomous vehicle, as well as additional or alternative state-related information.

Also, "results" of solving an optimization problem as the term is used herein should not be viewed as being limited to particular maxima, minima or other specific optimization results, and instead should be broadly construed as comprising one or more outputs and/or other related information generated by application of a shortest-path algorithm to a formulated optimization problem.

In some embodiments, separate optimization problems are formulated and solved separately for each of the autonomous vehicles 102 on an individual basis, although other arrangements are possible.

Accordingly, in some embodiments, the routing optimization system 110 of autonomous vehicle 102-1 is configured to manage STP planning for that particular vehicle alone and does not influence, nor is it influenced by, the individual STP planning for any other ones of the autonomous vehicles 102.

In some embodiments, obtaining information characterizing at least a portion of a routing environment of the autonomous vehicle 102-1 comprises populating one or more data structures characterizing possible transitions between a plurality of states as the autonomous vehicle 102-1 travels in the routing environment. For example, such information may be obtained at least in part from one or more of the routing information databases 112, which are illustratively accessible to one or more of the other autonomous vehicles 102. In some embodiments, the routing information databases 112 can provide the autonomous vehicle 102-1 with external data on predicted general traffic conditions (e.g., general traffic delays near on-ramps, congestion in terms of average speed slowdown, slowdowns due to inclement weather, etc.), as well as a wide variety of additional or alternative types of information characterizing the routing environment. At least portions of the routing information databases 112 in some embodiments may be implemented at least in part with one or more core devices and/or one or more edge devices of the network 104. For example, such

databases may be accessible via servers of the network 104. Additionally or alternatively, at least portions of the routing information databases 112 may be implemented at least in part within respective ones of the autonomous vehicles 102, such as within memories or other storage devices of the autonomous vehicles 102.

The routing optimization system 110 in formulating an optimization problem based at least in part on the obtained information illustratively formulates the optimization problem as an opportunistically stochastic shortest-path (OSSP) problem configured to support selection between a plurality of actions including one or more stochastic transitions and one or more deterministic transitions at each of a plurality of states.

Additionally or alternatively, the routing optimization system 110 is illustratively configured to formulate the optimization problem to incorporate a transition cost function that guarantees monotone causality.

For example, illustrative embodiments disclosed herein provide an OSSP problem formulation using a monotonically causal transition cost function that ensures that the OSSP problem can be effectively solved utilizing fast and non-iterative label-setting shortest-path algorithms.

In some embodiments, the OSSP framework provides a physically-realistic setting within which to pose STP determination problems since it provides an autonomous vehicle with options between deterministic transitions and stochastic transitions corresponding to how urgent the lane-change attempt is from its current position. At locations where an adjacent lane or merge opportunity is physically available, the OSSP formulation is illustratively configured to offer at least the following two deterministic transition options: moving to the next state in the same lane (no urgency, probability of success $p = 0$) or moving to the next state in the adjacent lane (full urgency, probability of success $p = 1$), and some number of stochastic transition options, where the chances of landing at one of those two successor states is characterized by the urgency level p in (0,1). If there is no physically available lane-change opportunity from the vehicle's current position, the OSSP formulation in some embodiments will only offer the deterministic "stay-in-

lane" action (no urgency, probability of success $p = 0$). On the other hand, if the lane ends at the vehicle's current position, the OSSP formulation will only offer the deterministic "forced change" action (full urgency, probability of success $p = 1$).

Accordingly, in some embodiments, the multiple levels of urgency for respective possible lane-change actions of the autonomous vehicle 102-1 illustratively comprise a plurality of discrete urgency levels.

In other embodiments, the multiple levels of urgency for respective possible lane-change actions of the autonomous vehicle 102-1 illustratively comprise a plurality of urgency levels selected from a continuum of urgency levels.

The routing optimization system 110 in solving the optimization problem utilizing a shortest-path algorithm in some embodiments is more particularly configured to select a particular one of a plurality of shortest-path algorithms implemented in the routing optimization system 110, and to solve the optimization problem utilizing the selected shortest-path algorithm.

In some embodiments, the selectable shortest-path algorithms comprise at least one label-setting shortest-path algorithm.

More particularly, the selectable shortest-path algorithms in some embodiments may comprise, for example, at least Dijkstra's algorithm and Dial's algorithm, although additional or alternative shortest-path algorithms can be used, such as Dijkstra-like algorithms and/or Dial-like algorithms. In some embodiments, the routing optimization system 110 implements only a single shortest-path algorithm.

In some embodiments, solving the optimization problem utilizing a shortest-path algorithm illustratively comprises generating in each of a plurality of stages of the shortest-path algorithm an optimal action and a corresponding cumulative cost, with the optimal action having associated therewith a particular one of the levels of urgency.

As a more particular example, the optimal action illustratively comprises a particular one of the lane-change actions and the associated one of the levels of urgency is indicative of a corresponding degree to which a velocity of the autonomous vehicle 102-1 can be adjusted,

illustratively at least in part by the vehicle controller 108, in conjunction with attempting the particular lane-change action.

In some embodiments, solving the optimization problem illustratively provides the optimal expected cost and optimal lane-change action at each of a plurality of positions corresponding to respective nodes of a road-network. Label-setting shortest-path algorithms are particularly useful in these and other scenarios since no successive iterations or successive executions of a given such algorithm at each individual position are required to determine the optimal expected cost and action at every position within the road-network. It is possible, however, that the algorithm may need to be re-executed under certain conditions, such as a significant disruption in the surrounding environment (e.g., unexpected road closures due to a car accident or deteriorating weather conditions), or the user changing the destination.

In some embodiments, generating at least one control signal for utilization in the autonomous vehicle 102-1 illustratively comprises generating a lane-change control signal or other type or arrangement of one or more control signals in the vehicle controller 108 of the autonomous vehicle 102-1. Such a control signal illustratively implements at least a portion of a particular lane-change action of the autonomous vehicle 102-1 and/or other related functionality associated with the particular lane-change action.

Additionally or alternatively, generating at least one control signal illustratively comprises generating at least one control signal for utilization in a driver-assistance system of the autonomous vehicle 102-1. Such a driver-assistance system may be part of the vehicle controller 108, or may be implemented as a separate system that comprises at least a portion of the vehicle controller 108. For example, a control signal generated for utilization in a driver-assistance system can be configured to cause the driver-assistance system to generate an audible and/or visible prompt of a suggested lane-change action to a human driver of a semi-autonomous vehicle. A wide variety of other types and arrangements of one or more control signals may be generated based at least in part on results of solving an optimization problem and a current state of the autonomous vehicle 102-1 as disclosed herein.

As another example, in an embodiment in which the autonomous vehicle 102-1 is operating in a fully-autonomous manner, the control signals may be comprise instructions produced in accordance with the STP and transmitted to operational control modules which are responsible for physically steering and controlling the vehicle.

As another example, the autonomous vehicle 102-1 can be operated in a semi-autonomous manner, in which it is at least partially controlled by a human driver, and the control signals can trigger behavioral suggestions that the human driver may or may not choose to implement. These behavioral suggestions may be, for example, displayed on a screen built into the vehicle itself, on a handheld GPS system, or on a mobile device via an application. Numerous other arrangements are possible.

It is therefore to be appreciated that the term "control signal" as used herein is intended to be broadly construed. In some embodiments, control signals are distinct from "actions," which are also intended to be broadly construed herein, in that the latter illustratively comprise behaviors prescribed or recommended by the STP, while control signals illustratively comprise instructions, commands or other signals utilized by and/or generated within the vehicle controller 108, which may comprise multiple operational control units of the autonomous vehicle 102-1, to physically implement those actions.

It should also be noted in this regard that solving the OSSP to determine the STP in some embodiments only yields one optimal lane-change action per each position in the road-network. However, the optimal lane-change action obtained may then be used to inform many other tasks handled by other subsystems within the autonomous vehicle 102-1 which are necessary to physically execute the optimal lane-change action.

The autonomous vehicle 102-1 or a portion thereof may be viewed as an example of what is more generally referred to herein as a processing platform comprising one or more processing devices, each comprising a processor coupled to a memory. Other types of processing platforms can be used in illustrative embodiments, such as a processing platform that comprises a combination of one or more processing devices of the autonomous vehicle 102-1, one or more core

devices of the core network 104A, and/or one or more edge devices of the edge network 104B. Accordingly, in some embodiments herein, routing-related functionality implemented for one or more of the autonomous vehicles can be distributed across multiple separate processing devices, possibly including one or more core devices and/or one or more edge devices, each comprising a processor coupled to a memory.

Detailed examples of particular implementations of vehicle controller 108 and routing optimization system 110 are described elsewhere herein.

Although the vehicle controller 108 and routing optimization system 110 are shown in the figure as both being deployed on the autonomous vehicle 102-1, other arrangements are possible. For example, the routing optimization system 110 in some embodiments can be deployed in whole or in part outside of the autonomous vehicle 102-1, such as on one or more core devices and/or one or more edge devices of the network 104. It is therefore possible, for example, that the routing optimization system 110 can be implemented at least in part in the cloud, illustratively utilizing one or more core devices, with corresponding optimization results being transmitted to the autonomous vehicle 102-1 via one or more edge devices. Numerous other arrangements of one or more processing devices can be used. Examples of such processing devices include computers, servers or other processing devices arranged to communicate over at least a portion of the network 104. The network 104 can comprise a combination of multiple networks of different types, such as, for example, a global computer network such as the Internet, a wide area network (WAN), a local area network (LAN), a satellite network, a telephone or cable network, a cellular network such as a 4G or 5G network, a wireless network implemented using a wireless protocol such as Bluetooth, WiFi or WiMAX, or various portions or combinations of these and other types of communication networks.

The autonomous vehicle 102-1 in the present embodiment further comprises a processor 120, a memory 122 and a network interface 124. The processor 120 is assumed to be operatively coupled to the memory 122 and to the network interface 124 as illustrated by the interconnections shown in the figure.

The processor 120 may comprise, for example, a microprocessor, an application-specific integrated circuit (ASIC), a field-programmable gate array (FPGA), a central processing unit (CPU), a graphics processing unit (GPU), a tensor processing unit (TPU), an arithmetic logic unit (ALU), a digital signal processor (DSP), or other similar processing device component, as well as other types and arrangements of processing circuitry, in any combination. At least a portion of the functionality of the autonomous vehicle 102-1, such as the routing-related operations of the vehicle controller 108 and the routing optimization system 110 as disclosed herein can be implemented using such circuitry.

The memory 122 stores software program code for execution by the processor 120 in implementing portions of the functionality of the autonomous vehicle 102-1. For example, at least portions of the functionality of vehicle controller 108 and routing optimization system 110 can be implemented using program code stored in memory 122.

A given such memory that stores such program code for execution by a corresponding processor is an example of what is more generally referred to herein as a processor-readable storage medium having program code embodied therein, and may comprise, for example, electronic memory such as SRAM, DRAM or other types of random access memory, flash memory, read-only memory (ROM), magnetic memory, optical memory, or other types of storage devices in any combination.

Articles of manufacture comprising such processor-readable storage media are considered embodiments of the present disclosure. The term "article of manufacture" as used herein should be understood to exclude transitory, propagating signals.

Other types of computer program products comprising processor-readable storage media can be implemented in other embodiments.

In addition, illustrative embodiments may be implemented in the form of integrated circuits comprising processing circuitry configured to implement processing operations associated with one or both of the vehicle controller 108 and the routing optimization system 110 as well as other related functionality.

The network interface 124 is configured to allow the autonomous vehicle 102-1 to communicate over the network 104 with other system elements, and may comprise one or more conventional transceivers.

It is to be appreciated that the particular arrangement of components and other system elements shown in FIG. 1 is presented by way of illustrative example only, and numerous alternative embodiments are possible. For example, other embodiments of information processing systems can be configured to implement lane-change timing determinations and other routing optimization functionality of the type disclosed herein.

Additional illustrative embodiments involving examples of lane-change determinations will now be described with reference to FIGS. 2, 3 and 4.

By way of example, a vehicle at any position A in an internal-lane on a multi-lane highway will have at most three possible positions it can transition to:

1. Position B_1 which is in the same lane as A.
2. Position B_2 which is in the right-adjacent lane.
3. Position B_3 which is in the left-adjacent lane.

All three of these are down the road relative to A. For illustrative purposes, in what follows we refer to decisions made at some position A in the left lane of a two-lane road, as shown in FIG. 2, but the general case can be described in a similar manner. As the vehicle moves through the road-network, its transitions are penalized by an instantaneous "cost." The cost of transitioning from one position to another is described by a mathematical function which incorporates time to destination, fuel use, safety, or a combination of other important physical factors. The STP produces a routing policy, which minimizes the expected cumulative cost (e.g., the sum of costs of individual transitions).

FIG. 2 shows an example lane change schematic for a two-lane road. As indicated above, a vehicle at position A has two transition possibilities: Continue driving in the current lane by transitioning to B_1 along the solid horizontal line, or change lanes by transitioning to B_2 along the solid diagonal line. The lanes are illustratively separated by ω meters, and the successor position

(B_1) in the vehicle's current lane is ℓ meters straight down the road from the vehicle's current position (A). The vehicle's successor position in the adjacent lane (B_2) is $h = \sqrt{\omega^2 + \ell^2}$ meters diagonally down the road from A. The vehicle cannot transition between positions connected by a gray dashed line.

In some embodiments, multiple LSM-urgency levels are provided in autonomous and semi-autonomous settings, as will now be described.

By way of example, the instructions produced by an STP algorithm specify whether to stay in the same lane $(A \longrightarrow B_1)$ or to attempt switching to an adjacent lane $(A \longrightarrow B_2)$. Possible actions in this context include the following, with varying probability of a lane-change:

1. α_0: Remain in the current lane, certainly transition to B_1.
2. $\tilde{\alpha}$: Tentative LSM $(A \longrightarrow B_2$ with a fixed probability $\tilde{p} \in (0,1)$; otherwise, $A \longrightarrow B_1)$.
3. $\alpha^{\#}$: Forced LSM: certainly transition $A \rightarrow B_2$ regardless of the cost.

Illustrative embodiments disclosed herein recognize that the level of effort used to carry out the LSM can vary as well – we therefore implement a lane-change urgency level and allow multiple tentative LSM actions $\tilde{\alpha}(p)$, parametrized by their respective urgency (probability of success) levels p. This interpretation also encodes the two deterministic options above since $\tilde{\alpha}(0) = \alpha_0$ and $\tilde{\alpha}(1) = \alpha^{\#}$.

This exemplary approach advantageously allows handling finitely many urgency levels $(0 = p_0 < p_1 < \cdots < p_{L-1} < p_L = 1)$ or even a continuous spectrum of urgency levels $p \in [0,1]$. In the fully-autonomous vehicle setting, the STP-optimal p_* encodes the allowable range of in-lane speed adjustments that might be used to make the lane-change possible. In the semi-autonomous setting, the STP-prescribed urgency level p_* can be communicated to the driver in audio or visual form – e.g., as a size (or color intensity) of a "Lane Change Needed" bar/signal on an assistive navigation panel. Such an assistive navigation panel may be viewed as an illustrative example of at least a portion of what is more generally referred to herein as a "driver-assistance system."

As mentioned above, given a specific destination D, the STP illustratively prescribes an optimal action to take at each location, with the goal of minimizing the expected cumulative cost (i.e., the sum of costs of individual actions taken until D is reached). This makes it necessary to describe the immediate cost of all available actions, parametrized by their LSM-urgency-level (or, equivalently, their actual successful lane-change probability) p. Suppose $K(p)$ is a function describing the cost of each available action $\tilde{a}(p)$. It is disclosed herein that efficient label-setting algorithms can be used to solve this STP problem provided:

1. $K(p)$ is convex (or convexified);
2. $K(p)$ is monotonically increasing; and
3. There exists some $\delta \geq 0$ such that:

$$K(p) \geq K(1)p + (1-p)\delta$$

for every available urgency level $p > 0$.

The first two conditions above are utilized for modeling the LSM and are not significantly restrictive. The third condition is utilized to make fast algorithms applicable to perform the STP correctly. Without it, re-planning would become much more time-consuming. If the third condition is satisfied with $\delta = 0$, Dijkstra's algorithm will be applicable in our more general models as well. But if that condition also holds with some $\delta > 0$, this makes it possible to use a parallelizable Dial's algorithm, which can be even more efficient on many graphs. In general, the wider is the "bucket width" δ, the faster Dial's algorithm will be. In models with a continuous spectrum of urgency levels $p \in [0,1]$, it can be shown that the largest usable bucket width in some embodiments is $\delta_{max} = K(1) - K'(1)$.

Illustrative embodiments disclosed herein further provide a flexible cost-modeling framework.

There are endless examples of transition cost functions which satisfy the criteria above and are also relevant in routing models. Below we provide several representative examples.

When there are only finitely many available actions, the transition cost structure is finite. By way of example, suppose that the vehicle has $L + 1$ total actions $\alpha_0, \ldots, \alpha_L$ to choose from, with

α_0 associated with $p_0 = 0$ (no urgency), α_L associated with $p_L = 1$ (complete urgency), and all other (intermediate) α_i associated with urgency levels $p_i \in (0,1)$ sorted in ascending order (i.e., $i < j$ implies $p_i < p_j$). In this exemplary setting, the transition cost associated with each available action α_i is illustratively determined by the cost of transitioning to the next position in the current lane Y_0, the associated urgency level in changing lanes p_i, and the additional penalty Y_i for attempting to change lanes at the associated urgency level. In some embodiments, the actions can be used progressively, with the next level of urgency tried only if the previous ones have failed. In other words,

$$K(p_0) = Y_0; \text{ and } K(p_i) = K(p_{i-1}) + (1 - p_{i-1})Y_i \text{ for } i = 1, \dots, L.$$

With $L = 2$, this yields the same "three actions" structure as previously described. We can certainly use it with higher k values too, and assuming that the third condition listed above is satisfied, we can compute the STP efficiently.

It should be noted that this example approach is somewhat flawed because it essentially assumes an "instant escalation" behavior. In reality, each tentative LSM action takes time and you cannot attempt multiple LSMs on a short stretch of the road between A and B_1. Multiple lane-change attempts within such a short distance would result in a less-smooth and/or less-comfortable ride for passengers and increased risk of collisions. In some implementations, LSMs can typically take, for example, anywhere from half a second to 6+ seconds to fully execute. If $\ell = 10m$ and the vehicle is traveling at 55 mph, this gives us only ≈ 0.4 seconds until we reach B_1. Performing multiple LSM attempts within this time is unrealistic and even dangerous. Thus, we believe that using $K(p)$ models that drop this "multiple-instant-attempts" assumption will be both safer and more reliable.

Since our example framework can accommodate a continuous spectrum of urgency levels between 0 and 1 (i.e., infinitely many available intermediate actions), it is also natural to use cost functions which are smooth and continuous. One simple example is

$$K(p) = \beta p^2 + \gamma$$

where β and γ are positive parameters, whose values reflect current congestion on the road or learned individual driver's behavior. This $K(p)$ always satisfies our first two conditions. If we take $\delta = 0$ to use Dijkstra's algorithm, this $K(p)$ also satisfies the third condition above as long as $\beta \leq \gamma$. This model has an additional advantage in that the optimal p can be computed analytically at each specific node A.

Another alternative in modeling $K(p)$ is to pass a smooth curve through a finite set of (urgency, cost) points, perhaps collected by studying the driver behavior and traffic patterns in adjacent lanes, making sure the resulting $K(p)$ satisfies the three conditions stipulated above. In some embodiments disclosed herein, this can be done through utilization of Rational Bezier Curves. The shape of a Rational Bezier Curve is determined by control points, and the curve is guaranteed to pass through its first and last control points. Given three (urgency, cost) pairs $(0, K_0), (\tilde{p}, \tilde{K})$, and $(1, K_1)$, the Rational Bezier Curve passing through them is given parametrically by

$$P(t) = \frac{2w_1(1-t)t\frac{(K_0-\delta)}{(K_1-\delta)} + w_2t^2}{w_0(1-t)^2 + 2w_1(1-t)t + w_2t^2}$$

$$K(t) = \frac{(1-t)^2w_0K_0 + 2w_1(1-t)tK_0 + K_1w_2t^2}{w_0(1-t)^2 + 2w_1(1-t)t + w_2t^2}$$

where t is a parameter between 0 and 1 that "sweeps out" the curve into its full, intended shape, and ω_i are positive numbers representing the "weight" or "influence" that the associated control point has in determining values along the transition cost curve. To determine the ω_1 value which guarantees that the Rational Bezier Curve passes through the point $(\tilde{p}, \tilde{K})$, illustrative embodiments disclosed herein solve a system of two nonlinear equations once for t and ω_1. This example solve is carried out before the main algorithm runs, and thus it adds no additional computation time in determining the optimal routing policy.

By way of illustrative example only, a conventional model known in the art is based on three such pairs, as follows:

$$(0, c_s),\ (\tilde{p}, c_s + c_{lc}),\ \text{and}\ \left(1, c_s + c_{lc} + (1-\tilde{p})c_{flc}\right).$$

Starting from the above 3-action model, a compatible continuous-urgency-spectrum model is obtained using a Rational Bezier Curve as follows:

$$P(t) = \frac{2\omega_1(1-t)t\left(\frac{c_s - \delta}{c_s + c_{lc} + (1-\tilde{p}_1)c_{flc} - \delta}\right) + \omega_2 t^2}{\omega_0(1-t)^2 + 2\omega_1(1-t)t + \omega_2 t^2}$$

$$K(t) = \frac{(1-t)^2\omega_0 c_s + 2\omega_1(1-t)tc_s + \omega_2 t^2\left(c_s + c_{lc} + (1-\tilde{p})c_{flc}\right)}{\omega_0(1-t)^2 + 2\omega_1(1-t)t + \omega_2 t^2}$$

It is to be appreciated that this is only an example, and numerous alternative arrangements may be used in other embodiments.

In some embodiments, a Rational Bezier Curve is constructed to always pass through the following two points: (0, stay in lane cost), (1, forced lane change cost). A third, intermediate point that the Rational Bezier Curve will pass through can then be selected according to one or more application-specific criteria. For example, in some applications, the selection of the third point will be informed by available traffic data which may be utilized to determine the probability of lane change success under the current conditions and compute a corresponding transition cost. Assuming that the third point satisfies the monotone causality condition of illustrative embodiments herein, and that the control points are selected as described herein, a label-setting method can be used to solve the OSSP in a highly efficient manner.

Illustrative embodiments disclosed herein advantageously provide more-nuanced STPs on complex road-networks. Such embodiments make it possible for us to generate nuanced STPs for autonomous vehicles navigating road-networks far more complex than the two-lane scenario described in FIG. 2. That two-lane scenario is just a building block for locally representing the transitions available in a larger graph. For example, these two-lane blocks may be combined to construct the corresponding road-network and determine the STP on a multi-lane highway, as will now be described in conjunction with FIG. 3.

FIG. 3 shows a road-network segment of a three-lane highway with an on ramp connecting to the highway. Part (A) of the figure shows a network representation with on-ramp connection at position 8. Vehicles at positions 1, 2, 3, 7, 8 and 9 only have the option to stay in lane or switch to the middle. Vehicles at positions 4, 5 and 6 have the option to stay in the current lane, switch to the left lane, or switch to the right lane. After making this decision, the vehicle then only has two possible positions it can end up in after a lane-change attempt: the successor in the current lane or the successor in the desired adjacent lane. In the merge lane, vehicles at positions 10 and 11 may only transition to the successor in the same lane (as indicated by the arrows) since the merge-lane is treated as a one lane entity which only connects to the main three-lane road at position 8. Part (B) of the figure shows a sample STP produced by our algorithm for an autonomous vehicle 500m away from its destination in the left-lane. The transitions are penalized according to the Rational Bezier Curve described above, where c_{right}, c_{mid}, and c_{left} are the stay-in-lane costs for the right, middle, and left lanes respectively. Congestion at the merge is very high, and the vehicle also takes this into account in STP planning.

Such two-lane blocks may also be used to represent the transitions available at junctions between large and small roundabouts in a roundabout network. This example is inspired by the "Magic Roundabout" in Swindon, UK. We thus adopt UK driving standards on this example network, as will be described below in conjunction with FIG. 4. The resulting STP automatically comes up with optimal backup plans. For example, when the outer roundabout has more vehicles on it, a vehicle approaching in the left lane of the SW entry, will try (with increasing urgency) to switch to the right lane and then continue counterclockwise - avoiding some of the heavier traffic by using the innermost roundabout. But if these LSM attempts fail due to increased traffic congestion in the right lane, the vehicle will proceed to its destination clockwise via the outermost roundabout.

FIG. 4 in part (A) thereof shows a graph representation of a road-network with five connected roundabouts. All possible transitions from each node are shown, and the arrow indicates the transition direction between each node. The target destination along the southeastern exit is

circled in black. Part (B) of the figure shows snapshots of a sample STP computed using an illustrative embodiment of an algorithm as disclosed herein, along each of the three entryways into the roundabout network. The transitions are penalized by the quadratic cost function $K(p) = \beta p^2 + \gamma$. Solid arrows in part (B) indicate a deterministic transition between two states (i.e., stay in lane actions and forced lane change actions), and dashed arrows in part (B) indicate that a lane change attempt with the specified urgency level p is optimal.

It is to be appreciated that the particular arrangements described above in conjunction with FIGS. 2, 3 and 4 are only examples, intended to illustrate various aspects of the present disclosure, and can be varied in other embodiments.

In some embodiments, routing optimization functionality as disclosed herein is implemented at least in part utilizing software program code, illustratively written in C++ and Python. By way of example, the C++ portion of the software is responsible for storing the provided road-network and executing Dijkstra's algorithm to determine the STP for a vehicle traveling through this network to a specified destination, and the Python portion of the software is responsible for processing the computational results produced by the C++ code and using them to generate a visualization of the STP. The C++ code and Python code make use of common external libraries and packages, illustratively including, in the C++ portion, Boost, and in the Python portion, Numpy, NetworkX and Matplotlib. Other types of software code and programming languages can be used in other embodiments.

The C++ code is illustratively configured to store the road-network representation for a variety of general and specific real-world examples as two-dimensional Boost multiarray structures. Each state in the road-network is assigned a unique label ranging from 1 to n, where n is the total number of states in the road-network, and we assign label "1" to the destination state. The stay-in-lane transition cost is denoted as c_s and the forced lane change transition cost is denoted as c_f.

In some embodiments, road-networks are illustratively represented as directed graphs. This ensures that traffic continues to flow in the correct direction and that vehicles are not able to unsafely and illegally switch their direction of motion.

We refer to the two-dimensional multiarray for a specific road-network as the Deterministic Transition Cost Matrix whose (i, j) entries are assigned as follows:

(i, j) entry $= 0 \longrightarrow$ no direct deterministic transition possible from state i to state j

(i, j) entry $= c_s \longrightarrow$ deterministic stay-in-lane transition possible from state i to state j

(i, j) entry $= c_f \longrightarrow$ deterministic forced lane change transition possible from state i to state j.

To compute the STP, the two-dimensional multiarray containing the transition information between each state in the network is passed to a function within the C++ code which is responsible for executing Dijkstra's algorithm to solve the routing problem and reporting the optimal expected cost at each state. When a state is processed in Dijkstra's algorithm, the C++ code executes an optimization routine to determine the optimal action in the following ways depending on whether there are finitely many available urgency levels to select from or a continuum of available urgency levels to select from:

1. *Finitely many available urgency levels*: For-loop over all available candidates to find the optimal action and optimal expected cumulative cost. This process may be improved with simple heuristics when there are a significantly large number of urgency levels to select from.

2. *Infinitely many available urgency levels*: The minimum corresponds to the action that makes the derivative of the cost equal zero. An analytical formula for the optimal action is used when it is available (this availability is dependent on the transition cost function). Otherwise, a bisection method is used when the transition cost function is a general polynomial or a rational function, such as the Rational Bezier Function, as described elsewhere herein.

Before processing the next state by Dijkstra's algorithm, information about the optimal action is stored in a two-dimensional multiarray, and information about the optimal cumulative cost at each node is stored in a vector. After Dijkstra's algorithm has solved the routing problem,

the two-dimensional multiarray containing information about the optimal action and the vector containing information about the optimal cumulative cost at each state are written directly to .txt files for processing.

Once the computation in C++ is completed, the Python visualization code processes the output .txt files. The Python visualization code reads the .txt files into lists and two-dimensional Numpy arrays as appropriate, and it uses plotting functions from NetworkX and Matplotlib to render the STP images shown in certain figures herein. The Python visualization code can also save the images automatically after generation in a user-selected format (e.g., a .png format, .jpg format, etc.).

It should be noted that the code arrangements described above are presented by way of illustrative example only, and should not be considered limiting in any way. Numerous alternative implementations are possible. For example, in embodiments that do not generate a visualization, the Python visualization code may be eliminated or replaced with other types of code performing other types of functionality relating to autonomous vehicle routing.

Additionally or alternatively, in other embodiments, the code can be adapted in a straightforward manner in order to, for example, implement Dial's algorithm, in addition to or in place of Dijkstra's algorithm, to solve the routing problem to determine the STP. It may be beneficial to implement Dial's algorithm in some embodiments, especially since Dial's algorithm is parallelizable and will likely result in additional performance gains on large road-networks. Further functionality could be added to generate lane-level road-network representations directly from available map data retrieved from a database, and code may be written to incorporate readings from real-time traffic data available on external databases or from sensors attached to the vehicle itself.

As indicated previously, additional details regarding the above-noted illustrative embodiments can be found elsewhere herein. The information processing systems, processing platforms, processing devices, networks, vehicle controllers, routing optimization systems and other systems and components disclosed herein in conjunction with FIGS. 1 through 4 are

illustratively configured in some embodiments to implement the example algorithms and other techniques that are described in detail below.

Additional illustrative embodiments will now be described with reference to FIGS. 5 through 8. Some of these embodiments are then further described with additional analysis in FIGS. 9 through 19.

Let $X = \{x_1, \ldots, x_n, t\}$ be the set of all states (nodes), and suppose t is the target (destination). Let $A = \bigcup_{x \in X} A(x)$ be the set of all controls (actions), where $A(x)$ is the set of controls available to the decision maker at state $x \in X$. Starting at the initial stage $k = 0$ from state $y_0 = x_i$, a decision maker chooses a control value $a \in A(x_i)$ which specifies a probability distribution $\boldsymbol{p}$ over possible successor nodes, where the probability of transitioning to node $x_j \in X$ from x_i is given by $p_{ij}(a)$. After selecting a control, the decision maker incurs a transition cost $C(x_i, a)$, and the next stage of the process begins.

Throughout the process, the decision maker builds a deterministic control policy which is a sequence of control mappings of the form $\mu: X \to A$ where $\mu(x_i) \in A(x_i), \forall x_i \in X$. The mapping $\mu_k(y_k)$ provides a rule for how the decision maker selects control a at stage k from state y_k. If the policy is independent of the initial state y_0 and does not change after each stage (i.e. $\mu_k = \mu, \forall k$), we say that the policy $\pi = \{\mu, \mu, \ldots\}$ is stationary.

In some embodiments, the following assumptions are made about the transition cost function and probabilities:

1. The target is absorbing (i.e. $p_{tt}(a) = 1, C(t, a) = 0, \forall a$).
2. $A(x_i)$ is compact $\forall i$.
3. $\forall i, C(x_i, a)$ is lower semicontinuous over A_i.
4. $\forall i, p_{ij}(a)$ are continuous over A_i.
5. There exists at least one stationary policy which results in arriving at the target with probability 1 from any starting state. We refer to a policy of this type as proper.

6. Polices which do not lead to arriving at t imply that at least one node will have infinite cost. The decision maker incurs a positive transition cost at every stage until they reach t, and the total expected cost accumulated starting from $y_0 = x_i$ under policy π is given by

$$\mathcal{J}(x_i, \pi) = \mathbb{E}\left(\sum_{k=0}^{\infty} C\big(y_k, \mu(y_k)\big)\right) \tag{1}$$

The decision maker's overall goal is to determine the optimal policy π^* which minimizes (1). The minimum expected cumulative cost starting from x_i over all policies is defined as the value function,

$$U(x_i) = \inf_{\pi}\{\mathcal{J}(x_i, \pi)\} \tag{2}$$

Based on the assumptions listed above, the infimum in (2) is achieved by a stationary optimal policy, and it becomes a minimum over the set of available controls. For notational ease, denote $U(x_i)$ as U_i and $A(x_i)$ as A_i. Assuming that the value function U_i is finite, the dynamic programming principle applies, and it satisfies

$$U_i = \min_{a \in A_i}\left\{C(x_i, a) + \sum_{k=1}^{n+1} p_{ik}(a) U_k\right\} \tag{3}$$

at all $x_i \in X \setminus \{t\}$ and

$$U_t = 0$$

at t.

In the following description, we impose the additional assumption that $C(x, a) > 0, \forall x \in X \setminus \{t\}$. This assumption is important for the applicability of label-setting methods to solve SSPs which will be discussed in detail below. Furthermore, since each action $a \in A(x_i)$ specifies a transition probability distribution $\boldsymbol{p}$ over possible successor nodes from node x_i, it is often illuminating to express $C(x_i, a)$ in terms of $\boldsymbol{p}$ as $C(\boldsymbol{p})$. We will utilize both notations, and we will rely heavily on the $C(\boldsymbol{p})$ viewpoint in description of some embodiments.

While it is standard to consider deterministic control policies, it is also possible to allow the decision maker to select randomized or mixed actions under a randomized control policy. In this setting, the control mapping is of the form $\tilde{\mu}: X \to \mathcal{P}(A)$ where $\mathcal{P}(A)$ contains all probability distributions over the set of available controls, A. At stage k and state $y_k = x_i$ the randomized control mapping $\tilde{\mu}_k(y_k)$ specifies a probability distribution $\lambda(a) \in \mathcal{P}(A_i)$ and control value of the form

$$\tilde{a} = \sum_{j=1}^{|A_i|} \lambda_j a_j$$

for all $a_j \in A_i$. It is also clear that the deterministic control policy is embedded in this framework. At each stage, it is possible for the decision maker at x_i to select a probability distribution $\lambda(a) = \boldsymbol{e}_k$ where $\boldsymbol{e}_k$ is a standard basis vector in $\mathbb{R}^{|A_i|}$.

We now introduce two definitions which will be useful in classifying which available actions are candidates for optimality. Throughout the proceeding discussion, consider an SSP where node $x \in X$ has m possible successor nodes, and let Ξ_m be the $(m-1)$-dimensional probability simplex.

Definition A. Dominated Point. Let $\mathcal{D}$ denote the set of all tuples $(\boldsymbol{p}, C(\boldsymbol{p}))$ associated with an available action at x. Let $\boldsymbol{p} \in \Xi_m$ be a probability distribution over the states to which it is possible to transition to from node x. We say that the action is dominated if there exists a collection of m unique points $\psi = \left\{\left(\boldsymbol{p}_j, C(\boldsymbol{p}_j)\right)\right\}_{j=1,\dots,m}$ which all belong to $\mathcal{D}$ and if there also exists a probability distribution λ over that set such that $\boldsymbol{p} = \sum_{j=1}^{m} \lambda_j \boldsymbol{p}_j$ holds and

$$C(\boldsymbol{p}) > \sum_{j=1}^{m} \lambda_j C(\boldsymbol{p}_j)$$

holds.

In other words, $a \in A$ is dominated when there exists a mixed action under a randomized control policy which yields a lower transition cost than the cost incurred by selecting the pure action.

Definition B. Convexified Cost. Suppose that $U_i < \infty$ for all $x_i \in X$. Let Ψ denote the set of all ψ for which there exists a probability distribution $\lambda \in \Xi_m$ such that

$$\boldsymbol{p} = \sum_{j=1}^{m} \lambda_j \boldsymbol{p}_j$$

holds. The convexified cost at $(\boldsymbol{p}, C(\boldsymbol{p}))$ is defined as

$$\check{C}(\boldsymbol{p}) = \min_{\pi \in \Pi} \left\{ \sum_{j=1}^{m} \lambda_j C(\overrightarrow{p_j}) \right\}$$

The convexification process becomes instrumental when the original cost function $C(\boldsymbol{p})$ is not convex on the probability simplex. In SSPs with non-convex cost functions, the convexification removes the non-convex portions of $C(\boldsymbol{p})$ and creates $\check{C}(\boldsymbol{p})$ by connecting the remaining convex portions of $C(\boldsymbol{p})$ together. $\check{C}(\boldsymbol{p})$ can then be used in place of $C(\boldsymbol{p})$ since the optimum in this cost-minimization problem will never occur on a non-convex portion of $C(\boldsymbol{p})$, and thus the value function U remains unchanged after convexification.

We now shift our attention to defining a particular subclass of SSPs. OSSPs as disclosed in some embodiments herein are SSPs in which every stochastically realizable path is also realizable using only deterministic controls.

Definition C. OSSP. Let $e^{ij} \in A_i$ denote an action which results in a deterministic transition from x_i to $x_j \in X$. We will refer to an SSP as an OSSP if

$$\exists a \in A_i \text{ such that } p_{ij}(a) > 0 \implies$$
$$\exists e^{ij} \in A_i \text{ such that } p_{ij}(e^{ij}) = 1$$

holds for all i and j.

It should be noted that the optimum in an OSSP in illustrative embodiments will never be attained at a dominated point. This may be verified for illustrative embodiments using Definitions

A and B in a straightforward convexity argument. In light of this and the convexification assumption, there is no advantage to using a randomized control policy over a deterministic control policy. As such, we focus solely on stationary deterministic control policies throughout the remainder of this discussion.

OSSPs are directly related to another important class of SSPs known as Multimode Stochastic Shortest Path (MSSP) Problems. In general MSSPs, each node $x \in X = \{x_1, \ldots, x_n, t\}$ has a set of modes $\mathcal{M}(x)$. Each mode $m \in \mathcal{M}(x)$ is characterized by a collection of possible successor nodes $m = \{y_1^m, \ldots, y_{|m|}^m\} \subset X$, and a transition cost function $C(x, a)$. Transitions to successor nodes in mode m occur according to a probability distribution $\boldsymbol{p} \in \Xi_{|m|}$, where $\Xi_{|m|}$ is the $(|m| - 1)$-dimensional probability simplex. At each stage, the decision maker chooses a control $a = (m, \boldsymbol{p}) \in A(x)$, and we assume that for every $\boldsymbol{p} \in \Xi_{|m|}$ and every $m \in \mathcal{M}(x)$, there exists an available control belonging to $A(x)$. The decision maker selects a control at each step so as to satisfy the equations

$$V^m(x) = \min_{\boldsymbol{p} \in \Xi_{|m|}} \left\{ C(x, (m, \boldsymbol{p})) + \sum_{i=1}^{|m|} p_i U(y_i^m) \right\}$$

and

$$U(x) = \min_{m \in \mathcal{M}(x)} \{V^m(x)\}$$

Thus, selecting an optimal control at each stage in an MSSP amounts to identifying both the optimal mode and the optimal transition probability distribution over the possible successor nodes in the corresponding mode. From the OSSP viewpoint, imposing a randomized control policy implies that at every stage, the decision maker now has infinitely many available randomized actions to select from.

It should also be noted that OSSPs in illustrative embodiments are equivalent to MSSPs under a randomized control policy.

Numerical methods for solving SSPs have been extensively studied over the past several decades. We begin our discussion of the applicability of label-setting methods to OSSPs with a review of value iterations and label-setting methods in the deterministic setting.

Value Iterations (VI) are a standard iterative method used to solve a system of coupled nonlinear equations for the value function in an SSP. Consider the SSP defined above, and assume that the value function at every node is finite. Let $U \in \mathbb{R}^{n+1}$ be a vector containing the value function values at each node. Using (3), define the operator T such that for any $W \in \mathbb{R}^{n+1}$, the i^{th} component of TW is given by

$$(TW)_i = \min_{a \in A_i} \left\{ C(x_i, a) + \sum_{k=1}^{n+1} p_k(x_i, a) + W_k \right\}$$

Assuming that all stationary policies are proper, T is a contraction mapping, and U is its unique fixed point. For any initial guess $W^0 \in \mathbb{R}^{n+1}$, the iteration

$$W^{k+1} = TW^k$$

converges to U as $k \to \infty$.

Whether or not the value iteration process described above converges in finitely many iterations hinges on properties of the SSP's transition probability graph. Under a stationary policy μ, we construct a directed graph $\mathcal{G}$ with nodes $X = \{x_1, \ldots, x_n, t\}$, and we draw an edge from x_i to x_j if $p_{ij}(\mu(x_i)) > 0$. If $\mathcal{G}$ under the optimal policy μ^* is acyclic, then the value iterations will converge to U after at most n iterations.

To motivate the discussion of label-setting methods in the SSP setting, we briefly review label-setting methods for deterministic shortest path problems. Suppose $N(x_i)$ is the set containing all $x \in X$ such that x_i is connected to x. By the dynamic programming principle, solving a deterministic SP amounts to determining

$$U_i = \min_{x_j \in N(x_i)} \{ C(x_i, x_j) + U_j \}$$

for all $x_i \in X \setminus \{t\}$ and $U(t) = 0$. If $\gamma << |X|$ is an upper bound on the node indegrees in the network, and the transition cost is always nonnegative, label-setting methods determine a

special ordering in which to compute $U(x)$ at each $x \in X$ such that U_i is never recomputed more than γ times. In this case, label-setting methods are an attractive alternative to the standard value iterations described above. At the start, label-setting methods mark the value at the target t as permanent and assign temporary value labels to each $x \in X \setminus \{t\}$. After computing value function candidate values for some $x \in X$ at each stage of the process, these methods are able to assign permanent value labels to one or more $x \in X$, and this process continues until all labels are designated permanent. As a result, although the node labels are changing through different stages of the algorithm, we refer to these label-setting methods as non-iterative.

Perhaps the most widely-known label-setting method is Dijkstra's algorithm, which computes the value function correctly at each node provided that the causality condition

$$U_i \geq U_j$$

holds when x_i depends on x_j. In other words, the value function must be decreasing along an optimal path to t from x_i that passes through x_j. The algorithm begins by setting the value at t equal to zero and initializing the value function at all other nodes to infinity. The target is assigned to a set of nodes with known value function values K, while all remaining $x_i \in X \setminus \{t\}$ comprise a set of nodes with "tentatively known" value function values, L. The first stage of Dijkstra's algorithm computes

$$U_i = C(x_i, t) + U_t$$

at all $x_i \in N(t)$, where $N(t)$ is the set of all nodes directly connected to the target. The Algorithm updates the value function value at the corresponding node if it is smaller than the previously assigned value. Let x^* denote the node belonging to L with the smallest updated value at the end of the first stage. This node moves from L to K, and the next stage proceeds by recomputing the value function at each $x_i \in N(x_*)$ according to

$$U_i = C(x_i, x^*) + U_*.$$

Once again, the node with the smallest updated value at the end of the stage moves to K, and this recompute and reassign process continues until all nodes have moved from L to K.

Convergence to the true value function is achieved once L is empty. The complexity of $O(n\log n)$ is achieved by utilizing a heap data structure to store the nodes in L.

When $C(x_i, x_j) > \delta$ for a given $\delta > 0$, it is possible to use Dial's algorithm instead to solve the deterministic SP. Dial's algorithm operates similarly to Dijkstra's algorithm, but instead of using a heap sort date structure and declaring the smallest of the tentatively known nodes as known in every stage, Dial's algorithm groups the nodes into buckets based on their tentative label values. Each bucket has a fixed width $\delta > 0$, and at each stage, the algorithm scans the buckets in increasing order and stops at the first non-empty bucket. All of the labels in this bucket are accepted as permanent, and those nodes are removed from the bucket. Then, the algorithm only recomputes the tentatively known values at the nodes which are connected to those from the current bucket which were just accepted as permanent. These nodes with newly updated tentatively known values are assigned to buckets as appropriate, and the process continues at the next non-empty bucket.

Whether or not a Dijkstra-like algorithm is applicable to solve an OSSP relies on the existence of a consistently improving optimal stationary policy. A consistently improving optimal policy μ^* can be defined as a stationary optimal policy such that for all i,

$$p_{ij}(\mu^*(x_i)) > 0 \Rightarrow U_i > U_j$$

That is, a consistently improving optimal policy μ^* is a stationary policy such that for action a and for all i,

$$p_i(x, a) > 0 \Rightarrow U_i > U_j.$$

Causal OSSPs are separated into two categories - explicitly causal and monotonically causal. An OSSP is illustratively considered explicitly causal if for every x_i and x_j in $X \setminus \{t\}$, if there exists a stationary policy μ such that there is a path from x_i to x_j in the transition probability graph under policy μ, then there cannot be a stationary policy π which allows the existence of a path from x_j to x_i in the transition probability graph under policy π. Not only does the causality condition hold, but Dijkstra's algorithm is unnecessary. In this case, the transition probabilities

and actions provide a direct way to reorder the nodes to compute the value function in at most n stages. On the other hand, we refer to OSSPs which are causal but not explicitly causal as monotonically causal.

The applicability of Dial's algorithm to solve OSSPs is similar in that it relies on the existence of a consistently δ-improving optimal policy. That is, for a given $\delta > 0$, the optimal policy μ^* is a stationary optimal policy such that for all i,

$$p_{ij}(\mu^*(\boldsymbol{x_i})) > \delta \Longrightarrow U_i > U_j + \delta$$

or for all i,

$$p_i(x, a) > \delta \Longrightarrow U_i > U_j$$

In this setting, we refer to OSSPs which are causal but not explicitly causal as monotonically δ-causal. Determining if an OSSP is monotonically (or monotonically δ) causal is more challenging since the existence of μ^* is not generally known prior to solving the OSSP. Instead, applicability of label-setting methods to monotonically causal problems may be guaranteed through conditions on the transition cost function and transition probabilities.

Consider an OSSP as defined in Definition C with state space $X = \{x_1, \ldots, x_n, t\}$, available controls A, and transition cost function $C(x, a) > 0, \forall x \in X$ with $C(t, a) = 0$. Suppose that node $x \in X \setminus \{t\}$ has m possible successor nodes, and let $\boldsymbol{p} \in \Xi_m$ denote a transition probability distribution over possible successor nodes.

Theorem A. Sufficient Causality Condition in OSSP. Suppose a^* is the optimal action at node x, and $\boldsymbol{p} \in \Xi_m$ is the associated optimal transition probability distribution. Let $\boldsymbol{e}_j$ be a standard basis vector in $\mathbb{R}^m$ and $c_j = C(\boldsymbol{e}_j)$. If for a given $\delta \geq 0$ and every $i = 1, \ldots, m, p_i > 0$ implies that the cost $C(\boldsymbol{p})$ satisfies

$$C(\boldsymbol{p}) \geq \sum_{j=1, j\neq i}^{m} c_j p_j + p_i \delta$$

then monotone δ-causality holds, and a label-setting method such as Dijkstra's algorithm or Dial's algorithm is applicable to solve the OSSP.

Let $W_j \in \mathbb{R}^m, j = 1, \ldots m$ be the value function at each of the possible successor nodes, and assume that $W_j < \infty$ for all j. Recall that the value function at x can be written as $U = C(\boldsymbol{p}) + \sum_{j=1}^{m} p_j W_j$. For notational ease, let $c^* = C(\boldsymbol{p})$. Solving for c^* in the definition of U and assuming that c^* satisfies the bound given in the Theorem A statement, we bound U from below as

$$U \geq \sum_{j=1, j\neq i}^{m} p_j(c_j + W_j) + p_i(W_i + \delta)$$

Since a^* optimal, we must have that $W_j + c_j > U, \forall j$. So, $\sum_{j=1, j\neq i}^{m} p_j(W_j + c_j) > \sum_{j=1, j\neq i}^{m} p_j U$. Combining this with the lower bound on U yields

$$U - \sum_{j=1, j\neq i}^{m} p_j U > p_i(W_i + \delta)$$

And since $1 - \sum_{j=1, j\neq i}^{m} p_j = p_i$,

$$p_i U > p_i(W_i + \delta)$$

Thus,

$$U > W_i + \delta$$

for any i. Causality clearly holds, and a label-setting method to solve the OSSP is applicable.

It has been shown that transition costs which are continuously differentiable, homogeneous of degree d, and only have partial derivatives which satisfy $\frac{\partial C}{\partial p_i} > (d-1)C(p) + \delta$ for all $i = 1, \ldots, m$ and all $\boldsymbol{p} \in \Xi_m$ guarantee absolute δ causality of the mode in an MSSP. While these conditions may seem rather restrictive, cost functions of this type appear often in many applications. Recall that $C(\boldsymbol{p})$ is homogeneous of degree d if $C(\boldsymbol{p}) = |a|^d C(\boldsymbol{p})$ holds for all $a \in \mathbb{R}, \boldsymbol{p} \in \Xi_m$. Further, when $C(\boldsymbol{p})$ is smooth, Euler's Homogeneous Function Theorem applies, and the cost function can be written as

$$C(\boldsymbol{p}) = \frac{1}{d} \sum_{i=1}^{m} p_i \frac{\partial C(\boldsymbol{p})}{\partial p_i}$$

We extend these results to show that transition cost functions in an OSSP which satisfy the criteria mentioned above also satisfy the causality condition in Theorem A when the number of possible successor nodes is two.

Theorem B. Suppose $m = 2$, and consider a transition cost function $C(\boldsymbol{p})$ which is convex, monotonically increasing, continuously differentiable on the probability simplex, homogeneous of degree d, and satisfies

$$\frac{\partial C(\boldsymbol{p})}{\partial p_j} - (d-1)C(\boldsymbol{p}) > \delta \geq 0$$

for all $\boldsymbol{p} \in \Xi_2$ and for $j = 1,2$. Let $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ be standard basis vectors in $\mathbb{R}^2$, and let $c_j = C(\boldsymbol{e}_j)$ for $j = 1,2$. Then, $p_i > 0$ implies that

$$C(\boldsymbol{p}) \geq \sum_{j=1, j\neq i}^{2} c_j p_j + p_i \delta$$

holds for $i = 1,2$ and for all $\boldsymbol{p} \in \Xi_2$.

We proceed by showing the contrapositive. Suppose that the condition in Theorem A is violated at $\boldsymbol{p} \in \Xi_2$. By Euler's Homogeneous Function Theorem, we have $C(\boldsymbol{p}) = \frac{1}{d}\sum_{k=1}^{2} p_k \frac{\partial C(\boldsymbol{p})}{\partial p_k}$, and

$$\frac{1}{d}\left(p_1 \frac{\partial C(\boldsymbol{p})}{\partial p_1} + p_2 \frac{\partial C(\boldsymbol{p})}{\partial p_2}\right) < c_1 p_1 + p_2 \delta = \frac{1}{d}\frac{\partial C(e_1)}{\partial p_1} + p_2 \delta \tag{4}$$

or

$$\frac{1}{d}\left(p_1 \frac{\partial C(\boldsymbol{p})}{\partial p_1} + p_2 \frac{\partial C(\boldsymbol{p})}{\partial p_2}\right) < c_2 p_2 + p_1 \delta = \frac{1}{d}\frac{\partial C(e_2)}{\partial p_2} + p_1 \delta \tag{5}$$

must hold. Since $p_1 + p_2 = 1$, let $p = p_2$ and $p_1 = 1 - p$. Restricting $C(p_1, p_2)$ to Ξ_2 yields $K(p) = C(1-p, p)$, where

$$K'(p) = -C_{p_1}(1-p, p) + C_{p_2}(1-p, p)$$

Convex functions are convex in all directions, so $K(p)$ must also be convex, and $K'(p)$ is monotonically increasing over the interval [0,1]. Assuming (5) holds, geometrically speaking, this implies that

$$K'(1) > K(1) - \delta \tag{6}$$

as the slope of the line tangent to $K(p)$ is greater than the slope of the restriction line imposed by the causality condition when $i = 1$. Inequality (6) is equivalent to

$$-\frac{\partial C(0,1)}{\partial p_1} + \frac{\partial C(0,1)}{\partial p_2} > \frac{1}{d}\frac{\partial C(0,1)}{\partial p_2} - \delta$$

But, since $C(\boldsymbol{p})$ homogenous of degree d, by definition,

$$-\frac{\partial C(0,1)}{\partial p_1} + dC(0,1) > C(0,1) - \delta$$

Rearranging yields

$$\frac{\partial C}{\partial p_1}(0,1) < (d-1)C(0,1) + \delta$$

Continuity of $\frac{\partial C(\boldsymbol{p})}{\partial p_2}$ for all $\boldsymbol{p}$ implies there exists a $\tilde{p} \in \Xi_2$ such that $p_2 < 1$ and $\frac{\partial C}{\partial p_1}(\tilde{p}) < (d-1)\,C(\tilde{p}) + \delta$. This directly violates the requirement that all partial derivatives of $C(\boldsymbol{p})$ must be $> \delta + (d-1)C(p)$ within Ξ_2.

Now, suppose instead that the monotone causality condition when $i = 2$ fails so that (4) holds. Following a geometric argument similar to the one taken above, in this case, the slope of the tangent line when $p = 0$ is less than the slope of the restriction line imposed by the causality condition when $i = 2$. Thus,

$$K'(0) < \delta - K(0)$$

which is equivalent to

$$-dC(1,0) + \frac{\partial C(1,0)}{\partial p_2} < \delta - C(1,0)$$

Since, $\frac{\partial C(\boldsymbol{p})}{\partial p_1}$ continuous for all $\boldsymbol{p}$, there exists $\tilde{p} \in \Xi_2$ such that $p_2 > 0$ and $\frac{\partial C(\tilde{p})}{\partial p_2} < (d-1)C(\tilde{p}) + \delta$. Once again, this contradicts the assumption that all partial derivatives of $C(\boldsymbol{p})$ are $> \delta + (d-1)C(p)$ in Ξ_2.

All in all only transition cost functions which satisfy the criteria set forth in Theorem B are also guaranteed to satisfy the causality condition of Theorem A.

Corollary A. Maximum Allowable δ in Dial's algorithm for $m = 2$: Let $m = 2$ and suppose that $C(\boldsymbol{p})$ satisfies the assumptions of Theorem B. Then, the maximum allowable $\delta_\star$ for monotone δ-causality is

$$\delta_\star = (1-d)C(0,1) + \frac{\partial C}{\partial p_1}(0,1)$$

The proof of Corollary A is straightforward and follows directly from the geometric argument presented in the proof of Theorem B.

Surprisingly, Theorem B does not generally hold in cases where $m > 2$. As a counterexample, suppose $m = 3$ and consider a cost function of the form

$$C(p_1, p_2, p_3) = A(p_1 + p_2 + p_3) + \sqrt{A^2 p_1^2 + B^2 p_2^2 + C^2 p_3^2} \tag{7}$$

It is clear that for any choice of $A, B, C \in \mathbb{R}_+$, this cost function satisfies the conditions set forth above. Take $A = B = C$ and $p_2 = p_3$. This restricts (7) to lie along the median of the triangle with vertices $(1,0,0)$, $(0,1,0)$, and $(0,0,1)$, and it becomes

$$C(p_1, p_2, p_3) = A(p_1 + 2p_2) + \sqrt{A^2(p_1^2 + 2p_2^2)}$$

To satisfy the sufficient monotone causality condition when $i = 1$, we require

$$A(p_1 + 2p_2) + \sqrt{A^2(p_1^2 + 2p_2^2)} \geq 2Ap_2 + 2Ap_2$$

Rearranging and squaring both sides yields

$$2A^2 p_1 p_2 \geq A^2 p_2^2$$

Noting that $p_2 = \frac{1-p_1}{2}$ we are left with

$$2p_1\left(\frac{1-p_1}{2}\right) \geq \left(\frac{1-p_1}{2}\right)^2$$

Setting the two quadratics equal to each other and solving for their intersection point reveals that the $i = 1$ monotone causality condition is violated at all $p_1 \geq 0.2$.

As indicated previously, OSSPs are utilized in vehicle routing problems in illustrative embodiments herein. As autonomous vehicles start to integrate into everyday traffic, routing frameworks which can account for the uncertainty associated with attempted lane-changes, merges, and highway exits are crucial for making these vehicles safer and more reliable. In general, decision-making during the driving process may be broken down into three categories: i) strategical (e.g., determining the turn-by-turn directions to arrive at the destination), ii) tactical (e.g., determining lane-changes en-route to the destination), and iii) operational (e.g., determining the exact trajectory and physically driving the vehicle). The autonomous vehicle operating system in some embodiments illustratively comprises multiple interconnected modules responsible for determining the Strategic Plan (SP), deciding upon a Tactical Plan (TP) which encompasses planned lane-changes and merges, and synthesizing this information with the onboard Operational Control (OC) algorithms which are responsible for computing a smooth trajectory for the vehicle to follow and executing the driving process along this trajectory.

Changing traffic conditions, weather, and unanticipated road closures render the outcome of any tactical decisions such as lane-changes inherently uncertain, and to ensure that the autonomous vehicle operates safely and reliably, the algorithm responsible for determining the TP must take this uncertainty into account when computing the lane-change policy. As a result, the routing module must also be able to recompute the TP quickly and efficiently so that the OC algorithms may adjust the planned trajectory accordingly. Thus, the ability to utilize fast computational methods to determine the TP is paramount to achieving high performance metrics and a better driving experience overall.

At a more nuanced level, the probability that a lane-change attempt is successful is related to how urgently the vehicle attempts to change lanes. As the lane-change urgency increases, the

controller becomes more willing to alter the vehicle's velocity in order to ensure that the lane-change maneuver is successful. To this end, the available actions may be instead interpreted as the driver selecting a lane-change action in accordance with their urgency to change lanes from state x. The stay-in-lane action corresponds to no urgency and a lane-change success probability $p = 0$. The forced lane-change action corresponds to full urgency and lane-change success probability $p = 1$. Available actions with stochastic outcomes correspond to intermediate urgency levels and success probability $p \in (0,1)$.

In reality, drivers naturally increase or decrease their lane-change urgency in response to changing traffic conditions or nearby infrastructure. As such, the lane-change urgency options available at each state exist on a continuous spectrum, and allowing the routing system to select from a wider range of lane-change actions may improve the autonomous vehicle's performance metrics overall. In an example framework disclosed herein, there are infinitely many available actions for the routing system to select from at state x, and each of the actions specify a lane-change success probability which reflects how urgently the vehicle should attempt to change lanes from its current position. To capture this continuous spectrum, it is illuminating to consider transition cost functions which are continuous, convex, and monotonically increasing in terms of the lane-change success probability p. While there are countless transition cost functions which meet the conditions mentioned above and satisfy the monotone causality condition stated in Theorem A, a simple, natural choice is a quadratic function

$$C(p) = Ap^2 + Bp + D$$

where $A, B, D \in \mathbb{R}_{0,+}$ are constants which may reflect current traffic conditions near the vehicle's current state or the driver's personal preferences. These constants can be selected to ensure that $C(p)$ satisfies Theorem A.

Approaches in illustrative embodiments can be extended to accommodate any finite number of stochastic actions which will capture a wider range of intermediate lane-change urgency levels. Suppose there are $k + 1$ available actions $a_i, i = 0, \ldots, k$ in which the attempted switch succeeds with fixed probability $p_i, i = 0, \ldots, k$ where $p_0 = 0, p_k = 1$, and $p_i \in (0,1)$ for $i =$

$1, \ldots, k-1$ with $p_1 < \cdots < p_{k-1}$. Each increasing urgency level when $i > 0$ has an associated additional cost penalty $K_i \in \mathbb{R}_+$, that the driver incurs upon a successful lane-change at that level. In some embodiments, a directly augmented cost formulation assumes that before arriving at the successor state, the vehicle attempts all intermediate urgency levels, up to and including the level they select. In other words, selecting action a_i, implies that the driver attempts to change lanes at all urgency levels 1 through $i-1$, and then attempts to change lanes at urgency level i if the previous change attempts failed. This corresponds to an escalating cost structure of the form

$$C(p_0) = c(x), \text{ and } C(p_i) = C(p_{i-1}) + (1 - p_{i-1})K_i, \; i = 1, \ldots, k.$$

With regards to Theorem A, this OSSP is monotonically δ-causal if the stay in lane cost $c(x)$ satisfies

$$c(x) \geq p_i C(p_k) + (1 - p_i)\delta - \sum_{j=1}^{i} (1 - p_{j-1})K_j \text{ for } i = 1, \ldots, k. \tag{8}$$

It is also possible to utilize data points from the discrete transition cost structure to construct a continuous transition cost function which passes through those desired points. As the vehicle's hardware scans the surrounding environment and gathers information about the local traffic conditions, the routing module may utilize this data to determine an approximate lane-change success probability $p_\star$ from state x. In turn, the routing module can compute a value for the stay in lane cost K_0, the tentative lane-change cost $K_\star$ using probability $p_\star$, and the forced lane-change cost K_1 from x. To construct the continuous function passing through the three points $(0, K_0), (p_\star, K_\star)$, and $(1, K_1)$, the system then builds a quadratic Rational Bezier Curve. These curves are characterized by three control points which are responsible for the curve's shape as well as defining the convex hull in which the curve is restricted to lie. Using the restriction lines imposed by the monotone causality condition presented in Theorem A, we select the control points $(0, K_0), \left(\frac{K_0 - \delta}{K_1 - \delta}, K_0\right)$, and $(1, K_1)$, and the resulting curve is given parametrically as

$$P(t) = \frac{2w_1(1-t)t\left(\frac{K_0 - \delta}{K_1 - \delta}\right) + w_2 t^2}{w_0(1-t)^2 + 2w_1(1-t)t + w_2 t^2} \tag{9}$$

$$C(t) = \frac{(1-t)^2 w_0 K_0 + 2w_1(1-t)tK_0 + w_2 t^2 K_1}{w_0(1-t)^2 + 2w_1(1-t)t + w_2 t^2} \tag{10}$$

where t is a parameter between 0 and 1 that "sweeps out" the curve into its full, intended shape, and w_i are positive numbers representing the influence that the associated control point has in determining values along the transition cost curve. Since Bezier Curves are guaranteed to pass through their first and last control points, we take $w_0 = w_2 = 1$ and construct a system of nonlinear equations for t and w_1 by setting $P(t) = p_\star$ and $C(t) = K_\star$. This system is quickly solved numerically to determine the w_1 value which guarantees that the curve passes through $(p_\star, K_\star)$, and the w_1 value is then set accordingly in (9) and (10) to compute the transition cost from x in the OSSP.

We now employ a Dijkstra-like algorithm to solve OSSPs arising in autonomous vehicle routing problems to determine the vehicle's optimal lane-change strategy for several different road networks and transition cost functions. In all examples, the distance between successor nodes in the same lane is ℓ, and the direction of traffic flow is indicated by arrows. Deterministic actions (e.g. continuing in the same lane or forcefully changing lanes) are indicated with solid transition arrows while stochastic actions are indicated with dashed transition arrows unless noted otherwise.

Consider as a first example a three lane highway, with four available actions. As a finite extension of an escalating cost formulation, suppose the routing system needs to determine an STP for a vehicle traveling to a destination in the left lane, and there are four available lane-change actions to select from. Each of the tentative lane-change attempts correspond to an urgency level reflective of how willing the vehicle is to adjust its velocity to successfully complete the lane change. As such, each urgency level is associated with a lane-change success probability $p \in \{p_0, p_1, p_2, p_3\}$, where $p_0 = 0$ (no lane change urgency), $p_3 = 1$ (complete lane change urgency) and $p_1 < p_2$. Let $c(x) = 1 + \sigma c_\ell$ be the cost of continuing in the current lane where $c_\ell \in \mathbb{R}_+$is a constant penalty for driving in the left lane, and $\sigma = 0,1,2$ in the right, middle, and left lanes respectively. The transition cost is given by

$$C(p_0) = c(x), \text{ and } C(p_i) = C(p_{i-1}) + (1 - p_{i-1})K_i,\ i = 1,2,3$$

where the constants K_i are the additional costs associated with attempting a lane-change at urgency level i. The vehicle must also account for an on-ramp merging into the right lane 1 km from the target. The vehicle is also subject to an additional penalty $[c_m]_{x_\#} \in \mathbb{R}_+$for driving through the merge location. Using the causality condition given in (8), a Dijkstra-like algorithm to solve the OSSP applies provided that $c(x) \geq p_i C(p_3) - \sum_{j=1}^{i}(1 - p_{j-1})K_j$ for $i = 1,2,3$.

To understand the effect that different available urgency level options has on the resulting STP, we set $\alpha = 0.01, \ell = 10$ m, $c_\ell = 0.25, p_1 = 1 - e^{-\alpha\ell}$, and $K_3 = 100$, but we vary p_2 and K_2. Since the transition cost at any level $i > 0$ in the escalating action framework is only determined by $c(x)$, the p_i's, and the K_i's, the cost associated with the stay in lane action and the less-urgent tentative lane-change action will remain fixed throughout, but the cost associated with the more-urgent tentative lane-change action and the forced lane-change action will change. Results for different values of (p_2, K_2) and c_m are shown in FIG. 5, which includes four distinct parts denoted (A), (B), (C) and (D).

FIG. 5 more particularly illustrates lane-change planning on a three-lane highway 1 km away from a destination in the left lane with four available actions and varying degrees of congestion at the merge location when (A) $(p_2, K_2) = (0.15,5), c_m = 50$, (B) $(p_2, K_2) = (0.5,40), c_m = 50$, (C) $(p_2, K_2) = (0.15,5), c_m = 180$, and (D) $(p_2, K_2) = (0.5,40), c_m = 180$. The resulting STP is affected by increases in congestion, as well as the spectrum of available urgency levels.

Both the anticipated congestion at the merge location and the range of urgency levels available to the routing system has a significant effect on the resulting STP. Although both STPs presented in FIG. 5 show that the vehicle should forcefully change lanes from the right lane to the middle lane in order to avoid driving through the merge congestion, there is a clear difference in urgency when attempting to change from the left lane to the middle lane and when attempting to leave the right lane before the merge. In parts (A) and (B) of FIG. 5, a vehicle traveling within this highway stretch is subject to moderate congestion at the merge location $(c_m = 50)$, and both STPs indicate that if the vehicle is traveling in the middle lane within 40 m of the merge location,

it should remain in its current lane. The primary differences between (A) and (B) are associated with the range of urgency levels available for the system to select from. In (B), vehicles in the left lane have the opportunity to attempt a lane change to the middle with much higher urgency than vehicles subject to the STP in (A), and they take advantage of this choice for at least 50 m. Now, when the congestion becomes very heavy and $c_m = 180$, vehicles in the right lane attempt to change to the middle lane with increasing urgency and will forcefully change lanes if necessary. In (D), the urgency level associated with $p_2 = 0.5$ is a bit too high, and the vehicle will only attempt to change lanes with an urgency level corresponding to $p_1 = 0.095$. However, in (C), the resulting STP takes advantage of the slightly more urgent p_2 option and the vehicle attempts a lane change between 20 m and 30 m away from the merge location.

The above example with four available actions illustrates two significant drawbacks of the finite action setting. First, the limited selection of urgency levels may result in sub-optimal driving performance. For example, the resulting STP may suggest a lane-change attempt at an urgency level less than what may be truly optimal if the vehicle were allowed to change lanes with any urgency level (e.g., the attempted lane changes from the left lane to the middle lane between 1020 m and 970 m from the target with urgency level corresponding to $p = 0.15$ in parts (A) and (C) of FIG. 5). This could lead to a slightly higher overall expected cumulative cost compared to settings in which the routing system has a wider range of available urgency levels to select from when determining the STP. Additionally, the escalating actions framework becomes less physically practical as the number of available urgency levels increases. Lane switch maneuvers can take anywhere from 0.5 to 6 + seconds to complete, and a vehicle traveling at typical highway speeds will not have enough time to safely execute multiple attempts before reaching the next road-network location in its current lane.

To remedy this, we now allow the routing module to select actions associated with a continuous spectrum of urgency levels, in an example of a three lane highway with infinitely many available actions. Since the Rational Bezier Curve formulation described previously is a direct extension of the finite action framework to the continuous setting and can also incorporate real

traffic data in constructing the transition cost curve, we illustrate the approach by determining a section of an STP for a vehicle driving westbound on East Wacker Drive between Michigan Avenue and State Street in Downtown Chicago, as illustrated in FIG. 6, which includes three distinct parts (A), (B) and (C).

FIG. 6 more particularly illustrates an example STP for a vehicle traveling on a section of Wacker Drive between Michigan Avenue and State Street in Downtown Chicago, IL for different values of left lane average velocity v_s. The target destination s 108 meters away from the location where traffic from N. Wabash Avenue joins the right lane of Wacker Drive, and the congestion cost c_m at this point is 2.5 seconds. Part (A) shows an Apple Maps map rendering containing the road segment, which is highlighted, and the approximate location of the destination is marked by the large round dot at the end of the highlighted segment. Parts (B) and (C) show resulting STP between 171 m and 90 m away from the destination in the left lane when $v_s = 16$ miles per hour and $v_s = 8$ miles per hour, respectively. In both instances, traffic from N. Wabash Avenue joins the right lane of Wacker Drive via the upward arrow connecting the merge lane segment with the main three-lane road. As the average travel speed in the left lane decreases following a traffic incident (e.g., a traffic accident), the vehicle exhibits more urgency in its attempts to change from the left lane to the middle lane.

The road segment illustrated in part (A) of FIG. 6 is 306 meters long and consists of three lanes of westbound traffic with additional vehicles turning into the right lane from N. Wabash Avenue approximately 108 meters from the intersection between East Wacker Drive and State Street. According to congestion data provided by the City of Chicago, the average speed of vehicles traveling along this road segment is 24mph, and thus it should take a vehicle on average 28.5 seconds to traverse the entire 306 m road segment from end to end with no lane changes, stops, or unusual slowdowns.

We impose a road-network along this segment of Wacker Drive in which successor nodes in the same lane are separated by $\ell = 9$ meters, and a traffic incident in the left lane has caused the average travel speed in the left lane and middle lane to drop to v_s and 20mph respectively.

The vehicle's destination is in the left lane at the intersection of Wacker Drive and State Street, and the routing module aims to minimize the vehicle's expected travel time to destination. As such, the stay in lane cost $c(x)$ in each lane is defined as the time it takes in seconds for the vehicle to travel $9m$ using the average travel speed in that lane. We assume that a voluntary lane change can take 3.4 seconds to fully execute, and forced lane changes may take an additional 10.2 seconds to execute when accounting for any additional slow down/wait time needed to ensure a successful lane change. We also assume that the congestion joining the right lane of Wacker Drive from N. Wabash Avenue will add an additional $c_m = 2.5$ seconds to the expected travel time of any vehicle passing through the merge location.

Assuming that the 3.4-second voluntary lane change succeeds in this area with probability $\hat{p} = 1 - e^{-0.2} \approx 0.393$, the transition costs associated with the three actions described above is given by

$$c(0) = c(x),\ c(\hat{p}) = c(x) + 3.4,\ c(1) = c(\hat{p}) + (1 - \hat{p})10.2.$$

The Rational Bezier Curve passing through those three points is given parametrically as

$$P(t) = \frac{2(1-t)t\tilde{p}_1 w_1 + t^2}{(1-t)^2 + 2(1-t)tw_1 + t^2}$$

$$C(t) = \frac{(1-t)^2\tilde{c}_0 + 2(1-t)t\tilde{c}_1 w_1 + t^2\tilde{c}_2}{(1-t)^2 + 2(1-t)tw_1 + t^2}$$

where $\tilde{p}_1 = \frac{c(x)-\delta}{c_f-\delta}$, $\tilde{c}_0 = \tilde{c}_1 = c(x)$, and $\tilde{c}_2 = c(1)$. To determine w_1 which ensures that the curve passes through the voluntary lane change point $(0.393, c(x) + 3.4)$, we set $P(t) = 0.393$ and $C(t) = c(x) + 3.4$, and we solve the resulting system of nonlinear equations for t and w_1. Since $c(x)$ changes based on the lane-selection, the system is solved for each lane prior to running the Dijkstra-like algorithm to determine the optimal policy, and the resulting parameter values are then used throughout the optimization routine when determining the transition cost. Results for different values of v_s are shown in FIG. 6.

The severity of the traffic slowdown in the left lane following the traffic incident has a significant effect on the resulting STP. When the average travel speed in the left lane is reduced

to 16mph, a vehicle in the left lane between 171 m and 90 m away from the destination chooses to remain in the left lane since the slowdown is not significant enough to make lane change attempts so close to the destination worthwhile. Further, when the vehicle is in the middle lane, the routing system is very reluctant to attempt a lane switch from the middle lane to the left lane to avoid greater slowdowns from the traffic incident. When v_s is reduced to $8mph$, a vehicle in the left lane will now very urgently attempt to join the middle lane, while a vehicle in the middle lane will choose to remain in its current lane. However, when a vehicle in the left lane reaches a state within approximately 117 m from the destination, the lane-change urgency drops dramatically in order to keep the vehicle in the left lane the closer it gets to the destination. In both cases, regardless of v_s, the continuum of available urgency levels results in a more-detailed and finer-tuned STP than if only the actions associated with the three original (p, c) points had been available.

FIG. 7 shows an example with multiple roundabouts and infinitely many available actions, similar to the example previously introduced in conjunction with FIG. 4, and includes distinct parts (A) and (B). Part (A) shows a schematic of a roundabout network with five connected traffic circles. Arrows indicate the direction of traffic flow and dashed edges indicate available transitions between roundabouts. We assume that the vehicle drives on the left-hand side of the road as is customary in the United Kingdom. Part (B) shows a graph version of the roundabout network with all possible transitions from each node shown. The direction of the arrow indicates the transition direction between each node, and the target destination along the southeastern exit is circled in black.

In the previous examples of FIGS. 5 and 6, the traffic only flows from left to right on a straight highway. The OSSPs describing those scenarios are explicitly causal, and a Dijkstra-like algorithm to solve them is unnecessary provided that the equation for U_i is solved after U_j if x_i depends on x_j. OSSPs on road networks with cycles are no longer explicitly causal, and a Dijkstra-like algorithm to solve them is necessary, provided that the usual conditions for absolute causality hold. With reference again to the above-noted Magic Roundabout in Swindon, UK, we consider

a roundabout network comprised of five one-lane roundabouts: three mini-roundabouts connected via an innermost roundabout and an outermost "ring" roundabout. The network also has three two-lane entry/exit streets which allow vehicles to enter and exit the roundabout network from either a mini-roundabout or the outermost ring. Traffic flows clockwise around the outermost and mini roundabouts, while traffic flows counterclockwise around the innermost roundabout. A schematic and graph interpretation of this roundabout network with the entry and exit lanes are shown in respective parts (A) and (B) of FIG. 7. The driver must plan their route to a target in the left lane of the southeastern exit road while also managing varying amounts of traffic around each of the three roundabouts and at the locations where the entry lanes join the roundabout network. Possible "lane-change" opportunities are indicated in the road network by arrows which also signify the direction of transition.

$$C(p) = Ap^2 + B + [c_m]_{x_M} + [c_m]_{x_O}$$

where $A, B \in \mathbb{R}_{0,+}$ reflect the congestion present within the current road segment. The constant $[c_m]_{x_M}$ is an additional penalty only applied at the locations x_M where a vehicle entering the roundabout network from the right lane may choose to merge into a mini-roundabout or merge into the outer ring. Similarly, the constant $[c_m]_{x_O}$ is an additional penalty only applied at the locations x_O where a vehicle entering the roundabout network from the left lane merges into the outer ring. It is easily verified that the causality condition established in Theorem A is satisfied when $A \leq B + [c_m]_{x_M} + [c_m]_{x_O}$. We set $[c_m]_{x_M} = 13$ and $[c_m]_{x_O} = 10$, and we focus on how the driver's optimal lane-change behavior at each of the three roundabout entryways changes as the amount of congestion around each roundabout type varies. We denote the congestion pairs as (A_O, B_O), (A_I, B_I) and (A_M, B_M) for the outermost, innermost, and mini roundabouts respectively, and the resulting optimal lane-change strategies approaching the roundabout network entrances are displayed in FIG. 8, which includes two distinct parts denoted (A) and (B).

FIG. 8 more particularly shows lane-change planning results with roundabout congestion levels in part (A) of $(A_O, B_O) = (2.5,5)$, $(A_I, B_I) = (4.75,9.5)$, and $(A_M, B_M) = (0.875,1.75)$, and in part (B) of $(A_O, B_O) = (4.75,9.5)$, $(A_I, B_I) = (2.5,5)$, and $(A_M, B_M) = (0.875,1.75)$. Once the

outermost roundabout becomes more expensive, vehicles entering the network via the northern entryway are less urgent in their lane change attempt to use the outermost roundabout even though that route is more direct. Similarly, when the innermost roundabout is most expensive, drivers entering the network from the southwestern roundabout are now more urgent in their lane change attempt to utilize the outer roundabout, even though this route is longer.

In both examples, roundabout change attempts are influenced both by network topology and the anticipated congestion within each of the roundabout types. These two examples directly highlight the dilemma that drivers face between following a more-direct route through the network with fewer required switch maneuvers and following a less direct route through the network involving more switch maneuvers. In both (A) and (B) of FIG. 8, drivers entering the network via the southeast (SE) entryway never have an incentive to do anything other than enter the southeastern mini roundabout and use it to turn into the southeastern exit lane to eventually switch into the left lane to reach the target. The most interesting decision points for drivers occur at the southwestern (SW) and northern (N) entryways. From a distance perspective, the shortest route to the target for drivers entering from SW is through the SW and SE mini roundabouts via the inner roundabout. On the other hand, the shortest route to the target for drivers entering from N is along the outer ring. As anticipated congestion around the roundabout types varies, the drivers' optimal strategies in the SW and N entryways varies. When $(A_O, B_O) = (2.5,5), (A_I, B_I) = (4.75,9.5)$, and $(A_M, B_M) = (0.875,1.75)$, the outer ring is not expensive enough for drivers entering from N to consider a route through the mini and inner roundabouts. However, for drivers entering through SW, the price of the innermost roundabout is high enough to compel them to attempt a lane-change with urgency level $p = 0.138$ which would take them along the longer route through thelane-change outer ring. Now, when $(A_O, B_O) = (4.75,9.5), (A_I, B_I) = (2.5,5)$, and $(A_M, B_M) = (0.875,1.75)$, these strategies flip. For drivers entering from SW, the route along the outer ring is not only longer but also significantly more expensive than the other route, so they now revert to taking the route through the mini and inner roundabouts with probability $p = 1$. On the other hand, for drivers entering from N, the most direct route along the outer ring is now

expensive enough that they will consider switching lanes with urgency level $p = 0.112$ to join the much less direct route through the mini and innermost roundabouts.

The above-described illustrative embodiments of FIGS. 1 through 8 disclose a useful class of SSPs referred to herein as OSSP problems. The OSSP transition probability structure provides the decision-maker with the option to choose between stochastic transitions and deterministic transitions as they deem most beneficial to minimizing their cumulative expected cost to the target destination. We proved a sufficient condition on the transition cost function which guarantees monotone causality and the applicability of label-setting methods to solve the OSSP. We also illustrated how OSSPs can be used to determine combined STPs for an autonomous vehicle navigating various road-networks. The resulting STPs capture the inherent stochasticity in the outcome of lane-change attempts, while taking into account how urgently the vehicle should attempt to change lanes under different circumstances.

The disclosed OSSPs are relevant in a wide variety of application areas. For example, illustrative embodiments can be configured to escalate the n-action cost structure presented above to determine whether or not it is actually optimal for the vehicle to attempt a lane change with a chosen level of urgency without trying all of the less-urgent levels first. Other embodiments can impose additional or alternative conditions on the transition cost function which guarantee monotone causality.

Illustrative embodiments will now be further described with reference to FIGS. 9 through 19. These embodiments, including the various numerical examples, are similar to those previously described, but certain terminology will be reintroduced and/or adjusted for use in subsequent description. Accordingly, it is to be appreciated that the terminology in the following description differs in some respects from that used elsewhere herein.

Let $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{x}_{n+1} = \boldsymbol{t}\}$ be the set of all states (nodes), where the last one encodes the target (destination). A sequence $(\boldsymbol{y}_k)_{k=0,1,\ldots}$ reflects a possible stochastic path, where each $\boldsymbol{y}_k \in X$ denotes the state of the process at the k-th stage (i.e., after k transitions). We use $A(\boldsymbol{x}_i)$ to denote the set of actions available at a state $\boldsymbol{x}_i \in X$ and $A = \mathsf{U}_{\boldsymbol{x}\in X} A(\boldsymbol{x})$ to denote the set of all

actions. Choosing any action $\boldsymbol{a} \in A(\boldsymbol{x}_i)$ incurs a known cost $C(\boldsymbol{x}_i, \boldsymbol{a})$ and yields a known probability distribution $p(\boldsymbol{x}_i, \boldsymbol{a}, \boldsymbol{x}_j) = p_{ij}(\boldsymbol{a})$ over the possible successor nodes for the next transition; i.e., if an action $\boldsymbol{a} \in A(\boldsymbol{y}_k)$ is chosen at the k-th stage of the process, then $\mathbb{P}(\boldsymbol{y}_{k+1} = \boldsymbol{x}_j) = p(\boldsymbol{y}_k, \boldsymbol{a}, \boldsymbol{x}_j)$ for all $\boldsymbol{x}_j \in X$.

The target node is assumed to be absorbing and no additional cost is incurred after we reach it; i.e., $p(\boldsymbol{t}, \boldsymbol{a}, \boldsymbol{t}) = 1$ and $C(\boldsymbol{t}, \boldsymbol{a}) = 0$ for all $\boldsymbol{a} \in A(\boldsymbol{t})$. The decision maker's goal is to choose actions to minimize the expected cumulative cost up to the target. This is done by optimizing over the set of control policies. A function $\mu: X \to A$ is a control mapping if $\mu(\boldsymbol{x}) \in A(\boldsymbol{x})$ for all $\boldsymbol{x} \in X$. A control policy is an infinite sequence of control mappings $\pi = (\mu_k)_{k=0,1,\ldots}$ to be used at respective stages of the process. The expected cost of using a policy π starting from any node $\boldsymbol{y}_0 = \boldsymbol{x}$ is defined as

$$\mathcal{J}(\boldsymbol{x}, \pi) = \mathbb{E}\left[\sum_{k=0}^{\infty} C(\boldsymbol{y}_k, \mu_k(\boldsymbol{y}_k)) \mid \boldsymbol{y}_0 = \boldsymbol{x}\right]$$

and the value function $U(\boldsymbol{x})$ denotes the result of minimizing the expected cost starting from $\boldsymbol{x}$:

$$U(\boldsymbol{x}) = \inf_{\pi} \mathcal{J}(\boldsymbol{x}, \pi)$$

With a slight abuse of notation, we also use the symbol μ to refer to a stationary policy $(\mu, \mu, \ldots)$ that uses the same control mapping μ at each stage of the process. The following five assumptions can guarantee that the infimum above is actually attained and that there exists an optimal stationary policy μ^* which attains this minimum for every starting state $\boldsymbol{x} \in X$:

(A1) $A(\boldsymbol{x}_i)$ is compact for all $\boldsymbol{x}_i$.

(A2) $C(\boldsymbol{x}_i, \boldsymbol{a})$ is lower semicontinuous over $A(\boldsymbol{x}_i)$ for all $\boldsymbol{x}_i$.

(A3) $p_{ij}(\boldsymbol{a})$ is continuous over $A(\boldsymbol{x}_i)$ for all $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$.

(A4) There exists at least one policy which results in arriving at the target with probability 1 from every starting state. We refer to policies of this type as proper and all others as improper.

(A5) Every improper policy yields an infinite expected cost starting from at least one node.

Throughout the following description, we will also impose a stronger cost-positivity assumption

(A5′) $C(\boldsymbol{x}, \boldsymbol{a}) > 0,\ \forall \boldsymbol{x} \in X \setminus \{\boldsymbol{t}\}, \boldsymbol{a} \in A(\boldsymbol{x})$.

This is a necessary condition for the applicability of label-setting methods, which will be discussed in detail below. We note that (A1) and (A2) make (A5) a trivial consequence of (A5′). If C is also bounded from above, the existence of a proper stationary policy implies that $U(\boldsymbol{x})$ is finite $\forall \boldsymbol{x} \in X$.

For simplicity of exposition, we will further assume the lack of self-transitions on $X \setminus \{\boldsymbol{t}\}$; i.e., (A6) $p(\boldsymbol{x}, \boldsymbol{a}, \boldsymbol{x}) = 0,\ \forall \boldsymbol{x} \in X \setminus \{\boldsymbol{t}\}, \boldsymbol{a} \in A(\boldsymbol{x})$.

This is not really restrictive since self-transitions can be removed without affecting the value function. Whenever $p_{ii}(\boldsymbol{a}) > 0$ for $\boldsymbol{x}_i \neq \boldsymbol{t}$, one can obtain an SSP with the same value function by switching to

$$\tilde{C}(\boldsymbol{x}_i, \boldsymbol{a}) = \frac{C(\boldsymbol{x}_i, \boldsymbol{a})}{1 - p_{ii}(\boldsymbol{a})};\ \text{and}\ \tilde{p}_{ij}(\boldsymbol{a}) = \begin{cases} 0 & \text{if } j = i, \\ \dfrac{p_{ij}(\boldsymbol{a})}{1 - p_{ii}(\boldsymbol{a})} & \text{if } j \neq i; \end{cases} \text{for } j = 1, \cdots, n+1.$$

For notational ease, denote $U(\boldsymbol{x}_i)$ as U_i and $A(\boldsymbol{x}_i)$ as A_i. The value function can be computed by solving a coupled system of dynamic programming equations

$$U_i = \min_{\boldsymbol{a} \in A_i} \left\{ C(\boldsymbol{x}_i, \boldsymbol{a}) + \sum_{j=1}^{n+1} p_{ij}(\boldsymbol{a}) U_j \right\}, \quad \text{for } i = 1, \ldots n \tag{11}$$
$$U_{n+1} = 0,$$

and the optimal control mapping μ^* can be obtained by using any action from the arg min above. In vector notation, $\boldsymbol{U} = [U_1, \ldots, U_n]^T$ is a fixed point of the operator $\mathcal{T}: \mathbb{R}^n \to \mathbb{R}^n$ defined componentwise as

$$(\mathcal{T}\boldsymbol{W})_i = \min_{\boldsymbol{a} \in A_i} \mathcal{F}_i(\boldsymbol{a}, \boldsymbol{W}),\ \text{where}\ \mathcal{F}_i(\boldsymbol{a}, \boldsymbol{W}) = C(\boldsymbol{x}_i, \boldsymbol{a}) + \sum_{j=1}^{n} p_{ij}(\boldsymbol{a}) W_j \tag{12}$$

This $\mathcal{F}_i(\boldsymbol{a}, \boldsymbol{W})$ represents the expected cost-to-go from $\boldsymbol{x}_i$ when using $\boldsymbol{a} \in A_i$ for the first transition and under the assumption that the remaining expected cost-to-go is encoded in $\boldsymbol{W} \in \mathbb{R}^n$.

The above definition of control policies selects state-dependent actions deterministically. It is well known that extending SSPs to randomized control policies (which select actions probabilistically) does not lower the value function. Nevertheless, we show below that this generalization is useful in pruning the set of actions and in proving the properties of OSSPs.

In the following, we will refer to all actions available in the original SSP as "pure" to distinguish them from the "randomized"/mixed actions allowed here. Suppose the set of pure actions A_i is finite and $\lambda = (\lambda_1, \ldots, \lambda_{|A_i|})$ is some probability distribution over it. Preparing for a transition from $\boldsymbol{y}_k = \boldsymbol{x}_i$, one can implement a mixed (or randomized) action $\boldsymbol{a}_\lambda$ by selecting each $\boldsymbol{a}_r \in A_i$ with probability λ_r. This requires extending the definition above to use

$$\mathcal{F}_i(\boldsymbol{a}_\lambda, \boldsymbol{W}) = \sum_{r=1}^{|A_i|} \lambda_r \mathcal{F}_i(\boldsymbol{a}_r, \boldsymbol{W})$$

and minimizing over all such λ's when computing $(\mathcal{T}\boldsymbol{W})_i$. Of course, this also includes all pure actions since λ might be prescribing a single $\boldsymbol{a}_r \in A_i$ with probability one. We will refer to the problem based on such randomized actions as rSSP. But it's easy to see that the value function of this rSSP is actually the same as in the original SSP since

$$\boldsymbol{a}_* \in \arg\min_{\boldsymbol{a} \in A_i} \mathcal{F}_i(\boldsymbol{a}, \boldsymbol{W}) \implies \mathcal{F}_i(\boldsymbol{a}_*, \boldsymbol{W}) \leq \mathcal{F}_i(\boldsymbol{a}_\lambda, \boldsymbol{W}) \; \forall \lambda \tag{13}$$

Thus, there exists an optimal stationary policy for rSSP that uses only pure actions. The same conclusions also hold for a general compact A_i, except that λ would need to be a probability measure.

We will say that two pure actions $\boldsymbol{a}, \tilde{\boldsymbol{a}} \in A_i$ are transition-equivalent if $p_{ij}(\boldsymbol{a}) = p_{ij}(\tilde{\boldsymbol{a}})$. Clearly, if they are transition-equivalent and $C(\boldsymbol{x}_i, \boldsymbol{a}) \geq C(\boldsymbol{x}_i, \tilde{\boldsymbol{a}})$, then $\boldsymbol{a}$ can be removed from A_i without affecting the value function. This pruning idea can also be extended to use randomized actions.

Definition 1. Dominated, Replaceable, and Useful Actions. An action $\boldsymbol{a} \in A_i$ is probabilistically dominated if there exists a set of actions $\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_l\} \subset A_i$ and a probability

distribution over them $\lambda = (\lambda_1, \ldots, \lambda_l)$ such that the corresponding mixed action $\boldsymbol{a}_\lambda$ achieves the same probability over successor nodes; i.e.,

$$p_{ij}(\boldsymbol{a}) = p_{ij}(\boldsymbol{a}_\lambda) = \sum_{r=1}^{l} \lambda_r p_{ij}(\boldsymbol{a}_r), \ \forall j \tag{14}$$

and the resulting expected cost of the first transition is lower with $\boldsymbol{a}_\lambda$; i.e.,

$$C(\boldsymbol{x}_i, \boldsymbol{a}) > C(\boldsymbol{x}_i, \boldsymbol{a}_\lambda) = \sum_{r=1}^{l} \lambda_r C(\boldsymbol{x}_i, \boldsymbol{a}_r)$$

We will use $\Lambda(\boldsymbol{a})$ to denote the set of randomized actions $\boldsymbol{a}_\lambda$ that satisfy (14) and are thus transition-equivalent to $\boldsymbol{a}$.

We will say that an action $\boldsymbol{a} \in A_i$ is replaceable if there exists a set of actions $\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_l\} \subset A_i$ and a probability distribution over them $\lambda = (\lambda_1, \ldots, \lambda_l)$ such that none of the $\boldsymbol{a}_r$'s are individually transition-equivalent to $\boldsymbol{a}$ but $\boldsymbol{a}_\lambda \in \Lambda(\boldsymbol{a})$; and the expected transition cost is the same; i.e., $C(\boldsymbol{x}_i, \boldsymbol{a}) = C(\boldsymbol{x}_i, \boldsymbol{a}_\lambda) = \sum_{r=1}^{l} \lambda_r C(\boldsymbol{x}_i, \boldsymbol{a}_r)$.

We will say that a pure action $\boldsymbol{a} \in A_i$ is (potentially) useful in an SSP if it is not probabilistically dominated or replaceable, with A_i^u denoting the set of all such actions. In view of (13), whenever $\boldsymbol{a} \in A_i$ is probabilistically dominated or replaceable, there will always exist another $\tilde{\boldsymbol{a}} \in A_i$ which is at least as good of a choice at $\boldsymbol{x}_i$.

Observation 1. Suppose $\boldsymbol{a} \in A_i$ is not useful. Then, for every $\boldsymbol{W} \in \mathbb{R}^n$, there exists another action $\tilde{\boldsymbol{a}} \in A_i \setminus \{\boldsymbol{a}\}$ such that $\mathcal{F}_i(\boldsymbol{a}, \boldsymbol{W}) \geq \mathcal{F}_i(\tilde{\boldsymbol{a}}, \boldsymbol{W})$. So, removing this $\boldsymbol{a}$ from A_i will not affect U_i regardless of the U_j values at any potential successor nodes. The SSP based on A_i^u's instead of A_i's will have exactly the same value function.

Proof: Suppose $\boldsymbol{a}$ is probabilistically dominated (or replaceable) by some $\boldsymbol{a}_\lambda$ and $\tilde{\boldsymbol{a}} \in \arg\min_{\boldsymbol{a}_r \in \mathcal{A}(\boldsymbol{a}_\gamma)} \mathcal{F}_i(\boldsymbol{a}_r, \boldsymbol{W})$, where $\mathcal{A}(\boldsymbol{a}_\lambda) \subset A_i$ is the set of pure actions selected by $\boldsymbol{a}_\lambda$ with positive probability. Then

$$\mathcal{F}_i(\boldsymbol{a}, \boldsymbol{W}) = C(\boldsymbol{x}_i, \boldsymbol{a}) + \sum_{j=1}^{n+1} p_{ij}(\boldsymbol{a}) W_j \geq \sum_{r=1}^{l} \lambda_r C(\boldsymbol{x}_i, \boldsymbol{a}_r) + \sum_{j=1}^{n+1} p_{ij}(\boldsymbol{a}) W_j =$$
$$\sum_{r=1}^{l} \lambda_r \mathcal{F}_i(\boldsymbol{a}_r, \boldsymbol{W}) \geq \mathcal{F}_i(\tilde{\boldsymbol{a}}, \boldsymbol{W}).$$

Since the above definitions are fully based on the cost $C(\boldsymbol{x}_i,\cdot)$, the set of "potentially useful" pure actions is defined locally for each $\boldsymbol{x}_i$ and independently of the global properties of the SSP. The corresponding pruning of the action space can be performed as a pre-processing step to speed up the computation of the value function, which might be particularly worthwhile if one needs to solve a family of related problems; e.g., differing only by the identity of the target $\boldsymbol{t} \in X$.

A geometric interpretation can be provided as follows. Once A_i no longer includes any transition-equivalent elements, it is often convenient to identify the remaining pure actions with their corresponding probability distributions over successor nodes. Suppose $Z_i = \{\boldsymbol{z}_1, \dots, \boldsymbol{z}_m\} \subset X$ is a set of all possible immediate successors of $\boldsymbol{x}_i$ and let

$$\Xi_m = \left\{\boldsymbol{\xi} = (\xi_1, \dots, \xi_m) \mid \xi_1 + \cdots + \xi_m = 1 \text{ and } \xi_j \geq 0 \; \forall j\right\}$$

denote the $(m-1)$-dimensional probability simplex. We can then assume that $A_i \subset \Xi_m$ and $\boldsymbol{\xi} \in A_i$ implies $p(\boldsymbol{x}_i, \boldsymbol{\xi}, \boldsymbol{z}_j) = \xi_j$ for $j = 1, \dots, m$. The cost of the next transition is then $C(\boldsymbol{x}_i, \boldsymbol{\xi})$ though we will also use the notation $C(\boldsymbol{\xi})$ whenever $\boldsymbol{x}_i \in X$ is clear from the context. Allowing the use of randomized actions in rSSP is equivalent to switching from A_i to its convex hull $A_i^{co} \subset \Xi_m$. Using the best randomized action in each transition equivalence class essentially replaces C with its lower convex envelope

$$\check{C}(\boldsymbol{\xi}) = \min_{\boldsymbol{a}_\lambda \in \Lambda(\boldsymbol{\xi})} C(\boldsymbol{a}_\lambda) \; \forall \boldsymbol{\xi} \in A_i^{co},$$

where $\boldsymbol{a}_\lambda$ is specified by some choice of $\lambda \in \Xi_m$ and $\{\boldsymbol{\xi}^1, \dots, \boldsymbol{\xi}^m\} \subset A_i$, which satisfy $\xi_j = \sum_{r=1}^m \lambda_r \xi_r^j$ for $j = 1, \dots, m$ while $C(\boldsymbol{a}_\lambda) = \sum_{r=1}^m \lambda_r C(\boldsymbol{\xi}^r)$. The resulting C is convex and thus continuous on the interior of A_i^{co} (and lower semi-continuous on ∂A_i^{co}). The pruning outlined above is possible because the minimum in (11) can only be attained at $\boldsymbol{\xi} \in A_i$ if $C(\boldsymbol{\xi}) = \check{C}(\boldsymbol{\xi})$. If an optimal $\boldsymbol{\xi}$ is replaceable, that same minimum will be also achieved by another (non-replaceable) action. An action $\boldsymbol{\xi} \in A_i$ is "useful" if and only if $(\boldsymbol{\xi}, C(\boldsymbol{\xi}))$ is an extreme point of the epigraph of $\check{C}$. FIG. 9 illustrates this for the case of $m = 2$, where there are only two possible successor nodes, and we can identify $\boldsymbol{\xi} = (1-p, p)$) with $p \in [0,1]$. In this case, the transition cost $C(\boldsymbol{x}, \boldsymbol{a})$ may

be represented instead by the mapping $K: A_i \to \mathbb{R}$, and its convexified version $\check{C}(\boldsymbol{\xi})$ by the function $\check{K}: [0,1] \to \mathbb{R}$.

FIG. 9 more specifically illustrates an example of (A) deterministic and (B) probabilistic pruning of $K(p(\boldsymbol{a}))$ when $m = 2$. In part (A) of the figure, the transition probabilities associated with each action in $A_i = [\underline{\boldsymbol{a}}, \overline{\boldsymbol{a}}] \cup \{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_4\}$ are indicated on the p-axis. During the deterministic pruning process, all actions $\boldsymbol{a} \in A_i$ that are transition-equivalent to another action $\tilde{\boldsymbol{a}} \in A_i$ with $K(p(\boldsymbol{a})) \geq K(p(\tilde{\boldsymbol{a}}))$ are removed along with the corresponding portion of the transition cost curve, as shown by dashes. In part (B) of the figure, following the deterministic pruning, we obtain $\check{K}(p)$ by taking the lower convex envelope of K (shown by a curve with solid and dashed portions), and the resulting A_i^{co} is indicated above the p-axis. Replaceable actions $\boldsymbol{a}'$ and $\boldsymbol{a}_1$ are also removed at this stage. The transition probabilities associated with the remaining useful pure actions A_i^u are indicated on the p-axis, and their corresponding transition cost values are also marked along $\check{K}$.

In the example above, $A_i = [\underline{\boldsymbol{a}}, \overline{\boldsymbol{a}}] \cup \{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_4\}$ and some of the pure actions from $[\underline{\boldsymbol{a}}, \overline{\boldsymbol{a}}]$ are transition-equivalent. Once these are removed (as indicated by the dashes in part (A) of FIG. 9), we can perform additional pruning based on randomized actions, as illustrated in part (B) of FIG. 9.

For any pure action $\boldsymbol{a} \in A_i$, we can define the set of potential successor nodes $\mathcal{I}(\boldsymbol{a}) = \{\boldsymbol{x}_j \in X \mid p_{ij}(\boldsymbol{a}) > 0\}$. We will say that a pure action $\boldsymbol{a} \in A$ is deterministic if its $\mathcal{I}(\boldsymbol{a})$ is a singleton - using such an $\boldsymbol{a}$ ensures a deterministic transition. We now focus on a specific subclass of SSPs, in which every stochastically realizable path is also realizable using only deterministic actions. Later, we will demonstrate their usefulness in approximating solutions of continuous optimal control problems and in routing of autonomous vehicles.

Definition 2. OSSP. We will refer to an SSP as an OSSP if

$$\exists \boldsymbol{a} \in A_i \text{ s.t. } p_{ij}(\boldsymbol{a}) > 0 \implies \exists \tilde{\boldsymbol{a}} \in A_i \text{ s.t. } p_{ij}(\tilde{\boldsymbol{a}}) = 1 \text{ holds for all } i \text{ and } j \qquad (15)$$

If the set of potential successor nodes $\mathcal{I}(\boldsymbol{x}_i) = \{\boldsymbol{x}_j \in X \mid \exists \boldsymbol{a} \in A_i \text{ s.t. } p_{ij}(\boldsymbol{a}) > 0\}$ has m elements, then the post-pruning A_i^u can be identified with a subset of Ξ_m, which in an OSSP will include all vertices of this probability simplex, encoding deterministic transitions. Correspondingly, the randomized-policy version of this problem (rOSSP), would have the action set $A_i^{co} = \Xi_m$.

We note that the OSSPs are also closely related to MSSPs. The modes in MSSPs are subsets of $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{t}\}$, describing the possible successor nodes under a particular class of actions. Each non-terminal node $\boldsymbol{x} \in X \setminus \{\boldsymbol{t}\}$ has its own set of such modes $\mathcal{S}(\boldsymbol{x})$, and each mode $s \in \mathcal{S}(\boldsymbol{x})$ is usually specified by enumerating the nodes in it; i.e., $s = \{\boldsymbol{z}_1^s, \ldots, \boldsymbol{z}_{|s|}^s\} \subset X \setminus \{\boldsymbol{x}\}$. MSSPs are built on two critical assumptions:

(M1) any available action has all of its successor nodes in one of these modes; i.e.,

$$\forall \boldsymbol{x}_i \in X, \boldsymbol{a} \in A(\boldsymbol{x}_i)\ \exists s \in \mathcal{S}(\boldsymbol{x}_i) \text{ s.t. } \mathcal{I}(\boldsymbol{a}) \subset s$$

(M2) every probability distribution over the set of successor nodes in any mode is achievable via some action; i.e.,

$$\forall \boldsymbol{x}_i \in X \setminus \{\boldsymbol{t}\}, s \in \mathcal{S}(\boldsymbol{x}_i), \text{ and } \boldsymbol{\xi} \in \Xi_{|s|}\ \exists \boldsymbol{a} \in A(\boldsymbol{x}_i) \text{ s.t. } p(\boldsymbol{x}_i, \boldsymbol{a}, \boldsymbol{z}_r) = \xi_r,\ \forall \boldsymbol{z}_r \in s,$$

where $\Xi_{|s|}$ is the $(|s| - 1)$-dimensional probability simplex.

Thus, a decision at each stage of an MSSP is twofold: a deterministic choice of a mode and a choice of a probability distribution over the possible successor nodes in that mode. This makes it natural to represent the actions as $\boldsymbol{a} = (s, \boldsymbol{\xi}) \in A(\boldsymbol{x})$, where $\boldsymbol{\xi} \in \Xi_{|s|}$, and then re-write the optimality equation (11) as

$$\begin{aligned} V^s(\boldsymbol{x}) &= \min_{\boldsymbol{\xi} \in \Xi_{|s|}} \left\{ C^s(\boldsymbol{x}, \boldsymbol{\xi}) + \sum_{r=1}^{|s|} \xi_r U(\boldsymbol{z}_r^s) \right\}, \\ U(\boldsymbol{x}) &= \min_{s \in \mathcal{S}(\boldsymbol{x})} V^s(\boldsymbol{x}), \\ U(\boldsymbol{t}) &= 0. \end{aligned} \qquad \forall \boldsymbol{x} \in X \setminus \{\boldsymbol{t}\}, \tag{16}$$

OSSP is a generalization of MSSP since the assumption (M2) implies the availability of deterministic transitions from $\boldsymbol{x}_i$ to each $\boldsymbol{z}_r \in s \in \mathcal{S}(\boldsymbol{x}_i)$. Unless an MSSP is fully deterministic,

it must also include a continuum of actions spanning entire probability simplexes corresponding to each mode. This clearly does not have to be the case in general OSSPs, for which each A_i^u might be a proper (or even finite) subset of the probability simplex.

On the other hand, for any given OSSP, it is easy to construct an MSSP with the same value function; e.g., by using the convexified cost $\check{C}$ instead of the original C. To have a single mode at every node, one can define $\mathcal{S}(\boldsymbol{x}) = \{s\}$ with $s = \mathcal{I}(\boldsymbol{x}) = \mathsf{U}_{\boldsymbol{a}\in A(\boldsymbol{x})}\mathcal{I}(\boldsymbol{a})$, and then use the convexified cost $\check{C}$ instead of the original C. The action set $A(\boldsymbol{x})$ can be then extended by adding all randomized actions to satisfy (M2).

It is worth noting that in the definition of $\check{C}$, the result will not change if the randomized actions are formed by only using pure actions $\tilde{\boldsymbol{a}}$ such that $\mathcal{I}(\tilde{\boldsymbol{a}}) \subset \mathcal{I}(\boldsymbol{a})$. (In all other cases, $\boldsymbol{a}_\lambda \notin \Lambda(\boldsymbol{x})$.) This suggests a natural approach to introduce "incomplete modes" in OSSPs. We will say that $s \subset \mathcal{I}(\boldsymbol{x})$ is an incomplete mode if $\exists \boldsymbol{a} \in A(\boldsymbol{x})$ s.t. $s = \mathcal{I}(\boldsymbol{a})$ but $\nexists \tilde{\boldsymbol{a}} \in A(\boldsymbol{x})$ s.t. $s \subsetneq \mathcal{I}(\tilde{\boldsymbol{a}})$. Each action $\boldsymbol{a} \in A(\boldsymbol{x})$ is associated with a mode $s \in \mathcal{S}(\boldsymbol{x})$ if $\mathcal{I}(\boldsymbol{a}) \subset s$. Thus, an action might be associated with multiple modes simultaneously.

To provide a concrete example, consider $A(\boldsymbol{x}) = \{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{10}\}$, where the first five actions are deterministic $\left(\mathcal{I}(\boldsymbol{a}_j) = \boldsymbol{x}_j, j = 1, \ldots, 5\right)$ while the rest of them are not:

$\mathcal{I}(\boldsymbol{a}_6) = \{\boldsymbol{x}_1, \boldsymbol{x}_2\}$, $\mathcal{I}(\boldsymbol{a}_7) = \{\boldsymbol{x}_2, \boldsymbol{x}_3\}$, $\mathcal{I}(\boldsymbol{a}_8) = \{\boldsymbol{x}_3, \boldsymbol{x}_4\}$, $\mathcal{I}(\boldsymbol{a}_9) = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3\}$, $\mathcal{I}(\boldsymbol{a}_{10}) = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_4\}$.

Based on the above definition, $\mathcal{I}(\boldsymbol{x}) = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_5\}$ and $\mathcal{S}(\boldsymbol{x}) = \{s_1, \ldots, s_4\}$ with $s_1 = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3\}$ (associated actions: $\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3, \boldsymbol{a}_6, \boldsymbol{a}_7, \boldsymbol{a}_9$), $s_2 = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_4\}$ (associated actions: $\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_4, \boldsymbol{a}_6, \boldsymbol{a}_{10}$), $s_3 = \{\boldsymbol{x}_3, \boldsymbol{x}_4\}$ (associated actions: $\boldsymbol{a}_3, \boldsymbol{a}_4, \boldsymbol{a}_8$), and $s_4 = \{\boldsymbol{x}_5\}$ (associated action: $\boldsymbol{a}_5$). Compared to MSSPs, these modes are "incomplete" since (M2) is violated. But this completeness can be restored without changing the value function by adding all randomized actions in each mode and using $\check{C}$ to define their cost.

Aspects of label-setting and Monotone Causality in OSSPs will now be described. Since the value function $\boldsymbol{U} = [U_1, \ldots, U_n]^T$ is a fixed point of the operator $\mathcal{T}$ defined in (12), the simplest

approach for computing $\boldsymbol{U}$ is through value iterations (VI), in which one starts with an initial guess $\boldsymbol{W}^0 \in \mathbb{R}^n$ and updates it iteratively by taking $\boldsymbol{W}^{k+1} = \mathcal{T}\boldsymbol{W}^k$. The operator $\mathcal{T}$ is generally not a contraction unless all stationary policies are known to be proper. But it has been shown that assumptions (A1)-(A5) guarantee that this fixed point is unique and $\boldsymbol{W}^k \to \boldsymbol{U}$ as $k \to \infty$ regardless of $\boldsymbol{W}^0$. Unfortunately, for a general SSP, this does not necessarily occur after any finite number of iterations. The convergence can be slow and the VI algorithm is often impractical for large problems. Finding computationally efficient alternatives to these basic value iterations has been an active research area in the last several decades. One possible direction is to use a Gauss-Seidel relaxation (GSVI), where the components of $\boldsymbol{W}^{k+1}$ are computed sequentially and the previously computed components are immediately used in computing the remaining ones; i.e.,

$$\boldsymbol{W}_i^{k+1} = \min_{\boldsymbol{a} \in A_i} \mathcal{F}_i\left(\boldsymbol{a}, \left[\boldsymbol{W}_1^{k+1}, \ldots, \boldsymbol{W}_{i-1}^{k+1}, \boldsymbol{W}_i^k, \ldots, \boldsymbol{W}_n^k\right]^T\right)$$

But the efficiency of this approach is heavily dependent on the ordering imposed on the nodes/states in X; so, in some implementations the algorithm alternates through several orderings that are likely to be efficient. In others, the so called label-correcting methods, the next node/state to be updated is determined dynamically, based on the current vector of tentative labels/values $\boldsymbol{W}$ and the history of updates up till that point. Some of these methods were originally developed for the deterministic shortest path setting, but have since been adapted to SSPs as well, particularly when used to discretize Hamilton-Jacobi equations.

Other notable approaches include topological value iterations (in which the topological structure imposed on X by A is taken into account to attempt splitting an SSP into a sequence of causally ordered subrpoblems), policy iterations (in which the goal is to produce an improving sequence of stationary policies (μ_k), with $\boldsymbol{U}$ recovered as a limit of their respective policy value vectors), and hybrid policy-value iterations.

Here we are primarily interested in a subclass of SSPs for which VI actually does converge in at most n iterations regardless of $\boldsymbol{W}^0$. Typically, this happens when at least one element of $\boldsymbol{W}^k$ becomes newly converged in each iteration. But even assuming the minimization in (12) can be

performed in $O(1)$ operations, the computational cost of a single value iteration is $O(n)$, yielding the overall cost of $O(n^2)$ up to convergence. In such SSPs, the same worst-case $O(n^2)$ computational cost also holds for all VI variants mentioned above, while the policy iterations would still require an infinite number of steps for full convergence. In contrast, our goal in illustrative embodiments is to obtain $\boldsymbol{U}$ in $O(n\log n)$ or $O(n)$ operations, bounding the number of "approximate U_i" updates based on that node's stochastic outdegree (i.e., $|\mathcal{J}(\boldsymbol{x}_i)|$) rather than on the overall number of non-terminal nodes $n \gg \max_{\boldsymbol{x}_i}|\mathcal{J}(\boldsymbol{x}_i)|$. For the deterministic shortest path problems, this cost reduction is accomplished by classical label-setting methods. We will describe the prior (implicit) conditions for their general SSP-applicability, and then derive new (explicit) conditions for their OSSP-applicability.

Classical shortest/cheapest path problems on directed graphs can be interpreted as a subclass of SSPs, in which all actions yield deterministic transitions. In that setting, $C_{ij} = C(\boldsymbol{x}_i, \boldsymbol{x}_j)$ encodes the cost of a direct $(\boldsymbol{x}_i \to \boldsymbol{x}_j)$ transition, with the set of potential successor nodes denoted $N_i = N(\boldsymbol{x}_i) = \{\boldsymbol{x}_j \in X \mid C_{ij} < \infty\}$. We will further assume that $C_{ij} \geq \delta \geq 0$. The value function satisfies the following dynamic programming equations:

$$U_i = \min_{\boldsymbol{x}_j \in N(\boldsymbol{x}_i)} \{C_{ij} + U_j\},\ \forall \boldsymbol{x}_i \in X \setminus \{\boldsymbol{t}\}$$

with $U(\boldsymbol{t}) = 0$. Efficient algorithms for solving these equations are well-known and covered in standard references, but we provide a quick overview for the sake of completeness.

The key idea of label-setting methods is to re-order iterations so that the tentative value of each node $\boldsymbol{x}_i$ is updated at most $|N_i|$ times. This yields a significant performance advantage over value iterations, particularly when the outdegree of nodes is bounded and relatively small, $\kappa = \max_i |N_i| \ll n$. Since the number of updates per node is bounded by κ, such methods are also often considered noniterative. The non-negativity of transition costs implies a monotone causality property:

$$U_i \text{ may depend on } U_j \text{ only if } U_i \geq U_j. \tag{17}$$

If all $C_{ij} > 0$ and nodes could be ordered monotonically based on U_i's, GSVI would converge in a single iteration. But since such an ordering is not known in advance, it has to be obtained at run-time. Dijkstra's algorithm accomplishes this by recomputing tentative labels V_i while maintaining two lists of nodes: the list of known/converged nodes $K = \{\boldsymbol{x}_i \mid V_i = U_i$ is confirmed$\}$ and the list of tentative nodes $L = X \setminus K$ for which $V_i \geq U_i$. The basic version of this algorithm starts with $V(\boldsymbol{t}) = U(\boldsymbol{t}) = 0, K = \{\boldsymbol{t}\}$, and $V_i = +\infty$ for all $\boldsymbol{x}_i \in L = X \setminus \{\boldsymbol{t}\}$. At each stage of Dijkstra's algorithm, the node $\bar{\boldsymbol{x}}$ with the current smallest finite value in L is moved to K, and other tentative nodes adjacent to it (i.e., all $\boldsymbol{x}_i \in L$ s.t. $\bar{\boldsymbol{x}} \in N_i$) are updated by setting $V_i := \min_{\boldsymbol{x}_j \in N_i \cap K}\{C_{ij} + V_j\}$, or, more efficiently, by using $V_i := \min(V_i, C(\boldsymbol{x}_i, \bar{\boldsymbol{x}}) + V(\bar{\boldsymbol{x}}))$. The algorithm terminates once $L = \emptyset$ or once all values remaining on L are infinite. In the latter case, the nodes remaining on L are not path-connected to $\boldsymbol{t}$. The algorithm terminates after at most κn updates and yields the correct U_i values even if $\delta = 0$. To identify $\bar{\boldsymbol{x}}$ efficiently, L is typically implemented using heap-sort data structures, resulting in an overall computational cost of $O(n\log n)$.

If $\delta > 0$, the nodes whose tentative values are less than δ apart cannot depend on each other. This makes it possible to use another label-setting method, known as Dial's algorithm, in which the tentative values are not sorted but instead placed into "buckets" of width δ. At each stage of the algorithm, all tentative nodes in the current smallest bucket are moved to K simultaneously, and all tentative labels of nodes adjacent to them are updated, switching these adjacent nodes to new buckets if necessary. Since inserting to and deleting from a bucket can be performed in $O(1)$ time, the overall computational complexity of Dial's algorithm is $O(n)$. Unlike Dijkstra's algorithm, Dial's algorithm is also naturally parallelizable. Nevertheless, which of them is more efficient in practice depends on the properties of the SP problem.

Avoiding iterative methods in general SSPs is usually harder. Given any stationary policy μ, one can define the corresponding dependency digraph G_μ by starting with the nodes X and adding arcs whenever direct transitions are possible under μ (i.e., adding an arc from $\boldsymbol{x}_i$ to $\boldsymbol{x}_j$

whenever $p_{ij}(\mu(\boldsymbol{x}_i)) > 0$.) We will call an SSP causal if there exists an optimal stationary policy μ^* whose dependency digraph G_{μ^*} is acyclic. Value iterations on any such causal SSP will converge to $\boldsymbol{U}$ after at most n iterations. If one were to use a reverse topological ordering of G_{μ^*} to sort X, the GSVI algorithm would converge in a single iteration. This property is trivially satisfied by special explicitly causal SSPs, in which the dependency digraph of every stationary policy is acyclic, making it easy to obtain the full $\boldsymbol{U}$ with only n node value updates regardless of the properties of C. This mirrors the fact that Dijkstra's algorithm and Dial's algorithm are not really needed to find deterministic shortest paths on acyclic graphs.

Going beyond explicitly causal SSPs, a natural question to consider is whether the label-setting methods might be applicable. We will say that an optimal stationary policy μ^* is consistently improving if, for all $\boldsymbol{x}_i \neq \boldsymbol{t}$,

$$p_{ij}(\mu^*(\boldsymbol{x}_i)) > 0 \Longrightarrow U_i > U_j \tag{18}$$

This property is a stochastic analogue of (17). If such μ^* is known to exist, (18) guarantees that an SSP-version of Dijkstra's algorithm will correctly produce $\boldsymbol{U}$ as its output. Although it might seem natural to pose a "$U_i \geq U_j$" condition in (18), to fully mirror (17), for non-deterministic actions, this condition turns out to be too weak and may result in Dijkstra's algorithm failing to converge to $\boldsymbol{U}$. In terms of implementation, the only difference from the deterministic SP case described previously, is that, once $\overline{\boldsymbol{x}}$ is moved to K, we would need to update all $\boldsymbol{x}_i$'s such that $\overline{\boldsymbol{x}} \in \mathcal{I}(\boldsymbol{x}_i)$ by using

$$V_i := \min\left(V_i, \min_{\substack{\boldsymbol{a}\in A_i \text{ s.t.} \\ \overline{\boldsymbol{x}}\in\mathcal{I}(\boldsymbol{a})\subset K}} \mathcal{F}_i(\boldsymbol{a}, \boldsymbol{V})\right) \tag{19}$$

Continuing this approach, we will say that an optimal stationary policy μ^* is consistently δ improving if, for some $\delta > 0$ and all $\boldsymbol{x}_i \neq \boldsymbol{t}$,

$$p_{ij}(\mu^*(\boldsymbol{x}_i)) \geq 0 \Longrightarrow U_i \geq U_j + \delta.$$

The existence of such a μ^* similarly guarantees the applicability of Dial's algorithm with buckets of width δ. Consistently δ-improving policies were previously defined with a strict inequality and $\delta \geq 0$. However, since the buckets have a positive width δ and $(U_j + \delta)$ is never in the same bucket as U_j, the above definition is more suitable to guarantee the applicability of Dial's algorithm. If we assume that the minimization in (19) can be performed in $O(1)$ operations and stochastic outdegrees $|\mathcal{J}(\boldsymbol{x}_i)|$ are bounded by a constant κ, then the overall computational cost will scale the same way as in the deterministic case: $O(n\log\, n)$ for Dijkstra's algorithm and $O(n)$ for Dial's algorithm.

We will say that a (non-explicitly causal) SSP is monotone (δ-) causal if at least one of its optimal stationary policies is consistently (δ-) improving. Unfortunately, both of the above criteria are implicit and hard to apply in practice since none of the optimal policies are known before $\boldsymbol{U}$ is computed. It is natural to search for sufficient monotone causality criteria that can be verified locally (e.g., based on the cost function properties at each node/state) without considering the global structure of the SSP. Such SSP examples have been used to apply Dijkstra's algorithm and Dial's algorithm to regular-grid semi-Lagrangian discretizations of isotropic optimal control problems. Related and more general algorithms were also developed, and Multimode SSPs were then introduced both to provide an overall structure for discussing the label-setting in PDE discretizations and to show that the monotone (δ-) causality can be guaranteed even for some SSPs unrelated to optimal control problems. The idea was to pose the sufficient conditions based on the cost C only, and have them verified on a mode-per-mode, node-per-node basis. Given the close relationship between the OSSPs and MSSPs, it is worth noting that some known criteria on the cost have either required C to be concave or posed conditions on its derivatives, while we allow C to be merely lower semi-continuous.

Aspects of monotone causality in OSSPs will now be further described. Consider an OSSP satisfying assumptions (A1)-(A4), (A5′), (A6), and (15). Suppose an action $\boldsymbol{a} \in A_i$ results in a list of possible successor nodes $\mathcal{J}(\boldsymbol{a}) = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\} \subset X \setminus \{\boldsymbol{x}_i\}$, with the respective transition probabilities $p(\boldsymbol{x}_i, \boldsymbol{a}, \boldsymbol{z}_j) = \xi_j > 0$ for $j = 1, \ldots, m$ and $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_m) \in \text{int}\,(\Xi_m)$. Of course,

these $m, \boldsymbol{z}_r$'s, and ξ_r's are always action-dependent. But for the sake of readability, we do not indicate this in the notation whenever a specific action is clear from the context.

Assuming this $\boldsymbol{a} \in A_i$ is not deterministic (i.e., $m > 1$) and choosing any specific $r \in \{1, \ldots, m\}$, we define $\boldsymbol{\zeta}_r = (\zeta_{r,1}, \ldots, \zeta_{r,m})$ to be an oblique (proportional) projection of $\boldsymbol{\xi}$ as follows

$$\zeta_{r,j} = \begin{cases} 0, & \text{if } j = r \\ \xi_j/(1-\xi_r), & \text{otherwise} \end{cases}$$

We note that, omitting its r-th zero component, $\boldsymbol{\zeta}_r$ can be thought of as a point in $\text{int}\left(\Xi_{(m-1)}\right)$, the set $\Lambda(\boldsymbol{\zeta}_r)$ of relaxed actions transition-equivalent to $\boldsymbol{\zeta}_r$ is nonempty, and the convexified cost of $\boldsymbol{\zeta}_r$ can be defined as $\check{C}(\boldsymbol{\zeta}_r) = \min_{a_\lambda \in \Lambda(\boldsymbol{\zeta}_r)} C(\boldsymbol{a}_\lambda)$.

Theorem 1. Sufficient Monotone δ-Causality Condition in OSSPs.

Suppose there exists a $\delta \geq 0$ such that, for all $\boldsymbol{x}_i \neq \boldsymbol{t}$ and $\boldsymbol{a} \in A_i^u$, if $\boldsymbol{a}$ is deterministic, then $C(\boldsymbol{x}_i, \boldsymbol{a}) \geq \delta$; and if $\boldsymbol{a}$ is not deterministic, then

$$C(\boldsymbol{x}_i, \boldsymbol{a}) \geq (1-\xi_r)\check{C}(\boldsymbol{\zeta}_r) + \xi_r \delta, \ \forall r \in \{1, \ldots, |\mathcal{J}(\boldsymbol{a})|\} \tag{20}$$

If these conditions are satisfied, this OSSP is monotone causal and Dijkstra's algorithm is applicable. If $\delta > 0$, the OSSP is monotone δ-causal and Dial's algorithm with buckets of width δ is also applicable.

Proof: Define the set of optimal actions $A_i^* = \arg\min_{\boldsymbol{a} \in A_i} \{C(\boldsymbol{x}_i, \boldsymbol{a}) + \sum_{j=1}^{n+1} p_{ij}(\boldsymbol{a}) U_j\}$, and suppose that $\boldsymbol{a}^* \in A_i^*$ yields the minimal number of successor nodes among them (i.e., $|\mathcal{J}(\boldsymbol{a}^\sharp)| \geq |\mathcal{J}(\boldsymbol{a}^*)|$ for all $\boldsymbol{a}^\sharp \in A_i^*$.) If $\boldsymbol{a}^*$ is deterministic, then $\mathcal{J}(\boldsymbol{a}^*) = \{\boldsymbol{z}_1\}$ and U_i depends only on $U(\boldsymbol{z}_1)$ while $U_i = C(\boldsymbol{x}_i, \boldsymbol{a}^*) + U(\boldsymbol{z}_1) \geq \delta + U(\boldsymbol{z}_1)$. For $\delta = 0$ this inequality is strict due to (A5′).

If $\boldsymbol{a}^*$ is not deterministic, its use leads to one of the successor nodes $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\}$ with respective probabilities $\xi_r > 0$, and U_i depends on all of the $U(\boldsymbol{z}_r)$ with $r \in \{1, \ldots, m\}$. Choosing any specific r and combining the optimality of $\boldsymbol{a}^*$ with (20), we see that

$$U_i = C(\boldsymbol{x}_i, \boldsymbol{a}^*) + \sum_{j=1}^{m} \xi_j U(\boldsymbol{z}_j) \geq (1-\xi_r)\check{C}(\boldsymbol{\zeta}_r) + \xi_r \delta + \sum_{j=1}^{m} \xi_j U(\boldsymbol{z}_j) \tag{21}$$

On the other hand, using $\boldsymbol{a}^*$ must be strictly better that using any randomized action $\boldsymbol{a}_\lambda \in \Lambda(\boldsymbol{\zeta}_r)$. Otherwise, the best pure action $\tilde{\boldsymbol{a}}$ among those potentially selected by $\boldsymbol{a}_\lambda$ would have to be at least as good as $\boldsymbol{a}^*$, leading to $\tilde{\boldsymbol{a}} \in A_i^*$ with $|\mathcal{I}(\tilde{\boldsymbol{a}})| \leq m-1$, which is a contradiction. Thus, we have $U_i < \mathcal{F}_i(\boldsymbol{a}_\lambda, \boldsymbol{U})$ for all $\boldsymbol{a}_\lambda \in \Lambda(\boldsymbol{\zeta}_r)$, which implies

$$U_i < \check{C}(\boldsymbol{\zeta}_r) + \sum_{j \neq r} \zeta_{r,j} U(\boldsymbol{z}_j) = \check{C}(\boldsymbol{\zeta}_r) + \frac{1}{(1-\xi_r)} \sum_{j \neq r} \xi_j U(\boldsymbol{z}_j)$$

Multiplying both sides by $(1-\xi_r)$ and rearranging,

$$U_i < (1-\xi_r)\check{C}(\boldsymbol{\zeta}_r) + \sum_{j \neq r} \xi_j U(\boldsymbol{z}_j) + \xi_r U_i \tag{22}$$

Combining (21) and (22), canceling the terms present on both sides of the inequality, and dividing by $\xi_r > 0$, we obtain $U_i > \delta + U(\boldsymbol{z}_r)$, which proves the monotone (δ-)causality of this OSSP.

The above condition is explicit and sharp in the sense to be outlined below in Theorem 3, but it requires checking the convexified cost of multiple oblique projections for each non-deterministic action. We now provide another sufficient condition, which is more restrictive but easier to verify.

FIG. 10 illustrates the geometric interpretation of (23) when $m = 2$ and, in part (A), $\delta = 0$ and in part (B), $\delta = 0.3$, for several classes of transition cost functions for which the OSSP theory described previously applies. These include two solid curves that are each smooth and convex, another solid curve that is smooth and nonconvex, and a dashed curve that is the latter's convexified version. In all cases, $K(p)$ is monotone (δ-)causal provided that the curve stays entirely above the two restriction lines on the interval $[0,1]$. In (A), the lowermost smooth and convex curve violates the $i = 2$ condition in (23) when $\delta = 0$, so it cannot be monotone (δ-) causal. While the other two curves satisfy (23) when $\delta = 0$, only the nonconvex curve (along with its convexified version) is monotone (δ-) causal for $\delta \leq 0.3$. The uppermost smooth and convex curve can only be monotone causal, as the restriction lines imposed by (23) coincide with its tangent lines at $p = 0$ and $p = 1$.

Theorem 2. Simplified Monotone δ-Causality Condition in OSSPs. Suppose there exists a $\delta \geq 0$ such that, for all $\boldsymbol{x}_i \neq \boldsymbol{t}, \boldsymbol{a} \in A_i^u$, and every $r \in \{1, \ldots, m\}$,

$$C(\boldsymbol{x}_i, \boldsymbol{a}) \geq \sum_{j=1, j\neq r}^{m} \xi_j C_j + \xi_r \delta \tag{23}$$

where $\mathcal{I}(\boldsymbol{a}) = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\}, p(\boldsymbol{x}_i, \boldsymbol{a}, \boldsymbol{z}_j) = \xi_j > 0$, and each C_j encodes the cost of a deterministic action $\boldsymbol{e}_j \in A_i$ transitioning to $\boldsymbol{z}_j$. If these conditions are satisfied, this OSSP is monotone (δ-) causal.

Proof: If $\boldsymbol{a}$ is deterministic and leads to $\boldsymbol{z}_r$, then (23) yields $C_r = C(\boldsymbol{x}_i, \boldsymbol{a}) \geq \delta$. If $\boldsymbol{a}$ is not deterministic, then all of these $\boldsymbol{e}_j$'s exist by the definition of OSSPs, and a randomized action $\boldsymbol{a}_\lambda$ selecting each $\boldsymbol{e}_j$ with probability $\xi_j/(1-\xi_r)$ is transition-equivalent to $\boldsymbol{\zeta}_r$. Thus, $C(\boldsymbol{a}_\lambda) = \frac{1}{(1-\xi_r)}\sum_{j=1, j\neq r}^{m} \xi_j C_j \geq \check{C}(\boldsymbol{\zeta}_r)$ and (23) implies (20).

Note that the two criteria (20) and (23) are equivalent when $m = 2$. The latter also has a simple geometric interpretation illustrated in FIG. 10, with $\boldsymbol{\xi} = (1-p, p) \in \Xi_2$ and the assumption that all actions $\boldsymbol{a} \in A_i$ lead to the same two successor states $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$. The oblique projections of any $\boldsymbol{\xi} \in \text{int}(\Xi_2)$ will correspond to deterministic transitions to $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$. This OSSP is monotone (δ-)causal if the graph of $K(p) = C(1-p, p)$ does not venture below the two straight lines $K = (1-p)C_1 + \delta p$ and $K = pC_2 + \delta(1-p)$.

For $m > 2$, the two criteria are only equivalent if $\sum_{j=1, j\neq r}^{m} \xi_j C_j = (1-\xi_r)\check{C}(\zeta_r)$ for each r, which is only true if $\check{C}$ is linear on $\partial\Xi_m$ (or, equivalently, if $A_i^u \cap \partial\Xi_m$ contains only deterministic actions). Unfortunately, this is often not the case for OSSPs arising in practical applications. Consider, for example, the case where $m = 3, A_i = \Xi_3$, and the cost function is $C(\boldsymbol{\xi}) = \sqrt{\xi_1^2 + \xi_2^2 + \xi_3^2}$. Theorem 1 guarantees its monotone causality: if $\xi_r > 0$ then $\check{C}(\boldsymbol{\zeta}_r) = C(\boldsymbol{\zeta}_r) = (1-\xi_r)^{-1}\sqrt{\sum_{j\neq r} \xi_j^2}$. Thus, $(C(\boldsymbol{\xi}))^2 - \left((1-\xi_r)C(\boldsymbol{\zeta}_r)\right)^2 = \xi_r^2 > 0$, which verifies that (20) holds for $\delta = 0$. On the other hand, a simpler criterion from Theorem 2 is not satisfied by

this C since (23) requires $\sqrt{\xi_1^2+\xi_2^2+\xi_3^2} \geq \sum_{j\neq r}\xi_j$, which is violated on a large part of Ξ_3, including at $\boldsymbol{\xi}=\left(\frac{1}{3},\frac{1}{3},\frac{1}{3}\right)$.

For an OSSP already known to be monotone causal, another question to ask is about the largest $\delta \geq 0$ for which it is monotone δ-causal. If this $\delta_\star = 0$, the problem can be only treated by Dijkstra's algorithm. Otherwise, $\delta_\star > 0$ specifies the largest bucket width usable in Dial's algorithm. With $m = 2$, the answer to the above question has a simple geometric interpretation.

Observation 2. Maximum Allowable δ when $m = 2$. Suppose that all non-deterministic actions in A_i lead to $\{\boldsymbol{z}_1, \boldsymbol{z}_2\}$ and satisfy (23) for $\delta = 0$. Letting $\boldsymbol{e}_1, \boldsymbol{e}_2 \in A_i$ be the deterministic actions such that $p(\boldsymbol{x}_i, \boldsymbol{e}_j, \boldsymbol{z}_j) = 1$, we will use the notation $C_j = C(\boldsymbol{x}_i, \boldsymbol{e}_j)$.

The largest δ for which the MC conditions are satisfied is then $\delta_\star = \min(\delta_1, \delta_2)$, where

$$\delta_1 = \min_{\boldsymbol{a}\in A_i\setminus\{e_2\}} \frac{C(\boldsymbol{x}_i,\boldsymbol{a}) - p(\boldsymbol{x}_i,\boldsymbol{a},\boldsymbol{z}_2)C_2}{p(\boldsymbol{x}_i,\boldsymbol{a},\boldsymbol{z}_1)} \text{ and } \delta_2 = \min_{\boldsymbol{a}\in A_i\setminus\{e_1\}} \frac{C(\boldsymbol{x}_i,\boldsymbol{a}) - p(\boldsymbol{x}_i,\boldsymbol{a},\boldsymbol{z}_1)C_1}{p(\boldsymbol{x}_i,\boldsymbol{a},\boldsymbol{z}_2)} \quad (24)$$

Furthermore, if $A_i = \Xi_2$ while $C(\boldsymbol{\xi}) = C(\boldsymbol{x}_i, \boldsymbol{\xi})$ is convex on Ξ_2 and differentiable at $\boldsymbol{e}_1 = (1,0)$ and $\boldsymbol{e}_2 = (0,1)$, then

$$\delta_1 = C(\boldsymbol{e}_2) + \frac{\partial C}{\partial \xi_1}(\boldsymbol{e}_2) - \frac{\partial C}{\partial \xi_2}(\boldsymbol{e}_2) \text{ and } \delta_2 = C(\boldsymbol{e}_1) - \frac{\partial C}{\partial \xi_1}(\boldsymbol{e}_1) + \frac{\partial C}{\partial \xi_2}(\boldsymbol{e}_1) \quad (25)$$

Proof: Since (23) already holds with $\delta = 0$ for all non-deterministic $\boldsymbol{a} \in A_i$, the ratios minimized in (24) are always non-negative. Based on assumptions (A1)-(A3), (A5′), the minima are attained and guarantee that (23) is satisfied for each $r = 1,2$ with $\delta = \delta_r$. Part (A) of FIG. 11 illustrates this idea and the restriction lines corresponding to $\delta_\star$ for an example of a monotone causal OSSP with $|A_i| = 5$.

FIG. 11 more particularly illustrates the geometric interpretation of $\delta_\star$ for monotone δ-causality when $m = 2$ for (A): a monotonecausal OSSP where $|A_i|$ is finite and the transition cost function is piecewise-continuous and convex, and (B): a monotone-causal OSSP where $A_i = \Xi_2$ and the transition cost function is convex and continuous, yet not everywhere differentiable. In part (A) of the figure, $|A_i| = 5$ and $\delta_\star$ is easily computed via an application of (24). In part (B) of

the figure, while $K(p)$ is not differentiable at $p = 0.25$, it is differentiable at $p = 0$ and $p = 1$, so the conditions of Observation 2 are still satisfied. Thus, $\delta_\star$ follows immediately from (25).

If $A_i = \Xi_2$, each action can be again identified with its probability distribution $\boldsymbol{\xi} = (1 - p, p)$ for $p \in [0,1]$. If $C(\boldsymbol{\xi})$ is convex then so is $K(p) = C(1 - p, p)$. So, the inequality (23) for $r = 1$ can be rewritten as $K(p) \geq \delta + p(K(1) - \delta)$, which is equivalent to $K'(1) \leq K(1) - \delta$ due to convexity (i.e., this is the requirement for the tangent at $p = 1$ to lie no lower than the δ-dependent restriction line; see the uppermost curve in part (B) of FIG. 11.) Thus, the largest δ that can guarantee this is $\delta_1 = K(1) - K'(1)$. The same argument for $r = 2$ results in $\delta_2 = K(0) + K'(0)$, and the chain rule yields (25).

Theorem 3. Sharpness of MC Condition. Suppose that the condition (20) of Theorem 1 is violated for some specific $\delta \geq 0, \boldsymbol{x}_i \neq \boldsymbol{t}$, non-deterministic action $\boldsymbol{a}$, and $r \in \{1, \ldots, m = |\mathcal{J}(\boldsymbol{a})|\}$. Then there exists a non-monotone δ-causal OSSP in which

1. $\boldsymbol{a} \in A_i$, with $\mathcal{J}(\boldsymbol{a}) = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\} \in X \setminus \{\boldsymbol{x}_i\}$ and $p(\boldsymbol{x}_i, \boldsymbol{a}, \boldsymbol{z}_j) = \xi_j > 0$ for $j = 1, \ldots, m$;
2. $\exists \tilde{\boldsymbol{a}} \in A_i$, with $\mathcal{J}(\tilde{\boldsymbol{a}}) = \mathcal{J}(\boldsymbol{a}) \setminus \{\boldsymbol{z}_r\}, C(\boldsymbol{x}_i, \tilde{\boldsymbol{a}}) = \check{C}(\boldsymbol{\zeta}_r)$, and

$$p(\boldsymbol{x}_i, \tilde{\boldsymbol{a}}, \boldsymbol{z}_j) = \zeta_{r,j} = \begin{cases} 0, & \text{if } j = r \\ \xi_j/(1 - \xi_r), & \text{otherwise} \end{cases} \quad \text{for } j = 1, \ldots, m$$

$$C(\boldsymbol{x}_i, \boldsymbol{a}) < (1 - \xi_r)C(\boldsymbol{x}_i, \tilde{\boldsymbol{a}}) + \xi_r \delta \tag{26}$$

4. $\boldsymbol{a}$ is the unique optimal action at $\boldsymbol{x}_i$;
5. $U(\boldsymbol{x}_i) < U(\boldsymbol{z}_r) + \delta$.

For $\delta = 0$, this implies that any label-setting method will produce a wrong answer for this OSSP. (U_i will be computed incorrectly since $\boldsymbol{z}_r$ will be still in L by the time V_i is last updated.) For $\delta > 0$, this means that a wrong answer would be produced by Dial's algorithm with the bucket width δ.

Proof: To construct the simplest such OSSP example, we will assume that there is only one action $\hat{\boldsymbol{a}}$ available at every $\boldsymbol{x} \notin \{\boldsymbol{x}_i, \boldsymbol{t}\}$, and that $\hat{\boldsymbol{a}}$ leads to an immediate termination deterministically; i.e., $p(\boldsymbol{x}, \hat{\boldsymbol{a}}, \boldsymbol{t}) = 1$. This essentially allows us to prescribe any values to $\boldsymbol{x} \in X \setminus$

$\{\boldsymbol{x}_i, \boldsymbol{t}\}$ since $U(\boldsymbol{x}) = C(\boldsymbol{x}, \hat{\boldsymbol{a}})$. (The resulting OSSP will be thus explicitly causal, but this does not imply the monotone causality.) At $\boldsymbol{x}_i$, we will consider the minimal relevant set of actions $A_i = \{\boldsymbol{a}, \tilde{\boldsymbol{a}}, \boldsymbol{e}_1, \ldots, \boldsymbol{e}_m\}$, where $p(\boldsymbol{x}_i, \boldsymbol{e}_j, \boldsymbol{z}_j) = 1$, with costs $C(\boldsymbol{x}_i, \boldsymbol{a})$ and $C(\boldsymbol{x}_i, \tilde{\boldsymbol{a}})$ already prescribed above and satisfying (26), while all $C(\boldsymbol{x}_i, \boldsymbol{e}_j)$ costs will be prescribed later.

The key quantity of interest in this proof is $B = U(\boldsymbol{z}_r) - \sum_{j=1, j\neq r}^{m} \zeta_{r,j} U(\boldsymbol{z}_j)$. To ensure that $\boldsymbol{a}$ is optimal and $\tilde{\boldsymbol{a}}$ is not, we must have

$$U(\boldsymbol{x}_i) = C(\boldsymbol{x}_i, \boldsymbol{a}) + \sum_{j=1}^{m} \xi_j U(\boldsymbol{z}_j) < C(\boldsymbol{x}_i, \tilde{\boldsymbol{a}}) + \sum_{j=1, j\neq r}^{m} \zeta_{r,j} U(\boldsymbol{z}_j)$$

which is equivalent to $B < \bar{B} = \frac{C(\boldsymbol{x}_i, \tilde{\boldsymbol{a}}) - C(\boldsymbol{x}_i, \boldsymbol{a})}{\xi_r}$. The condition $U(\boldsymbol{x}_i) < U(\boldsymbol{z}_r) + \delta$ is equivalent to $B > \underline{B} = \frac{C(\boldsymbol{x}_i, \boldsymbol{a}) - \delta}{1 - \xi_r}$. These bounds on B can be satisfied simultaneously since (26) is equivalent to $\underline{B} < \bar{B}$. For example, we can choose $U(\boldsymbol{z}_r) = C(\boldsymbol{z}_r, \hat{\boldsymbol{a}}) = 2\delta + (\underline{B} + \bar{B})/2$ and $U(\boldsymbol{z}_j) = C(\boldsymbol{z}_j, \hat{\boldsymbol{a}}) = 2\delta$ for all $j \neq r$. Finally, the non-optimality of all $\boldsymbol{e}_j$'s (i.e., $U(\boldsymbol{x}_i) < C(\boldsymbol{x}_j, \boldsymbol{e}_j) + U(\boldsymbol{z}_j), \forall j \neq r$) is easy to ensure by selecting $C(\boldsymbol{x}_j, \boldsymbol{e}_j)$ to be sufficiently large once all U values are already selected.

To illustrate the connection to causality criteria previously derived for MSSPs, we now turn to homogeneous transition cost functions, which often arise naturally in applications. Recall that $H: \mathbb{R}^m \mapsto \mathbb{R}$ is homogeneous of degree d if $H(\beta \boldsymbol{q}) = |\beta|^d H(\boldsymbol{q})$ holds for all $\beta \in \mathbb{R}, \boldsymbol{q} = (q_1, \ldots., q_m) \in \mathbb{R}^m$. Further, when $H(\boldsymbol{q})$ is smooth, Euler's Homogeneous Function Theorem applies and $H(\boldsymbol{q}) = \frac{1}{d} \sum_{j=1}^{m} q_j \frac{\partial H}{\partial q_j}(\boldsymbol{q}), \forall \boldsymbol{q} \in \mathbb{R}^m$.

Focusing on a single node $\boldsymbol{x}$ and one of its modes $s = \{\boldsymbol{z}_1^s, \ldots, \boldsymbol{z}_{|s|}^s\} \subset X \setminus \{\boldsymbol{x}\}$, in MSSPs one can choose any $\boldsymbol{\xi} \in \Xi_{|s|}$ for the desired probability distribution over s. The corresponding cost $C^s(\boldsymbol{x}, \boldsymbol{\xi})$ can be separately defined for each $\boldsymbol{x}$ and $s \in \mathcal{M}(\boldsymbol{x})$; when they are clear from context, it is convenient to abbreviate the notation to $C(\boldsymbol{\xi})$ and $m = |s|$. An MSSP

is guaranteed to be monotone δ-causal if, in its every mode, $C(\boldsymbol{\xi})$ is continuously differentiable, homogeneous of degree d and satisfies

$$\xi_j > 0 \implies \frac{\partial C}{\partial \xi_j}(\boldsymbol{\xi}) - (d-1)C(\boldsymbol{\xi}) > \delta \geq 0,\ \forall \boldsymbol{\xi} \in \Xi_m, j \in \{1, \ldots, m\} \qquad (27)$$

Note that this covers the example $C(\boldsymbol{\xi}) = \sqrt{\xi_1^2 + \xi_2^2 + \xi_3^2}$ over Ξ_3 already considered previously. This $C(\boldsymbol{\xi})$ is homogeneous of degree one with $\frac{\partial C}{\partial \xi_i}(\boldsymbol{\xi}) = \xi_i / \sqrt{\xi_1^2 + \xi_2^2 + \xi_3^2} > 0$ whenever $\xi_i > 0$. Accordingly, this C is monotone causal (with $\delta = 0$), which can be also concluded from Theorem 1.

Below we show that, for $m = 2$, the condition (27) implies (23).

Theorem 4. Monotone δ-Causality for Homogeneous C($\boldsymbol{\xi}$) when m $= 2$. Suppose $m = 2$, and consider a transition cost function $C(\boldsymbol{\xi})$ which is convex, continuously differentiable, homogeneous of degree d, and satisfying (27). Then it also satisfies

$$C(\boldsymbol{\xi}) \geq \max(C_1\xi_1 + \delta\xi_2, \delta\xi_1 + C_2\xi_2)\ \forall \boldsymbol{\xi} \in \Xi_2$$

where C_1 and C_2 are the shorthands for the cost of deterministic actions $\boldsymbol{e}_1 = (1,0)$ and $\boldsymbol{e}_2 = (0,1)$.

Proof: Define $K: [0,1] \mapsto \mathbb{R}$ by setting $K(p) = C(1-p, p)$, as in FIG. 10. Arguing by contradiction, assume that, for some $\bar{\boldsymbol{\xi}} = (1-\bar{p}, \bar{p})$, we have $C(\bar{\boldsymbol{\xi}}) < \delta\bar{\xi}_1 + C_2\bar{\xi}_2$ or, equivalently, $K(\bar{p}) < \delta + \bar{p}(K(1) - \delta)$. By the convexity of K, this inequality must also hold for all $p \in [\bar{p}, 1)$ and implies $K'(1) > K(1) - \delta$. From the chain rule, this is equivalent to

$$-\frac{\partial C}{\partial \xi_1}(\boldsymbol{e}_2) + \frac{\partial C}{\partial \xi_2}(\boldsymbol{e}_2) > C(\boldsymbol{e}_2) - \delta$$

By Euler's Homogeneous Function Theorem, $dC(\boldsymbol{e}_2) = \frac{\partial C}{\partial \xi_2}(\boldsymbol{e}_2)$; thus,

$$\frac{\partial C}{\partial \xi_1}(\boldsymbol{e}_2) < (d-1)C(\boldsymbol{e}_2) + \delta$$

Since C is continuously differentiable, this inequality also holds for $\boldsymbol{\xi} \in \Xi_2$ sufficiently close to $\boldsymbol{e}_2$, which contradicts (27).

If we instead assume $C(\bar{\boldsymbol{\xi}}) < C_1\bar{\xi}_1 + \delta\bar{\xi}_2$ (or, equivalently, $K(\bar{p}) < K(0) + \bar{p}(\delta - K(0))$), a similar argument leads to $K'(0) < \delta - K(0)$, yielding the same contradiction for $\boldsymbol{\xi} \in \Xi_2$ near $\boldsymbol{e}_1$.

SSPs are also quite useful in approximating solutions of stationary Hamilton-Jacobi-Bellman (HJB) partial differential equations (PDEs), which arise naturally in areas as diverse as robotics, anomaly detection, optimal control, computational geometry, image registration, photolithography, and exploration geophysics. We will focus on a simple example of a time-optimal control problem in an open domain $\Omega \subset \mathbb{R}^m$. With $\boldsymbol{y}(t)$ encoding the position at the time t, the dynamics is prescribed by $\boldsymbol{y}'(t) = f(\boldsymbol{y}(t), \boldsymbol{a}(t))\boldsymbol{a}(t)$, where $\boldsymbol{a}(t) \in S_{m-1} = \{\boldsymbol{a} \in \mathbb{R}^m | |\boldsymbol{a}| = 1\}$ is a unit vector describing our chosen direction of motion, while the speed $f: \Omega \times S_{m-1} \mapsto \mathbb{R}_+$generally depends both on the position and the direction. We will generally assume that f is Lipschitz continuous and will leverage the properties of the corresponding speed profile $\mathcal{V}_f(\boldsymbol{x}) = \{f(\boldsymbol{x}, \boldsymbol{a})\boldsymbol{a} \mid \boldsymbol{a} \in S_{m-1}\}$, which describes all velocities achievable at $\boldsymbol{x}$. It is contained in a spherical shell with the inner and outer radii $F_1(\boldsymbol{x}) = \min_a f(\boldsymbol{x}, \boldsymbol{a})$ and $F_2(\boldsymbol{x}) = \max_a f(\boldsymbol{x}, \boldsymbol{a})$. We will further assume that $\exists F_1, F_2 \in \mathbb{R}$ such that $0 < F_1 \leq F_1(\boldsymbol{x}) \leq F_2(\boldsymbol{x}) < F_2$ holds for all $\boldsymbol{x} \in \Omega$. In addition to the $\boldsymbol{a}(\cdot)$-dependent time-to- $\partial\Omega$, we also incur a Lipschitz continuous exit-penalty $q: \partial\Omega \mapsto \mathbb{R}_{+,0}$. The value function $u(\boldsymbol{x})$ encodes the optimal overall time to exit starting from $\boldsymbol{x}$, formally defined by taking the infimum over all measurable controls $\boldsymbol{a}(\cdot)$. Our assumptions guarantee that u will be Lipschitz-continuous on Ω; moreover, if it is also smooth, a standard argument shows that u must also satisfy the HJB equation

$$\begin{aligned} H(\nabla u(\boldsymbol{x}), \boldsymbol{x}) = \min_{\boldsymbol{a} \in S_{m-1}} \{(\nabla u(\boldsymbol{x}) \cdot \boldsymbol{a}) f(\boldsymbol{x}, \boldsymbol{a})\} + 1 = 0, \ \boldsymbol{x} \in \Omega \\ u(\boldsymbol{x}) = q(\boldsymbol{x}), \ \boldsymbol{x} \in \partial\Omega \end{aligned} \tag{28}$$

More generally, even if u is not smooth, it can be recovered as a unique viscosity solution of this PDE. If we further assume that $\mathcal{V}_f(\boldsymbol{x})$ is convex for all $\boldsymbol{x}$, this guarantees the existence of time-optimal trajectories, which are encoded by the characteristic curves of (28).

A frequently encountered subclass of problems has isotropic dynamics (i.e., with $f(\boldsymbol{x}, \boldsymbol{a}) = f(\boldsymbol{x})$) and allows for further simplification. In this case, $\mathcal{V}_f(\boldsymbol{x})$ is always just a ball, the optimal

direction is known to be $\boldsymbol{a}_* = -\nabla u/|\nabla u|$, the optimal trajectories coincide with the gradient lines of u, and (28) reduces to an Eikonal equation $|\nabla u(\boldsymbol{x})|f(x) = 1$.

Efficient numerical methods for both the Eikonal equations and the general (anisotropic) HJB (28) have been an active area of research in the last 30 years. Many of such methods start with a semi-Lagrangian discretization on a grid or simplicial mesh X over $\bar{\Omega}$. Suppose $\mathcal{S}(\boldsymbol{x})$ is the set of simplexes used to build a computational stencil at $\boldsymbol{x}$, with each $s \in \mathcal{S}(\boldsymbol{x})$ having vertices $\boldsymbol{x}, \boldsymbol{z}_1^s, \ldots, \boldsymbol{z}_m^s$. (For $m = 2$ and a Cartesian grid, two examples of such stencils are shown in FIG. 12.)

FIG. 12 more particularly illustrates two computational stencils based on a uniform Cartesian grid in $\mathbb{R}^2$. The value function update is computed at $\boldsymbol{x}$, and $\tilde{\boldsymbol{x}}_\xi$ is the new position after traveling in a straight line along the chosen direction of motion $\boldsymbol{a}_\xi$ with speed $f(\boldsymbol{x}, \boldsymbol{a}_\xi)$. In part (A) of the figure, the modes are essentially quadrants: $\mathcal{S}(\boldsymbol{x}) = \{(\boldsymbol{x}_1, \boldsymbol{x}_3), (\boldsymbol{x}_3, \boldsymbol{x}_5), (\boldsymbol{x}_5, \boldsymbol{x}_7), (\boldsymbol{x}_7, \boldsymbol{x}_1)\}$. In (B) of the figure, each quadrant is split into two modes: $\mathcal{S}(\boldsymbol{x}) = \{(\boldsymbol{x}_1, \boldsymbol{x}_2), (\boldsymbol{x}_2, \boldsymbol{x}_3), (\boldsymbol{x}_3, \boldsymbol{x}_4), (\boldsymbol{x}_4, \boldsymbol{x}_5), (\boldsymbol{x}_5, \boldsymbol{x}_6), (\boldsymbol{x}_6, \boldsymbol{x}_7), (\boldsymbol{x}_7, \boldsymbol{x}_8), (\boldsymbol{x}_8, \boldsymbol{x}_1)\}$.

For a chosen s, any $\boldsymbol{\xi} \in \Xi_m$ defines a "waypoint" $\tilde{\boldsymbol{x}}_\xi = \sum_j \xi_j \boldsymbol{z}_j^s$ on the opposite face of the simplex and the corresponding direction of motion $\boldsymbol{a}_\xi = (\tilde{\boldsymbol{x}}_\xi - \boldsymbol{x})/|\tilde{\boldsymbol{x}}_\xi - \boldsymbol{x}|$. Assuming we choose that direction and follow it from $\boldsymbol{x}$ to $\tilde{\boldsymbol{x}}_\xi$, the time it takes will be approximately $|\tilde{\boldsymbol{x}}_\xi - \boldsymbol{x}|/f(\boldsymbol{x}, \boldsymbol{a}_\xi)$ and the remaining minimal time from there on can be approximated by linear interpolation as $u(\tilde{\boldsymbol{x}}_\xi) \approx \sum_j \xi_j u(\boldsymbol{z}_j^s)$. Using $U(\boldsymbol{x})$ for the function approximating the true solution u at meshpoints/gridpoints, this yields the following first-order accurate semi-Lagrangian discretization:

$$\begin{aligned} U(\boldsymbol{x}) &= \min_{s \in \mathcal{S}} \min_{\xi \in \Xi_m} \left\{ \frac{|\tilde{\boldsymbol{x}}_\xi - \boldsymbol{x}|}{f(\boldsymbol{x}, \boldsymbol{a}_\xi)} + \sum_{j=1}^{m} \xi_j U(\boldsymbol{z}_j^s) \right\}, \quad \forall \boldsymbol{x} \in X \cap \Omega \\ U(\boldsymbol{x}) &= q(\boldsymbol{x}), \qquad\qquad\qquad\qquad\qquad\qquad\quad \forall \boldsymbol{x} \in X \cap \partial\Omega. \end{aligned} \tag{29}$$

Assuming that $\partial\Omega$ is well resolved by a family of computational grids $\{X^h\}$, it is well-known that $U^h \to u$ uniformly under grid refinement (i.e., as $|X^h| \to +\infty$ with

$\left(\max_{\boldsymbol{x}\in X^h}\max_{s\in S(\boldsymbol{x})}\max_j\left|\boldsymbol{z}_j^s-\boldsymbol{x}\right|\right)\to 0$). In one approach, the same system of equations can be obtained by choosing a suitable controlled Markov process on X. Indeed, taking $C^s(\boldsymbol{x},\boldsymbol{\xi})=\left|\tilde{\boldsymbol{x}}_{\boldsymbol{\xi}}-\boldsymbol{x}\right|/f\left(\boldsymbol{x},\boldsymbol{a}_{\boldsymbol{\xi}}\right)$, it is easy to see that the above is actually an MSSP defined in (16) with modes $s=\{\boldsymbol{z}_1^s,\ldots,\boldsymbol{z}_m^s\}$. The nodes $\boldsymbol{x}\in\partial\Omega$ are assumed to have a single deterministic action only, leading to at the cost $q(\boldsymbol{x})$. In another approach, the same system can be also derived through finite differences, approximating directional derivatives of u for each $\left(\boldsymbol{z}_j^s-\boldsymbol{x}\right)$. Since the discretized system (29) is nonlinear and coupled, it has been traditionally treated with value iterations, but there was always a significant interest in speeding them up, often mirroring the label-correcting and label-setting methods on graphs. Here we focus on the latter, hybrid versions, and parallelization.

The key label-setting ideas were first developed in the isotropic case. If $C^s\left(\boldsymbol{x},\boldsymbol{a}_{\boldsymbol{\xi}}\right)$ is monotone (δ-)causal for each s, the system (29) may be solved using noniterative label-setting methods. It has been shown that any Eikonal discretized in a semi-Lagrangian way on the stencil in part (A) of FIG. 12 (or its m-dimensional version) can be solved by Dijkstra's algorithm. The same result was independently derived in an Eulerian (finite-difference) framework. In the context of isotropic front propagation problems, a Dijkstra-like Fast Marching Method was introduced and generalized to handle higher-order accurate discretizations and arbitrary simplicial meshes both in $\mathbb{R}^m$ and on manifolds. It has also been shown that Dial's algorithm with bucket width $\delta=\frac{h}{F_2\sqrt{2}}$ is applicable to an Eikonal discretized on the stencil in part (B) of FIG. 12 while its 3D version can be treated with the bucket width $\delta=\frac{h}{F_2\sqrt{3}}$. Other Dial-like approaches have been defined. In all these cases, the proofs have heavily relied on specific discretization approaches and stencil choices. The MSSPs have later provided a unifying framework, in which it is easy to prove all related δ-MC conditions based on the properties of $C^s(\boldsymbol{x},\boldsymbol{\xi})$.

Building label-setting methods to correctly solve the system (29) in the anisotropic setting is significantly more challenging since characteristics of the HJB PDE no longer coincide with the gradient lines of u. Aside from the small set of problems with "grid-aligned anisotropy," the

causality is generally not guaranteed on usual stencils of the type shown in FIG. 12. It is possible to handle this problem via Dijkstra-like Ordered Upwind Methods (OUMs), in which the stencil is extended dynamically (at run-time) just enough to ensure the monotone causality. In OUMs, the range of stencil extension is governed by an anisotropy coefficient $\Upsilon(\boldsymbol{x}) = F_2(\boldsymbol{x})/F_1(\boldsymbol{x})$, which is equal to one (no stencil extension) in the isotropic case. The anisotropy coefficient can be used to bound from above the angle between $-\nabla u(\boldsymbol{x})$ and the optimal direction of motion $\boldsymbol{a}_*(\boldsymbol{x})$. This is why the stencil extension proportional to $\Upsilon(\boldsymbol{x})$ is sufficient to ensure the monotone causality. However, later OUMs were also extended to approximate the solutions of HJB PDEs in which both $F_2(\boldsymbol{x})$ and $\Upsilon(\boldsymbol{x})$ might be infinite.

The resulting algorithms are indifferent to the exact speed profile, using the same dynamic stencil extension for all $\mathcal{V}_f(\boldsymbol{x})$ that share the same $F_1(\boldsymbol{x})$ and $F_2(\boldsymbol{x})$. This yields the computational cost of $O(\Upsilon^{m-1} n \log n)$, where $n = |X|$ and $\Upsilon = \max_{\boldsymbol{x}} \Upsilon(\boldsymbol{x})$. Subsequently, a number of papers have aimed to reduce this stencil extension and pre-build a static causal stencil for each $\boldsymbol{x}$ based on the finer properties of $\mathcal{V}_f(\boldsymbol{x})$. A general anisotropic δ-MC criterion has been derived and extended to non-smooth $\mathcal{V}_f(\boldsymbol{x})$. Starting with a uniform Cartesian grid, MC-stencils can be built for elliptic and ellipsoidal speed profiles for $m = 2,3$ with the uniform bound on stencil cardinality $|\mathcal{I}(\boldsymbol{x})|$ independent of Υ. A related approach was also developed to build reduced-cardinality MC-stencils for general (smooth and convex) speed profiles in 2D and 3D. But in all of these cases, the MC criteria were based on the properties of the Hamiltonian H, on implicit conditions for the derivatives of f, or on analytic properties of a function embedding $\mathcal{V}_f$ as a level set. Below we show that the criteria for OSSP developed herein yield surprisingly simple δ-MC conditions for 2D and 3D stencils, using only the basic geometric properties of $\mathcal{V}_f$.

Throughout the remainder of this description, we make use of the following notational conventions. Without loss of generality, we assume that the coordinate system is centered at $\boldsymbol{x} = \mathbf{0}$ and will mostly focus on a single simplex $s \in \mathcal{S}(\boldsymbol{x})$ with vertices $\mathbf{0}, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_m$. We will use $\mathcal{V}_f^s = \{\boldsymbol{v} \in \mathcal{V}_f \mid \boldsymbol{v} = \sum_{j=1}^m \theta_j \boldsymbol{v}_j, \text{ all } \theta_j \geq 0\}$ to denote the part of $\mathcal{V}_f$ falling into this simplex. The

speed, velocity, and cost in the direction $\boldsymbol{a}_{\xi} = \tilde{\boldsymbol{x}}_{\xi}/|\tilde{\boldsymbol{x}}_{\xi}|$ will be denoted by $f(\boldsymbol{\xi}) = f(\boldsymbol{x}, \boldsymbol{a}_{\xi}), \boldsymbol{v}(\boldsymbol{\xi}) = f(\boldsymbol{\xi})\boldsymbol{a}_{\xi}$, and $C(\boldsymbol{\xi}) = C^{s}(\boldsymbol{\xi}) = |\tilde{\boldsymbol{x}}_{\xi}|/f(\boldsymbol{\xi})$ respectively. A similar notation $(f_j, \boldsymbol{v}_j,$ and $C_j = |\boldsymbol{z}_j|/f_j)$ will be also used for the speed, velocity, and cost in each direction $\boldsymbol{z}_j/|\boldsymbol{z}_j|$.

Theorem 5. If $\mathcal{V}_f(\boldsymbol{x})$ is convex then the function $C(\boldsymbol{\xi})$ is convex in every simplex $s \in \mathcal{S}(\boldsymbol{x})$.

Proof: Suppose $\boldsymbol{\eta}_1, \boldsymbol{\eta}_2 \in \Xi_m$ and $\boldsymbol{\phi} = \xi_1\boldsymbol{\eta}_1 + \xi_2\boldsymbol{\eta}_2$ for some $\boldsymbol{\xi} = (\xi_1, \xi_2) \in \Xi_2$. The corresponding waypoints on the simplex $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m)$ are $\tilde{\boldsymbol{x}}_{\eta_i} = \sum_{j=1}^{m} \eta_{i,j}\boldsymbol{z}_j$ for $i = 1,2$ and $\tilde{\boldsymbol{x}}_{\boldsymbol{\phi}} = \xi_1\tilde{\boldsymbol{x}}_{\boldsymbol{\eta}_1} + \xi_2\tilde{\boldsymbol{x}}_{\boldsymbol{\eta}_2}$. The respective directions are $\boldsymbol{a}_{\eta_i} = \tilde{\boldsymbol{x}}_{\boldsymbol{\eta}_i}/|\tilde{\boldsymbol{x}}_{\boldsymbol{\eta}_i}|$ and $\boldsymbol{a}_{\phi} = \tilde{\boldsymbol{x}}_{\phi}/|\tilde{\boldsymbol{x}}_{\phi}|$. The velocity corresponding to $\boldsymbol{a}_{\phi}$ is

$$\boldsymbol{v}(\boldsymbol{\phi}) = f(\boldsymbol{\phi})\frac{\tilde{\boldsymbol{x}}_{\phi}}{|\tilde{\boldsymbol{x}}_{\boldsymbol{\phi}}|} = \frac{f(\boldsymbol{\phi})}{|\tilde{\boldsymbol{x}}_{\boldsymbol{\phi}}|}\left(\xi_1\tilde{\boldsymbol{x}}_{\boldsymbol{\eta}_1} + \xi_2\tilde{\boldsymbol{x}}_{\boldsymbol{\eta}_2}\right) = \theta_1\boldsymbol{v}(\boldsymbol{\eta}_1) + \theta_2\boldsymbol{v}(\boldsymbol{\eta}_2)$$

where $\theta_1, \theta_2 \geq 0$ since $\boldsymbol{v}(\phi)$ belongs to the angular sector formed by $\boldsymbol{v}(\boldsymbol{\eta}_1)$ and $\boldsymbol{v}(\boldsymbol{\eta}_2)$.

Since $\boldsymbol{v}(\boldsymbol{\eta}_i) = f(\boldsymbol{\eta}_i)\frac{\tilde{\boldsymbol{x}}_{\eta_i}}{|\tilde{\boldsymbol{x}}_i|}$, we see that $\frac{f(\phi)\xi_i}{|\tilde{\boldsymbol{x}}_{\boldsymbol{\phi}}|} = \theta_i\frac{f(\boldsymbol{\eta}_i)}{|\tilde{\boldsymbol{x}}_i|}$ for $i = 1, 2$. By definition of C, this is equivalent to

$$\theta_1 = \frac{\xi_1 C(\boldsymbol{\eta}_1)}{C(\boldsymbol{\phi})} \text{ and } \theta_2 = \frac{\xi_2 C(\boldsymbol{\eta}_2)}{C(\boldsymbol{\phi})} \tag{30}$$

The convexity of $\mathcal{V}_f(\boldsymbol{x})$ implies $\theta_1 + \theta_2 \geq 1$ or, equivalently, $\xi_1 C(\boldsymbol{\eta}_1) + \xi_2 C(\boldsymbol{\eta}_2) \geq C(\boldsymbol{\phi})$.

We also note that, if $f(\boldsymbol{\xi})$ is written as a function of direction (i.e., homogeneous of degree zero) then $C(\boldsymbol{\xi}) = |\tilde{\boldsymbol{x}}_{\xi}|/f(\boldsymbol{\xi})$ is homogeneous of degree one, making MC conditions such as (27) applicable. But we will see that Theorems 1 and 2 are more useful in relating MC to geometry.

Aspects of monotone causality in $\mathbb{R}^2$ will now be described in more detail.

Theorem 6. Sufficient Causality Condition on $\mathcal{V}_f$ in $\mathbb{R}^2$. $C(\boldsymbol{\xi})$ is monotone causal on the simplex formed by $(\boldsymbol{x} = \boldsymbol{0}, \boldsymbol{z}_1, \boldsymbol{z}_2)$ if $\mathcal{V}_f^s(\boldsymbol{x})$ is fully contained in a parallelogram with vertices at $\boldsymbol{0}, \boldsymbol{v}_1, \boldsymbol{v}_2,$ and $\boldsymbol{v}_1 + \boldsymbol{v}_2$.

If the above property holds for every $s \in \mathcal{S}(\boldsymbol{x})$ (i.e., if the entire $\mathcal{V}_f(\boldsymbol{x})$ is contained in the union of such parallelograms), then the entire stencil is causal at $\boldsymbol{x}$. If this is the case $\forall \boldsymbol{x} \in X$, then Dijkstra's algorithm is applicable.

Proof: Using the same argument as in Theorem 5 but with $m = 2$ and deterministic $\boldsymbol{\eta}_i = \boldsymbol{e}_i$ for $i = 1,2$, we see that $\boldsymbol{\phi} = \boldsymbol{\xi}$ and $\tilde{\boldsymbol{x}}_{\boldsymbol{\eta}_i} = \boldsymbol{z}_i$. Thus, (30) reduces to

$$\theta_1 = \frac{\xi_1 C_1}{C(\boldsymbol{\xi})} \text{ and } \theta_2 = \frac{\xi_2 C_2}{C(\boldsymbol{\xi})} \tag{31}$$

If $\boldsymbol{v}(\boldsymbol{\xi})$ lies in that parallelogram, this means that $\theta_i \leq 1$ for $i = 1,2$. Combined with (31), this is equivalent to (23) with $\delta = 0$ and Theorem 2 applies.

FIG. 13 shows bounding parallelograms for several speed profiles on the 8-point stencil shown in part (B) of FIG. 12. The speed profile is also depicted in each of the corresponding parts of the figure. The vectors along directions $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are shown, and the velocities $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ along their respective directions $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are also shown. The speed profiles in parts (A), (B) and (C) are as follows: (A) isotropic circular speed profile, (B) octagonal speed profile, and (C) non-convex speed profile. The speed profiles in parts (D) and (E) are tilted elliptical speed profiles, with part (E) more particularly illustrating the result of reducing the standard 8-point stencil used in the previous examples to a 6-point stencil by removing any superfluous nodes $\boldsymbol{z}_j$ which are not crucial to the monotone-causality of the stencil. The speed profile in part (F) is a tilted elliptical speed profile with an aspect ratio twice that of the ellipse in panels (D) and (E). For this aspect ratio, monotone-causality no longer holds on the standard 8-point stencil. The condition in Theorem 6 is violated on two simplexes in $\mathcal{S}(\boldsymbol{x})$.

The standard 8-point stencil is guaranteed to be monotone causal at $\boldsymbol{x}$ for all examples of speed profiles in parts (A) to (D) of FIG. 13. The non-convex example in part (C) of FIG. 13 might seem paradoxical. It is well-known that replacing $\mathcal{V}_f$ with its convex hull does not change the value function since the velocities which are not on the convex hull boundary are never optimal for any ∇u. Thus, this monotone causal OSSP is hardly "occasionally stochastic" since the deterministic transitions will never be chosen. This is the artifact of our simplex-by-simplex

approach of enforcing MC - the non-convexity (and the irrelevance of a large number of directions) is not known if we can only consider the portion of $\mathcal{V}_f$ falling between $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$. Part (E) of FIG. 13 illustrates MC-stencil reduction by merging two pairs of simplexes from (D). An MC-stencil based on only 6 Cartesian grid points can be found for any elliptic $\mathcal{V}_f$, but for ellipses with larger aspect ratio and generic orientation such causal stencils become increasingly non-local. Part (F) of FIG. 13 shows that even a local 8-point stencil is not MC when the ellipse's aspect ratio is sufficiently large.

When $\mathcal{V}_f$ is known to be convex, checking the causality condition (31) becomes a lot simpler, since it can be replaced by a pair of inequalities verified at $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$. The following is an alternative geometric proof of the already known criterion.

Proposition 1. Causality and Tangent Information in $\mathbb{R}^2$. Consider a simplex formed by $(\boldsymbol{x} = \boldsymbol{0}, \boldsymbol{z}_1, \boldsymbol{z}_2)$. Suppose $\mathcal{V}_f$ is convex and smooth at $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, with $\hat{\boldsymbol{n}}_1$ and $\hat{\boldsymbol{n}}_2$ denoting the respective outward normal vectors to $\mathcal{V}_f$. Then $C(\boldsymbol{\xi})$ is guaranteed to be monotone causal within this simplex if

$$\boldsymbol{v}_1 \cdot \hat{\boldsymbol{n}}_2 \geq 0 \text{ and } \boldsymbol{v}_2 \cdot \hat{\boldsymbol{n}}_1 \geq 0 \tag{32}$$

Proof: By convexity, the angular sector of $\mathcal{V}_f$ is contained in a quadrilateral formed by $\boldsymbol{v}_1, \boldsymbol{v}_2$, and two respective tangent lines T_1 and T_2. The inequalities (32) ensure that this quadrilateral is fully contained in the parallelogram with vertices $(\boldsymbol{0}, \boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_1 + \boldsymbol{v}_2)$; see part (A) of FIG. 15.

In the Eikonal case, it is well known that the stencil is MC at $\boldsymbol{x}$ if and only if each angle between $(\boldsymbol{z}_i - \boldsymbol{x})$ and $(\boldsymbol{z}_j - \boldsymbol{x})$ is non-obtuse in each $s \in \mathcal{S}(\boldsymbol{x})$. Since the isotropic $\mathcal{V}_f$ is just a ball, in 2D this is a simple consequence of (32). For smooth convex anisotropic speed profiles in 2D, the above criterion also provides an easy to verify alternative to the "negative-gradient-acuteness" condition.

Aspects of monotone causality in $\mathbb{R}^3$ will now be described in further detail. It might seem intuitive that the 3D version of the MC condition in Theorem 6 should require the portion of $\mathcal{V}_f$

falling into the simplex $(\boldsymbol{x} = \boldsymbol{0}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)$ to lie within a parallelepiped $\Pi = \left\{\sum_{j=1}^{3} \theta_j \boldsymbol{v}_j \mid \theta_j \in [0,1] \text{ for } j = 1,2,3\right\}$. However, this condition turns out to be insufficient. As a motivating example, consider a positive orthant based on unit vectors $\boldsymbol{z}_j = \boldsymbol{e}_j$ with $C_j = 1$ for $j = 1,2,3$ and just 4 available directions of motion: $\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3$, and $\hat{\boldsymbol{\xi}} = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$. In this case, Π is just a unit cube and selecting $f(\hat{\boldsymbol{\xi}}) = \sqrt{3}$ ensures $\left\{\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3, \boldsymbol{v}(\hat{\boldsymbol{\xi}})\right\} \subset \Pi$. The relevant part of $\mathcal{V}_f$ will be a union of 3 triangles with vertices $\boldsymbol{v}(\hat{\boldsymbol{\xi}}), \boldsymbol{v}_i, \boldsymbol{v}_j$. To show that this simplex is not MC for the described $\mathcal{V}_f$, it is enough to find some $U(\boldsymbol{z}_j)$ values such that $\hat{\boldsymbol{\xi}}$ is uniquely optimal but $U(\boldsymbol{x}) < \max_j U(\boldsymbol{z}_j)$. It is easy to check that this is the case with $U(\boldsymbol{z}_1) = 0, U(\boldsymbol{z}_2) = 0.25, U(\boldsymbol{z}_3) = 1.5$, resulting in $U(\boldsymbol{x}) = \frac{11}{12} < U(\boldsymbol{z}_3)$.

The correct sufficient condition is more subtle and uses oblique projections. Before stating it, we need to introduce some additional notation. If (i, j, k) is any permutation of $(1,2,3)$, we define the parallelograms $\Pi^k = \left\{\theta_1 \boldsymbol{v}_i + \theta_2 \boldsymbol{v}_j \mid \theta_1, \theta_2 \in [0,1]\right\}$ and the parts of $\mathcal{V}_f^s$ in those same planes $\mathcal{V}_f^k = \left\{\left(\theta_1 \boldsymbol{v}_i + \theta_2 \boldsymbol{v}_j\right) \in \mathcal{V}_f \mid \theta_1, \theta_2 \geq 0\right\}$. Finally, we also define the corresponding generalized semiinfinite cylinder as $\Psi^k = \left\{\theta_1 \boldsymbol{v}_k + \theta_2 \boldsymbol{v} \mid \boldsymbol{v} \in \mathcal{V}_f^k, \theta_1 \geq 0, \theta_2 \in [0,1]\right\}$.

Theorem 7. Causality Condition on $\mathcal{V}_f$ in $\mathbb{R}^3$. A simplex s is guaranteed to be monotone causal if $\mathcal{V}_f$ is convex, $\mathcal{V}_f^s \subset (\Psi^1 \cap \Psi^2 \cap \Psi^3)$, and $\mathcal{V}_f^r \subset \Pi^r$ for $r = 1,2,3$.

Proof: We will prove that the resulting C satisfies the conditions of Theorem 1 with $\delta = 0$. For deterministic actions $\boldsymbol{e}_i$ this follows directly from (A5′).

If $\boldsymbol{\xi} \in \mathcal{V}_f^1$, then $\mathcal{I}(\boldsymbol{\xi}) = 2$ and both of its oblique projections are deterministic: $\boldsymbol{\zeta}_2 = \boldsymbol{e}_3$ and $\boldsymbol{\zeta}_3 = \boldsymbol{e}_2$. So, (20) is equivalent to $C(\boldsymbol{\xi}) \geq \xi_2 C_2$ and $C(\boldsymbol{\xi}) \geq \xi_3 C_3$. But both of these are guaranteed by $\mathcal{V}_f^1 \in \bar{\Pi}^1$; see Theorem 6. The same argument also covers the cases $\boldsymbol{\xi} \in \mathcal{V}_f^2$ and $\boldsymbol{\xi} \in \mathcal{V}_f^3$.

If $\boldsymbol{\xi} \in \text{int}\,(\Xi_3)$, then $\boldsymbol{v}(\boldsymbol{\xi}) = \sum_{j=1}^{3} \theta_j \boldsymbol{v}_j$. A similar derivation to the one in Theorems 5 and 6 shows

$$\theta_j = \frac{\xi_j C_j}{C(\boldsymbol{\xi})} = \xi_j \frac{f(\boldsymbol{\xi})}{|\tilde{\boldsymbol{x}}_{\boldsymbol{\xi}}|} \frac{|\boldsymbol{z}_j|}{f_j} > 0 \;\forall j \tag{33}$$

So, for any $r \in \{1,2,3\}$, we have the oblique projection $\boldsymbol{\zeta}_r$, the corresponding waypoint $\tilde{\boldsymbol{x}}_{\zeta_r} = \sum_{j\neq r} \zeta_{r,j}\boldsymbol{z}_j$, and velocity $\boldsymbol{v}(\boldsymbol{\zeta}_r) = \sum_{j\neq r} \frac{\zeta_{r,j}\boldsymbol{z}_j}{|\tilde{\boldsymbol{x}}_{\zeta_r}|} f(\boldsymbol{\zeta}_r)$. Thus,

$$\begin{aligned}\boldsymbol{v}(\boldsymbol{\xi}) &= \theta_r \boldsymbol{v}_r + \sum_{j\neq r} \theta_j \boldsymbol{v}_j = \theta_r \boldsymbol{v}_r + \frac{f(\boldsymbol{\xi})}{|\tilde{\boldsymbol{x}}_\xi|} \sum_{j\neq r} \xi_j \frac{|\boldsymbol{z}_j|}{f_j} \boldsymbol{v}_j = \theta_r \boldsymbol{v}_r + \frac{f(\boldsymbol{\xi})}{|\tilde{\boldsymbol{x}}_\xi|} \sum_{j\neq r} \xi_j \boldsymbol{z}_j \\ &= \theta_r \boldsymbol{v}_r + (1-\xi_r)\frac{f(\boldsymbol{\xi})}{|\tilde{\boldsymbol{x}}_\xi|} \sum_{j\neq r} \zeta_{r,j}\boldsymbol{z}_j = \theta_r \boldsymbol{v}_r + \left[(1-\xi_r)\frac{f(\boldsymbol{\xi})}{|\tilde{\boldsymbol{x}}_\xi|}\frac{|\tilde{\boldsymbol{x}}_{\zeta_r}|}{f(\boldsymbol{\zeta}_r)}\right]\boldsymbol{v}(\boldsymbol{\zeta}_r), \\ &= \theta_r \boldsymbol{v}_r + \left[(1-\xi_r)\frac{C(\boldsymbol{\zeta}_r)}{C(\boldsymbol{\xi})}\right]\boldsymbol{v}(\boldsymbol{\zeta}_r)\end{aligned}$$

where the sequence of equalities uses (33), the definition of $\boldsymbol{v}_j$, the definition of $\boldsymbol{\zeta}_r$, the representation of $\boldsymbol{v}(\boldsymbol{\zeta}_r)$, and the formula for C. Since $\mathcal{V}_f^s \subset \Psi^r$, the coefficient in front of $\boldsymbol{v}(\boldsymbol{\zeta}_r)$ has to be less than or equal to 1. By Theorem 5 , the convexity of $\mathcal{V}_f^s$ guarantees $\check{C}(\boldsymbol{\zeta}_r) = C(\boldsymbol{\zeta}_r)$, and thus $C(\boldsymbol{\xi}) \geq (1-\xi_r)\check{C}(\boldsymbol{\zeta}_r)$.

Another interesting question is the amount of anisotropic distortion compatible with the monotone causality of a standard Cartesian stencil in $\mathbb{R}^m$. For $m = 2$ and the stencil in part (A) of FIG. 12, $\mathcal{V}_f(\boldsymbol{x})$ simply needs to lie in a rectangle whose vertices are at $(-f_W, f_N), (f_E, f_N), (f_E, -f_S), (-f_W, -f_S)$. Here the subscripts indicate directions of motion and the coordinate system is centered at $\boldsymbol{x}$.

But as we already saw, with $m = 3$ and the standard 6 point stencil, the answer is more complicated and depends on $f(\boldsymbol{a})$ in all directions parallel to coordinate planes (comprising $\mathcal{V}_f^k$'s). For the example mentioned above, $(\Psi^1 \cap \Psi^2 \cap \Psi^3) = \{(v_1, v_2, v_3) \in \mathbb{R}^3_{+,0} \mid \max(v_1 + v_2, v_2 + v_3, v_1 + v_3) \leq 1\}$. Thus, Theorem 7 guarantees that this simplex is MC only if $f(\hat{\boldsymbol{\xi}}) \leq \sqrt{3}/2$. For another interesting 3D example, consider a speed profile that is known to be isotropic (i.e., $f(\boldsymbol{a}) = F$) but only for $\boldsymbol{a}$'s that are parallel to coordinate planes (x_1, x_2), (x_2, x_3), and (x_1, x_3). Theorem 7 guarantees that the standard 6-point stencil will be monotone causal if $\mathcal{V}_f(\boldsymbol{x})$ lies within a tricylinder $(\Psi^1 \cap \Psi^2 \cap \Psi^3) = \{(v_1, v_2, v_3) \in \mathbb{R}^3 \mid \max(v_1^2 + v_2^2, v_2^2 + v_3^2, v_1^2 + v_3^2) \leq F^2\}$.

We also note that in principle the argument in Theorem 7 can be recursively extended to any $m > 3$. The second half of the proof can be adopted for any m, but the conditions on $\partial\Xi_m$ will be more complicated. For example, for $m = 4$, the current theorem provides conditions for all 3-dimensional faces of Ξ_4. We do not further describe this extension because the criteria would not be as geometrically suggestive and because grid discretizations of (28) are most commonly used when $m = 2$ or 3.

Similarly to the 2D case, for a smooth convex speed profile, the conditions of Theorem 7 replaced by criteria verified on the facets of the simplex only. The following Proposition 2 is a 3D analog of Proposition 1. It also provides an explicit and easier to verify version of the 3D causality condition derived.

Let $\hat{\boldsymbol{n}}(\tilde{\boldsymbol{v}})$ denote an outward pointing normal vector to $\mathcal{V}_f$ at any $\tilde{\boldsymbol{v}} \in \mathcal{V}_f^r$. The support hyperplane to $\mathcal{V}_f$ at $\tilde{\boldsymbol{v}}$ is specified by $(\boldsymbol{v} - \tilde{\boldsymbol{v}}) \cdot \hat{\boldsymbol{n}}(\tilde{\boldsymbol{v}}) = 0$. We define the r-th "tangential envelope" of $\mathcal{V}_f^s$ as

$$\Phi^r = \left\{\boldsymbol{v} = \theta_1\boldsymbol{v}_1 + \theta_2\boldsymbol{v}_2 + \theta_3\boldsymbol{v}_3 \mid \theta_i \geq 0 \text{ for } i = 1,2,3 \text{ and } (\boldsymbol{v} - \tilde{\boldsymbol{v}}) \cdot \hat{\boldsymbol{n}}(\tilde{\boldsymbol{v}}) \leq 0 \text{ for all } \tilde{\boldsymbol{v}} \in \mathcal{V}_f^r\right\}$$

Proposition 2. Normal MC verification in 3D. Suppose that the speed profile $\mathcal{V}_f^s$ is convex and smooth. A simplex $s \in \mathcal{S}(\boldsymbol{x})$ with vertices $(\boldsymbol{x} = \boldsymbol{0}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)$ and corresponding velocities $(\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3)$ is guaranteed to be monotone causal if

1. $\boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}(\boldsymbol{v}_j) \geq 0, \ \forall i, j \in \{1,2,3\}$;
2. $\boldsymbol{v}_r \cdot \hat{\boldsymbol{n}}(\tilde{\boldsymbol{v}}) \geq 0, \ \forall r \in \{1,2,3\}, \tilde{\boldsymbol{v}} \in \mathcal{V}_f^r$.

Proof: By Proposition 1, the first condition implies that $\mathcal{V}_f^r \subset \Pi^r$ for all $r \in \{1,2,3\}$.

We note that $\mathcal{V}_f^s \subset \Phi^r$ due to convexity of $\mathcal{V}_f$ and that $\partial\Phi^r$ is tangential to $\mathcal{V}_f$ along the $\mathcal{V}_f^r \subset \partial\Psi^r$. Thus, $\mathcal{V}_f^s \subset \Psi^r$ if and only if $\Phi^r \subset \Psi^r$, which is ensured by the second condition above. This shows that s satisfies the criteria listed in Theorem 7.

Aspects of monotone δ-Causality in $\mathbb{R}^2$ will now be further described. The MC conditions derived above are fully based on $\mathcal{V}_f$ and the set of directions represented in a particular simplex;

i.e., the distances $|\boldsymbol{z}_i - \boldsymbol{x}|$ were irrelevant in specifying the MC-constrained set for $\mathcal{V}_f^s$. This is not the case for monotone δ-causality as we show below for $m = 2$.

Theorem 8. δ-Causality Condition on $\mathcal{V}_f$ in $\mathbb{R}^2$. Suppose that a simplex $s \in \mathcal{S}(\boldsymbol{x})$ formed by vertices $\boldsymbol{x} = \boldsymbol{0}, \boldsymbol{z}_1$, and $\boldsymbol{z}_2$ is monotone causal for a specific $\mathcal{V}_f(\boldsymbol{x})$ and that $0 < \delta \leq \min(C_1 = \frac{|z_1|}{f_1}, C_2 = \frac{|z_2|}{f_2})$. This simplex is monotone δ-causal if $\mathcal{V}_f^s$ is contained in a quadrilateral with vertices at $\boldsymbol{x} = \boldsymbol{0}, \boldsymbol{v}_1, \boldsymbol{v}_2$, and $\boldsymbol{w}(\delta) = \theta_1^{\#}(\delta)\boldsymbol{v}_1 + \theta_2^{\#}(\delta)\boldsymbol{v}_2$, where

$$\theta_1^{\#}(\delta) = \frac{C_1(C_2 - \delta)}{C_1 C_2 - \delta^2} \text{ and } \theta_2^{\#}(\delta) = \frac{C_2(C_1 - \delta)}{C_1 C_2 - \delta^2} \tag{34}$$

Proof: For every $\boldsymbol{\xi} \in \Xi_2$, the corresponding velocity is $\boldsymbol{v}(\boldsymbol{\xi}) = \theta_1 \boldsymbol{v}_1 + \theta_2 \boldsymbol{v}_2$, where $\theta_1, \theta_2 \geq 0$. By Theorem 1, monotone δ-causality of $C(\boldsymbol{\xi})$ is guaranteed if

$$C(\boldsymbol{\xi}) \geq \xi_1 C_1 + \xi_2 \delta \text{ and } C(\boldsymbol{\xi}) \geq \xi_1 \delta + \xi_2 C_2$$

Dividing through by $C(\boldsymbol{\xi})$ and using (31), we see that the equivalent conditions are

$$1 \geq \theta_1 + \frac{\theta_2 \delta}{C_2} \text{ and } 1 \geq \theta_2 + \frac{\theta_1 \delta}{C_1}$$

Each of these inequalities specifies a half-plane where $\boldsymbol{v}(\boldsymbol{\xi})$ is allowed to lie, with two restriction lines L_1 (passing through the points $\boldsymbol{v}_1$ and $\frac{C_2}{\delta}\boldsymbol{v}_2 = \frac{1}{\delta}\boldsymbol{z}_2$) and L_2 (passing through the points $\boldsymbol{v}_2$ and $\frac{C_1}{\delta}\boldsymbol{v}_1 = \frac{1}{\delta}\boldsymbol{z}_1$). Solving for the intersection point $\boldsymbol{w}(\delta)$, we obtain (34).

It is easy to see that this quadrilateral is fully contained in the MC parallelogram Π and that $\boldsymbol{w}(\delta) \to (\boldsymbol{v}_1 + \boldsymbol{v}_2)$ as $\delta \to 0$. If $\delta = \min(C_1, C_2)$, one of the restriction lines coincides with the straight line connecting $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, and the δ-MC region becomes a triangle $(\boldsymbol{0}, \boldsymbol{v}_1, \boldsymbol{v}_2)$, ensuring that either $\frac{z_1}{|z_1|}$ or $\frac{z_2}{|z_2|}$ will be always optimal.

Similarly to the MC setting, if $C^s(\boldsymbol{\xi})$ is δ-causal for all $\boldsymbol{s} \in \mathcal{S}(\boldsymbol{x})$, then the entire stencil is δ-MC at $\boldsymbol{x}$ and the speed profile $\mathcal{V}_f(\boldsymbol{x})$ is inscribed within the union of bounding δ-MC quadrilaterals which encompass all possible directions of motion. FIG. 14 displays this result for two speed profiles and the 8-point stencil depicted in part (B) of FIG. 12.

FIG. 14 more specifically shows bounding δ-MC "sunflowers" for the 8-point stencil (grid spacing $h = 1$ for $|\boldsymbol{z}_1| = h, |\boldsymbol{z}_2| = h\sqrt{2}$) found by fixing $f_i(\boldsymbol{x}) = f(\boldsymbol{x}, \boldsymbol{a}_{\boldsymbol{e}_i})$ for 8 directions $(\boldsymbol{x}_i - \boldsymbol{x})$ and varying δ. The f_i are chosen to match an isotropic speed profile in part (A) and an anisotropic elliptical speed profile in part (B). The direction vectors corresponding to $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are shown, as are the velocity vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$. The bounding sunflower corresponding to the maximum $\Delta(\boldsymbol{x})$ applicable for this $(\mathcal{V}_f$, stencil) combination is also shown.

A subsequent question relates to the largest δ such that a specific simplex is δ-MC. The answer can be found from Observation 2, but the geometric derivation provided below is more natural.

Theorem 9. Max δ for Smooth, Strictly Convex $\mathcal{V}_f$ in $\mathbb{R}^2$. Suppose that $m = 2$ and $\mathcal{V}_f(\boldsymbol{x})$ is smooth and strictly convex. Suppose also that a simplex $s \in \mathcal{S}(\boldsymbol{x})$ formed by $\boldsymbol{x} = \boldsymbol{0}, \boldsymbol{z}_1$, and $\boldsymbol{z}_2$ is *MC* for this $\mathcal{V}_f(\boldsymbol{x})$. The maximal δ for which this simplex is δ-MC can be computed as $\Delta(\boldsymbol{x}, s) = \min\{\delta_1, \delta_2\}$, where

$$\delta_1 = C_2 \frac{\boldsymbol{v}_2 \cdot \hat{\boldsymbol{n}}_1}{\boldsymbol{v}_1 \cdot \hat{\boldsymbol{n}}_1} \text{ and } \delta_2 = C_1 \frac{\boldsymbol{v}_1 \cdot \hat{\boldsymbol{n}}_2}{\boldsymbol{v}_2 \cdot \hat{\boldsymbol{n}}_2} \tag{35}$$

and $\hat{\boldsymbol{n}}_i$'s denote normal vectors to $\mathcal{V}_f(\boldsymbol{x})$ at respective $\boldsymbol{v}_i$'s.

Proof: Starting with the MC-parallelogram, gradually increasing δ moves the lines L_1, L_2 (described in Theorem 8) and shrinks the δ-MC quadrilateral. For a convex and smooth speed profile, this process can be continued until one of the L_i's becomes tangential to $\mathcal{V}_f(\boldsymbol{x})$ at $\boldsymbol{v}_i$. Suppose this happens with $i = 1$ first, when $\delta = \delta_1$. The line L_1 passes through the points $\boldsymbol{v}_1$ and $\frac{C_2}{\delta_1}\boldsymbol{v}_2$. Tangentiality of L_1 means that $\left(\frac{C_2}{\delta_1}\boldsymbol{v}_2 - \boldsymbol{v}_1\right) \cdot \hat{\boldsymbol{n}}_1 = 0$, which is equivalent to the expression in (35). The case where L_2 becomes tangential first similarly yields the expression for δ_2.

If either of these lines is already tangential to $\mathcal{V}_f(\boldsymbol{x})$ when $\delta = 0$, this means that the simplex is MC but not δ-MC for any $\delta > 0$; see part (B) of FIG. 15. To find the largest δ such that the entire stencil is δ-MC at $\boldsymbol{x}$, we define $\Delta(\boldsymbol{x}) = \min_{s \in \mathcal{S}(\boldsymbol{x})} \Delta(\boldsymbol{x}, s)$. Similarly, let $\Delta(X) = \min_{\boldsymbol{x} \in X} \Delta(\boldsymbol{x})$. For an MC discretization, if $\Delta(X) > 0$ then Dial's algorithm with the bucket width

$\Delta(X)$ is also applicable. We note that, since $C_i = \frac{|z_i|}{f_i}$, for any fixed speed function $f(\boldsymbol{x}, \boldsymbol{a})$, the allowed bucket width $\Delta(X)$ will be shrinking under grid refinement.

FIG. 15 more particularly shows simplex s and causality conditions on $\mathcal{V}_f^s$ for three different convex speed profiles. Tangent lines to $\mathcal{V}_f$ at are labeled T_1 and T_2 respectively and plotted using dashed arrows. $\boldsymbol{v}_1, \boldsymbol{v}_2$, and α are equivalent in all three examples. $\mathcal{V}_f^s$ in (A) is satisfies the monotone causality conditions posed in Theorem 6 and the monotone δ-causality conditions posed in Theorem 9. The corresponding sides of the resulting bounding quadrilateral are shown in solid lines. In this case, L_2 (shown in dotted lines) is tangential to $\mathcal{V}_f^s$, and $C(\boldsymbol{\xi})$ is monotone δ-causal in this simplex with $\delta = \delta_2$. In (B), $\mathcal{V}_f^s$ only satisfies the conditions of Theorem 6 and $C(\boldsymbol{\xi})$ cannot be monotone δ-causal, as the side of the parallelogram opposite to $\boldsymbol{v}_1$ is tangential to $\mathcal{V}_f$. Finally in (C), $\mathcal{V}_f^s$ violates the $\boldsymbol{v}_1 \cdot \hat{\boldsymbol{n}}_2 \geq 0$ condition in Proposition 1 and $C(\boldsymbol{\xi})$ cannot be monotone causal on this simplex.

Theorem 9 also provides a simpler derivation for some of the results previously found for specific 2D-stencils. For example, in the isotropic case $\mathcal{V}_f$ is just a ball; so, $\boldsymbol{v}_i$ and $\hat{\boldsymbol{n}}_i$ are parallel. If β is an angle between $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$, (35) reduces to $\delta_1 = C_2 \cos\beta = \frac{|z_2|}{f(\boldsymbol{x})}\cos\beta$ and $\delta_2 = C_1 \cos\beta = \frac{|z_1|}{f(\boldsymbol{x})}\cos\beta$. In the 8-point stencil of part (B) of FIG. 12, $|\boldsymbol{z}_1| = h$, $|\boldsymbol{z}_2| = h\sqrt{2}$, and $\beta = \pi/4$. Thus, $\delta_1 = \frac{h}{f(\boldsymbol{x})}, \delta_2 = \frac{h}{\sqrt{2}f(\boldsymbol{x})}$, and $\Delta(\boldsymbol{x}) = \delta_2$. As a result, $\Delta(X) = \frac{h}{F_2\sqrt{2}}$, which matches a known derived bucket width. General 2D stencils and triangulated meshes can be treated similarly, using upper bounds on $|\boldsymbol{z}_i|$'s and β.

As indicated previously, routing modules in autonomous vehicles have multiple interacting levels of planning and control which work together to bring the vehicle from its starting destination s to a predetermined destination $\boldsymbol{t}$. Typically, the routing module first produces a Strategic Plan (SP), which is a set of deterministic, cost-minimizing, turn-by-turn directions from s to $\boldsymbol{t}$, where costs illustratively include, for example, time, fuel consumption, toll charges, passengers' comfort or a combination of these factors. The SP is usually precomputed based on traffic and weather

conditions using some label-setting method to identify the deterministic cheapest path on a graph representing the road network. After that, the system may determine a Tactical Plan (TP) outlining the planned timing and location of lane switch maneuvers (LSMs) necessary to execute the SP. Finally, the SP and TP are communicated to the Operational Control (OC) module which is responsible for planning the vehicle's continuous trajectory, steering, and dynamically accelerating/decelerating in order to follow the planned route. OC occurs in real time and is heavily constrained by safety overrides based on the actual observed vehicles. If a planned LSM fails, this often makes it necessary to follow a suboptimal path and repeatedly recompute both SP and TP.

The overall performance can be clearly improved by directly modeling the probabilistic nature of LSMs and minimizing the expected cost to target. Such an approach can be characterized as computing a hybrid STP, also referred to herein as a combined stochastic STP. The routing problem can be cast as an SSP in which the states correspond to nodes within a lane-level road network such as the examples in part (A) of FIG. 16 and part (B) of FIG. 7. Using the terminology herein, such a model is an example of an OSSP in which $|s| \leq 2$ for all $s \in \mathcal{S}(\boldsymbol{x}), \boldsymbol{x} \in X$. Deterministic modes (with $|s| = 1$) are used to model the actions that normally don't fail (e.g., moving forward wherever lane changes are not allowed or available, or turning at an intersection without any lane changes). On the other hand, a possible switch to each specific adjacent lane is encoded using a separate mode with $|s| = 2$. Each such mode may be associated with exactly three LSM actions: the deterministic stay in lane, the deterministic "forced" LSMs, and one tentative LSM which succeeds with probability $\tilde{p} \in (0,1)$. Part (B) of FIG. 16 shows an example of the LSMs available to the vehicle at $\boldsymbol{x}$ in mode $s_1 = \{\boldsymbol{x}_j, \boldsymbol{x}_k\}$. The vehicle may deterministically transition to $\boldsymbol{x}_j$, deterministically transition to $\boldsymbol{x}_k$, or use a stochastic transition leading to $\boldsymbol{x}_k$ or $\boldsymbol{x}_j$ with respective probabilities $\tilde{p}$ and $(1 - \tilde{p})$.

FIG. 16 in part (A) thereof shows an example lane-level road network representation of a three-lane highway. Lanes are discretized into cells of length D meters, and each node marks the center of a cell. The vehicle travels from the starting point s to the destination $\boldsymbol{t}$ via a series of

planned LSMs. Part (B) of FIG. 16 illustrates actions available at node $\boldsymbol{x}$ in mode $s_1 = \{\boldsymbol{x}_j, \boldsymbol{x}_k\} \in \mathcal{S}(\boldsymbol{x})$. The vehicle may continue driving in the current lane and directly transition to $\boldsymbol{x}_j$ (solid horizontal arrow), forcefully switch lanes and directly transition to $\boldsymbol{x}_k$ (solid downward arrow), or attempt a tentative lane change (dashed downward arrow). The other mode available at $\boldsymbol{x}$ is $s_2 = \{\boldsymbol{x}_j, \boldsymbol{x}_i\}$, encoding a possible switch to another lane.

The vehicle's transitions are penalized as

$$K(0) = g(\boldsymbol{x}),\ K(\tilde{p}) = g(\boldsymbol{x}) + \tilde{p}g_1,\ \text{and}\ K(1) = g(\boldsymbol{x}) + g_1 + (1-\tilde{p})g_2 \tag{36}$$

where $g(\boldsymbol{x})$ is the cost of traveling to the next node in the same lane, $g_1 > 0$ is the additional cost incurred if a tentative LSM is successful, and $g_2 > g_1$ is a large penalty incurred for forcing the lane switch. For an unforced/tentative LSM, the transition probability is modeled as $\tilde{p} = 1 - e^{-\alpha D}$ where D is the distance to the next node in the same lane, and α is the success rate determined from local traffic data. The cost $K(p)$ in (36) is monotonically increasing in p based on the assumption that the instantaneous cost of changing lanes will always be higher than staying in the current one.

It is known that a Dijkstra-like algorithm is applicable to this SSP provided

$$g(\boldsymbol{x}) \geq \alpha D g_2 \tag{37}$$

The same result also follows as a direct application of Theorem 2 , which for $m = 2$ is equivalent to Theorem 1. Since $K(p)$ is monotonically increasing on $[0,1]$, we only need to check the inequality (23) for $r = 1$, which requires

$$K(\tilde{p}) = g(\boldsymbol{x}) + \tilde{p}g_1 \geq \tilde{p}[g(\boldsymbol{x}) + g_1 + (1-\tilde{p})g_2] \tag{38}$$

or, equivalently, $g(\boldsymbol{x}) \geq \tilde{p}g_2$. By the convexity of exponentials, $\alpha D \geq \tilde{p}$, and so (37) implies (38). We note that, outside of MSSPs, this is the first known example of a monotone causal OSSP. Moreover, if (37) holds, one can use Observation 2 to show that this OSSP is also monotone δ-causal with $\Delta = \min_x g(\boldsymbol{x}) - \tilde{p}g_2$.

The theory developed above significantly widens the scope of STP models that can be treated with label-setting methods. For example, we can now similarly account for a larger number of tentative LSM actions and different cost models with monotonically increasing and convex

$K(p)$. The OSSP framework also allows for a useful reinterpretations of the lane change success probability in terms of the lane change urgency, which reflects the degree of controller's willingness to alter the vehicle's velocity to ensure a successful lane change. It might seem that the urgency of an LSM should be fully determined by the vehicle's distance to its next preplanned turn, which has to be executed from the lane we are trying to switch to. But if similarly cheap alternate routes are available, missing a planned turn or highway exit may only have a very minor effect on the vehicle's total time (or other relevant expenditures) up to $\boldsymbol{t}$. Thus, an accurate assessment of urgency should take into account the global road network structure and traffic patterns.

In reality, drivers gradually increase or decrease LSM urgency in response to the local traffic conditions or nearby infrastructure. Thus, urgency exists on a continuous spectrum in which the stay-in-lane maneuver corresponds to no urgency ($p = 0$), the forced lane change maneuver corresponds to full urgency ($p = 1$), and all LSMs with stochastic outcomes correspond to intermediate urgency levels ($p \in (0,1)$).

A simple generalization of an example model is to allow for more than one (but finitely many) intermediate LSMs, which might be executed progressively at $\boldsymbol{x}$. The cost of a forced LSM in (37) reveals that upon selecting the forced lane change maneuver, the vehicle attempts to switch lanes at the intermediate level first, and then only forces the LSM should that initial attempt fail. As the LSM urgency increases between the first attempt and second attempt, the LSMs themselves in this framework may also be described as escalating.

To be more precise, suppose there are $L + 1$ available LSMs with associated success probabilities $p_\ell \in \mathcal{P}$ such that $0 = p_0 < \cdots < p_\ell < \cdots < p_L = 1$. The stay in lane cost is $K(p_0) = g(\boldsymbol{x})$, and maneuvers with $p_\ell > 0$ are subject to additional penalties $Y_\ell > 0$ which are monotonically increasing in ℓ and incurred upon a successful lane change at the corresponding urgency level. If these actions are executed in escalating manner, their expected cost has the form

$$\begin{aligned} K(p_\ell) &= p_{\ell-1}K(p_{\ell-1}) + (1 - p_{\ell-1})[K(p_{\ell-1}) + Y_\ell] \\ &= K(p_{\ell-1}) + (1 - p_{\ell-1})Y_\ell, \ \ell = 1, \ldots, L \end{aligned} \tag{39}$$

The convexity of K is ensured when $\frac{K(p_L)-K(p_\ell)}{1-p_\ell} \geq \frac{K(p_L)-K(p_{\ell-1})}{1-p_{\ell-1}}$ holds for all $\ell = 1, \ldots L-1$. By direct application of Theorem 2, the OSSP will be monotone (δ-)causal when

$$K(p_{\ell-1}) + (1-p_{\ell-1})Y_\ell \geq p_\ell[K(p_{L-1}) + (1-p_{L-1})Y_L] + (1-p_\ell)\delta,\ \ell = 1, \ldots, L-1$$

We note that an exact model can be recovered by taking $L = 2, p_1 = \tilde{p}, Y_1 = \tilde{p}g_1$, and $Y_2 = g_2 + g_1$.

Whether or not the above progressive escalation framework is realistic is debatable: a lane change may take six or more seconds to complete and attempting multiple consecutive LSMs before reaching the successor node in the same lane might be impossible. But our framework can be similarly used for any increasing sequence of costs $K_\ell = K(p_\ell)$. Due to monotonicity of K, the inequality (23) has to be enforced for $r = 1$ only and will hold automatically for $r = 2$. The lane switch mode will be (δ-)monotone causal as long as

$$K(p) \geq pK(1) + (1-p)\delta \tag{40}$$

holds for every attainable $p < 1$. This criterion works with models of finitely many urgency levels ($p \in \{p_0, \ldots, p_L\}$) and also for models with a continuous spectrum of urgency levels ($p \in [0,1]$); see also FIG. 10. The maximum allowable δ can be similarly computed using Observation 2.

We also mention two other LSM-cost models illustrated by the numerical experiments described below. For the continuous urgency spectrum, one of the simplest (monotone, convex) cost models is quadratic

$$K(p) = \beta_s(\boldsymbol{x})p^2 + \gamma_s(\boldsymbol{x})$$

where $\beta_s(\boldsymbol{x}), \gamma_s(\boldsymbol{x}) > 0$ are constants which may reflect traffic conditions near $\boldsymbol{x}$ or learned driver behavior. Using this $K(p)$ in (40) and simplifying, we see that this lane switch mode will be δ-monotone causal as long as $\beta_s(\boldsymbol{x}) + \delta \leq \gamma_s(\boldsymbol{x})$. Thus, Dijkstra's algorithm will be applicable if $\beta_s(\boldsymbol{x}) \leq \gamma_s(\boldsymbol{x})$ for all $\boldsymbol{x} \in X, s \in \mathcal{S}(\boldsymbol{x})$. If this inequality is strict, we can also use Dial's algorithm with buckets of width $\Delta(X) = \min_{x \in X} \min_{s \in \mathcal{S}(x)} (\gamma_s(\boldsymbol{x}) - \beta_s(\boldsymbol{x}))$.

Alternatively, we can start with an (increasing, convex) sequence of urgency levels and associated costs (p_ℓ, K_ℓ) learned from local traffic data or from the vehicle's performance analytics, and then extend it to a continuous urgency spectrum $p \in [0,1]$ by defining $K(p)$ through a suitable interpolation.

For example, when $L = 2$, a natural choice of interpolant is a quadratic Rational Bézier Curve (RBC) with suitable control points and weights chosen to ensure that the resulting smooth approximation to $K(p)$ is monotone, convex, and monotone (δ-)causal by Theorem 2. The RBC is described parametrically as $\boldsymbol{B}(t) = (p(t), K(t))$, and the entire curve sits within the convex hull of a given set of $L + 1$ control points. The curve's shape is governed by those control points and a set of corresponding weights $\{\omega_0, \ldots, \omega_L\}$ which determine how much the curve bends toward each control point. Assuming that the original three-point (p_ℓ, K_ℓ) sequence is δ-MC, we can take the first control point to be $(0, K(0))$ and the last control point to be $(1, K(1))$, with the remaining control point chosen to lie at the intersection of the lines $K = K_0$ and $K = pK_2 + (1-p)\delta$ to ensure that the resulting curve is both monotonically increasing and monotone (δ-)causal by Theorem 2. Since RBCs are guaranteed to pass through their first and last control points, we can set $\omega_0 = \omega_2 = 1$, and we construct a system of two linear equations (one for $K(t) = K_1$ and one for $p(t) = p_1$) to compute t and the value of ω_1 which ensures that the RBC also passes through (p_1, K_1) as required. We note that, for this choice of control points, Theorem 2 guarantees that this OSSP will be monotone causal though not δ-MC for any $\delta > 0$.

Evaluating the usefulness (and the optimal urgency) of merging to an adjacent lane is the essential part of computing the value function. For example, starting from the point $\boldsymbol{x}$ in part (B) of FIG. 16, we need to find p that minimizes $\big(K(p) + (1-p)U_j + pU_k\big)$. For a continuous spectrum of p values, the availability of $K'(p)$ can be used to perform this minimization either analytically (e.g., with the quadratic K model) or semi-analytically (e.g., with the RBC-based K model described above) to ensure the computational efficiency.

A number of numerical examples, similar to those described elsewhere herein, will now be further described. We use a Dijkstra-like algorithm to determine the optimal STP for a vehicle

traveling through several different lane-level road networks. In all examples, $D = 10$ meters, and the direction of traffic flow between successor nodes is indicated by arrows. When visualizing optimal STPs, we use solid arrows to indicate when a deterministic LSM is optimal and dashed arrows to indicate when the optimal LSM is deterministic. The color of such arrows indicates the optimal p^*, whose value (rounded to 3 decimal places) is also shown as a label.

A first example involves a three-lane highway and four urgency levels. We first determine the STP for an autonomous vehicle traveling on a 1500 meter section of a three-lane highway. The target offramp $\boldsymbol{t}$ is located in the left lane, but the vehicle is incentivized to move in the right lane as much as possible. This is a common preference in routing autonomous trucks; we accomplish this by defining the cost of moving in the current lane as $g(\boldsymbol{x}) = D\sigma_i$ where $\sigma_i = 1 + i\varepsilon$, the lanes are enumerated from right to left ($i = 0,1,2$), and $\varepsilon > 0$ is a fixed penalty. We also model an onramp, with additional traffic entering the right lane from a merge lane at node $\boldsymbol{x}_{\#}$, exactly 1 km from the target. The right-lane nodes within 10 meters of $\boldsymbol{x}_{\#}$ experience an additional cost $\mu = 35$ in their $g(\boldsymbol{x})$ to account for moderate congestion due to the traffic merging in from the onramp. Nodes in the left and right lanes have only one mode, while nodes in the middle lane have two (switch to the left or to the right). In each mode, the system may choose between four available LSMs of increasing urgency with associated success probabilities $0 = p_0 < p_1 < p_2 < p_3 = 1$. $K_\ell = K(p_\ell)$ is given by (39) with $L = 3, p_1 = \tilde{p}, Y_1 = \tilde{p}g_1, p_2 = 0.2$, $Y_2 = 2$, and $Y_3 = 40$. We also set $\alpha = 0.01, \varepsilon = 0.1$, and $g_1 = 3$. A plot of $K(p_\ell)$ for vehicles driving in the right lane is displayed in part (A) of FIG. 17. The resulting $K(p_\ell)$ is monotone δ-causal by Theorem 2 with $\delta_\star \approx 4.095$.

FIG. 17 more particularly illustrates lane-change planning on a three-lane highway between 970 meters and 1040 meters away from $\boldsymbol{t}$. Part (A) of the figure shows the finite cost function $K(p)$ for vehicles traveling in the right lane (except within 10 meters from $\boldsymbol{x}_{\#}$ on either side) for the $L = 3$ escalating LSMs framework described above. C_1 is the stay in lane cost, $K(0)$, and C_2 is the forced LSM cost, $K(1)$. Part (B) of the figure shows the STP when there are four available LSMs per mode. The edges are colored corresponding to the success probability

associated with the optimal LSM urgency level at $\boldsymbol{x}$. Part (C) of the figure shows the deterministic SP when the stay in lane cost is $g(\boldsymbol{x})$ and the lane change cost is equal to $K(1)$.

We compute the STP on this stretch of a highway and show the optimal policy for a smaller segment (between 970 meters and 1040 meters away from $\boldsymbol{t}$) in part (B) of FIG. 17. It illustrates how additional intermediate LSMs allow the vehicle to dynamically adjust the urgency of its attempts in response to anticipated locations of higher cost. For example, vehicles in the right lane prefer to temporarily switch left as they approach the onramp with an increasing LSM urgency as they get closer. At $\boldsymbol{x}_{\#}$ this urgency decreases again (we have already suffered through most of the delay), and beyond $\boldsymbol{x}_{\#}$ those traveling on the right will prefer to continue in their current lane until they are much closer to the target. The vehicles traveling in the center lane will only start trying to switch right after passing the onramp. However, even after $\boldsymbol{x}_{\#}$ their urgency level to switch right remains low - this reflects the fact that the target offramp is already not too far and on the left.

We also show the deterministic optimal SP on the same highway segment in part (C) of FIG. 17. Following these instructions would result in much more aggressive lane switching from the central and left lanes and, not surprisingly, would yield a higher overall cost to target. Comparing SP and STP over all nodes in this network, STP results in the median, average, and maximum cost reduction of 5.23%, 5.49%, and 15.65% respectively.

A second example involves a three-lane highway and a continuous urgency spectrum. Extending the previous example, we now use Rational Bezier Curves to construct a continuous cost function through the points $(p_0 = 0, K_0), (p_2, K_2)$, and $(p_3 = 1, K_3)$. These three points are defined for each lane in Example 1, and we know this action set is monotone causal by Theorem 2); see part (A) of FIG. 17.

For this stretch of a three-lane highway, there are four RBCs to consider - one RBC for each lane (since $g(\boldsymbol{x})$ is lane-dependent) and one RBC for the slow-down (onramp merge) zone, which includes the right-lane nodes within 10 meters of $\boldsymbol{x}_{\#}$. Each RBC has control points $\left\{(p_0, K_0), \left(\frac{K_0}{K_1}, K_0\right), (p_3, K_3)\right\}$ and weights $\{\omega_0 = 1, \omega_1, \omega_2 = 1\}$, with ω_1 chosen to make the RBC

satisfy $K(p_2) = K_2$. The resulting cost function for vehicles traveling in the right lane (except at the locations within 10 meters of $\boldsymbol{x}_{\#}$) is displayed in part (A) of FIG. 18.

FIG. 18 illustrates lane-change planning on a three-lane highway between 970 meters and 1040 meters away from $\boldsymbol{t}$ in the left lane. Part (A) of the figure shows the Rational Bezier cost function through three cost estimate datapoints for vehicles traveling in the right lane (except 10 meters from the merge location, $\boldsymbol{x}_{\#}$), $C_1 = K(0), C_2 = K(1)$, and the point corresponding to the intermediate LSM from the dataset is also shown. Tangency of $K(p) = C_2 p$ to the cost curve prohibits monotone δ-causality by Theorem 2. Part (B) of the figure shows the resulting STP under the RBC cost structure between 970 meters and 1040 meters away from $\boldsymbol{t}$.

Since $K(p_1)$ under the RBC model is fairly close to K_1 defined previously, it is instructive to compare the RBC-based optimal STP shown in part (B) of FIG. 18 with the STP based on just 4 discrete LSMs in part (B) of FIG. 17. While the schematic representation of these policies is similar, the optimal behavior is significantly more nuanced in the case of RBCs, with gradual urgency build-ups highlighting the advantages of continuous spectrum models.

A third example involves multiple roundabouts and a continuous urgency spectrum. We note that in both previous examples the dependency digraph $\mathcal{G}_\mu$ is acyclic for every stationary policy. This is due to the simplicity of the underlying road network (all arrows pointing rightward), which represented a stretch of a single highway. As a result, the OSSPs are explicitly causal and (11) could be solved efficiently in a single Gauss-Seidel iteration (sweeping through the nodes from right to left) regardless of the properties of $K(p)$. But most road networks contain cycles and thus are not explicitly causal, making the MC properties (and applicability of label-setting methods) far more important. Non-trivial cycles are ubiquitous due to numerous road intersections on a large map, but can also be found in smaller examples with ring roads and roundabouts. The above-noted "magic" (or interconnected) roundabouts used in several cities of the United Kingdom present drivers with particularly interesting strategic choices when parts of the roundabout are clogged with traffic.

An example of such an arrangement was previously described in conjunction with FIG. 7, which in part (A) shows a schematic of roundabout network with five connected traffic circles. Arrows indicate the direction of traffic flow and dashed edges indicate available transitions between roundabouts. We assume that the vehicle drives on the left-hand side of the road as is customary in the United Kingdom. Part (B) of FIG. 7 shows lane-level road network representation of the roundabout network with all possible transitions from each node shown. The arrows show available node-to-node transitions, with crossed arrows indicating stochastic lane changes while all other transitions are deterministic. The destination $\boldsymbol{t}$ is along the southeastern exit and identified by a circled node.

In this example, which includes five interconnected roundabouts - one outer (moving clockwise), one inner (moving counterclockwise), and three miniature roundabouts which allow vehicles to switch between the outer and the inner roundabouts. To avoid terminological confusion, we will further refer to all of them as "rings," reserving the term "Roundabout" for their interconnected combination. We assume there are three roads running into/out of this Roundabout and that the target is in the left lane of the southeastern exit, identified by the circled node in part (B) of FIG. 7. For a driver approaching the Roundabout, the main strategic decision is whether it is worth changing lanes before the entry point and, if the answer is yes, how the urgency of LSM should vary as we get closer. Entering from the left lane leads to a clockwise trajectory; entering on the right is worthwhile if you want to transition to the inner roundabout and travel counterclockwise at least at first. The basic tradeoff is usually between following the most direct route to $\boldsymbol{t}$ and limiting exposure to heavy congestion.

As in the previous example, wherever a lane change is possible (as indicated by crossed arrows in part (B) of FIG. 7), the vehicle has access to a complete urgency spectrum of LSMs, but here our transition cost is quadratic in p :

$$K(p) = \beta(\boldsymbol{x})p^2 + \gamma(\boldsymbol{x}) + [f]$$

where $\beta(\boldsymbol{x}), \gamma(\boldsymbol{x}) > 0$ reflect the congestion present within the road segment (e.g., entryways, outer ring, inner ring, mini-ring) at $\boldsymbol{x}$. The positive constant f is only added at the

entrance-nodes from each of the roads (into the mini and outer rings respectively). These constants are chosen to reflect any waiting that the vehicle must endure as a result of current traffic conditions before being able to enter the roundabout. The causality condition established in Theorem 2 is satisfied when $\beta(\boldsymbol{x}) \leq \gamma(\boldsymbol{x}) + [f]$ at all $\boldsymbol{x} \in X$. Our goal is to highlight the effect of traffic congestion on the outer and inner rings; so, for the sake of simplicity, we assume that $\beta(\boldsymbol{x}) = \gamma(\boldsymbol{x}) = 1$ on all two-lane roads leading to/from the Roundabout and on the three mini-rings. The resulting $K(p)$ is shown in part (A) of FIG. 19. For the Roundabout entry nodes, we also set $f = 5$ in all experiments.

FIG. 19 in part (A) thereof shows $K(p)$ along the two-lane entryways to the roundabout network. Here, $\beta(\boldsymbol{x}) = \gamma(\boldsymbol{x}) = 1$, and $K(p)$ is only monotone causal by Theorem 2. Parts (B) and (C) of the figure show optimal STPs at the southwestern and northern entryways when (B) the inner ring has the highest congestion, and (C) the outer ring has the highest congestion. Thick arrows of a first shading indicate a direct entry to the outer ring while thick arrows of a second shading indicate a direct entry into a mini ring. In both cases, vehicles tackle the tradeoff between taking a more direct route to $\boldsymbol{t}$ and the amount of congestion they are willing to encounter.

For all deterministic transitions, we assume the stay-in-lane cost $K = \gamma(\boldsymbol{x})$ based on the local level of congestion. We focus on its effect on the optimal policy for the cars approaching along the northern and southwestern roads. In the first experiment, the inner ring is much more congested ($K = \gamma(\boldsymbol{x}) = 6.8$) than the outer ring ($K = 3$). As shown in part (B) of FIG. 19, it is optimal for the approaching cars to enter the Roundabout from the left lane and travel along the outer ring. Those cars that approach the Roundabout from the Southwest in the right lane will attempt LSMs to the left with an increasing urgency as they get closer. In the second experiment, the situation is essentially reversed with $K = \gamma(\boldsymbol{x}) = 3$ on the inner ring and $K = 5.2$ on the outer. As shown in part (C) of FIG. 19, the approaching cars now have a clear preference to enter through the right lane; this is largely true even for cars coming from the North, for which the counter-clockwise path (through the inner ring) is preferred despite being longer.

These and other embodiments disclosed herein introduce a class of OSSP problems and prove several sufficient "monotone causality" conditions to guarantee the applicability of efficient label-setting methods. The disclosed approaches have important applications both in discrete and continuous optimization. For example, given an anisotropic inhomogeneous speed function f for motion in a continuous domain, the deterministic time-to-target minimized over all feasible paths can be found as a viscosity solution of the corresponding stationary HamiltonJacobi-Bellman PDE. A first-order accurate semi-Lagrangian discretization of that PDE can be re-interpreted as an OSSP, and our MC criteria can be then used to check which discretization stencils are compatible with Dijkstra's algorithm (in 2D and 3D) and with Dial's algorithm (in 2D only). Importantly, the conditions we developed are expressed in terms of simple geometric properties of the anisotropic speed profile $\mathcal{V}_f$. However, our current analysis does not provide any guidance for finding the best MC-causal stencil for each $\mathcal{V}_f$ (since causal properties need to be balanced against a possible increase in local truncation errors). Restricted versions of this problem are known, but many aspects still remain open for general speed profiles. This balancing act is even more delicate with Dial's algorithm, where increasing the angular resolution of a stencil decreases that stencil's spatial locality but allows using a larger bucket width.

In the discrete setting, we have demonstrated the usefulness of OSSPs in optimizing the lane-change routing of autonomous vehicles. Extending prior work, we showed that Dijkstra's algorithm and Dial's algorithm are applicable to a much broader class of vehicle routing models that include multiple intermediate urgency levels (or even a continuous urgency spectrum) of lane-change maneuvers and a variety of cost functions. We note that the same approach is also useful in a semi-autonomous context; our STP could be also used by human drivers and the recommended lane-change urgency levels could be communicated (e.g., as indicator bars of varying length or color) through assistive navigation hardware or software.

OSSP models capture the inherent uncertainty of lane change maneuvers, but if the traffic conditions significantly change from those used to formulate the problem, the entire optimal policy has to be recomputed on the fly. The usability of label-setting methods makes such occasional

online replanning possible. But both Dijkstra's algorithm and Dial's algorithm compute the optimal policy for reaching the specific target $\boldsymbol{t}$ from each starting node of the road network. However, most of that network is probably irrelevant for a vehicle starting at a specific location s. In deterministic path planning, such single source/single target problems are often solved by an even more efficient A* method, which uses a "consistent heuristic" to restrict the computations to a smaller (implicitly defined) neighborhood of the optimal path from s to $\boldsymbol{t}$. While such a consistent heuristic is unavailable for general SSPs, it can be constructed in OSSPs with positive $\underline{C} = \min_{\boldsymbol{x}_i \in X} \min_{\boldsymbol{a} \in \Lambda(\boldsymbol{x}_i)} C(\boldsymbol{x}_i, \boldsymbol{a})$. But since realistic road networks include short lane segments, this $\underline{C}$ is usually quite small, making the consistent heuristic very conservative and yielding little computational savings. We believe that a more promising approach is to explore the use of inconsistent heuristics (ideally, with a bound on suboptimality of resulting routing policies). It might be possible to use the "asymptotically causal" domain restriction techniques developed for discretizations of HJB equations, with the true optimality of trajectories recovered only in the limit (under the grid refinement).

Other embodiments can be implemented in other contexts. For example, some embodiments can be configured to apply label-setting to "stochastic shortest path games," in which the probability distribution over the successor nodes at each stage depends on the actions chosen by two antagonistic players. We note that some results on the applicability of Dijkstra's algorithm to deterministic games on graphs have been developed, but the case of general stochastic games on graphs remains open.

It should also be understood that the particular arrangements shown and described in conjunction with FIGS. 1 through 19 are presented by way of illustrative example only, and numerous alternative embodiments are possible. The various embodiments disclosed herein should therefore not be construed as limiting in any way. Numerous alternative arrangements of routing optimization and associated autonomous vehicle control can be utilized in other embodiments. Those skilled in the art will also recognize that alternative processing operations and associated system configurations can be used in other embodiments.

It is therefore possible that other embodiments may include additional or alternative system elements, relative to the entities of the illustrative embodiments. Accordingly, the particular system configurations and associated algorithm implementations can be varied in other embodiments.

A given processing device or other component of an information processing system as described herein is illustratively configured utilizing a corresponding processing device comprising a processor coupled to a memory. The processor executes software program code stored in the memory in order to control the performance of processing operations and other functionality. The processing device also comprises a network interface that supports communication over one or more networks.

The processor may comprise, for example, a microprocessor, an ASIC, an FPGA, a CPU, a GPU, a TPU, an ALU, a DSP, or other similar processing device component, as well as other types and arrangements of processing circuitry, in any combination. For example, at least a portion of the functionality of at least one routing optimization system, and its associated routing optimization algorithms for use in autonomous vehicle control applications, as provided utilizing a processing platform comprising one or more processing devices as disclosed herein, can be implemented using such circuitry.

The memory stores software program code for execution by the processor in implementing portions of the functionality of the processing device. A given such memory that stores such program code for execution by a corresponding processor is an example of what is more generally referred to herein as a processor-readable storage medium having program code embodied therein, and may comprise, for example, electronic memory such as SRAM, DRAM or other types of random access memory, ROM, flash memory, magnetic memory, optical memory, or other types of storage devices in any combination.

As mentioned previously, articles of manufacture comprising such processor-readable storage media are considered embodiments of the present disclosure. The term “article of manufacture” as used herein should be understood to exclude transitory, propagating signals.

Other types of computer program products comprising processor-readable storage media can be implemented in other embodiments.

In addition, embodiments of the present disclosure may be implemented in the form of integrated circuits comprising processing circuitry configured to implement processing operations associated with autonomous vehicle routing.

An information processing system as disclosed herein may be implemented using one or more processing platforms, or portions thereof.

For example, one illustrative embodiment of a processing platform that may be used to implement at least a portion of an information processing system comprises cloud infrastructure including virtual machines implemented using a hypervisor that runs on physical infrastructure. Such virtual machines may comprise respective processing devices that communicate with one another over one or more networks.

The cloud infrastructure in such an embodiment may further comprise one or more sets of applications running on respective ones of the virtual machines under the control of the hypervisor. It is also possible to use multiple hypervisors each providing a set of virtual machines using at least one underlying physical machine. Different sets of virtual machines provided by one or more hypervisors may be utilized in configuring multiple instances of various components of the information processing system.

Another illustrative embodiment of a processing platform that may be used to implement at least a portion of an information processing system as disclosed herein comprises a plurality of processing devices which communicate with one another over at least one network. Each processing device of the processing platform is assumed to comprise a processor coupled to a memory. A given such network can illustratively include, for example, a global computer network such as the Internet, a WAN, a LAN, a satellite network, a telephone or cable network, a cellular network such as a 4G or 5G network, a wireless network implemented using a wireless protocol such as Bluetooth, WiFi or WiMAX, or various portions or combinations of these and other types of communication networks.

Again, these particular processing platforms are presented by way of example only, and an information processing system may include additional or alternative processing platforms, as well as numerous distinct processing platforms in any combination, with each such platform comprising one or more computers, servers, storage devices or other processing devices.

A given processing platform implementing a routing optimization system as disclosed herein can alternatively comprise a single processing device, such as a computer or other processing device. It is also possible in some embodiments that one or more such system elements can run on or be otherwise supported by cloud infrastructure or other types of virtualization infrastructure.

It should therefore be understood that in other embodiments different arrangements of additional or alternative elements may be used. At least a subset of these elements may be collectively implemented on a common processing platform, or each such element may be implemented on a separate processing platform.

Also, numerous other arrangements of autonomous vehicles, computers, servers, storage devices or other components are possible in an information processing system. Such components can communicate with other elements of the information processing system over any type of network or other communication media.

As indicated previously, components of the system as disclosed herein can be implemented at least in part in the form of one or more software programs stored in memory and executed by a processor of a processing device. For example, certain functionality disclosed herein can be implemented at least in part in the form of software.

The particular configurations of information processing systems described herein are exemplary only, and a given such system in other embodiments may include other elements in addition to or in place of those specifically shown, including one or more elements of a type commonly found in a conventional implementation of such a system.

For example, in some embodiments, an information processing system may be configured to utilize the disclosed techniques to provide additional or alternative functionality in other contexts.

It should again be emphasized that the embodiments of the present disclosure as described herein are intended to be illustrative only. Other embodiments of the disclosure can be implemented utilizing a wide variety of different types and arrangements of information processing systems, networks and processing devices than those utilized in the particular illustrative embodiments described herein, and in numerous alternative processing contexts. In addition, the particular assumptions made herein in the context of describing certain embodiments need not apply in other embodiments. These and numerous other alternative embodiments will be readily apparent to those skilled in the art.

Claims

What is claimed is:

1. A method comprising:

obtaining information characterizing at least a portion of a routing environment of an autonomous vehicle;

formulating an optimization problem based at least in part on the obtained information, the optimization problem incorporating multiple levels of urgency for respective possible lane-change actions of the autonomous vehicle;

solving the optimization problem utilizing a shortest-path algorithm; and

generating at least one control signal for utilization in the autonomous vehicle based at least in part on results of solving the optimization problem and a current state of the autonomous vehicle;

wherein the method is performed by at least one processing device comprising a processor coupled to a memory.

2. The method of claim 1 wherein obtaining information characterizing at least a portion of a routing environment of an autonomous vehicle comprises populating one or more data structures characterizing possible transitions between a plurality of states as the autonomous vehicle travels in the routing environment.

3. The method of claim 1 wherein formulating an optimization problem based at least in part on the obtained information comprises formulating the optimization problem as an opportunistically stochastic shortest-path problem configured to support selection between a plurality of actions including one or more stochastic transitions and one or more deterministic transitions at each of a plurality of states.

4. The method of claim 1 wherein formulating an optimization problem based at least in part on the obtained information comprises formulating the optimization problem to incorporate a transition cost function that guarantees monotone causality.

5. The method of claim 1 wherein the multiple levels of urgency for respective possible lane-change actions of the autonomous vehicle comprise a plurality of discrete urgency levels.

6. The method of claim 1 wherein the multiple levels of urgency for respective possible lane-change actions of the autonomous vehicle comprise a plurality of urgency levels selected from a continuum of urgency levels.

7. The method of claim 1 wherein solving the optimization problem utilizing a shortest-path algorithm comprises:

selecting a particular one of a plurality of shortest-path algorithms implemented in a routing optimization system of the autonomous vehicle; and

solving the optimization problem utilizing the selected shortest-path algorithm;

wherein the selected shortest-path algorithm comprises a label-setting shortest path algorithm.

8. The method of claim 7 wherein the plurality of shortest-path algorithms implemented in the routing optimization system of the autonomous vehicle comprise at least Dijkstra's algorithm and Dial's algorithm.

9. The method of claim 1 wherein solving the optimization problem utilizing a shortest-path algorithm comprises generating in each of a plurality of stages of the shortest-path algorithm an optimal action and a corresponding cumulative cost, the optimal action having associated therewith a particular one of the levels of urgency.

10. The method of claim 9 wherein the optimal action comprises a particular one of the lane-change actions and the associated one of the levels of urgency is indicative of a corresponding degree to which a velocity of the autonomous vehicle can be adjusted in conjunction with attempting the particular lane-change action.

11. The method of claim 1 wherein the autonomous vehicle comprises one of a fully-autonomous vehicle and a semi-autonomous vehicle.

12. The method of claim 1 wherein generating at least one control signal for utilization in the autonomous vehicle comprises at least one of:

generating at least one control signal in a vehicle controller of the autonomous vehicle; and

generating at least one control signal for utilization in a driver-assistance system of the autonomous vehicle.

13. The method of claim 1 wherein said at least one processing device comprises at least a portion of the autonomous vehicle.

14. The method of claim 1 wherein said at least one processing device comprises at least a portion of at least one of a core device and an edge device of at least one network configured to communicate with the autonomous vehicle.

15. A system comprising:

at least one processing device comprising a processor coupled to a memory;

the processing device being configured:

to obtain information characterizing at least a portion of a routing environment of an autonomous vehicle;

to formulate an optimization problem based at least in part on the obtained information, the optimization problem incorporating multiple levels of urgency for respective possible lane-change actions of the autonomous vehicle;

to solve the optimization problem utilizing a shortest-path algorithm; and

to generate at least one control signal for utilization in the autonomous vehicle based at least in part on results of solving the optimization problem and a current state of the autonomous vehicle.

16. The system of claim 15 wherein formulating an optimization problem based at least in part on the obtained information comprises formulating the optimization problem as an opportunistically stochastic shortest-path problem configured to support selection between a plurality of actions including one or more stochastic transitions and one or more deterministic transitions at each of a plurality of states.

17. The system of claim 15 wherein formulating an optimization problem based at least in part on the obtained information comprises formulating the optimization problem to incorporate a transition cost function that guarantees monotone causality.

18. A computer program product comprising a non-transitory processor-readable storage medium having stored therein program code of one or more software programs, wherein the program code, when executed by at least one processing device comprising a processor coupled to a memory, causes the processing device:

to obtain information characterizing at least a portion of a routing environment of an autonomous vehicle;

to formulate an optimization problem based at least in part on the obtained information, the optimization problem incorporating multiple levels of urgency for respective possible lane-change actions of the autonomous vehicle;

to solve the optimization problem utilizing a shortest-path algorithm; and

to generate at least one control signal for utilization in the autonomous vehicle based at least in part on results of solving the optimization problem and a current state of the autonomous vehicle.

19. The computer program product of claim 18 wherein formulating an optimization problem based at least in part on the obtained information comprises formulating the optimization problem as an opportunistically stochastic shortest-path problem configured to support selection between a plurality of actions including one or more stochastic transitions and one or more deterministic transitions at each of a plurality of states.

20. The computer program product of claim 18 wherein formulating an optimization problem based at least in part on the obtained information comprises formulating the optimization problem to incorporate a transition cost function that guarantees monotone causality.

FIG. 1

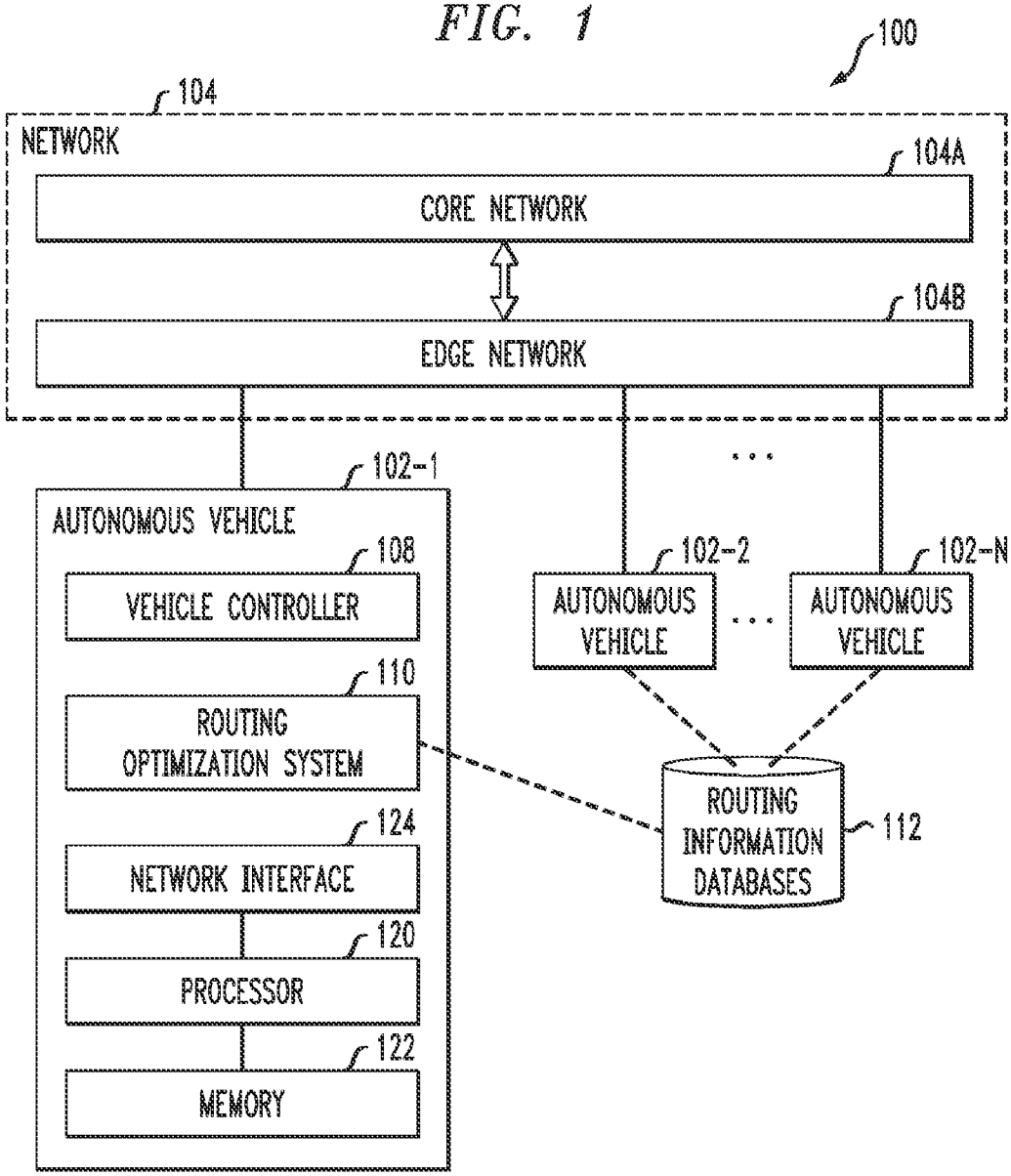

100
104
NETWORK
104A
CORE NETWORK
104B
EDGE NETWORK
102-1
AUTONOMOUS VEHICLE
108
VEHICLE CONTROLLER
110
ROUTING OPTIMIZATION SYSTEM
124
NETWORK INTERFACE
120
PROCESSOR
122
MEMORY
102-2
AUTONOMOUS VEHICLE
102-N
AUTONOMOUS VEHICLE
ROUTING INFORMATION DATABASES
112

FIG. 2

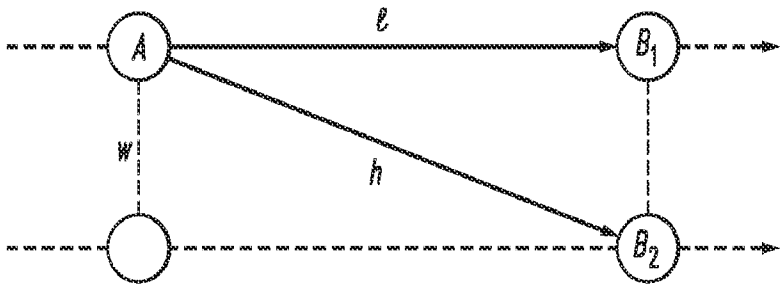
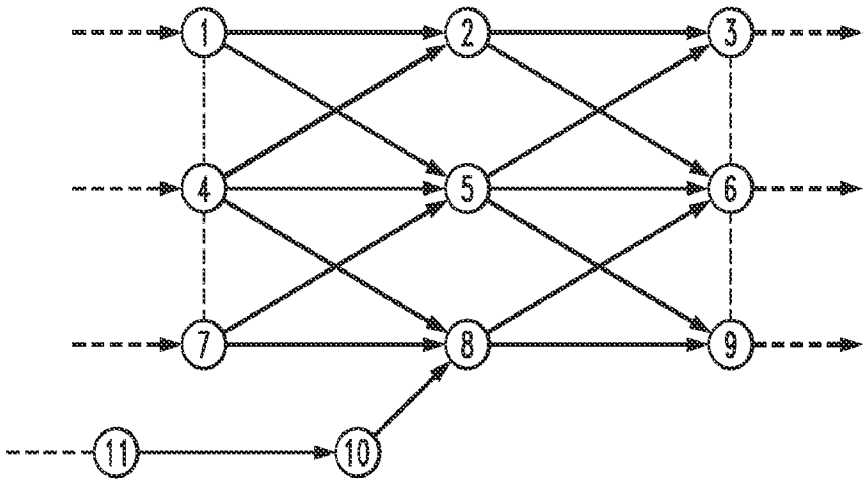

A
B1
B2
ℓ
w
h

FIG. 3

(A)


1
2
3
4
5
6
7
8
9
10
11

(B)

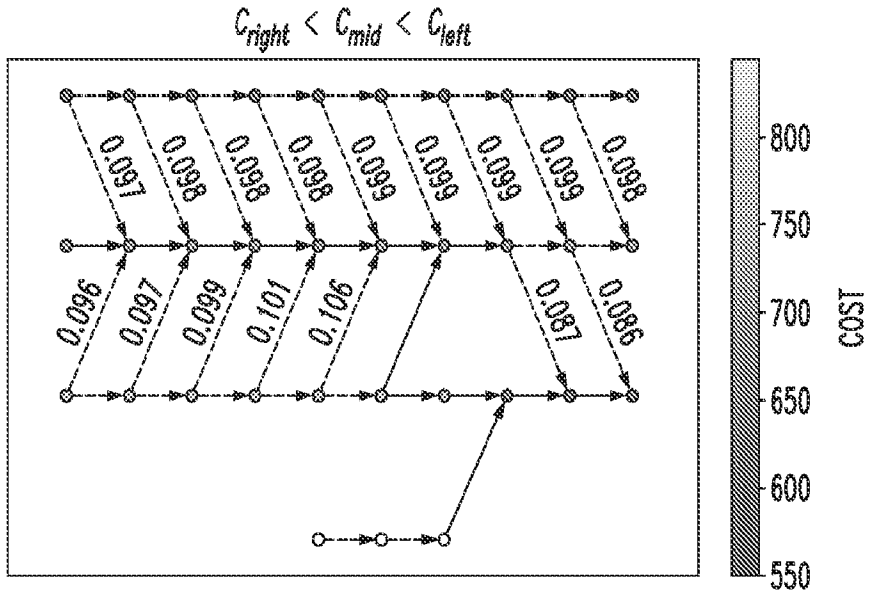

$C_{right} < C_{mid} < C_{left}$
0.097
0.098
0.098
0.098
0.099
0.099
0.099
0.099
0.098
0.096
0.097
0.099
0.101
0.106
0.087
0.086
800
750
700
650
600
550
COST

FIG. 4

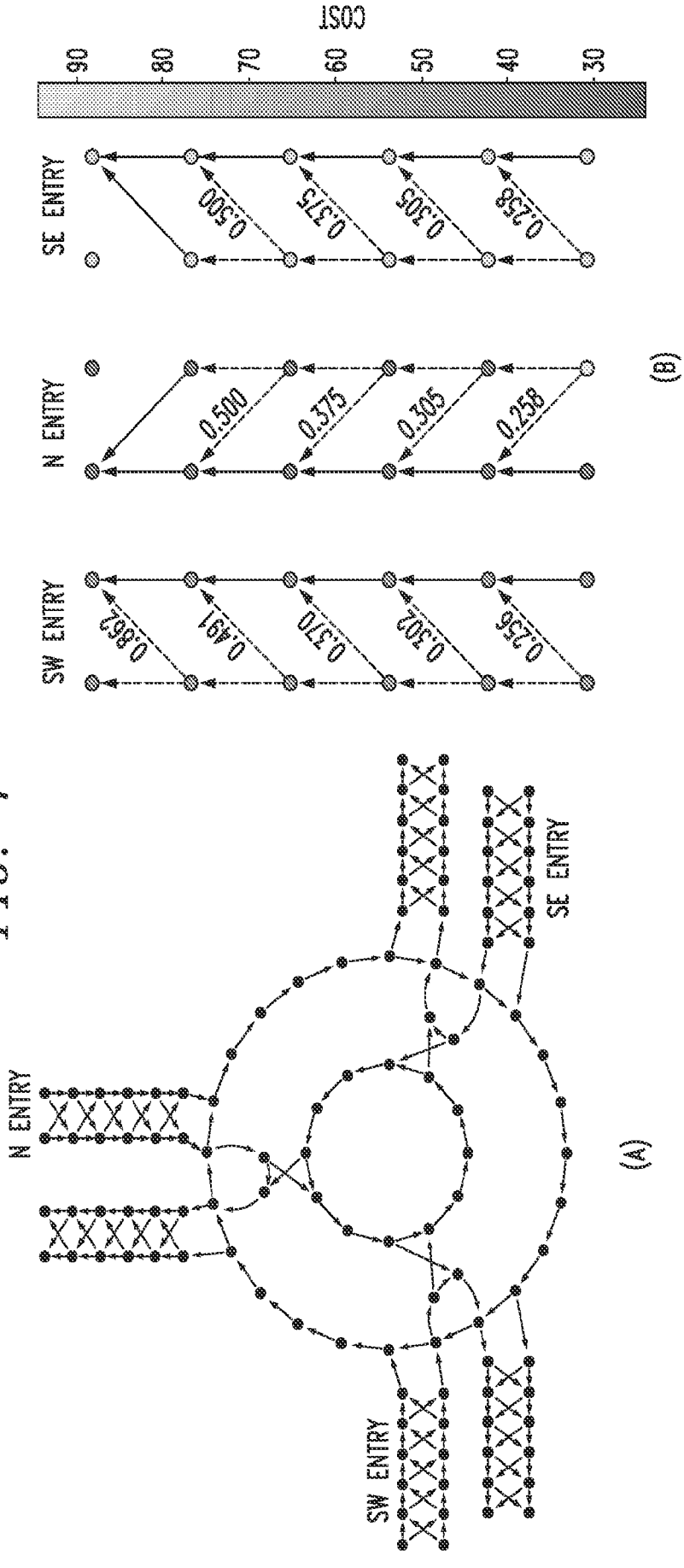

COST
90
80
70
60
50
40
30
SE ENTRY
0.500
0.375
0.305
0.258
N ENTRY
0.500
0.375
0.305
0.258
SW ENTRY
0.862
0.491
0.370
0.302
0.256
(B)
N ENTRY
SE ENTRY
SW ENTRY
(A)

FIG. 5

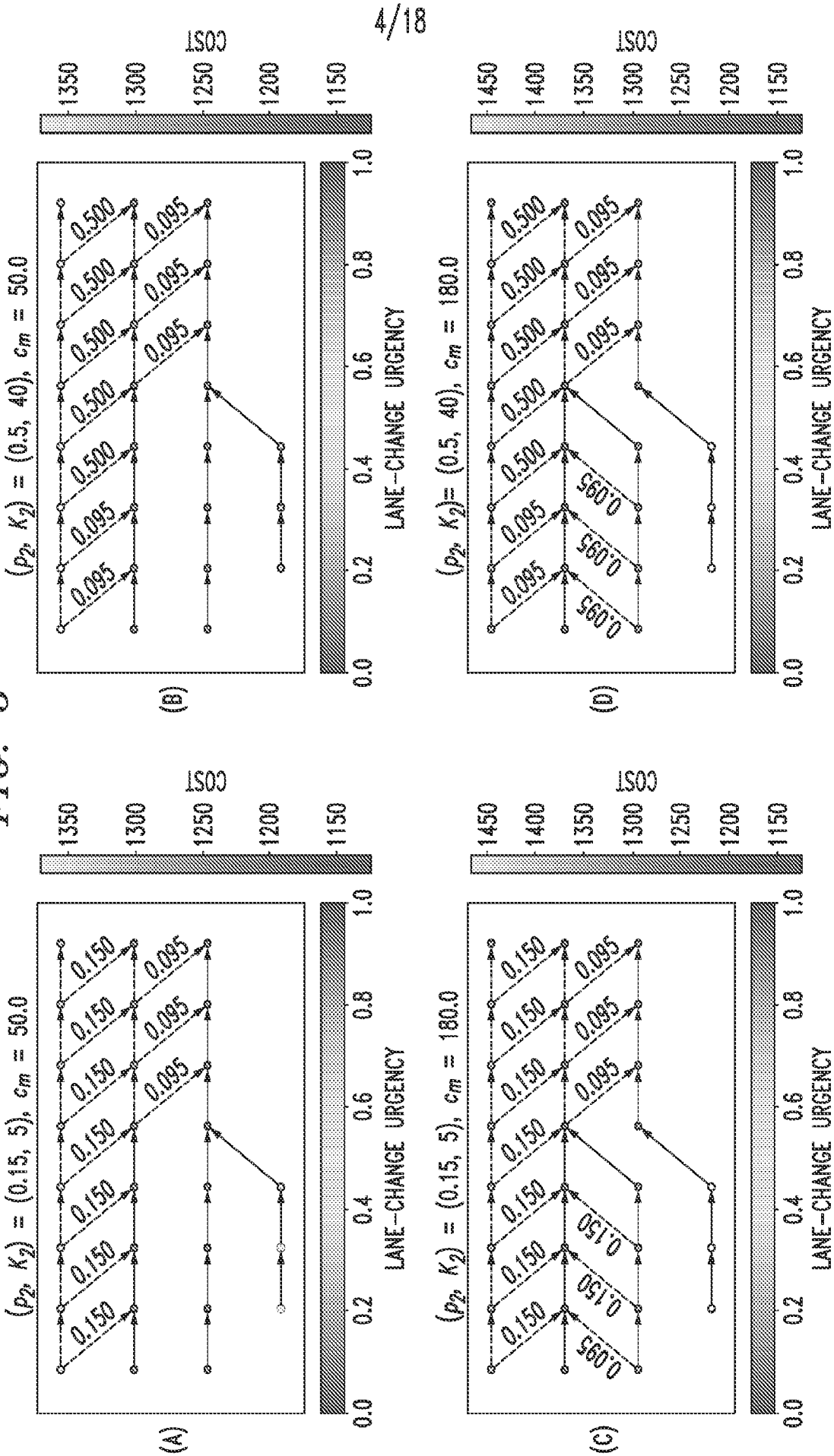
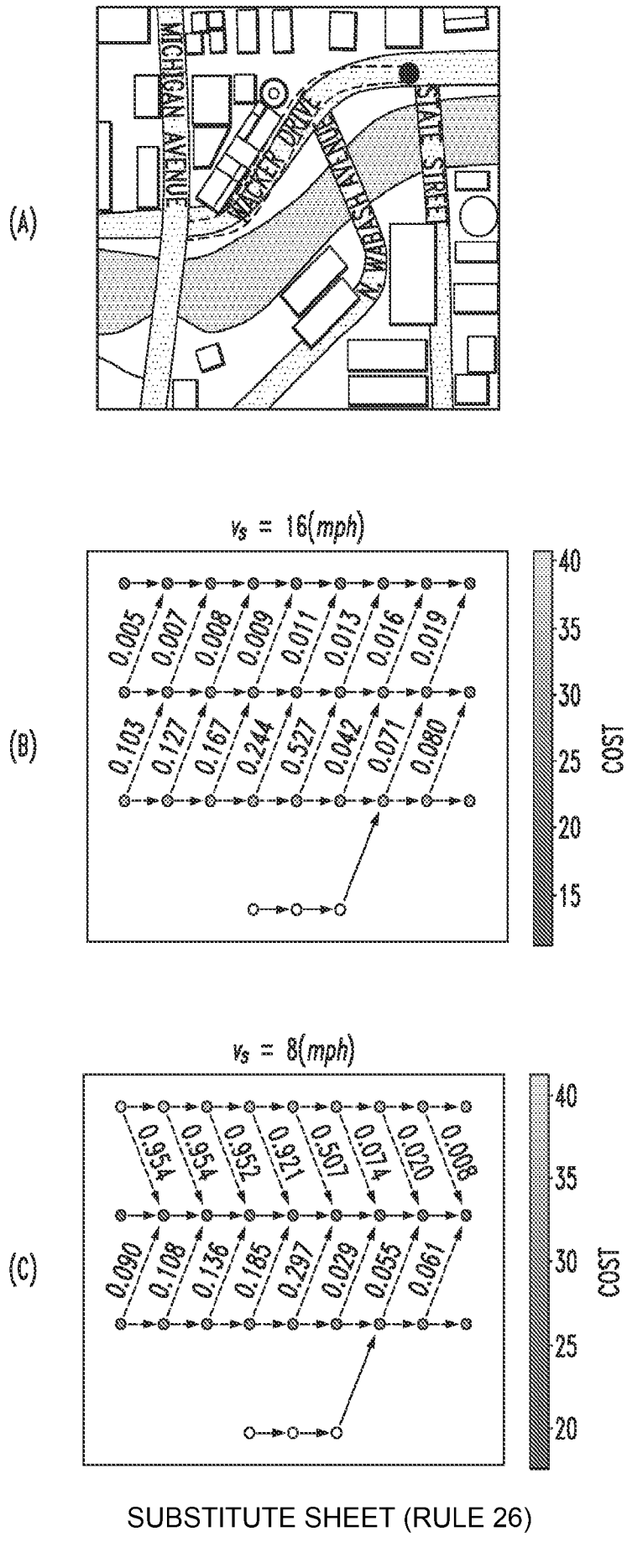

(A)
(B)
(C)
(D)
COST
LANE-CHANGE URGENCY
$(\rho_2, K_2) = (0.15, 5), c_m = 50.0$
$(\rho_2, K_2) = (0.5, 40), c_m = 50.0$
$(\rho_2, K_2) = (0.15, 5), c_m = 180.0$
$(\rho_2, K_2) = (0.5, 40), c_m = 180.0$
0.150
0.095
0.500
0.0
0.2
0.4
0.6
0.8
1.0
1150
1200
1250
1300
1350
1400
1450

FIG. 6


(A)
MICHIGAN AVENUE
WACKER DRIVE
N. WABASH AVENUE
STATE STREET
(B)
v_s = 16(mph)
0.005
0.007
0.008
0.009
0.011
0.013
0.016
0.019
0.103
0.127
0.167
0.244
0.527
0.042
0.071
0.080
40
35
30
25
20
15
COST
(C)
v_s = 8(mph)
0.954
0.954
0.952
0.921
0.507
0.074
0.020
0.008
0.090
0.108
0.136
0.185
0.297
0.029
0.055
0.061
40
35
30
25
20
COST

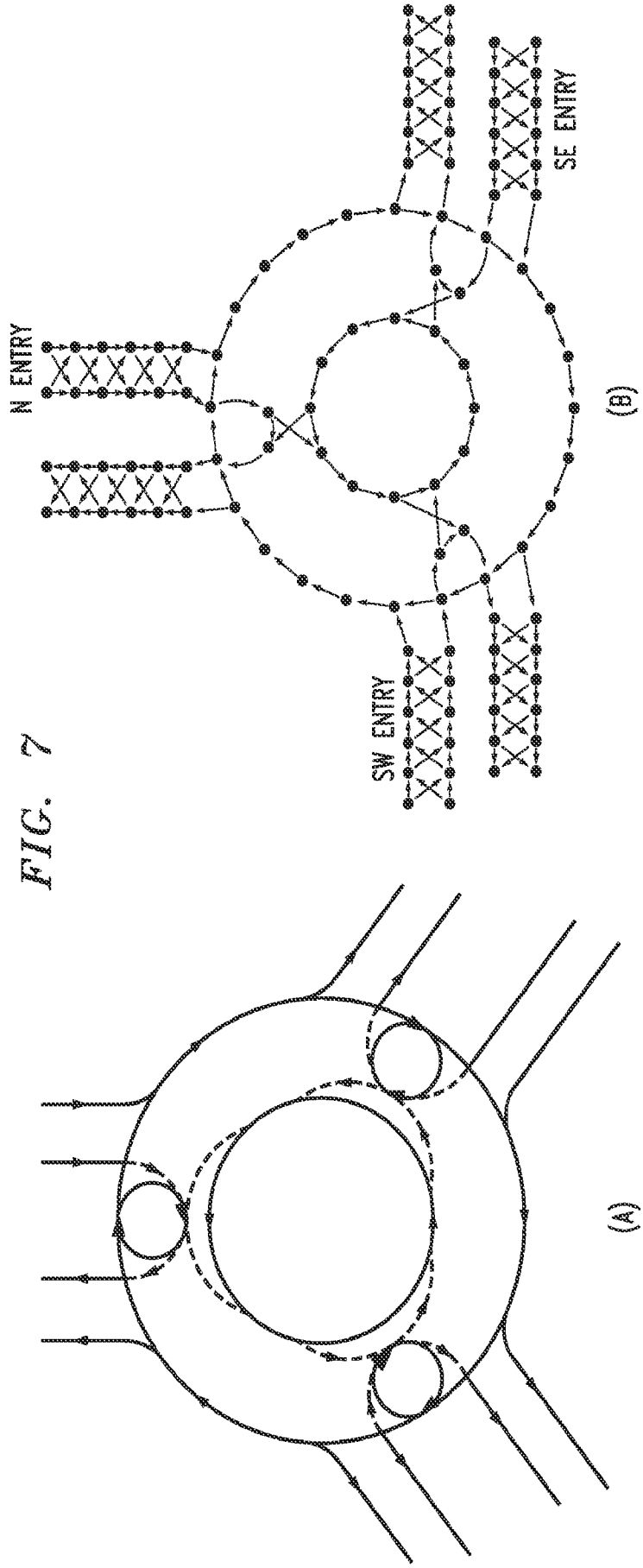
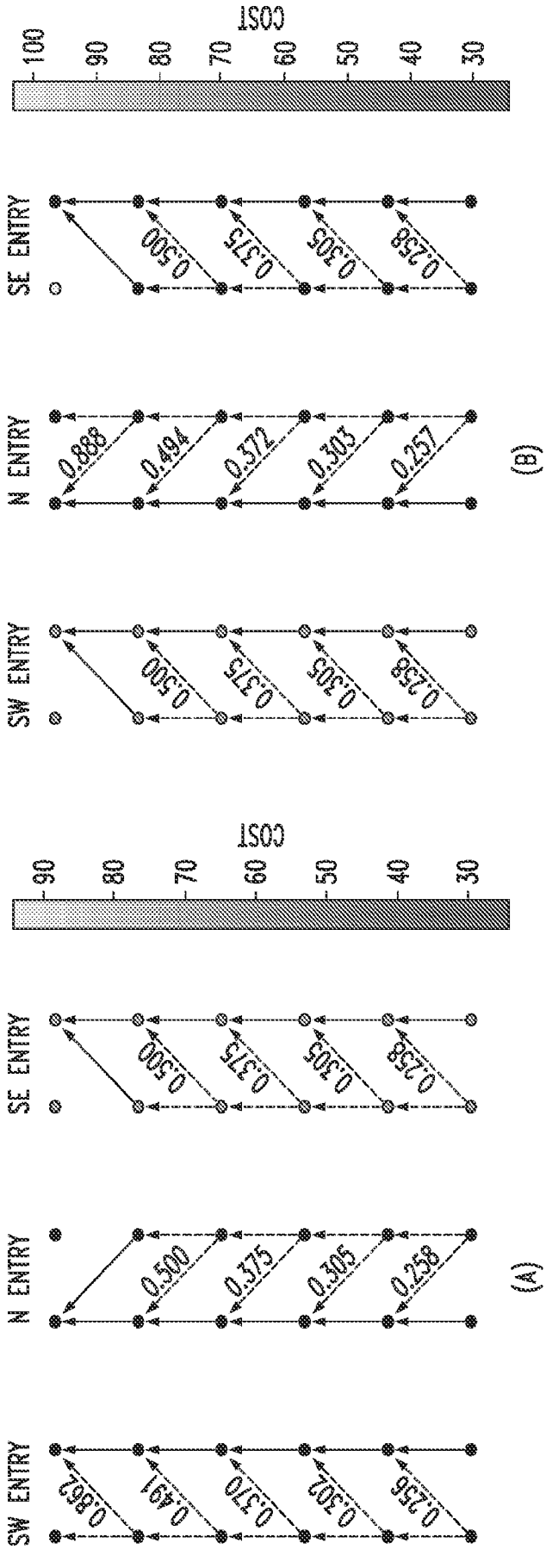


FIG. 7

FIG. 8


COST
100
90
80
70
60
50
40
30
SE ENTRY
0.500
0.375
0.305
0.258
N ENTRY
0.888
0.494
0.372
0.303
0.257
(B)
SW ENTRY
0.500
0.375
0.305
0.258
COST
90
80
70
60
50
40
30
SE ENTRY
0.500
0.375
0.305
0.258
N ENTRY
0.500
0.375
0.305
0.258
(A)
SW ENTRY
0.862
0.491
0.370
0.302
0.256

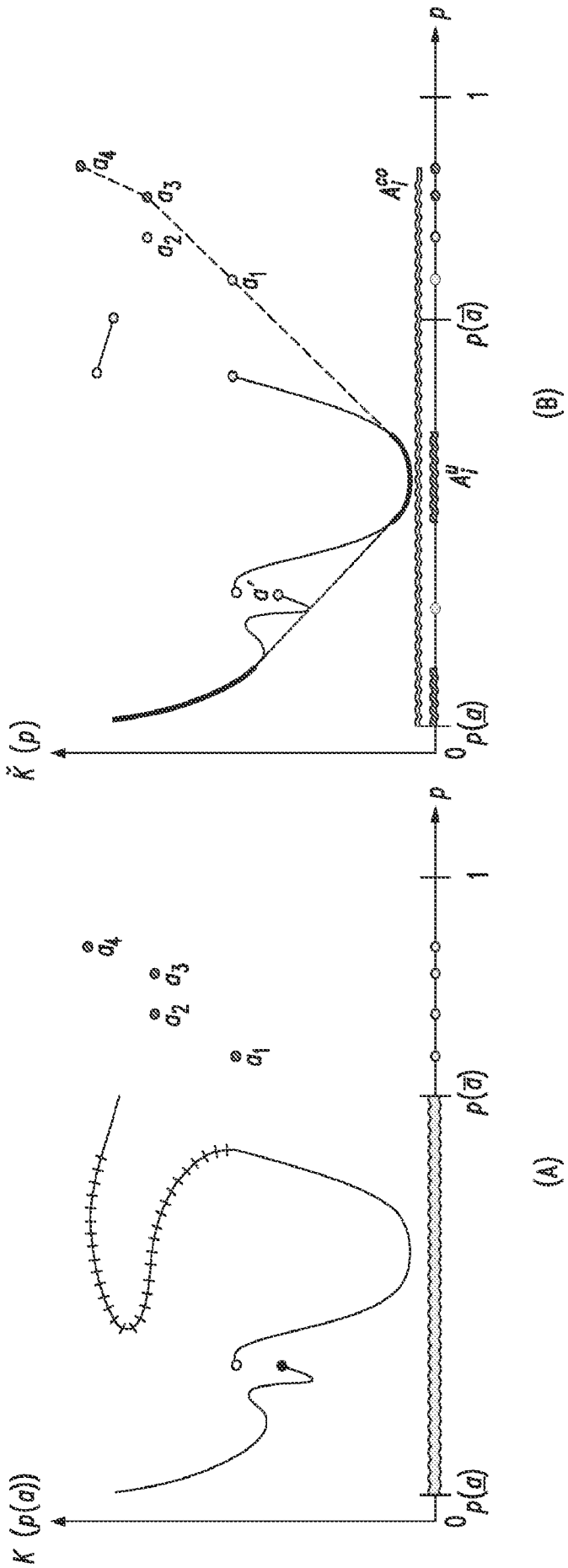
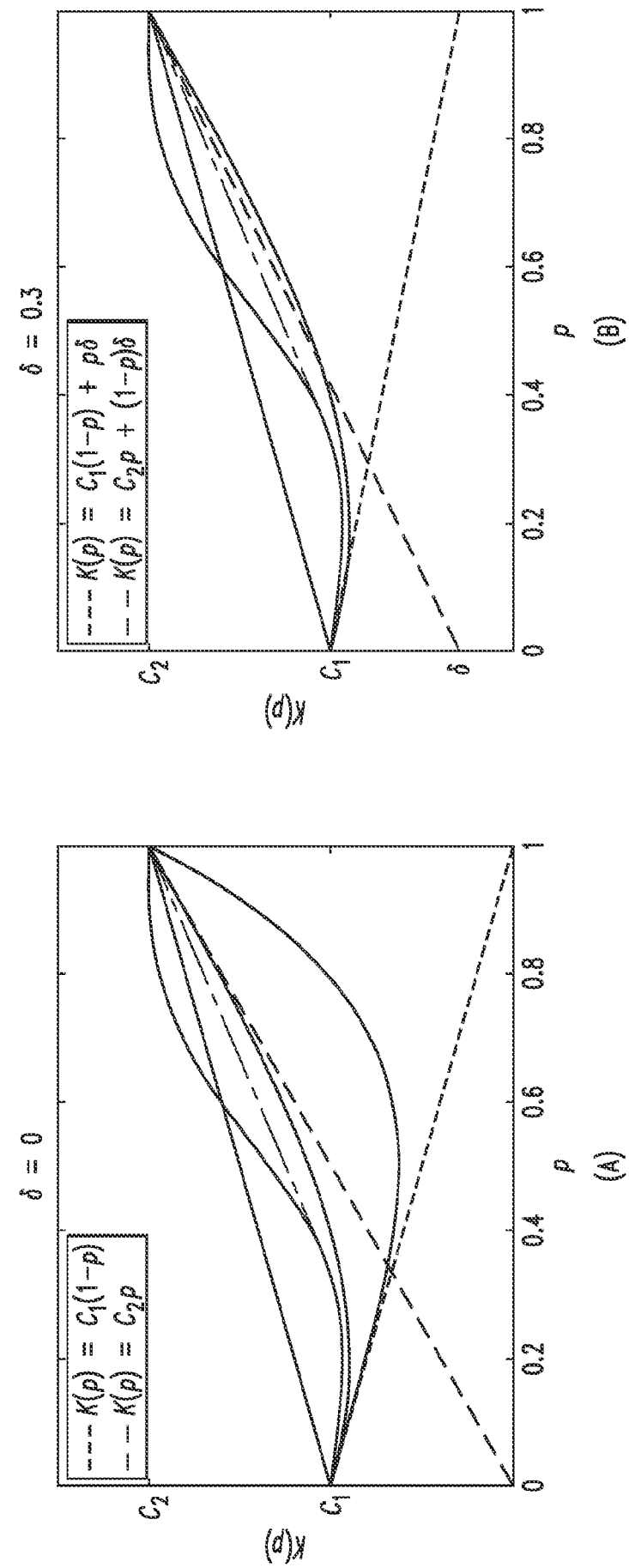


FIG. 9

FIG. 10


δ = 0
K(p) = C1(1−p)
K(p) = C2p
C2
C1
κ(p)
0
0.2
0.4
0.6
0.8
1
p
(A)
δ = 0.3
K(p) = C1(1−p) + pδ
K(p) = C2p + (1−p)δ
C2
C1
δ
κ(p)
0
0.2
0.4
0.6
0.8
1
p
(B)

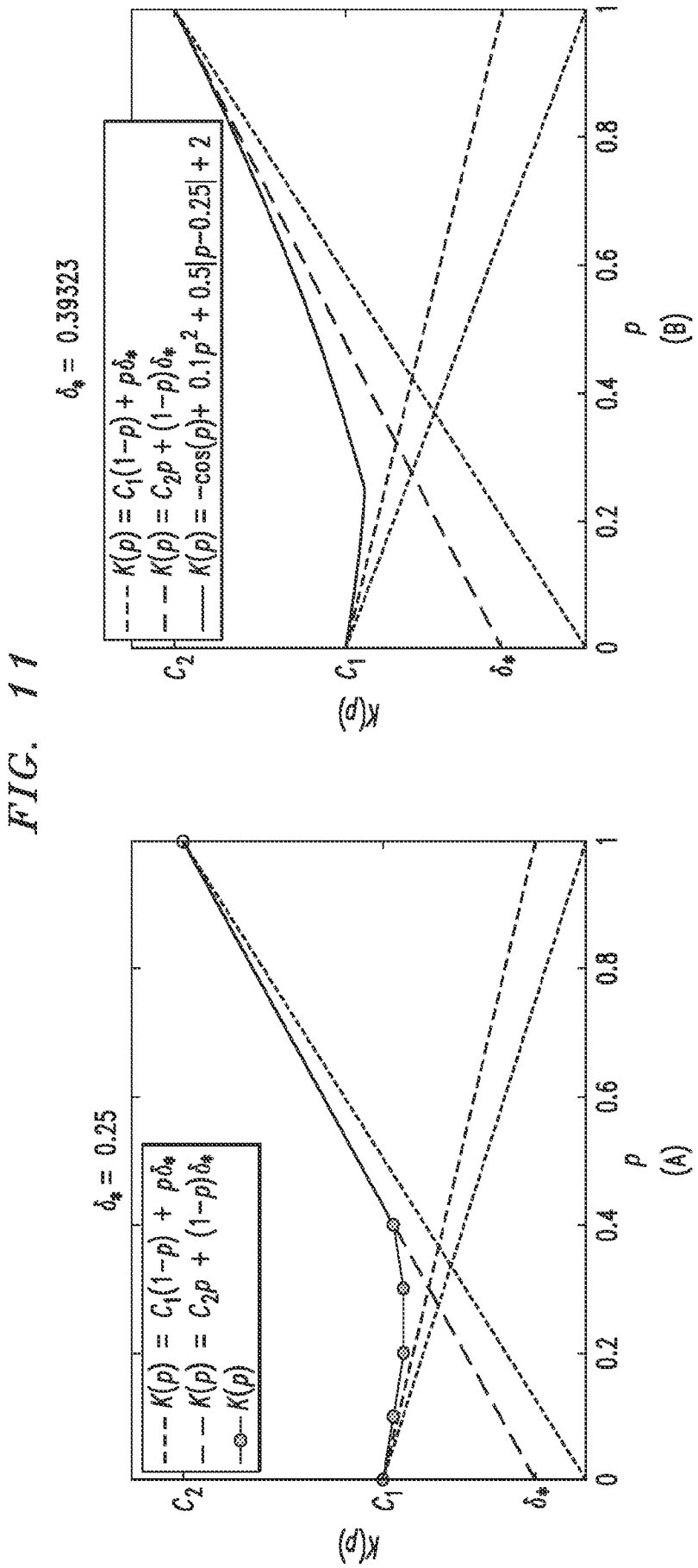
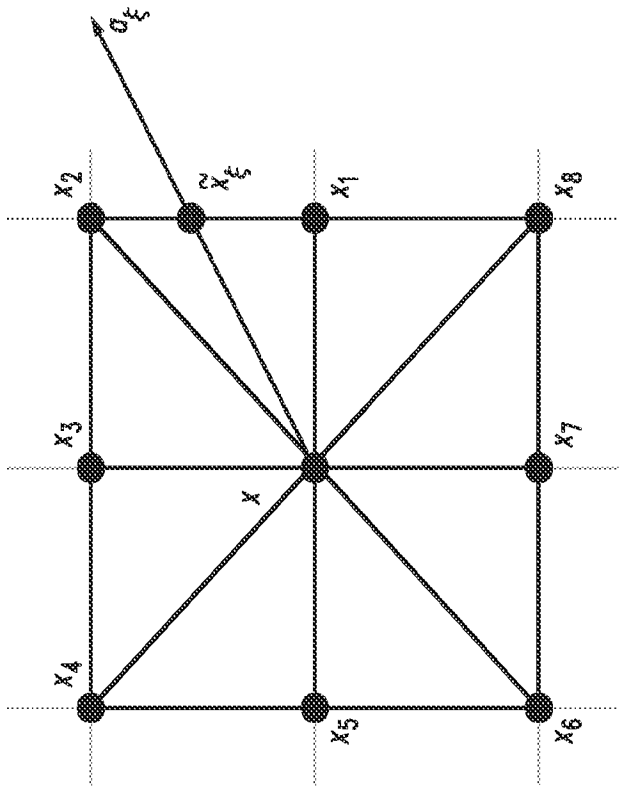
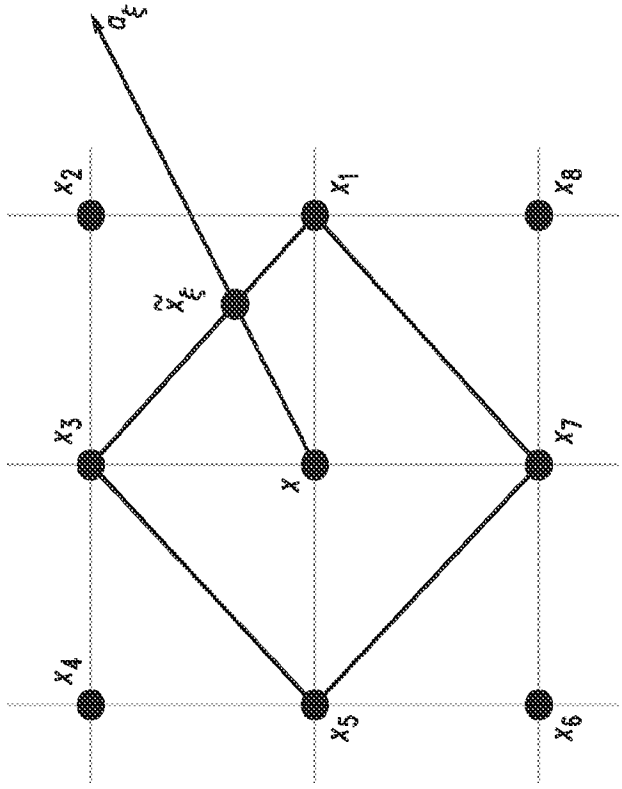

δ* = 0.25
K(p) = C1(1−p) + pδ*
K(p) = C2p + (1−p)δ*
K(p)
C2
C1
δ*
K(δ)
p
(A)
δ* = 0.39323
K(p) = C1(1−p) + pδ*
K(p) = C2p + (1−p)δ*
K(p) = −cos(p)+ 0.1p^2 + 0.5|p−0.25| + 2
C2
C1
δ*
K(δ)
p
(B)

FIG. 11


a_ξ
$\tilde{x}_\xi$
x_1
x_2
x_3
x_4
x_5
x_6
x_7
x_8
x

(B)


a_ξ
$\tilde{x}_\xi$
x_1
x_2
x_3
x_4
x_5
x_6
x_7
x_8
x

(A)

FIG. 12

FIG. 13

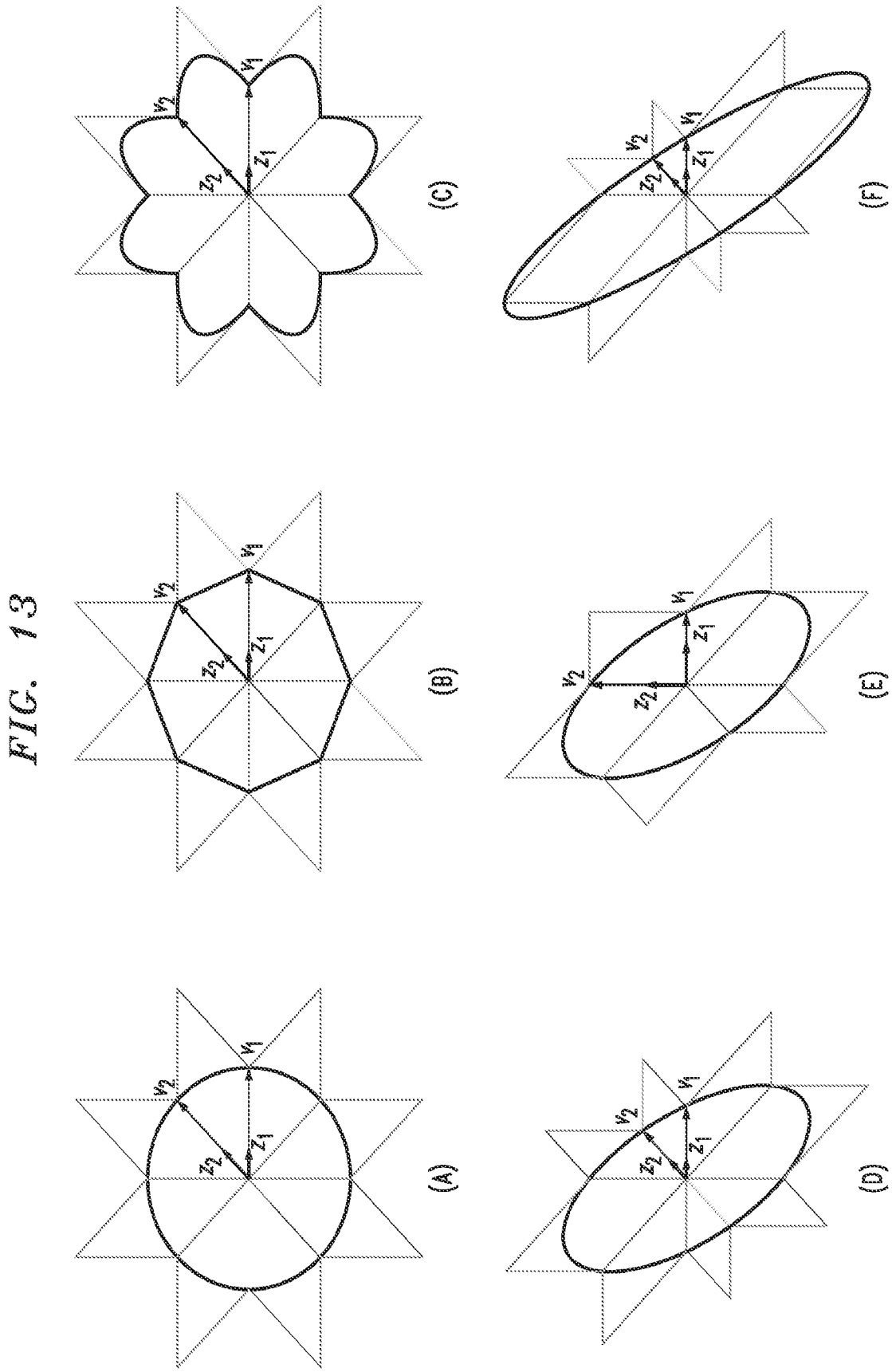
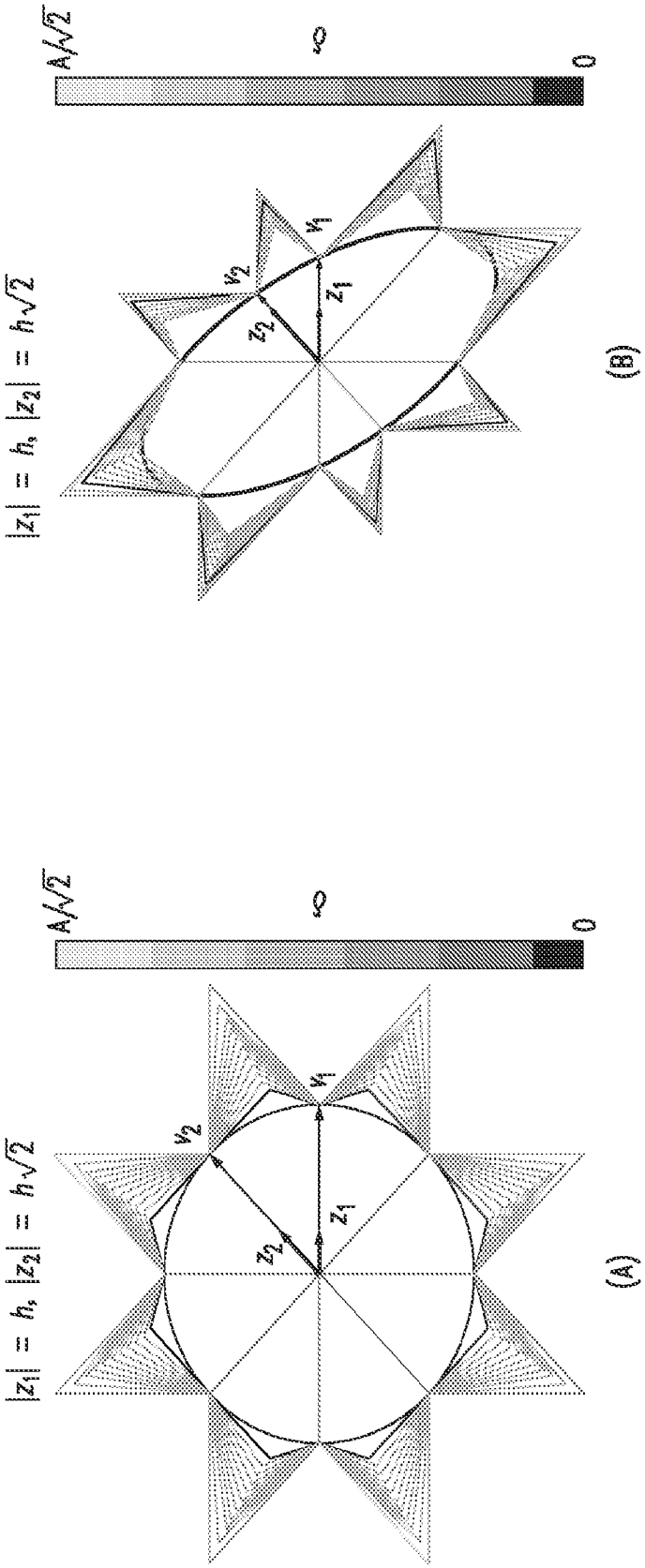


FIG. 14



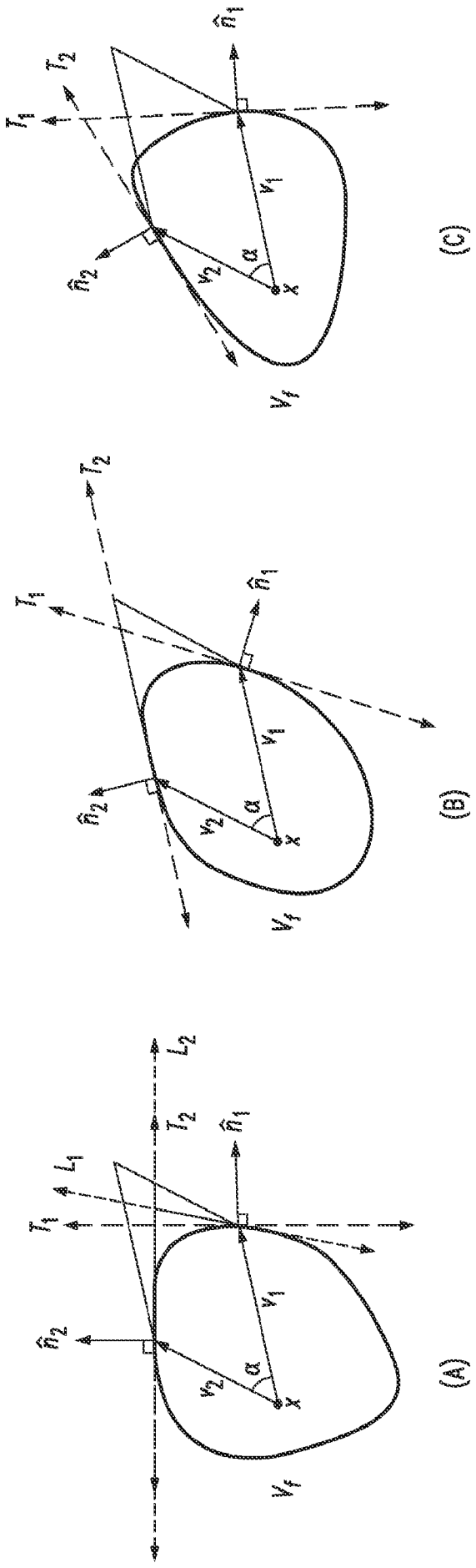


FIG. 15

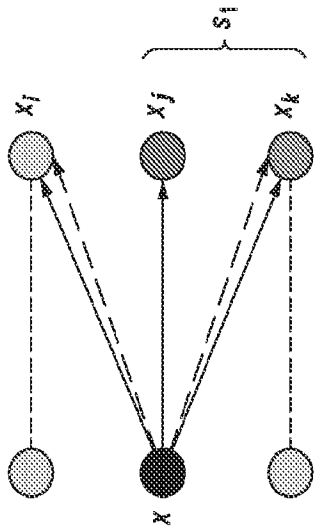


FIG. 16

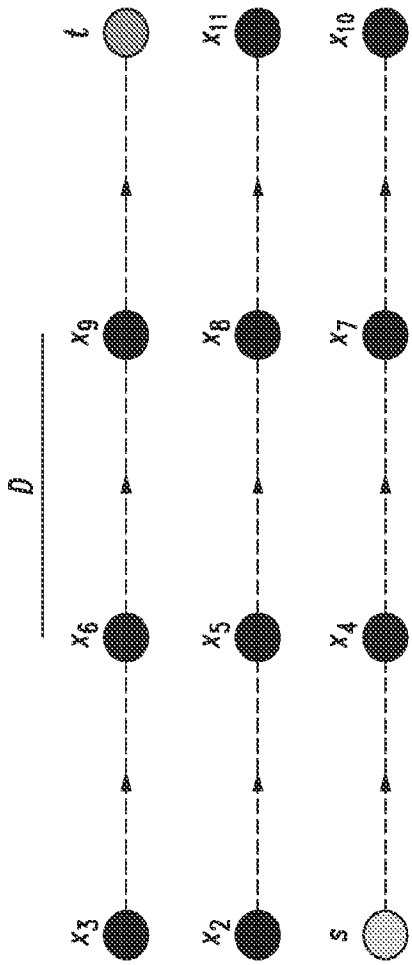


FIG. 17

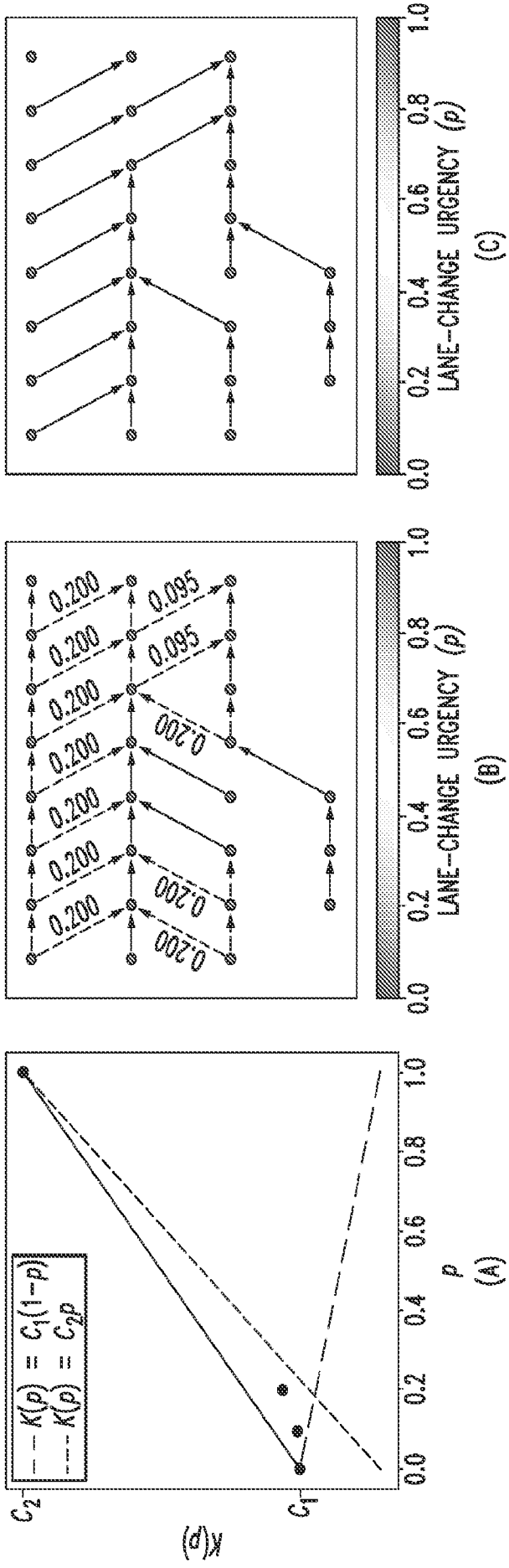

0.200
0.095
LANE-CHANGE URGENCY (ρ)
0.0
0.2
0.4
0.6
0.8
1.0
(C)
(B)
(A)
ρ
κ(ρ)
c_2
c_1
$\kappa(\rho) = c_1(1-\rho)$
$\kappa(\rho) = c_2\rho$

FIG. 18

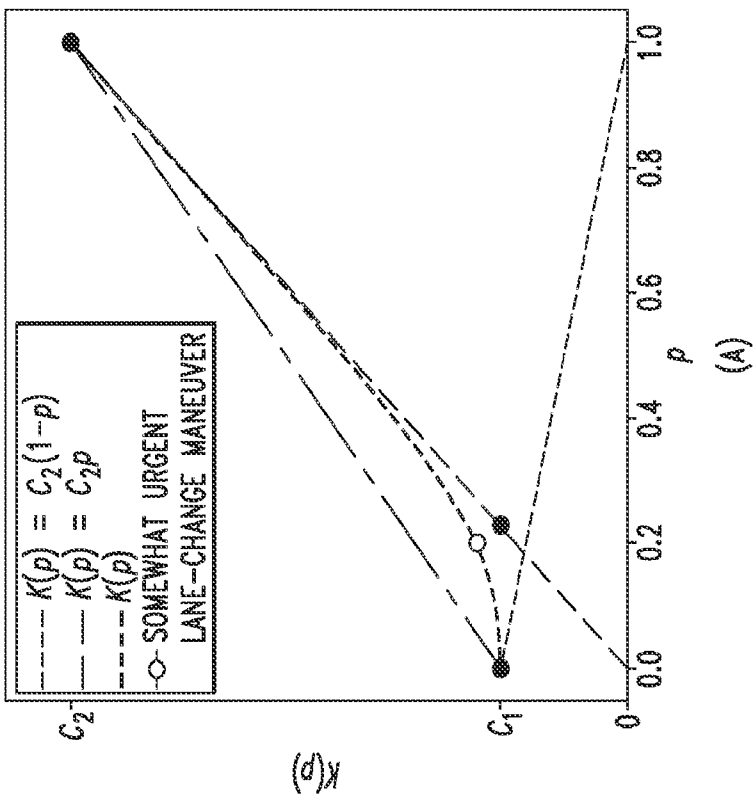
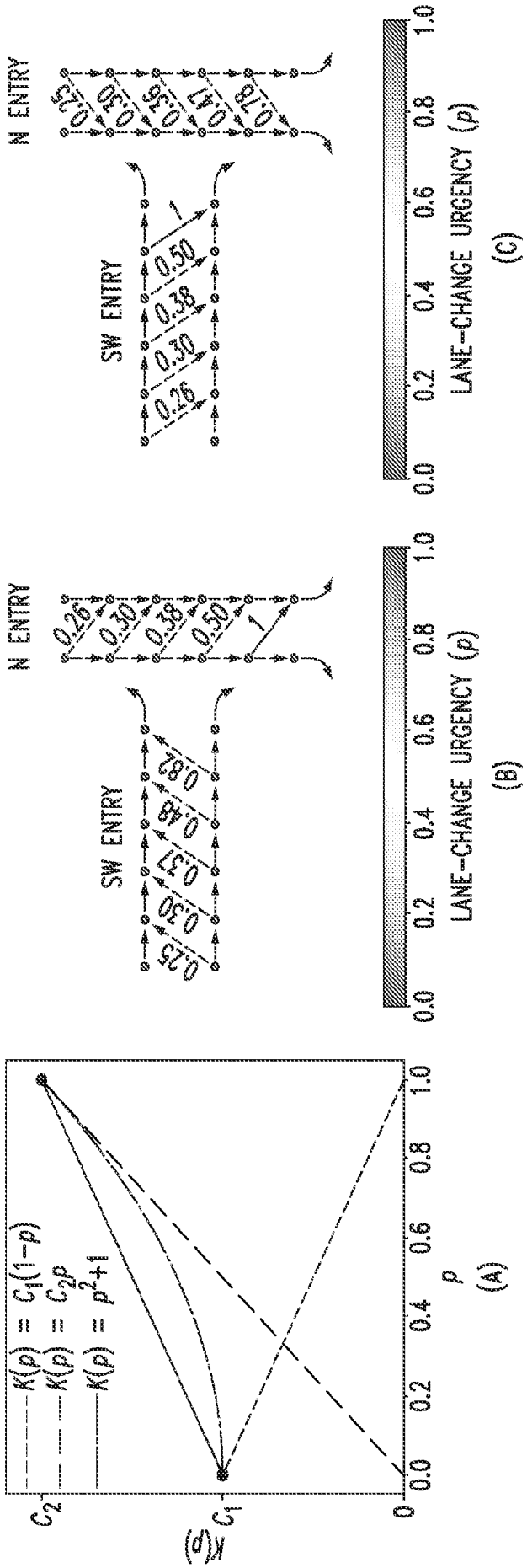

$K(\rho)$
C_2
C_1
0
0.0
0.2
0.4
0.6
0.8
1.0
ρ
(A)
$K(\rho) = C_2(1-\rho)$
$K(\rho) = C_2\rho$
$K(\rho)$
SOMEWHAT URGENT LANE-CHANGE MANEUVER
0.19
0.19
0.19
0.18
0.18
0.17
0.17
0.14
0.14
0.37
0.96
0.30
0.24
0.0
0.2
0.4
0.6
0.8
1.0
LANE-CHANGE URGENCY (ρ)
(B)

FIG. 19


N ENTRY
0.25
0.30
0.35
0.47
0.78
SW ENTRY
1
0.50
0.38
0.30
0.26
0.0
0.2
0.4
0.6
0.8
1.0
LANE-CHANGE URGENCY (ρ)
(C)
N ENTRY
0.26
0.30
0.38
0.50
1
SW ENTRY
0.78
0.48
0.37
0.30
0.25
0.0
0.2
0.4
0.6
0.8
1.0
LANE-CHANGE URGENCY (ρ)
(B)
K(ρ) = c1(1−ρ)
K(ρ) = c2ρ
K(ρ) = ρ²+1
c2
c1
K(ρ)
0
0.0
0.2
0.4
0.6
0.8
1.0
ρ
(A)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 23/80288

A. CLASSIFICATION OF SUBJECT MATTER

IPC - INV. B60W 60/00 (2024.01)
ADD. G05D 1/00, G06V 20/58 (2024.01)

CPC - INV. B60W 60/001

ADD. G05D 1/0088, B60W 30/18163, G06V 20/588

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
See Search History document

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
See Search History document

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
See Search History document

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X --- Y	US 2019/0258251 A1 (NVIDIA CORPORATION), 22 August 2019 (22.08.2019), entire document, especially Abstract; para [0120]-[0121], [0127], [0131], [0174], [0262], [0470], [0515], [0592], [0636], [0648], [0653], [0774]	1-2, 11-15, 18 ---------- 3-10, 16-17, 19-20
Y	US 2019/0333381 A1 (MOBILEYE VISION TECHNOLOGIES LTD.), 31 October 2019 (31.10.2019), entire document, especially Abstract; para [0232], [0319]	3, 16, 19
Y	CLAWSON. SHORTEST PATH PROBLEMS: DOMAIN RESTRICTION, ANYTIME PLANNING, AND MULTI-OBJECTIVE OPTIMIZATION. January 2017 (01.2017), [online], [retrieved on 2024-01-27]. Retrieved from the Internet <https://ecommons.cornell.edu/items/0e5663e6-ea3d-4a1a-8253-7c28e42010a0>, excerpts, especially Abstract; pg 4, 20, 25, 63	4, 7-10, 17, 20
Y	US 10,434,935 B1 (NISSAN NORTH AMERICA, INC.), 08 October 2019 (08.10.2019), entire document, especially Abstract; col 20, ln 43-52; col 36, ln 64 to col 37, ln 8	5-6

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"D" document cited by the applicant in the international application

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
23 January 2023	MAR 0 8 2024
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-8300	Authorized officer Kari Rodriquez Telephone No. PCT Helpdesk: 571-272-4300