US Patent & Trademark Office
Patent Public Search | Text View

United States Patent Application Publication 20250265846
Kind Code A1
Publication Date August 21, 2025
Inventor(s) Schroeter; Derik et al.

LANDMARK MATCHING IN ENVIRONMENT RECONSTRUCTION SYSTEMS AND APPLICATIONS

Abstract

Approaches presented herein provide for the matching and alignment of features in different instances of sensor data captured for an environment. At least one embodiment provides for accurate identification of matching landmarks between two or more tracks obtained from sensor-equipped machines. Track information can be collected to identify a number of landmarks within a region, and edges can be determined between landmarks that are within a maximum or determined distance from one another, forming edges that extend from one landmark to other landmarks within that distance to create a landmark graph. Landmark graphs for multiple tracks may be compared to identify corresponding edges. A set of corresponding edges for individual landmarks can be selected and counted to determine whether the edges between the different tracks satisfy a correspondence criterion or exceeds a correspondence threshold value, which is indicative of a matching landmark between the different tracks.

Inventors: **Schroeter; Derik (Newark, CA), Wu; Mengxi (San Jose, CA), Liu; Tian (Shanghai, CN)**

Applicant: **Nvidia Corporation** (Santa Clara, CA)

Family ID: **1000007817661**

Appl. No.: **18/595880**

Filed: **March 05, 2024**

Foreign Application Priority Data

WO	PCT/CN2024/077901	Feb. 21, 2024

Publication Classification

Int. Cl.: **G06V20/56** (20220101)

U.S. Cl.:

CPC **G06V20/56** (20220101);

Background/Summary

CROSS-REFERENCE TO RELATED APPLICATIONS [0001] This application claims priority to PCT Application Serial No. PCT/CN2024/077901 filed Feb. 21, 2024, and entitled “LANDMARK MATCHING IN ENVIRONMENT RECONSTRUCTION SYSTEMS AND APPLICATIONS,” which is hereby incorporated herein in its entirety and for all purposes.

BACKGROUND
[0002] There are various operations—such as may relate to autonomous or semi-autonomous navigation, as well as robotic simulation—where it can be desirable to generate or reconstruct a realistic digital and/or virtual environment that complies with real-world rules and constraints. As an example, maps—such as high definition (HD) maps, standard definition (SD) maps, navigation maps, etc.—are widely relied upon for semi-autonomous and autonomous operations. Autonomous and semi-autonomous vehicles and machines may rely on these maps, as well as real time sensor data, for navigation, localization, path or route planning, and/or other operations. In many instances, accurate map data depends in part upon sensor data captured by vehicles driving along various roadways or thoroughfares. In order to ensure accuracy of the information, multiple passes or tracks of data are captured for each section of roadway, which can help to ensure that relevant landmarks are represented in the sensor data, and not obstructed by another vehicle or object, and can also help to identify permanent versus temporary objects in a region. Unfortunately, sensor data often comes with some amount of error or imprecision which must be accounted for. Prior approaches assumed an underlying rigid transform between sets of corresponding features, but such an assumption does not hold in the context of landmark matching. Further, prior nearest neighbor-based approaches are very resource intensive and can result in delays to map updates, which can be problematic for tasks such as autonomous navigation.

Description

BRIEF DESCRIPTION OF THE DRAWINGS
[0003] Various embodiments in accordance with the present disclosure will be described with reference to the drawings, in which:
[0004] FIG. **1** illustrates example data produced at stages of an environment reconstruction process, according to at least one embodiment;
[0005] FIGS. **2**A and **2**B illustrate a capture and display of sensor data for an environment, according to at least one embodiment;
[0006] FIGS. **3**A-**3**F illustrates example landmark graphs that can be generated, according to at least one embodiment;
[0007] FIG. **4**A illustrates an example aligned landmark graph, according to at least one embodiment;
[0008] FIG. **4**B illustrates an example set of landmark matches between two tracks, according to at least one embodiment;
[0009] FIG. **5**A illustrates an example environment reconstruction system, according to at least one embodiment;
[0010] FIG. **5**B illustrates an example map generation system, according to at least one embodiment;

[0011] FIG. **6** illustrates components of a distributed system that can be used to match and align landmarks for an environment, according to at least one embodiment;
[0012] FIG. **7** illustrates an example process for matching and aligning landmarks in multiple data tracks, according to at least one embodiment;
[0013] FIG. **8** illustrates an example data center system, according to at least one embodiment;
[0014] FIG. **9** is a block diagram illustrating a computer system, according to at least one embodiment;
[0015] FIG. **10** is a block diagram illustrating a computer system, according to at least one embodiment;
[0016] FIG. **11** illustrates a computer system, according to at least one embodiment;
[0017] FIG. **12** illustrates a computer system, according to at least one embodiment;
[0018] FIG. **13** illustrates exemplary integrated circuits and associated graphics processors, according to at least one embodiment;
[0019] FIGS. **14**A, **14**B illustrate exemplary integrated circuits and associated graphics processors, according to at least one embodiment;
[0020] FIG. **15** illustrates a computer system, according to at least one embodiment;
[0021] FIG. **16**A illustrates a parallel processor, according to at least one embodiment;
[0022] FIG. **16**B illustrates a partition unit, according to at least one embodiment;
[0023] FIG. **17** illustrates at least portions of a graphics processor, according to one or more embodiments.
[0024] FIG. **18**A illustrates an example of an autonomous vehicle, according to at least one embodiment;
[0025] FIG. **18**B illustrates an example of camera locations and fields of view for the autonomous vehicle of FIG. **18**A, according to at least one embodiment;
[0026] FIG. **18**C is a block diagram illustrating an example system architecture for the autonomous vehicle of FIG. **18**A, according to at least one embodiment; and
[0027] FIG. **18**D is a diagram illustrating a system for communication between cloud-based server(s) and the autonomous vehicle of FIG. **18**A, according to at least one embodiment.

DETAILED DESCRIPTION

[0028] In the following description, various embodiments will be described. For purposes of explanation, specific configurations and details are set forth in order to provide a thorough understanding of the embodiments. However, it will also be apparent to one skilled in the art that the embodiments may be practiced without the specific details. Furthermore, well-known features may be omitted or simplified in order not to obscure the embodiment being described.
[0029] The systems and methods described herein may be used by, without limitation, non-autonomous vehicles or machines, semi-autonomous or autonomous vehicles or machines (e.g., in one or more advanced driver assistance systems (ADAS), one or more in-vehicle infotainment systems, one or more emergency vehicle detection systems), piloted and un-piloted robots or robotic platforms, warehouse vehicles, off-road vehicles, vehicles coupled to one or more trailers, flying vessels, boats, shuttles, emergency response vehicles, motorcycles, electric or motorized bicycles, aircraft, construction vehicles, trains, underwater craft, remotely operated vehicles such as drones, and/or other vehicle types. Further, the systems and methods described herein may be used for a variety of purposes, by way of example and without limitation, for machine control, machine locomotion, machine driving, synthetic data generation, generative AI, model training or updating, perception, augmented reality, virtual reality, mixed reality, robotics, security and surveillance, simulation and digital twinning, autonomous or semi-autonomous machine applications, deep learning, environment simulation, data center processing, conversational AI, light transport simulation (e.g., ray-tracing, path tracing, etc.), collaborative content creation for 3D assets, generative AI, cloud computing, and/or any other suitable applications.
[0030] Disclosed embodiments may be comprised in a variety of different systems such as

automotive systems (e.g., an in-vehicle infotainment system for an autonomous or semi-autonomous machine, a perception system for an autonomous or semi-autonomous machine), systems implemented using a robot, aerial systems, medial systems, boating systems, smart area monitoring systems, systems for performing deep learning operations, systems for performing simulation operations, systems for performing digital twin operations, systems implemented using an edge device, systems incorporating one or more virtual machines (VMs), systems for performing synthetic data generation operations, systems implemented at least partially in a data center, systems for performing conversational AI operations, systems implementing one or more language models—such as large language models (LLMs), systems for performing generative AI operations (e.g., using one or more language models), systems for performing light transport simulation, systems for performing collaborative content creation for 3D assets, systems implemented at least partially using cloud computing resources, and/or other types of systems.

[0031] Approaches in accordance with various illustrative embodiments provide for the matching of landmarks contained in different datasets. In particular, various embodiments provide for accurate identification of matching landmarks between two or more tracks obtained from sensor-equipped vehicles. Due at least in part to the error or imprecision in various sensors, different tracks within the same region may have some differences or offsets of landmark locations represented in those tracks of data. While prior approaches attempt to generate a rigid transform to fit different tracks into a common frame of reference, approaches in accordance with various embodiments can attempt to identify landmark correspondences between different tracks, and use data for those correspondences to aggregate the track data to a common reference. In at least one embodiment, track information can be collected to identify a number of landmarks within a region, and edges can be determined between landmarks that are within a maximum or determined distance from one another—such as within a maximum radius from each other—forming edges that extend from one landmark to other landmarks within that distance to create a landmark graph. Landmark graphs for multiple at least partially overlapping tracks may then be compared to identify corresponding edges. For example, a corresponding edge may be one that extends from a “same” or “common” landmark because the edges may be approximately the same length (e.g., within a threshold length difference), starting from approximately the same location of the common landmark, and in generally the same direction (e.g., within a threshold angular range). A set of corresponding edges for individual landmarks can be selected and counted, or otherwise accounted for, to determine whether the corresponding edges between the different tracks satisfies a correspondence criterion or exceeds a correspondence threshold value, for example, which is indicative of a matching landmark between the different tracks. A transform can then be performed for these matching landmarks to cause them to be aligned to a common frame of reference. These aligned landmark matches can then be provided for use in various operations, such as to generate or update map data for a region, or to perform auto-labeling for various landmarks, among other such options.

[0032] Variations of this and other such functionality can be used as well within the scope of the various embodiments as would be apparent to one of ordinary skill in the art in light of the teachings and suggestions contained herein.

[0033] FIG. **1** illustrates an example data processing flow that can be implemented in an environment representation and/or reconstruction system in accordance with at least one embodiment. In this example, sensor data **104** (or other raw data captured or representative of an environment) is obtained with respect to a specific environment **102**. The environment can be any appropriate physical environment, such as an indoor or outdoor environment that may include any number of different types of objects or elements. The sensor data can include data captured or obtained using any of a number of different types of sensors, as may include cameras, LiDAR systems, RADARs, sonic sensors, distance sensors, and the like. Additional data may be obtained that relates to the environment **102** as well in various embodiments, as may relate to basic map

data, contextual data, motion data, or other such data, which may also be obtained for virtual, augmented, or enhanced environments. In this example, the sensor data **104** (and any other available and useful data) can be used to generate an initial representation **106** of the environment **102**. In at least one embodiment, this may include a point cloud representation of the environment **102** generated by analyzing and aggregating the sensor data **104** that may have been captured by multiple sensors in order to generate a single, n-dimensional (e.g., 2D, 3D, or 4D) representation of the environment. Other initial representations can be generated as well, as may depend at least in part upon the type of sensor data provided. If image data is provided, the image data may be analyzed to attempt to determine feature and depth information, which can be combined from multiple images from different viewpoints to attempt to generate at least a 3D representation of the environment **102**, or at least objects and shapes within that environment.

[0034] This initial representation **106** of the environment **102** can be analyzed to attempt to determine specific aspects **108** of the environment. For example, a point cloud can be analyzed to attempt to determine the categories (or types) of objects represented in the environment, as may relate to roadways, traffic signs, sidewalks, buildings, and the like. The representation can also be analyzed to attempt to determine the locations of these objects in the environment, as may be defined using a set of 3D coordinates relative to a determined origin location. The initial representation **106** can also be analyzed to attempt to determine various relationships between these objects, such as where a crosswalk crosses specific lanes or where a stop sign is associated with a specific lane and indicates an expected behavior. Once these determined aspects **108** are obtained, these aspects can be used to generate an object-based representation **110** of the environment **102**. Various other types of representations can be generated as well within the scope of various embodiments. As illustrated, the object-based representation **110** will not be a comprehensive description of the environment **102** in this example, but will instead focus on the types of objects or features of the environment that are potentially relevant to a particular task. For autonomous driving, for example, the object-based representation may include objects such as road lanes, crosswalks, intersections, and the like, but may not include objects that may not be directly relevant to driving, as may include buildings, billboards, mailboxes, and other such objects, except to the extent those objects may be relevant to a specific operation or task. In this example, the object-based representation **110** also does not include vehicles, pedestrians, or other movable objects that will only be in specific locations in the environment **102** at specific times, but any or all of these and other such objects could be included in the representation as well within the scope of various embodiments.

[0035] From this object-based representation, an object graph **112** can be generated that provides a different representation of the environment **102**. An advantage of the object graph **112** is that it is relatively lightweight, and can be used to compactly describe aspects of the environment **102** that are important for a particular task or operation. For example, such an object graph **112** could be provided to a map generator in order to generate an HD map (or other such map or representation) that can be provided to an autonomous vehicle to make navigation decisions. Such an object graph **112** can also be provided as input to an environment generator that can generate a realistic 3D virtual environment that can be used for tasks such as robotic simulation or digital world recreation. A large number of object graphs can be stored to represent a number of different environments, which can require significantly less memory or storage capacity than sensor data, such as a large number of high-resolution images. Such object graphs can also be analyzed quickly to allow for (near) real-time operations, such as autonomous navigation or control.

[0036] As part of such a map generation process, sensor data may be captured from a large number of vehicles using a variety of different sensors, or types of sensors. Each of these sensors may have different amounts of imprecision or error. Further, the accuracy of the sensor data can be impacted by things such as environmental conditions, obstructions, object motion, vehicle motion, and the like. Further still, there will be some imprecision in the determined location of the vehicle due to

imprecision in location determined by a GPS system, for example, where the strength of the GPS signal may vary by location or condition, in addition to the inherent accuracy limitations of the system itself. Based in part on these and other such factors, the sensor data obtained from various vehicles, or even different instances of the same vehicle, will have some differences in location data for various objects or landmarks in a region.

[0037] In at least one embodiment, tracks of sensor data can be obtained from various vehicles during operation. “Tracks” as used herein refer to low bandwidth, feature-based data streams collected by sensor-equipped vehicles operating in a region, such as by driving along roads in that region. Tracks may comprise information relating to ego-motion of the respective ego-vehicle and geo-positional data (e.g., GPS data for the ego-vehicle), as well as data corresponding to local landmarks (e.g., signs, signals, or poles), lane dividers, parking spaces, RADAR features, and the like. A set of tracks may be received or obtained that correspond to a given region, or sub-region. In order to obtain an accurate representation of the landmarks, lane dividers, and other objects or aspects of the region, an attempt can be made to align these tracks, such as with respect to a prior map of the region and/or with respect to each other in a common frame of reference. Alignment can involve accurately identifying correspondences between features present in multiple tracks. These feature or landmark correspondences can be used with other information, such as ego-motion, geo-position, and other constraints to align and register these tracks together in a common coordinate frame. In at least one embodiment, an optimization-based approach to alignment can be used, which may iterate over the data from the various tracks. Once proper alignment is obtained then the aligned track data can be used for purposes such as map creation and auto labelling.

[0038] FIG. **2**A illustrates an example view **200** of a vehicle **202** navigating through a region of an environment. The vehicle can include a number of sensors (e.g., LiDAR, RADAR, camera, distance, and the like) that are able to capture data about landmarks in the region that are within a view **214** or capture range of at least one sensor. This may include 3D point data for features associated with various objects near the vehicle. The captured sensor data can be analyzed (along with other relevant information) to attempt to identify landmarks in the sensor data. The identification can be performed using any appropriate algorithm or machine learning model, for example, and can include a bounding box or other representation to be used for analysis. In the example of FIG. **2**A, landmarks may be identified such as may correspond to a traffic pole **204**, a traffic light **206**, or a traffic sign **208**, which can be three-dimensional (3D) in nature. Other objects or features may be identified as well, such as crosswalks **212** or lane markers **210**, but those will generally be substantially 2D in nature and can be treated differently than landmarks as discussed elsewhere herein.

[0039] In order to ensure to capture data for all relevant landmarks, as well as to provide additional measurements or positional determinations for those landmarks to help account for error or imprecision in the sensor data, multiple vehicles (or passes of the same vehicle) can occur through this region to attempt to capture multiple instances of sensor data for each landmark. The vehicle could travel in either direction in this example, and for road with additional lanes could travel in any of the lanes. Even when traveling in the same lane, different vehicles or passes along that route will not follow the exact same trajectory, even if remaining in the same lane. As a result, sensor data from different tracks will show landmarks in different positions, as illustrated in example view **250** in FIG. **2**B. As mentioned, some of this will be due to the differences in the locations of the vehicles or sensors for each pass. As illustrated, there can also be differences due to error or inaccuracies in the captured sensor data. These errors will not all be in the same direction or of the same measure. For example, the difference in position between a first representation **252** and a second representation **254** of a traffic light is illustrated to show differences in height, which is not illustrated by other landmarks, such that this is likely due to error in the sensor data. If the sensor data were accurate and offsets were only due to sensor or vehicle position, then landmark matching and alignment could be relatively straightforward. When each landmark may have imprecision in

any random direction in any given track, however, the alignment and matching becomes more difficult. Further, the ability of any landmark to have an unknown amount of imprecision in any given direction can make it difficult to determine an appropriate frame of reference to use consistently for the sensor data for all relevant tracks.

[0040] One approach to landmark matching and alignment, and in particular pairwise feature-based track matching, involves finding corresponding features between one or two tracks of data (as may be contained within one or more data streams) with one or more track-overlaps. Identified correspondences can be used for various tasks or operations, such as to formulate and solve a non-linear 3D pose graph optimization problem. Features in this context can include 3D landmarks such as signs, signals, poles, and waitlines, as well as 3D lane-dividers. Landmarks are features with well-defined 3D positions whereas lane-dividers only provide lateral constraints. In at least one embodiment, each feature can be associated with a single pose, and features can be expressed by means of relative coordinates with respect to their associated poses, such that feature correspondences imply constraints between these associated poses. Additional constraints can come from ego-motion, which constrain subsequent poses within a track, and geo-location (e.g., GPS) measurements, which constrain poses with respect to a global/common reference-frame. Track overlaps can refer to two ranges of poses from one or two tracks that are assumed to locally overlap, such that the tracks are assumed to go along the same road in the same or opposite direction. Track overlaps can be used as additional inputs to a feature matching and alignment process in at least one embodiment.

[0041] It is generally assumed that the prior global poses for tracks were determined by fusing ego-motion with GPS-measurements. How well two tracks initially align using these prior approaches can then depend heavily on the accuracy of GPS-measurements, which in turn depends on fixed and variable environmental factors as well as on the type of GPS. As a result, the translation between two tracks is somewhat arbitrary subject to some maximum displacement, but is commonly larger than the distance between nearby landmarks. Similarly, the local rotation between two tracks is assumed to be bounded by some maximum rotation, which, however, cannot be assumed to be arbitrarily small. In addition, 3D landmarks lack strong descriptive properties, in particular for poles and waitlines, which are primarily vertical and horizontal line-segments, respectively. The size and/or extents of landmarks can vary significantly. Likewise, orientations for signs and signals are not reliably estimated, and can vary significantly, in particular when the trajectories go around turns in opposite directions.

[0042] Various prior approaches used variants of an iterative closest points (ICP) or similar approach that assumes an underlying rigid transform between sets of corresponding features. Such an assumption often does not hold in the context of landmark matching. Approaches in accordance with various embodiments do not make such an assumption, which allows these approaches to be highly robust to ambiguities and more invariant to factors such as translation and rotation between tracks. In at least one embodiment, correspondence or “matching” is solved for track pairs with known overlaps. A landmark graph can be created for each track, or at least a subset of received tracks for a region, where the nodes of the graph correspond to individual landmarks. The edges of the graph then each connect two landmarks in the region. In at least one embodiment, each landmark of a graph will have an edge connected to every other landmark of the graph, or at least landmarks that satisfy a given connection criterion—such as having at least a minimum level of confidence or being within a maximum or threshold distance of each other. Pairs of tracks can be compared to attempt to locate correspondences between landmark edges, such as by using the edge geometry and landmark properties. A given landmark edge match, or correspondence, implies two landmark matches or correspondences. In at least one embodiment, however, multiple matching landmark edges adjacent to a given landmark are analyzed and/or inspected. A landmark or correspondence between tracks is determined to be established if a majority, or minimum number or percentage, of the adjacent matching landmark edges imply the same landmark match.

[0043] FIG. **3**A illustrates an example landmark graph **300** that can be generated in accordance with at least one embodiment. This graph is not a complete landmark graph, but only shows landmark edges **304** from a single landmark **302** to other landmarks **306** of this track for a given region. When analyzing data for a given track, a process in accordance with at least one embodiment can attempt to find matches or correspondences. This can involve forming landmark-pair-segments (LPSs) separately for the landmarks in each track. These LPSs can then be matched pairwise between two or more tracks, with identified LPS matches being used to determine a set of landmark matches. Such an approach can compute the geometric criteria using three-dimensional (3D) or two-dimensional (2D) points where a dimension such as altitude is ignored due to its lower accuracy.

[0044] Given the landmarks from the track overlap range of a single track, landmark pair segments (LPSs) can be formed from pairs of landmarks. Pairs are used—rather than triplets or quadruplets, for example—in at least one embodiment to limit the implied combinatorial explosion. If locally there are 30 landmarks then this implies 435 pairs, 4,060 triplets, and 27,405 quadruplets. Further, matching of pairs can be much simpler and faster than matching triplets or quadruplets. Still, even with pairs there can be quadratic growth with respect to the resulting number of possible pairs. To make this process scale linearly with the number of landmarks, an approach in accordance with at least one embodiment will use only a number N of the closest landmarks to form LPSs. The value N may be determined experimentally or set by a user or application, among other such options. In one example implementation, N is set to 30, but the appropriate number can vary based upon various factors, such as region size, landmark density, resource capacity, and the like. In addition, LPSs may be subject to minimum length and maximum length criteria. LPSs can also be created separately for individual tracks. As mentioned, FIG. **3**A illustrates a landmark graph **300** for a single track of data, which can be compared to graphs for other tracks.

[0045] FIGS. **3**B and **3**C illustrate zoomed-in views **310**, **320**, and FIG. **3**C illustrates a top-down, zoomed out view of a set of LPS matches that can be determined in accordance with at least one embodiment. The LPSs **332**, **334** adjacent to the poles are illustrated in black and gray, respectively, for a given overlapping track pair. As mentioned, there can be an attempt to match two unordered sets scales quadratically with the number of elements in these sets. To this end, similar to LPS creation discussed above, only the N (e.g., **30**) closest landmark pair segments (LPSs) are considered for matching. It should be understood, however, that the number of landmarks and the number of landmark pair segments can differ as well in various embodiments. An applicable distance threshold can be selected or determined that is comparably large, as tracks can have relatively large initial displacements, along with possible rotation. The distance can be measured between LPS centers. For the purpose of pairwise LPS matching, the landmarks of two LPSs can first be ordered such that the dot-product of the LPS orientations is larger than zero, such that the LPS direction vectors point roughly in the same direction. LPS matches that satisfy the matching criteria can be used for subsequent landmark-matching.

[0046] As another example, FIG. **3**D illustrates a landmark graph **330** for landmarks of a first track, and FIG. **3**E illustrates a landmark graph **340** for landmarks of a second track. These graphs can be compared using landmark pairs, such as those within a given distance range of each other, to attempt to find matching landmark pairs, and matching landmark pair segments. FIG. **3**F illustrates a landmark graph **350** that represents only the matching landmark pair segments between the two tracks. This graph can be compared to other, similar graphs, and aligned to a common frame of reference as discussed elsewhere herein.

[0047] LPS matching can consider various matching criteria. In at least one embodiment, this can include a relative length difference check. Given LPS-lengths L.sub.1 and L.sub.2, the relative length-difference ΔL is below ΔL.sub.threshold, as may be given by:

[00001] $$L = \frac{\text{.Math. } L_1 - L_2 \text{ .Math.}}{\max(L_1, L_2)} < L_{\text{threshold}}$$

[0048] Another example criterion is an angle difference check. Given the normalized LPS direction

vectors v.sub.1 and v.sub.2, (in a common 2D frame of reference), the angle-difference Δα must be below Δα.sub.threshold, or in practice, cos(Δα) must be above cos (Δα.sub.threshold), as may be given by:

[00002] $\cos(\Delta\alpha) = v_1 \text{ .Math. } v_2 > \cos(\Delta\alpha_{\text{threshold}})$

The threshold used can be relatively large to account for the possible rotation between the tracks, as well as orientation-differences caused by inaccurate landmark-positions. The corresponding landmarks should match as per prior ordering in at least one embodiment.

[0049] In this example, the LPSs are to pass a parallelogram check with respect to the vectors between the start points and the end points, respectively. Given LPS-points (p.sub.1,1, p.sub.1,2) and (p.sub.2,1, p.sub.2,2), where p.sub.1,i corresponds to p.sub.2,i as per prior ordering, the start vectors and end vectors can be given by:

[00003] $v_{\text{start}} = p_{2,1} - p_{1,1} \quad v_{end} = p_{2,2} - p_{1,2}$

Here, the test asserts that:

[00004] $$\begin{aligned} &\text{.Math. } v_{\text{start}} - v_{end} \text{ .Math. } < \text{StartEnd}_{\text{threshold}} \\ &\cos(v_{\text{start}} \text{ .Math. } v_{end}) > \cos(\quad_{\text{threshold}}) \end{aligned}$$

[0050] Matching between landmarks can consider landmark-type information, such as specific sign-type, and geometric measurements like width, height, and normal, as applicable. The geometric matching criteria can be formulated as a relative length difference check and angle difference check, both are the same as used above, though with different thresholds, as may be given by:

[00005] $$\begin{aligned} &L = \frac{\text{.Math. } L_1 - L_2 \text{ .Math.}}{\max(L_1, L_2)} > L_{\text{threshold}} \\ &\cos(\Delta\alpha) = v_1 \text{ .Math. } v_2 > \cos(\quad_{\text{threshold}}) \end{aligned}$$

[0051] A variety of checks can be applied, subject to the matching-parameters for specific landmark-types. This can include, for example, checks for poles, such as where pole types must match, pole lengths satisfy relative length difference checks, and pole directions satisfy angle difference checks. with pole-directions. For signs, checks can include checking that sign types match, sign colors match, relatively length different checks are satisfied separately for width/height and height-above-ground, and/or that sign normals satisfy angle difference checks, among other such options. For signals, these can include relative length difference checks separately for width/height and height-above-ground, as well as angle difference checks with signal-normals. For waitlines, these can include relative length difference checks with the wait line lengths, as well as angle difference checks with wait line directions.

[0052] Taking such an approach, each LPS match can effectively imply two potential landmark matches, and multiple LPS-matches can imply the same landmark matches. To determine landmark matches that are consistent with all (or at least most) of their adjacent LPS matches and the implied adjacent landmark matches, the landmark matching information implied by LPS matches can first be organized for all landmarks from both tracks. This can include, for each landmark, collecting all adjacent LPS matches (i.e., LPS matches that include the landmark), and grouping the adjacent LPS matches according to the landmark match implied for the given landmark. Landmark matches that do not pass a general sanity check (or other such) can be discarded. For each remaining landmark match, the list of implied adjacent landmark matches can be stored, and sorted in decreasing order by number of implied adjacent landmark matches.

[0053] The following provides an illustration of example matching information per landmark: [0054] Landmark M from track 1 [0055] Landmark I from track 2 [0056] List of implied adjacent landmark-matches. [0057] Landmark J from track 2 [0058] List of implied adjacent landmark-matches. The size is equal or less than the above due to the sorting. [0059] Landmark K from track 2 [0060] List of implied adjacent landmark-matches. The size is equal or less than the above due to the sorting.

[0061] In at least one embodiment, all landmark matches that are kept in the matching information must pass certain sanity checks. For example, the landmark matches must be supported by at least two adjacent LPS-matches, and must not imply the same poses (which would not form a valid or useful constraint) or the same landmarks. This can happen if the two tracks are the same, and the respective track overlap ranges comprise some of the exact same poses. For opposite direction tracks, landmarks cannot match if they are from different sides (left/right) of the trajectory. For same direction tracks, poles cannot match if they are from different sides (left/right) of the trajectory, while such a criterion may be unable to be enforced for other landmark-types (e.g., signs or signals) as they often hang over the middle of the road, and, thus, can be between same direction tracks. FIG. **4**A illustrates a top-down, zoomed out view **400** of a set of LPS matches that can be determined in accordance with at least one embodiment.

[0062] Such matching information can be used to generate a landmark match graph, where nodes present landmark-matches, and edges come from LPS-matches. Such a graph can contain some number of incorrect landmark matches, as may be implied by incorrect LPS matches. Between the correct landmark matches, however, the graph should be consistent. In at least one embodiment, an example process can be used to determine the correct landmark matches, and followed by a consistency check with respect to implied adjacent landmark-matches. In such a process, a landmark match can be established for each landmark L1 in track 1 if certain criteria are met. This can include that, for the first landmark match in the list, the number of adjacent landmark matches must be above a determined threshold, and there must be a unique maximum, in that the number of adjacent landmark matches must be larger than the second largest, if it exists. More specifically, the ratio between second and first largest must be below a determined threshold. For the matching landmark L2 from track 2, it must refer to L1 as the best landmark-match. There must also be a unique maximum, in that the number of adjacent landmark matches must be larger than the second largest, if it exists. More specifically, the ratio between second and first largest must be below a determined threshold. It should be noted that the number of adjacent landmark-matches for L2 is the same as for L1 due to the symmetry of LPS matches.

[0063] Afterwards, a consistency check can be performed based on the assumption that the majority of landmark matches established above are correct. For each landmark-match, the number of adjacent landmark matches are counted that are considered to be correct according to the above criteria. A landmark match can be discarded if the ratio of the number of correct and overall landmark matches falls below a determined threshold. This ratio is one if all landmark matches are considered to be correct, and zero if all are considered to be wrong. FIG. **4**B illustrates a set **450** of landmark matches between two tracks at an intersection. In this example, traffic signs **452** are illustrated in black, and traffic signals are illustrated in gray.

[0064] As mentioned, the alignment of landmarks between different tracks of data can allow these tracks to be aligned and/or registered with each other and/or with respect to prior map data for a region. Once so aligned, the track data can be provided for use in various purposes, such as for map creation, auto-labeling, or other such purposes. This aligned data can be used in a process such as that described with respect to FIG. **1** to generate or update a map representation of a region. While discussed using sensors on a vehicle driving on a roadway, to be used to generate high quality map data for navigating vehicles, the sensor data can be captured and used to generate maps or representations of other environments as well, such as factories or warehouses in which robots will operate. In such an example, one or more robots can make one or more passes through at least a portion of the warehouse while capturing sensor data, and the sensor data for each of those passes can correspond to a track of data. In order to correlate the data and improve the accuracy, the data can be aligned to a common frame of reference using matching landmarks, as discussed and suggested herein. Using multiple tracks can also help to ensure that landmarks are captured that may not be represented in all tracks, and that temporary objects are not to be interpreted as landmarks even though they may appear in one or more (but not all or a majority of the) tracks.

[0065] FIG. **5**A illustrates an example environment reconstruction pipeline **500** that can be used to generate a representation of an environment in accordance with at least one embodiment. Rather than requiring at least some amount of manual interaction, such an approach can automatically generate a representation from a variety of different types of input data. In this example, a capture device **502** can include, or be associated with, one or more sensors **504**, **506** that can capture or generate information about an environment **508**. The capture device **502** can include any device, system, or component that is able to obtain sensor data from one or more sensors and either process that sensor data or transmit that sensor data for processing, as may include a portable computer, a smart phone, a vehicle with data processing capability, or a robotic assembly, among other such options. The sensors can include any appropriate type of sensor that is able to capture or generate useful information about an environment, including sensors such as cameras, infrared (IR) sensors, ultrasonic sensors, depth sensors, LiDAR systems, RADAR systems, or other such sensors or data capture elements. The environment **508** can include an environment in which the capture device **502** is located, or that is within a capture distance of one or more sensors **504**, **506**.
[0066] In this example, the capture device **502** can provide the sensor data to be analyzed by a feature extraction module **510**. As mentioned, the feature extraction can be performed as part of a machine learning model, such as may be used by at least one alignment and optimization module **512**, or by a separate model or algorithm, among other such options. In at least one embodiment, the feature extraction module **510** may include an encoder that can extract features from the various instances of sensor data and encode those features as embeddings or points in a latent space **514**. The environment **508** in at least one embodiment can be represented by a set of embeddings or points in latent space, which may then be represented by one or more feature vectors corresponding to those individual embeddings. The latent space **514** may be an n-dimensional latent space, where each environment (or state of an environment) can correspond to a point (or vector) in the n-dimensional latent space. For algorithm-based approaches, the feature data may instead be stored as point cloud data or other such representations as discussed and suggested elsewhere herein.
[0067] In this example, at least a relevant portion of the feature data (in an appropriate form), as may correspond to two tracks of sensor data, can be provided as input to an alignment and optimization module **512**. Various types of embeddings or representations can be used within the scope of various embodiments. In at least one embodiment, each object (e.g., landmark or lane marker) in the environment can be represented, as discussed previously. Such a representation can specify not only the type of object, but can also represent various features or aspects of that object that can facilitate matching or other such operations.
[0068] The alignment and optimization module **512** can use this input to attempt to match and align landmarks or other features of the environment. In this example, the module might receive other input as well that may help to make more accurate matches. For example, the module might receive a prior or partial map or environment representation, which can help with consistency of representations over time, such as where the environment is being reconstructed for a vehicle moving through an environment and comparing the inferences for each time point can help to improve accuracy by reducing noise or removing false positives (or at least flagging inferences that do not make sense based on a prior determination, such as where an object type has changed or suddenly appeared out of nowhere). Various other types of input can be provided as well. For example, a user might use a client device **518**, such as a desktop computer or notebook computer, to provide input that can guide the generation of the tokenized text string. For example, the client device might **518** provide contextual information that can help to guide the generation. Contextual information might include, for example, a type of environment, such as indication of an urban or rural setting, which can help the module to apply the appropriate set of rules. The contextual information might indicate the state or country in which the sensor data was captured, as different states or countries often have different traffic or behavior rules, such as which lanes vehicles are allows to turn into at an intersection, which types of traffic signs or signals are used, types of lane

markers, etc.
[0069] Once matched and aligned features—such as landmarks—are output by the alignment and optimization module **512**, that output can be provided to various components for various tasks. In some embodiments, a reconstruction of the environment **508** might be performed by a reconstruction module **516** or system, such as to generate (or update) a high-definition map or 3D digital model of the environment **508**. In some embodiments, the output might be provided to a control or navigation system for an autonomous vehicle or robot to allow decisions to be made about how to move or interact with respect to objects in the environment. In this example, the initial capture device **502** might be on or part of a vehicle, or may in some embodiments be the vehicle (or robot, etc.) itself. The reconstruction of the environment can be provided back to the capture device for use in performing specific tasks. For example, if the capture device is an autonomous vehicle or driver assistance system, the reconstruction (or in some embodiments the tokenized text string) can be provided back to the capture device—which captured the initial sensor data using associated sensors **504**, **506**—to perform operations such as to make navigation or operation decisions based in part on the reconstruction.
[0070] In at least one embodiment, the reconstruction can be provided to a client device **518** for presentation or analysis, which may be the same client device that instructed the reconstruction. The client device **518** can analyze the reconstructed environment for accuracy and completeness in some embodiments, or can perform various operations or simulations with respect to the environment. The client device **518** may also provide additional information, such as context, to the reconstruction module to use to generate the environment. For example, the client device might instruct the reconstruction module **516** to generate multiple reconstructions of the same environment **508** using the same landmark data, but in different formats or using different criteria. This may include, for a simulation example, versions of the same environment in Europe versus Asia (which can impact the language and style used), and so forth. During model training, the environment reconstruction and/or aligned landmark match data can be compared against appropriate ground truth data in order to determine a loss value and update the parameters for the appropriate model.
[0071] In this example, the feature extraction and language generation operations may be part of the same or separate models. For example, a first model (e.g., an encoder) might take the sensor data as input and output a set of embeddings or latent feature vectors as output that can then be provided as input to a generative model (e.g., a generative deep learning model). In another embodiment, a generative model may include feature extraction or analysis capability, and can generate aligned feature match output without any intermediate or other steps to process or analyze the input sensor data. A generative model can be trained to take input from any of various stages of a representation generation pipeline. For example, a language model can take the raw sensor data as input, or can take as input an initial representation (e.g., a point cloud) generated by analyzing that sensor data using a separate module, system, component, model, algorithm, or process. Similarly, the model might take in determined aspects or information as may relate to the semantics, topology, or geometry of an environment, or might take as input an object-based representation generated for the environment, among other such options. In at least some embodiments, the type of input to be used may depend at least in part upon the system in which the generative model to be used, as different systems may already provide specific outputs to be used. In at least one embodiment, a generative model might take the raw sensor data and such an intermediate representation as input, in order to attempt to provide more accurate or consistent representations. In some embodiments, multiple generative models may be used. For example, a first model might be used to determine aspects of an environment that are then to be fed as input to another generative model.
[0072] In at least one embodiment, an environment generation and/or reconstruction system can work with various data formats, and can perform reformatting or restricting as appropriate. For

example, data might be received in map, object, or graph format and can be converted to a common format or representation for processing. Similarly, such a common format or representation can be used to generate any of these or other such representations of an environment.

[0073] In one example, a reconstruction model can be used to generate or correct a representation such as a high definition (HD) map. An HD map generally is a type of map used for tasks such as autonomous driving, which may contain details or information that are not typically included in, or associated with, a conventional map. In an example HD map, individual sections of a roadway are encoded separately. These encodings can differentiate regions corrupting to different lanes in an intersection, for example, as well as potentially options for navigating on those lanes. Such information can be helpful in an intersection where there may not be painted or explicit lane markers for each available lane in each direction. This information helps a navigation system to function more like a human would, having the ability to understand implicit information based on context, but using previous systems these aspects needed to be hard coded and were thus limited in scope and difficult to scale. Each feature in the road can be represented by a node in a graph associated with the HD map. A generative model can take this information, and can make corrections or additions based on its understanding of the relationships and semantics of the environment.

[0074] Approaches in accordance with various embodiments can provide for augmentation of perception data with local map information. Such approaches can provide for robust alignment of features in an environment **508**. As an example, FIG. **5**B illustrates an example environment representation generation or reconstruction system **530** that can be used in accordance with at least one embodiment. It should be understood that reference numbers can be carried over between figures for similar elements, but such usage should not be interpreted as a limitation on scope of the various embodiments. In this example, a capture device **502** can use sensors **504**, **506** to capture sensor data (or obtain other such observations) pertaining to an environment **508** using an approach such as that described with respect to FIG. **5**A, although other mechanisms or approaches can be used to obtain such data as well within the scope of the various embodiments. Further, additional data can be used to attempt to perceive information about at least a portion of the environment **508** as discussed in more detail elsewhere herein.

[0075] In this example, sensor data from the capture device **502** (along with potentially other observations) is provided to a perception module **532**. The capture device **502** may perform at least some amount of processing of the sensor data before providing it to the perception module **532**, as may include noise reduction, aggregation, correlation, redundant data point removal, and the like. The sensor data may be provided in any appropriate form(s), as may include image data, 3D point cloud data, feature vectors, and so on. An additional benefit to such an approach is that it can provide for enhanced system robustness, being able to identify and account for individual sensor inaccuracies or even failure. For example, a perception module **532** can perform at least some amount of processing of the data from individual sensors, and can determine when the data from one sensor is unreliable and should be modified, weighted by a lower amount, or discarded. In at least one embodiment, a perception module **532** can attempt to correct a value received from an individual sensor, based on data from other sensors or related sources, among other such options.

[0076] A perception module **532** in at least one embodiment can perform tasks including those discussed in more detail elsewhere herein, such as to extract features from the sensor data and attempt to identify objects in the environment, as well as to determine relevant information about those objects. Feature extraction or feature inference may be performed by an encoder in at least one embodiment to extract and encode features that may be relatively low-level and may not have a clear sematic meaning attached. The features may be used to generate a relatively universal and/or generic representation of the sensor data. The sensor or perception data can be interpreted and/or correlated in the cross-attention layer(s) of one or more trained models. Such a model can attempt to correlate related features to allow objects to be represented using shapes, such as may be

comprised of lines, triangles, or polygons, and can recognize and associate semantic information with the represented objects. In at least one embodiment, a model may analyze a feature vector including appearance information for an object, without any higher-level structure information, and attempt to determine various attributes relating to semantics, relationships, topology, geometry, and the like. An encoder thus may just attempt to represent the sensor data as faithfully and accurately as possible using a subset of points or embeddings, in a way that is friendly to downstream processing. A model (or other such component) receiving these features or embeddings can then attempt to make sense of these encodings using domain-specific knowledge.

[0077] Sensor data can be extracted and/or encoded in a number of different ways. For example, there may be one encoder per sensor so that each sensor can output a respective token stream that can be input to a model. A trained model can then fuse the information in the parallel streams with the map data as discussed herein. In other embodiments, sensor data fusion can be performed before generating the token stream. As an example, a point cloud representation of the environment around a vehicle can be generated using data captured by sensors around the vehicle, and this point cloud representation can be analyzed to generate the token stream. Correlation of the sensor data can be performed in such a way as calibration information for the sensors may already be available in many instances, such that position data can be determined with respect to a consistent coordinate system or frame of reference. The model can then analyze the consistent 3D representation to generate a single representation in at least one embodiment. The correlation of the sensor data can also address issues relating to multi-modality, as any of a number of different approaches can be used to interpret and correlate data from different types of sensors. For example, algorithms are available that can correlate appearance features extracted from a camera image with position data obtained from a LiDAR system, etc.

[0078] In at least one embodiment, a perception module **532** may attempt to identify only specific objects of interest, or types of objects, in order to reduce environment perception to a more manageable task. For navigation of a vehicle, for example, this may include detection of static and/or dynamic objects relevant to driving, as may include lane boundaries, traffic signals, other vehicles on the roadway, pedestrians within a range of the vehicle, and so forth. The perception module may determine that there are static objects away from the roadway, or what are otherwise unlikely to impact navigation, and may either classify those objects as unimportant or exclude those objects from identification, among other such options. In at least one embodiment, a perception module **532** will attempt to determine at least a relevant position of specific types of objects with respect to the ego vehicle, if not an absolute position with respect to some geographic origin or reference plane, point, or coordinate system. For objects that may be in motion, such as vehicles on the same roadway as the ego vehicle, this may include a position at a specific point in time or a range of positions over a window of time, such as a window having a length corresponding to the capture or refresh rate of the relevant sensor(s) used to determine the position. For objects in motion, the perception module **532** may also attempt to determine a direction and/or rate of motion, such as velocity, acceleration, or deceleration, as may be based in part on position or motion information determined for a prior point or window in time. In at least one embodiment, a perception module **532** can produce an accurate recreation or representation of the environment in which the ego vehicle is operating, in order to allow a control module **546**, process, or module to determine instructions for safely operating the vehicle within that environment to achieve a desired goal, such as to navigate the vehicle safely to a target destination.

[0079] As mentioned, in at least some embodiments, a vehicle—such as an autonomous or semi-autonomous vehicle or machine—can operate based on this perception data. In order to provide for a more accurate perception of at least a relevant portion of the environment **508**, however, a system in accordance with at least one embodiment can attempt to augment or improve accuracy of this perception data using local map information. In the example environment representation generation or reconstruction system **530** of FIG. **5**B, a mapping module **538** can access map data stored to a

map repository **540** or other such location. The map repository **540** may be available on the vehicle or accessible over a wireless data connection, for example, where relevant map data can be pre-fetched by the vehicle based on a current and/or anticipated location of the vehicle, such as for a given minimum distance of the vehicle or along a current navigation route. Pre-fetching can be used to attempt to ensure that the relevant map data is available even if the wireless network connection is weak, spotty, or otherwise unreliable or unavailable in a given location or region. The mapping module **538** in this example can work with a localization module **534** to attempt to determine a current geographic location of the vehicle. The localization module **534** can contain, or communicate with, at least one system, sensor, device, component, process, service, or other such mechanism to determine a location of the ego vehicle. This may include, for example, use of a GPS (or GNSS) system **536** that uses satellite-based radio signals to perform geolocation anywhere a sufficiently strong signal is able to be received from at least a minimum number of satellites, such as at least three or four satellites. A benefit of GPS is that it can be highly accurate, does not require an outgoing data transmission from the ego vehicle, and does not require an active network connection, such as an Internet or cellular connection. A GPS system must generally have an unobstructed transmission path from the minimum number of satellites, however, which may not be possible in certain locations, such as cities with tall buildings, tunnels, or mountainous regions. Other geolocation mechanisms can be used as well in other embodiments, such as those that make determinations based at least in part upon signals transmitted from earth or recognizable features in the nearby environment **508**, among other such options. A GPS receiver will typically be on the vehicle while other approaches might use components not on the vehicle, although latency and connectivity can then become problematic in certain situations. In at least one embodiment, the localization module **534** can attempt to improve or stabilize the location data from the GPS (or other such system) using other available information, such as the velocity and direction of travel of the vehicle, the locations of nearby objects, signal noise reduction, and so on. In this example, the mapping module **538** can receive geolocation data from the localization module **534**, and can determine the current location of the ego vehicle with respect to the stored map data. In at least one embodiment, this can be used to obtain and/or pre-fetch local map data for a current geolocation of the ego vehicle.

[0080] At least a selected portion of the perception data from the perception module **532**, and the geolocation and/or local map data from the mapping module **538**, can be provided as input to a map generator module **544**. The map generator can determine features in the perception module **532** and align those with features in prior (or other) tracks of data for the environment, in order to perform feature matching and alignment. The map generator module **544** can then use the aligned feature matches to generate updated map data where appropriate, which can be provided to a control module **546**. The control module **546** can then make operational decisions for a device or system, such as a vehicle or robotic assembly, indicating how to operate in the environment **508** given the current conditions. Such an approach may be useful in dynamic environments that are constantly changing, such as warehouses, where the map data can be updated dynamically using features in the sensor data (or perception data) that have been matched and aligned with previously-identified features in the environment **508**.

[0081] Aspects of various approaches presented herein can be lightweight enough to execute in various locations, such as on a device, such as a client device that includes a personal computer or gaming console, in real time. Such processing can be performed on, or for, content that is generated on, or received by, that client device or received from an external source, such as streaming data or other content received over at least one network from a cloud server or third-party service, among other such options. In some instances, at least a portion of the processing, generation, compositing, and/or determination of this content may be performed by one of these other devices, systems, or entities, then provided to the client device (or another such recipient) for presentation or another such use.

[0082] As an example, FIG. **6** illustrates an example network configuration **600** that can be used to provide, generate, modify, encode, process, fuse, and/or transmit generated data or other such content. In at least one embodiment, a client device **602** can generate or receive data for a session using components of a content application **604** on the client device **602** and data stored locally on that client device. In at least one embodiment, a content application **624** executing on a computer or processor **620** (e.g., a cloud server or control system) may initiate a session associated with at least one client device **602** (e.g., a vehicle or robot), as may use a session manager and user data stored in a user database **636**, and can cause content such as map data (e.g., implicit and/or explicit object representations or maps) from an asset repository **634** to be determined by a content manager **626**. A content manager **626** may work with at least one trained language module **628** perform tasks such as to extract features from sensor data, identify objects in the environment, or determine matching features in tracks of data, among other such options. The content manager **626** may also work with a perception module **632** to process the raw sensor data for use in feature matching, as well as a mapping module **630** for generating or updating map data based in part upon aligned feature matches. At least a portion of the generated map data (or aligned feature match data, etc.) can be transmitted to the client device **602** using an appropriate transmission manager **622** to send by download, streaming, or another such transmission channel. An encoder may be used to encode and/or compress at least some of this data before transmitting to the client device **602**. In at least one embodiment, the client device **602** receiving such content can provide this content to a corresponding content application **604**, which may also or alternatively include a graphical user interface **610** and content manager **612** for use in providing, synthesizing, rendering, compositing, modifying, or using content for presentation, navigation, control, (or other purposes) on or by the client device **602**. The content application **604** can also include a language module **614** that can perform various tasks, such as may relate to matching and aligning features and/or updating map data. In some embodiments, the computer/processor **620** and client device **602** may be able to communicate directly without needing to transmit data over a network **640**, in order to avoid issues with latency and availability, etc. A decoder may also be used to decode data received over the network **640** for presentation via client device **602**, such as map content through a display device **606** and audio, such as sounds and music, through at least one audio playback device **608**, such as speakers or headphones. In at least one embodiment, at least some of this content may already be stored on, rendered on, or accessible to client device **602** such that transmission over network **640** is not required for at least that portion of content, such as where that content (e.g., map data) may have been previously downloaded or stored locally on a hard drive or optical disk. In at least one embodiment, a transmission mechanism such as data streaming can be used to transfer this content from the computer/processor **620**, or user database **636**, to client device **602**. In at least one embodiment, at least a portion of this content can be obtained, enhanced, and/or streamed from another source, such as a third-party service **660** or other client device **650**, that may also include a content application for generating, updating, enhancing, or providing map content. In at least one embodiment, portions of this functionality can be performed using multiple computing devices, or multiple processors within one or more computing devices, such as may include a combination of CPUs and GPUs (Graphics Processing Unit).

[0083] FIG. **7** illustrates an example process **700** that can be performed to match and align landmarks (or other features) in captured sensor data, in accordance with at least one embodiment. It should be understood that for this and other processes presented herein that there may be additional, fewer, or alternative steps performed or similar or alternative orders, or at least partially in parallel, within the scope of the various embodiments unless otherwise specifically stated. Further, although this and other examples herein will be discussed with respect to driving or navigation domains and environments, there can be other types of domains, environments, and representations used and/or generated as well within the scope of various embodiments. In this example process, tracks (or other streams or sets) of data can be received **702** that each include a

set of observations corresponding to a region. These observations can include features, 3D points, or other aspects captured using one or more sensors, or generated using data from the one or more sensors. This may include point cloud data or other such observations. Features (or other data or aspects) extracted from these observations can be analyzed **704** to identify a set of landmarks in each track of data. This can include determining aspects such as size, shape, location, boundary, characteristics, labels, and the like.

[0084] There may be many tracks of data for a region that are at least partially overlapping, and pair-wise comparisons among the tracks can be performed. A pair of tracks can be selected **706** for each such comparison, and pairs of landmarks can be identified **708** within each of those tracks. For a given landmark, the landmark pairs can include up to a maximum number of other landmarks within a given distance range from the landmark, such as at least a minimum distance away but no more than a maximum distance away. Once these landmark pairs are identified, landmarks can be selected **710** that are associated with one or more (or multiple) identified landmark pairs in each of the pair of tracks. For each of these selected landmarks, the sets of landmark pairs from the pair of tracks that correspond to that selected landmark can be compared **712**, and it may be determined **714** that a number of corresponding landmark pairs for a given selected object meets or exceeds a specified correspondence threshold. The given landmark can then be identified **716** as a matching or corresponding landmark between the pair of tracks. The matching landmark can be aligned **718** to a common frame of reference for the region, and data for the aligned, matching landmark can be provided **720** for use in generating or updating a set of map data, auto-labelling data, or performing another such task or operation.

[0085] Taking such an approach, matches or edges are only considered that have an unambiguous correspondence. The matching between landmarks of a pair of tracks should match in both directions. As mentioned, in order to maintain high performance and reduce compute requirements, landmark pairs may only be considered that are within a given distance range, and only up to a maximum number of pairs per landmark. Such a process can attempt to identify implied matches, such as by analyzing adjacent edges, and not take matches directly out of a given landmark graph. Once landmarks are determined to correspond to each other, a line can be drawn to establish the transform needed to transform the tracks onto a same coordinate plane or other frame of reference. In at least one embodiment, a process can attempt to aggregate over all adjacent edges for a given landmark.

[0086] In this example, these client devices can include any appropriate computing devices, as may include a desktop computer, notebook computer, set-top box, streaming device, gaming console, smartphone, tablet computer, VR headset, AR goggles, wearable computer, or a smart television. Each client device can submit a request across at least one wired or wireless network, as may include the Internet, an Ethernet, a local area network (LAN), or a cellular network, among other such options. In this example, these requests can be submitted to an address associated with a cloud provider, who may operate or control one or more electronic resources in a cloud provider environment, such as may include a data center or server farm. In at least one embodiment, the request may be received or processed by at least one edge server, that sits on a network edge and is outside at least one security layer associated with the cloud provider environment. In this way, latency can be reduced by allowing the client devices to interact with servers that are in closer proximity, while also improving security of resources in the cloud provider environment.

[0087] In at least one embodiment, such a system can be used for performing graphical rendering operations. In other embodiments, such a system can be used for other purposes, such as for providing image or video content to test or validate autonomous machine applications, or for performing deep learning operations. In at least one embodiment, such a system can be implemented using an edge device or may incorporate one or more Virtual Machines (VMs). In at least one embodiment, such a system can be implemented at least partially in a data center or at least partially using cloud computing resources.

Data Center

[0088] FIG. **8** illustrates an example data center **800**, in which at least one embodiment may be used. In at least one embodiment, data center **800** includes a data center infrastructure layer **810**, a framework layer **820**, a software layer **830** and an application layer **840**.

[0089] In at least one embodiment, as shown in FIG. **8**, data center infrastructure layer **810** may include a resource orchestrator **812**, grouped computing resources **814**, and node computing resources ("node C.R.s") **816**(**1**)-**816**(N), where "N" represents a positive integer (which may be a different integer "N" than used in other figures). In at least one embodiment, node C.R.s **816**(**1**)-**816**(N) may include, but are not limited to, any number of central processing units ("CPUs") or other processors (including accelerators, field programmable gate arrays (FPGAs), graphics processors, etc.), memory storage devices **818**(**1**)-**818**(N) (e.g., dynamic read-only memory, solid state storage or disk drives), network input/output ("NW I/O") devices, network switches, virtual machines ("VMs"), power modules, and cooling modules, etc. In at least one embodiment, one or more node C.R.s from among node C.R.s **816**(**1**)-**816**(N) may be a server having one or more of above-mentioned computing resources.

[0090] In at least one embodiment, grouped computing resources **814** may include separate groupings of node C.R.s housed within one or more racks (not shown), or many racks housed in data centers at various geographical locations (also not shown). In at least one embodiment, separate groupings of node C.R.s within grouped computing resources **814** may include grouped compute, network, memory or storage resources that may be configured or allocated to support one or more workloads. In at least one embodiment, several node C.R.s including CPUs or processors may grouped within one or more racks to provide compute resources to support one or more workloads. In at least one embodiment, one or more racks may also include any number of power modules, cooling modules, and network switches, in any combination.

[0091] In at least one embodiment, resource orchestrator **812** may configure or otherwise control one or more node C.R.s **816**(**1**)-**816**(N) and/or grouped computing resources **814**. In at least one embodiment, resource orchestrator **812** may include a software design infrastructure ("SDI") management entity for data center **800**. In at least one embodiment, resource orchestrator **812** may include hardware, software or some combination thereof.

[0092] In at least one embodiment, as shown in FIG. **8**, framework layer **820** includes a job scheduler **822**, a configuration manager **824**, a resource manager **826** and a distributed file system **828**. In at least one embodiment, framework layer **820** may include a framework to support software **832** of software layer **830** and/or one or more application(s) **842** of application layer **840**. In at least one embodiment, software **832** or application(s) **842** may respectively include web-based service software or applications, such as those provided by Amazon Web Services, Google Cloud and Microsoft Azure. In at least one embodiment, framework layer **820** may be, but is not limited to, a type of free and open-source software web application framework such as Apache Spark™ (hereinafter "Spark") that may utilize distributed file system **828** for large-scale data processing (e.g., "big data"). In at least one embodiment, job scheduler **822** may include a Spark driver to facilitate scheduling of workloads supported by various layers of data center **800**. In at least one embodiment, configuration manager **824** may be capable of configuring different layers such as software layer **830** and framework layer **820** including Spark and distributed file system **828** for supporting large-scale data processing. In at least one embodiment, resource manager **826** may be capable of managing clustered or grouped computing resources mapped to or allocated for support of distributed file system **828** and job scheduler **822**. In at least one embodiment, clustered or grouped computing resources may include grouped computing resources **814** at data center infrastructure layer **810**. In at least one embodiment, resource manager **826** may coordinate with resource orchestrator **812** to manage these mapped or allocated computing resources.

[0093] In at least one embodiment, software **832** included in software layer **830** may include software used by at least portions of node C.R.s **816**(**1**)-**816**(N), grouped computing resources **814**,

and/or distributed file system **828** of framework layer **820**. In at least one embodiment, one or more types of software may include, but are not limited to, Internet web page search software, e-mail virus scan software, database software, and streaming video content software.

[0094] In at least one embodiment, application(s) **842** included in application layer **840** may include one or more types of applications used by at least portions of node C.R.s **816**(**1**)-**816**(N), grouped computing resources **814**, and/or distributed file system **828** of framework layer **820**. In at least one embodiment, one or more types of applications may include, but are not limited to, any number of a genomics application, a cognitive compute, application and a machine learning application, including training or inferencing software, machine learning framework software (e.g., PyTorch, TensorFlow, Caffe, etc.) or other machine learning applications used in conjunction with one or more embodiments.

[0095] In at least one embodiment, any of configuration manager **824**, resource manager **826**, and resource orchestrator **812** may implement any number and type of self-modifying actions based on any amount and type of data acquired in any technically feasible fashion. In at least one embodiment, self-modifying actions may relieve a data center operator of data center **800** from making possibly bad configuration decisions and possibly avoiding underutilized and/or poor performing portions of a data center.

[0096] In at least one embodiment, data center **800** may include tools, services, software or other resources to train one or more machine learning models or predict or infer information using one or more machine learning models according to one or more embodiments described herein. For example, in at least one embodiment, a machine learning model may be trained by calculating weight parameters according to a neural network architecture using software and computing resources described above with respect to data center **800**. In at least one embodiment, trained machine learning models corresponding to one or more neural networks may be used to infer or predict information using resources described above with respect to data center **800** by using weight parameters calculated through one or more training techniques described herein.

[0097] In at least one embodiment, data center may use CPUs, application-specific integrated circuits (ASICs), GPUs, FPGAs, or other hardware to perform training and/or inferencing using above-described resources. Moreover, one or more software and/or hardware resources described above may be configured as a service to allow users to train or performing inferencing of information, such as image recognition, speech recognition, or other artificial intelligence services.

[0098] Inference and/or training logic **815** are used to perform inferencing and/or training operations associated with one or more embodiments. In at least one embodiment, inference and/or training logic **815** may be used in system FIG. **8** for inferencing or predicting operations based, at least in part, on weight parameters calculated using neural network training operations, neural network functions and/or architectures, or neural network use cases described herein.

[0099] Embodiments presented herein can provide for the automated matching of landmarks in tracks of sensor data.

Computer Systems

[0100] FIG. **9** is a block diagram illustrating an exemplary computer system, which may be a system with interconnected devices and components, a system-on-a-chip (SOC) or some combination thereof formed with a processor that may include execution units to execute an instruction, according to at least one embodiment. In at least one embodiment, a computer system **900** may include, without limitation, a component, such as a processor **902** to employ execution units including logic to perform algorithms for process data, in accordance with present disclosure, such as in embodiment described herein. In at least one embodiment, computer system **900** may include processors, such as PENTIUM® Processor family, Xeon™, Itanium®, XScale™ and/or StrongARM™, Intel® Core™, or Intel® Nervana™ microprocessors available from Intel Corporation of Santa Clara, California, although other systems (including PCs having other microprocessors, engineering workstations, set-top boxes and like) may also be used. In at least one

embodiment, computer system **900** may execute a version of WINDOWS operating system available from Microsoft Corporation of Redmond, Wash., although other operating systems (UNIX and Linux, for example), embedded software, and/or graphical user interfaces, may also be used.

[0101] Embodiments may be used in other devices such as handheld devices and embedded applications. Some examples of handheld devices include cellular phones, Internet Protocol devices, digital cameras, personal digital assistants ("PDAs"), and handheld PCs. In at least one embodiment, embedded applications may include a microcontroller, a digital signal processor ("DSP"), system on a chip, network computers ("NetPCs"), set-top boxes, network hubs, wide area network ("WAN") switches, or any other system that may perform one or more instructions in accordance with at least one embodiment.

[0102] In at least one embodiment, computer system **900** may include, without limitation, processor **902** that may include, without limitation, one or more execution units **908** to perform machine learning model training and/or inferencing according to techniques described herein. In at least one embodiment, computer system **900** is a single processor desktop or server system, but in another embodiment, computer system **900** may be a multiprocessor system. In at least one embodiment, processor **902** may include, without limitation, a complex instruction set computer ("CISC") microprocessor, a reduced instruction set computing ("RISC") microprocessor, a very long instruction word ("VLIW") microprocessor, a processor implementing a combination of instruction sets, or any other processor device, such as a digital signal processor, for example. In at least one embodiment, processor **902** may be coupled to a processor bus **910** that may transmit data signals between processor **902** and other components in computer system **900**.

[0103] In at least one embodiment, processor **902** may include, without limitation, a Level 1 ("L1") internal cache memory ("cache") **904**. In at least one embodiment, processor **902** may have a single internal cache or multiple levels of internal cache. In at least one embodiment, cache memory may reside external to processor **902**. Other embodiments may also include a combination of both internal and external caches depending on particular implementation and needs. In at least one embodiment, a register file **906** may store different types of data in various registers including, without limitation, integer registers, floating point registers, status registers, and an instruction pointer register.

[0104] In at least one embodiment, execution unit **908**, including, without limitation, logic to perform integer and floating point operations, also resides in processor **902**. In at least one embodiment, processor **902** may also include a microcode ("ucode") read only memory ("ROM") that stores microcode for certain macro instructions. In at least one embodiment, execution unit **908** may include logic to handle a packed instruction set **909**. In at least one embodiment, by including packed instruction set **909** in an instruction set of a general-purpose processor, along with associated circuitry to execute instructions, operations used by many multimedia applications may be performed using packed data in processor **902**. In at least one embodiment, many multimedia applications may be accelerated and executed more efficiently by using a full width of a processor's data bus for performing operations on packed data, which may eliminate a need to transfer smaller units of data across that processor's data bus to perform one or more operations one data element at a time.

[0105] In at least one embodiment, execution unit **908** may also be used in microcontrollers, embedded processors, graphics devices, DSPs, and other types of logic circuits. In at least one embodiment, computer system **900** may include, without limitation, a memory **920**. In at least one embodiment, memory **920** may be a Dynamic Random Access Memory ("DRAM") device, a Static Random Access Memory ("SRAM") device, a flash memory device, or another memory device. In at least one embodiment, memory **920** may store instruction(s) **919** and/or data **921** represented by data signals that may be executed by processor **902**.

[0106] In at least one embodiment, a system logic chip may be coupled to processor bus **910** and

memory **920**. In at least one embodiment, a system logic chip may include, without limitation, a memory controller hub ("MCH") **916**, and processor **902** may communicate with MCH **916** via processor bus **910**. In at least one embodiment, MCH **916** may provide a high bandwidth memory path **918** to memory **920** for instruction and data storage and for storage of graphics commands, data, and textures. In at least one embodiment, MCH **916** may direct data signals between processor **902**, memory **920**, and other components in computer system **900** and to bridge data signals between processor bus **910**, memory **920**, and a system I/O interface **922**. In at least one embodiment, a system logic chip may provide a graphics port for coupling to a graphics controller. In at least one embodiment, MCH **916** may be coupled to memory **920** through high bandwidth memory path **918** and a graphics/video card **912** may be coupled to MCH **916** through an Accelerated Graphics Port ("AGP") interconnect **914**.

[0107] In at least one embodiment, computer system **900** may use system I/O interface **922** as a proprietary hub interface bus to couple MCH **916** to an I/O controller hub ("ICH") **930**. In at least one embodiment, ICH **930** may provide direct connections to some I/O devices via a local I/O bus. In at least one embodiment, a local I/O bus may include, without limitation, a high-speed I/O bus for connecting peripherals to memory **920**, a chipset, and processor **902**. Examples may include, without limitation, an audio controller **929**, a firmware hub ("flash BIOS") **928**, a wireless transceiver **926**, a data storage **924**, a legacy I/O controller **923** containing user input and keyboard interfaces **925**, a serial expansion port **927**, such as a Universal Serial Bus ("USB") port, and a network controller **934**. In at least one embodiment, data storage **924** may comprise a hard disk drive, a floppy disk drive, a CD-ROM device, a flash memory device, or other mass storage device.

[0108] In at least one embodiment, FIG. **9** illustrates a system, which includes interconnected hardware devices or "chips", whereas in other embodiments, FIG. **9** may illustrate an exemplary SoC. In at least one embodiment, devices illustrated in FIG. **9** may be interconnected with proprietary interconnects, standardized interconnects (e.g., PCIe) or some combination thereof. In at least one embodiment, one or more components of computer system **900** are interconnected using compute express link (CXL) interconnects.

[0109] Inference and/or training logic **815** are used to perform inferencing and/or training operations associated with one or more embodiments. In at least one embodiment, inference and/or training logic **815** may be used in system FIG. **9** for inferencing or predicting operations based, at least in part, on weight parameters calculated using neural network training operations, neural network functions and/or architectures, or neural network use cases described herein.

[0110] Embodiments presented herein can provide for the automated matching of landmarks in tracks of sensor data.

[0111] FIG. **10** is a block diagram illustrating an electronic device **1000** for utilizing a processor **1010**, according to at least one embodiment. In at least one embodiment, electronic device **1000** may be, for example and without limitation, a notebook, a tower server, a rack server, a blade server, a laptop, a desktop, a tablet, a mobile device, a phone, an embedded computer, or any other suitable electronic device.

[0112] In at least one embodiment, electronic device **1000** may include, without limitation, processor **1010** communicatively coupled to any suitable number or kind of components, peripherals, modules, or devices. In at least one embodiment, processor **1010** is coupled using a bus or interface, such as a I.sup.2C bus, a System Management Bus ("SMBus"), a Low Pin Count (LPC) bus, a Serial Peripheral Interface ("SPI"), a High Definition Audio ("HDA") bus, a Serial Advance Technology Attachment ("SATA") bus, a Universal Serial Bus ("USB") (versions **1**, **2**, **3**, etc.), or a Universal Asynchronous Receiver/Transmitter ("UART") bus. In at least one embodiment, FIG. **10** illustrates a system, which includes interconnected hardware devices or "chips", whereas in other embodiments, FIG. **10** may illustrate an exemplary SoC. In at least one embodiment, devices illustrated in FIG. **10** may be interconnected with proprietary interconnects, standardized interconnects (e.g., PCIe) or some combination thereof. In at least one embodiment,

one or more components of FIG. **10** are interconnected using compute express link (CXL) interconnects.

[0113] In at least one embodiment, FIG. **10** may include a display **1024**, a touch screen **1025**, a touch pad **1030**, a Near Field Communications unit ("NFC") **1045**, a sensor hub **1040**, a thermal sensor **1046**, an Express Chipset ("EC") **1035**, a Trusted Platform Module ("TPM") **1038**, BIOS/firmware/flash memory ("BIOS, FW Flash") **1022**, a DSP **1060**, a drive **1020** such as a Solid State Disk ("SSD") or a Hard Disk Drive ("HDD"), a wireless local area network unit ("WLAN") **1050**, a Bluetooth unit **1052**, a Wireless Wide Area Network unit ("WWAN") **1056**, a Global Positioning System (GPS) unit **1055**, a camera ("USB 3.0 camera") **1054** such as a USB 3.0 camera, and/or a Low Power Double Data Rate ("LPDDR") memory unit ("LPDDR3") **1015** implemented in, for example, an LPDDR3 standard. These components may each be implemented in any suitable manner.

[0114] In at least one embodiment, other components may be communicatively coupled to processor **1010** through components described herein. In at least one embodiment, an accelerometer **1041**, an ambient light sensor ("ALS") **1042**, a compass **1043**, and a gyroscope **1044** may be communicatively coupled to sensor hub **1040**. In at least one embodiment, a thermal sensor **1039**, a fan **1037**, a keyboard **1036**, and touch pad **1030** may be communicatively coupled to EC **1035**. In at least one embodiment, speakers **1063**, headphones **1064**, and a microphone ("mic") **1065** may be communicatively coupled to an audio unit ("audio codec and class D amp") **1062**, which may in turn be communicatively coupled to DSP **1060**. In at least one embodiment, audio unit **1062** may include, for example and without limitation, an audio coder/decoder ("codec") and a class D amplifier. In at least one embodiment, a SIM card ("SIM") **1057** may be communicatively coupled to WWAN unit **1056**. In at least one embodiment, components such as WLAN unit **1050** and Bluetooth unit **1052**, as well as WWAN unit **1056** may be implemented in a Next Generation Form Factor ("NGFF").

[0115] Inference and/or training logic **815** are used to perform inferencing and/or training operations associated with one or more embodiments. In at least one embodiment, inference and/or training logic **815** may be used in system FIG. **10** for inferencing or predicting operations based, at least in part, on weight parameters calculated using neural network training operations, neural network functions and/or architectures, or neural network use cases described herein.

[0116] Embodiments presented herein can provide for the automated matching of landmarks in tracks of sensor data.

[0117] FIG. **11** illustrates a computer system **1100**, according to at least one embodiment. In at least one embodiment, computer system **1100** is configured to implement various processes and methods described throughout this disclosure.

[0118] In at least one embodiment, computer system **1100** comprises, without limitation, at least one central processing unit ("CPU") **1102** that is connected to a communication bus **1110** implemented using any suitable protocol, such as PCI ("Peripheral Component Interconnect"), peripheral component interconnect express ("PCI-Express"), AGP ("Accelerated Graphics Port"), HyperTransport, or any other bus or point-to-point communication protocol(s). In at least one embodiment, computer system **1100** includes, without limitation, a main memory **1104** and control logic (e.g., implemented as hardware, software, or a combination thereof) and data are stored in main memory **1104**, which may take form of random access memory ("RAM"). In at least one embodiment, a network interface subsystem ("network interface") **1122** provides an interface to other computing devices and networks for receiving data from and transmitting data to other systems with computer system **1100**.

[0119] In at least one embodiment, computer system **1100**, in at least one embodiment, includes, without limitation, input devices **1108**, a parallel processing system **1112**, and display devices **1106** that can be implemented using a conventional cathode ray tube ("CRT"), a liquid crystal display ("LCD"), a light emitting diode ("LED") display, a plasma display, or other suitable display

technologies. In at least one embodiment, user input is received from input devices **1108** such as keyboard, mouse, touchpad, microphone, etc. In at least one embodiment, each module described herein can be situated on a single semiconductor platform to form a processing system.
[0120] Inference and/or training logic **815** are used to perform inferencing and/or training operations associated with one or more embodiments. In at least one embodiment, inference and/or training logic **815** may be used in system FIG. **11** for inferencing or predicting operations based, at least in part, on weight parameters calculated using neural network training operations, neural network functions and/or architectures, or neural network use cases described herein.
[0121] Embodiments presented herein can provide for the automated matching of landmarks in tracks of sensor data.
[0122] FIG. **12** illustrates a computer system **1200**, according to at least one embodiment. In at least one embodiment, computer system **1200** includes, without limitation, a computer **1210** and a USB stick **1220**. In at least one embodiment, computer **1210** may include, without limitation, any number and type of processor(s) (not shown) and a memory (not shown). In at least one embodiment, computer **1210** includes, without limitation, a server, a cloud instance, a laptop, and a desktop computer.
[0123] In at least one embodiment, USB stick **1220** includes, without limitation, a processing unit **1230**, a USB interface **1240**, and USB interface logic **1250**. In at least one embodiment, processing unit **1230** may be any instruction execution system, apparatus, or device capable of executing instructions. In at least one embodiment, processing unit **1230** may include, without limitation, any number and type of processing cores (not shown). In at least one embodiment, processing unit **1230** comprises an application specific integrated circuit ("ASIC") that is optimized to perform any amount and type of operations associated with machine learning. For instance, in at least one embodiment, processing unit **1230** is a tensor processing unit ("TPC") that is optimized to perform machine learning inference operations. In at least one embodiment, processing unit **1230** is a vision processing unit ("VPU") that is optimized to perform machine vision and machine learning inference operations.
[0124] In at least one embodiment, USB interface **1240** may be any type of USB connector or USB socket. For instance, in at least one embodiment, USB interface **1240** is a USB 3.0 Type-C socket for data and power. In at least one embodiment, USB interface **1240** is a USB 3.0 Type-A connector. In at least one embodiment, USB interface logic **1250** may include any amount and type of logic that enables processing unit **1230** to interface with devices (e.g., computer **1210**) via USB connector **1240**.
[0125] Inference and/or training logic **815** are used to perform inferencing and/or training operations associated with one or more embodiments. In at least one embodiment, inference and/or training logic **815** may be used in system FIG. **12** for inferencing or predicting operations based, at least in part, on weight parameters calculated using neural network training operations, neural network functions and/or architectures, or neural network use cases described herein.
[0126] Embodiments presented herein can provide for the automated matching of landmarks in tracks of sensor data.
[0127] FIG. **13** illustrates exemplary integrated circuits and associated graphics processors that may be fabricated using one or more IP cores, according to various embodiments described herein. In addition to what is illustrated, other logic and circuits may be included in at least one embodiment, including additional graphics processors/cores, peripheral interface controllers, or general-purpose processor cores.
[0128] FIG. **13** is a block diagram illustrating an exemplary system-on-a-chip (SOC) integrated circuit **1300** that may be fabricated using one or more IP cores, according to at least one embodiment. In at least one embodiment, SOC integrated circuit **1300** includes one or more application processor(s) **1305** (e.g., CPUs), at least one graphics processor **1310**, and may additionally include an image processor **1315** and/or a video processor **1320**, any of which may be

a modular IP core. In at least one embodiment, SOC integrated circuit **1300** includes peripheral or bus logic including a USB controller **1325**, a UART controller **1330**, an SPI/SDIO controller **1335**, and an I.sup.22S/I.sup.22C controller **1340**. In at least one embodiment, SOC integrated circuit **1300** can include a display device **1345** coupled to one or more of a high-definition multimedia interface (HDMI) controller **1350** and a mobile industry processor interface (MIPI) display interface **1355**. In at least one embodiment, storage may be provided by a flash memory subsystem **1360** including flash memory and a flash memory controller. In at least one embodiment, a memory interface may be provided via a memory controller **1365** for access to SDRAM or SRAM memory devices. In at least one embodiment, some integrated circuits additionally include an embedded security engine **1370**.

[0129] Inference and/or training logic **815** are used to perform inferencing and/or training operations associated with one or more embodiments. In at least one embodiment, inference and/or training logic **815** may be used in SOC integrated circuit **1300** for inferencing or predicting operations based, at least in part, on weight parameters calculated using neural network training operations, neural network functions and/or architectures, or neural network use cases described herein.

[0130] Embodiments presented herein can provide for the automated matching of landmarks in tracks of sensor data.

[0131] FIGS. **14**A-**14**B illustrate exemplary integrated circuits and associated graphics processors that may be fabricated using one or more IP cores, according to various embodiments described herein. In addition to what is illustrated, other logic and circuits may be included in at least one embodiment, including additional graphics processors/cores, peripheral interface controllers, or general-purpose processor cores.

[0132] FIGS. **14**A-**14**B are block diagrams illustrating exemplary graphics processors for use within an SoC, according to embodiments described herein. FIG. **14**A illustrates an exemplary graphics processor **1410** of a system on a chip integrated circuit that may be fabricated using one or more IP cores, according to at least one embodiment. FIG. **14**B illustrates an additional exemplary graphics processor **1440** of a system on a chip integrated circuit that may be fabricated using one or more IP cores, according to at least one embodiment. In at least one embodiment, graphics processor **1410** of FIG. **14**A is a low power graphics processor core. In at least one embodiment, graphics processor **1440** of FIG. **14**B is a higher performance graphics processor core. In at least one embodiment, each of graphics processors **1410**, **1440** can be variants of computer system **1200** of FIG. **12**.

[0133] In at least one embodiment, graphics processor **1410** includes a vertex processor **1405** and one or more fragment processor(s) **1415**A-**1415**N (e.g., **1415**A, **1415**B, **1415**C, **1415**D, through **1415**N-**1**, and **1415**N). In at least one embodiment, graphics processor **1410** can execute different shader programs via separate logic, such that vertex processor **1405** is optimized to execute operations for vertex shader programs, while one or more fragment processor(s) **1415**A-**1415**N execute fragment (e.g., pixel) shading operations for fragment or pixel shader programs. In at least one embodiment, vertex processor **1405** performs a vertex processing stage of a 3D graphics pipeline and generates primitives and vertex data. In at least one embodiment, fragment processor(s) **1415**A-**1415**N use primitive and vertex data generated by vertex processor **1405** to produce a framebuffer that is displayed on a display device. In at least one embodiment, fragment processor(s) **1415**A-**1415**N are optimized to execute fragment shader programs as provided for in an OpenGL API, which may be used to perform similar operations as a pixel shader program as provided for in a Direct 3D API.

[0134] In at least one embodiment, graphics processor **1410** additionally includes one or more memory management units (MMUs) **1420**A-**1420**B, cache(s) **1425**A-**1425**B, and circuit interconnect(s) **1430**A-**1430**B. In at least one embodiment, one or more MMU(s) **1420**A-**1420**B provide for virtual to physical address mapping for graphics processor **1410**, including for vertex

processor **1405** and/or fragment processor(s) **1415**A-**1415**N, which may reference vertex or image/texture data stored in memory, in addition to vertex or image/texture data stored in one or more cache(s) **1425**A-**1425**B. In at least one embodiment, one or more MMU(s) **1420**A-**1420**B may be synchronized with other MMUs within a system, including one or more MMUs associated with one or more application processor(s) **1405**, image processors **1415**, and/or video processors **1420** of FIG. **14**A, such that each processor **1405**-**1420** can participate in a shared or unified virtual memory system. In at least one embodiment, one or more circuit interconnect(s) **1430**A-**1430**B enable graphics processor **1410** to interface with other IP cores within SoC, either via an internal bus of SoC or via a direct connection.

[0135] In at least one embodiment, graphics processor **1440** includes one or more shader core(s) **1455**A-**1455**N (e.g., **1455**A, **1455**B, **1455**C, **1455**D, **1455**E, **1455**F, through **1455**N-**1**, and **1455**N) as shown in FIG. **14**B, which provides for a unified shader core architecture in which a single core or type or core can execute all types of programmable shader code, including shader program code to implement vertex shaders, fragment shaders, and/or compute shaders. In at least one embodiment, a number of shader cores can vary. In at least one embodiment, graphics processor **1440** includes an inter-core task manager **1445**, which acts as a thread dispatcher to dispatch execution threads to one or more shader cores **1455**A-**1455**N and a tiling unit **1458** to accelerate tiling operations for tile-based rendering, in which rendering operations for a scene are subdivided in image space, for example to exploit local spatial coherence within a scene or to optimize use of internal caches.

[0136] Embodiments presented herein can provide for the automated matching of landmarks in tracks of sensor data.

[0137] FIG. **15** is a block diagram illustrating a computing system **1500** according to at least one embodiment. In at least one embodiment, computing system **1500** includes a processing subsystem **1501** having one or more processor(s) **1502** and a system memory **1504** communicating via an interconnection path that may include a memory hub **1505**. In at least one embodiment, memory hub **1505** may be a separate component within a chipset component or may be integrated within one or more processor(s) **1502**. In at least one embodiment, memory hub **1505** couples with an I/O subsystem **1511** via a communication link **1506**. In at least one embodiment, I/O subsystem **1511** includes an I/O hub **1507** that can enable computing system **1500** to receive input from one or more input device(s) **1508**. In at least one embodiment, I/O hub **1507** can enable a display controller, which may be included in one or more processor(s) **1502**, to provide outputs to one or more display device(s) **1510**A. In at least one embodiment, one or more display device(s) **1510**A coupled with I/O hub **1507** can include a local, internal, or embedded display device.

[0138] In at least one embodiment, processing subsystem **1501** includes one or more parallel processor(s) **1512** coupled to memory hub **1505** via a bus or other communication link **1513**. In at least one embodiment, communication link **1513** may use one of any number of standards based communication link technologies or protocols, such as but not limited to PCI Express, or may be a vendor-specific communications interface or communications fabric. In at least one embodiment, one or more parallel processor(s) **1512** form a computationally focused parallel or vector processing system that can include a large number of processing cores and/or processing clusters, such as a many-integrated core (MIC) processor. In at least one embodiment, some or all of parallel processor(s) **1512** form a graphics processing subsystem that can output pixels to one of one or more display device(s) **1510**A coupled via I/O hub **1507**. In at least one embodiment, parallel processor(s) **1512** can also include a display controller and display interface (not shown) to enable a direct connection to one or more display device(s) **1510**B. In at least one embodiment, parallel processor(s) **1512** include one or more cores, such as graphics cores **1500** discussed herein.

[0139] In at least one embodiment, a system storage unit **1514** can connect to I/O hub **1507** to provide a storage mechanism for computing system **1500**. In at least one embodiment, an I/O switch **1516** can be used to provide an interface mechanism to enable connections between I/O hub

1507 and other components, such as a network adapter **1518** and/or a wireless network adapter **1519** that may be integrated into platform, and various other devices that can be added via one or more add-in device(s) **1520**. In at least one embodiment, network adapter **1518** can be an Ethernet adapter or another wired network adapter. In at least one embodiment, wireless network adapter **1519** can include one or more of a Wi-Fi, Bluetooth, near field communication (NFC), or other network device that includes one or more wireless radios.

[0140] In at least one embodiment, computing system **1500** can include other components not explicitly shown, including USB or other port connections, optical storage drives, video capture devices, and like, may also be connected to I/O hub **1507**. In at least one embodiment, communication paths interconnecting various components in FIG. **15** may be implemented using any suitable protocols, such as PCI (Peripheral Component Interconnect) based protocols (e.g., PCI-Express), or other bus or point-to-point communication interfaces and/or protocol(s), such as NV-Link high-speed interconnect, or interconnect protocols.

[0141] In at least one embodiment, parallel processor(s) **1512** incorporate circuitry optimized for graphics and video processing, including, for example, video output circuitry, and constitutes a graphics processing unit (GPU), e.g., parallel processor(s) **1512** includes graphics core **1500**. In at least one embodiment, parallel processor(s) **1512** incorporate circuitry optimized for general purpose processing. In at least embodiment, components of computing system **1500** may be integrated with one or more other system elements on a single integrated circuit. For example, in at least one embodiment, parallel processor(s) **1512**, memory hub **1505**, processor(s) **1502**, and I/O hub **1507** can be integrated into a system on chip (SoC) integrated circuit. In at least one embodiment, components of computing system **1500** can be integrated into a single package to form a system in package (SIP) configuration. In at least one embodiment, at least a portion of components of computing system **1500** can be integrated into a multi-chip module (MCM), which can be interconnected with other multi-chip modules into a modular computing system.

[0142] Inference and/or training logic **815** are used to perform inferencing and/or training operations associated with one or more embodiments. In at least one embodiment, inference and/or training logic **815** may be used in system FIG. **15** for inferencing or predicting operations based, at least in part, on weight parameters calculated using neural network training operations, neural network functions and/or architectures, or neural network use cases described herein.

[0143] Embodiments presented herein can provide for the automated matching of landmarks in tracks of sensor data.

Processors

[0144] FIG. **16**A illustrates a parallel processor **1600** according to at least one embodiment. In at least one embodiment, various components of parallel processor **1600** may be implemented using one or more integrated circuit devices, such as programmable processors, application specific integrated circuits (ASICs), or field programmable gate arrays (FPGA). In at least one embodiment, illustrated parallel processor **1600** is a variant of one or more parallel processor(s) **1512** shown in FIG. **15** according to an exemplary embodiment. In at least one embodiment, a parallel processor **1600** includes one or more graphics cores **1500**.

[0145] In at least one embodiment, parallel processor **1600** includes a parallel processing unit **1602**. In at least one embodiment, parallel processing unit **1602** includes an I/O unit **1604** that enables communication with other devices, including other instances of parallel processing unit **1602**. In at least one embodiment, I/O unit **1604** may be directly connected to other devices. In at least one embodiment, I/O unit **1604** connects with other devices via use of a hub or switch interface, such as a memory hub **1605**. In at least one embodiment, connections between memory hub **1605** and I/O unit **1604** form a communication link **1613**. In at least one embodiment, I/O unit **1604** connects with a host interface **1606** and a memory crossbar **1616**, where host interface **1606** receives commands directed to performing processing operations and memory crossbar **1616** receives commands directed to performing memory operations.

[0146] In at least one embodiment, when host interface **1606** receives a command buffer via I/O unit **1604**, host interface **1606** can direct work operations to perform those commands to a front end **1608**. In at least one embodiment, front end **1608** couples with a scheduler **1610** (which may be referred to as a sequencer), which is configured to distribute commands or other work items to a processing cluster array **1612**. In at least one embodiment, scheduler **1610** ensures that processing cluster array **1612** is properly configured and in a valid state before tasks are distributed to a cluster of processing cluster array **1612**. In at least one embodiment, scheduler **1610** is implemented via firmware logic executing on a microcontroller. In at least one embodiment, microcontroller implemented scheduler **1610** is configurable to perform complex scheduling and work distribution operations at coarse and fine granularity, enabling rapid preemption and context switching of threads executing on processing array **1612**. In at least one embodiment, host software can prove workloads for scheduling on processing cluster array **1612** via one of multiple graphics processing paths. In at least one embodiment, workloads can then be automatically distributed across processing array cluster **1612** by scheduler **1610** logic within a microcontroller including scheduler **1610**.
[0147] In at least one embodiment, processing cluster array **1612** can include up to "N" processing clusters (e.g., cluster **1614**A, cluster **1614**B, through cluster **1614**N), where "N" represents a positive integer (which may be a different integer "N" than used in other figures). In at least one embodiment, each cluster **1614**A-**1614**N of processing cluster array **1612** can execute a large number of concurrent threads. In at least one embodiment, scheduler **1610** can allocate work to clusters **1614**A-**1614**N of processing cluster array **1612** using various scheduling and/or work distribution algorithms, which may vary depending on workload arising for each type of program or computation. In at least one embodiment, scheduling can be handled dynamically by scheduler **1610**, or can be assisted in part by compiler logic during compilation of program logic configured for execution by processing cluster array **1612**. In at least one embodiment, different clusters **1614**A-**1614**N of processing cluster array **1612** can be allocated for processing different types of programs or for performing different types of computations.
[0148] In at least one embodiment, processing cluster array **1612** can be configured to perform various types of parallel processing operations. In at least one embodiment, processing cluster array **1612** is configured to perform general-purpose parallel compute operations. For example, in at least one embodiment, processing cluster array **1612** can include logic to execute processing tasks including filtering of video and/or audio data, performing modeling operations, including physics operations, and performing data transformations.
[0149] In at least one embodiment, processing cluster array **1612** is configured to perform parallel graphics processing operations. In at least one embodiment, processing cluster array **1612** can include additional logic to support execution of such graphics processing operations, including but not limited to, texture sampling logic to perform texture operations, as well as tessellation logic and other vertex processing logic. In at least one embodiment, processing cluster array **1612** can be configured to execute graphics processing related shader programs such as but not limited to, vertex shaders, tessellation shaders, geometry shaders, and pixel shaders. In at least one embodiment, parallel processing unit **1602** can transfer data from system memory via I/O unit **1604** for processing. In at least one embodiment, during processing, transferred data can be stored to on-chip memory (e.g., parallel processor memory **1622**) during processing, then written back to system memory.
[0150] In at least one embodiment, when parallel processing unit **1602** is used to perform graphics processing, scheduler **1610** can be configured to divide a processing workload into approximately equal sized tasks, to better enable distribution of graphics processing operations to multiple clusters **1614**A-**1614**N of processing cluster array **1612**. In at least one embodiment, portions of processing cluster array **1612** can be configured to perform different types of processing. For example, in at least one embodiment, a first portion may be configured to perform vertex shading and topology

generation, a second portion may be configured to perform tessellation and geometry shading, and a third portion may be configured to perform pixel shading or other screen space operations, to produce a rendered image for display. In at least one embodiment, intermediate data produced by one or more of clusters **1614**A-**1614**N may be stored in buffers to allow intermediate data to be transmitted between clusters **1614**A-**1614**N for further processing.

[0151] In at least one embodiment, processing cluster array **1612** can receive processing tasks to be executed via scheduler **1610**, which receives commands defining processing tasks from front end **1608**. In at least one embodiment, processing tasks can include indices of data to be processed, e.g., surface (patch) data, primitive data, vertex data, and/or pixel data, as well as state parameters and commands defining how data is to be processed (e.g., what program is to be executed). In at least one embodiment, scheduler **1610** may be configured to fetch indices corresponding to tasks or may receive indices from front end **1608**. In at least one embodiment, front end **1608** can be configured to ensure processing cluster array **1612** is configured to a valid state before a workload specified by incoming command buffers (e.g., batch-buffers, push buffers, etc.) is initiated.

[0152] In at least one embodiment, each of one or more instances of parallel processing unit **1602** can couple with a parallel processor memory **1622**. In at least one embodiment, parallel processor memory **1622** can be accessed via memory crossbar **1616**, which can receive memory requests from processing cluster array **1612** as well as I/O unit **1604**. In at least one embodiment, memory crossbar **1616** can access parallel processor memory **1622** via a memory interface **1618**. In at least one embodiment, memory interface **1618** can include multiple partition units (e.g., partition unit **1620**A, partition unit **1620**B, through partition unit **1620**N) that can each couple to a portion (e.g., memory unit) of parallel processor memory **1622**. In at least one embodiment, a number of partition units **1620**A-**1620**N is configured to be equal to a number of memory units, such that a first partition unit **1620**A has a corresponding first memory unit **1624**A, a second partition unit **1620**B has a corresponding memory unit **1624**B, and an N-th partition unit **1620**N has a corresponding N-th memory unit **1624**N. In at least one embodiment, a number of partition units **1620**A-**1620**N may not be equal to a number of memory units.

[0153] In at least one embodiment, memory units **1624**A-**1624**N can include various types of memory devices, including dynamic random access memory (DRAM) or graphics random access memory, such as synchronous graphics random access memory (SGRAM), including graphics double data rate (GDDR) memory. In at least one embodiment, memory units **1624**A-**1624**N may also include 3D stacked memory, including but not limited to high bandwidth memory (HBM), HBM2e, or HDM3. In at least one embodiment, render targets, such as frame buffers or texture maps may be stored across memory units **1624**A-**1624**N, allowing partition units **1620**A-**1620**N to write portions of each render target in parallel to efficiently use available bandwidth of parallel processor memory **1622**. In at least one embodiment, a local instance of parallel processor memory **1622** may be excluded in favor of a unified memory design that utilizes system memory in conjunction with local cache memory.

[0154] In at least one embodiment, any one of clusters **1614**A-**1614**N of processing cluster array **1612** can process data that will be written to any of memory units **1624**A-**1624**N within parallel processor memory **1622**. In at least one embodiment, memory crossbar **1616** can be configured to transfer an output of each cluster **1614**A-**1614**N to any partition unit **1620**A-**1620**N or to another cluster **1614**A-**1614**N, which can perform additional processing operations on an output. In at least one embodiment, each cluster **1614**A-**1614**N can communicate with memory interface **1618** through memory crossbar **1616** to read from or write to various external memory devices. In at least one embodiment, memory crossbar **1616** has a connection to memory interface **1618** to communicate with I/O unit **1604**, as well as a connection to a local instance of parallel processor memory **1622**, enabling processing units within different processing clusters **1614**A-**1614**N to communicate with system memory or other memory that is not local to parallel processing unit **1602**. In at least one embodiment, memory crossbar **1616** can use virtual channels to separate

traffic streams between clusters **1614**A-**1614**N and partition units **1620**A-**1620**N.
[0155] In at least one embodiment, multiple instances of parallel processing unit **1602** can be provided on a single add-in card, or multiple add-in cards can be interconnected. In at least one embodiment, different instances of parallel processing unit **1602** can be configured to interoperate even if different instances have different numbers of processing cores, different amounts of local parallel processor memory, and/or other configuration differences. For example, in at least one embodiment, some instances of parallel processing unit **1602** can include higher precision floating point units relative to other instances. In at least one embodiment, systems incorporating one or more instances of parallel processing unit **1602** or parallel processor **1600** can be implemented in a variety of configurations and form factors, including but not limited to desktop, laptop, or handheld personal computers, servers, workstations, game consoles, and/or embedded systems.
[0156] FIG. **16**B is a block diagram of a partition unit **1620** according to at least one embodiment. In at least one embodiment, partition unit **1620** is an instance of one of partition units **1620**A-**1620**N of FIG. **16**A. In at least one embodiment, partition unit **1620** includes an L2 cache **1621**, a frame buffer interface **1625**, and a ROP **1626** (raster operations unit). In at least one embodiment, L2 cache **1621** is a read/write cache that is configured to perform load and store operations received from memory crossbar **1616** and ROP **1626**. In at least one embodiment, read misses and urgent write-back requests are output by L2 cache **1621** to frame buffer interface **1625** for processing. In at least one embodiment, updates can also be sent to a frame buffer via frame buffer interface **1625** for processing. In at least one embodiment, frame buffer interface **1625** interfaces with one of memory units in parallel processor memory, such as memory units **1624**A-**1624**N of FIG. **16**A (e.g., within parallel processor memory **1622**).
[0157] In at least one embodiment, ROP **1626** is a processing unit that performs raster operations such as stencil, z test, blending, etc. In at least one embodiment, ROP **1626** then outputs processed graphics data that is stored in graphics memory. In at least one embodiment, ROP **1626** includes compression logic to compress depth or color data that is written to memory and decompress depth or color data that is read from memory. In at least one embodiment, compression logic can be lossless compression logic that makes use of one or more of multiple compression algorithms. In at least one embodiment, a type of compression that is performed by ROP **1626** can vary based on statistical characteristics of data to be compressed. For example, in at least one embodiment, delta color compression is performed on depth and color data on a per-tile basis.
[0158] In at least one embodiment, ROP **1626** is included within each processing cluster (e.g., cluster **1614**A-**1614**N of FIG. **16**A) instead of within partition unit **1620**. In at least one embodiment, read and write requests for pixel data are transmitted over memory crossbar **1616** instead of pixel fragment data. In at least one embodiment, processed graphics data may be displayed on a display device, such as one of one or more display device(s) **1510** of FIG. **15**, routed for further processing by processor(s) **1602**, or routed for further processing by one of processing entities within parallel processor **1600** of FIG. **16**A.
[0159] FIG. **17** is a block diagram of a processing system, according to at least one embodiment. In at least one embodiment, system **1700** includes one or more processor(s) **1702** and one or more graphics processor(s) **1708**, and may be a single processor desktop system, a multiprocessor workstation system, or a server system having a large number of processor(s) **1702** or processor core(s) **1707**. In at least one embodiment, system **1700** is a processing platform incorporated within a system-on-a-chip (SoC) integrated circuit for use in mobile, handheld, or embedded devices. In at least one embodiment, one or more graphics processor(s) **1708** include one or more graphics cores **1500**.
[0160] In at least one embodiment, system **1700** can include, or be incorporated within a server-based gaming platform, a game console, including a game and media console, a mobile gaming console, a handheld game console, or an online game console. In at least one embodiment, system **1700** is a mobile phone, a smart phone, a tablet computing device or a mobile Internet device. In at

least one embodiment, processing system **1700** can also include, couple with, or be integrated within a wearable device, such as a smart watch wearable device, a smart eyewear device, an augmented reality device, or a virtual reality device. In at least one embodiment, processing system **1700** is a television or set top box device having one or more processor(s) **1702** and a graphical interface generated by one or more graphics processor(s) **1708**.

[0161] In at least one embodiment, one or more processor(s) **1702** each include one or more processor core(s) **1707** to process instructions which, when executed, perform operations for system and user software. In at least one embodiment, each of one or more processor core(s) **1707** is configured to process a specific instruction sequence **1709**. In at least one embodiment, instruction sequence **1709** may facilitate Complex Instruction Set Computing (CISC), Reduced Instruction Set Computing (RISC), or computing via a Very Long Instruction Word (VLIW). In at least one embodiment, processor core(s) **1707** may each process a different instruction sequence **1709**, which may include instructions to facilitate emulation of other instruction sequences. In at least one embodiment, processor core(s) **1707** may also include other processing devices, such a Digital Signal Processor (DSP).

[0162] In at least one embodiment, processor(s) **1702** includes a cache memory **1704**. In at least one embodiment, processor(s) **1702** can have a single internal cache or multiple levels of internal cache. In at least one embodiment, cache memory is shared among various components of processor(s) **1702**. In at least one embodiment, processor(s) **1702** also uses an external cache (e.g., a Level-3 (L3) cache or Last Level Cache (LLC)) (not shown), which may be shared among processor core(s) **1707** using known cache coherency techniques. In at least one embodiment, a register file **1706** is additionally included in processor(s) **1702**, which may include different types of registers for storing different types of data (e.g., integer registers, floating point registers, status registers, and an instruction pointer register). In at least one embodiment, register file **1706** may include general-purpose registers or other registers.

[0163] In at least one embodiment, one or more processor(s) **1702** are coupled with one or more interface bus(es) **1710** to transmit communication signals such as address, data, or control signals between processor(s) **1702** and other components in system **1700**. In at least one embodiment, interface bus(es) **1710** can be a processor bus, such as a version of a Direct Media Interface (DMI) bus. In at least one embodiment, interface bus(es) **1710** is not limited to a DMI bus, and may include one or more Peripheral Component Interconnect buses (e.g., PCI, PCI Express), memory busses, or other types of interface busses. In at least one embodiment processor(s) **1702** include an integrated memory controller **1716** and a platform controller hub **1730**. In at least one embodiment, memory controller **1716** facilitates communication between a memory device and other components of system **1700**, while platform controller hub (PCH) **1730** provides connections to I/O devices via a local I/O bus.

[0164] In at least one embodiment, a memory device **1720** can be a dynamic random access memory (DRAM) device, a static random access memory (SRAM) device, flash memory device, phase-change memory device, or some other memory device having suitable performance to serve as process memory. In at least one embodiment, memory device **1720** can operate as system memory for system **1700**, to store data **1722** and instructions **1721** for use when one or more processor(s) **1702** executes an application or process. In at least one embodiment, memory controller **1716** also couples with an optional external graphics processor **1712**, which may communicate with one or more graphics processor(s) **1708** in processor(s) **1702** to perform graphics and media operations. In at least one embodiment, a display device **1711** can connect to processor(s) **1702**. In at least one embodiment, display device **1711** can include one or more of an internal display device, as in a mobile electronic device or a laptop device, or an external display device attached via a display interface (e.g., DisplayPort, etc.). In at least one embodiment, display device **1711** can include a head mounted display (HMD) such as a stereoscopic display device for use in virtual reality (VR) applications or augmented reality (AR) applications.

[0165] In at least one embodiment, platform controller hub **1730** enables peripherals to connect to memory device **1720** and processor(s) **1702** via a high-speed I/O bus. In at least one embodiment, I/O peripherals include, but are not limited to, an audio controller **1746**, a network controller **1734**, a firmware interface **1728**, a wireless transceiver **1726**, touch sensors **1725**, a data storage device **1724** (e.g., hard disk drive, flash memory, etc.). In at least one embodiment, data storage device **1724** can connect via a storage interface (e.g., SATA) or via a peripheral bus, such as a Peripheral Component Interconnect bus (e.g., PCI, PCI Express). In at least one embodiment, touch sensors **1725** can include touch screen sensors, pressure sensors, or fingerprint sensors. In at least one embodiment, wireless transceiver **1726** can be a Wi-Fi transceiver, a Bluetooth transceiver, or a mobile network transceiver such as a 3G, 4G, or Long Term Evolution (LTE) transceiver. In at least one embodiment, firmware interface **1728** enables communication with system firmware, and can be, for example, a unified extensible firmware interface (UEFI). In at least one embodiment, network controller **1734** can enable a network connection to a wired network. In at least one embodiment, a high-performance network controller (not shown) couples with interface bus(es) **1710**. In at least one embodiment, audio controller **1746** is a multi-channel high definition audio controller. In at least one embodiment, system **1700** includes an optional legacy I/O controller **1740** for coupling legacy (e.g., Personal System 2 (PS/2)) devices to system **1700**. In at least one embodiment, platform controller hub **1730** can also connect to one or more Universal Serial Bus (USB) controller(s) **1742** connect input devices, such as keyboard and mouse **1743** combinations, a camera **1744**, or other USB input devices.

[0166] In at least one embodiment, an instance of memory controller **1716** and platform controller hub **1730** may be integrated into a discreet external graphics processor, such as external graphics processor **1712**. In at least one embodiment, platform controller hub **1730** and/or memory controller **1716** may be external to one or more processor(s) **1702**. For example, in at least one embodiment, system **1700** can include an external memory controller **1716** and platform controller hub **1730**, which may be configured as a memory controller hub and peripheral controller hub within a system chipset that is in communication with processor(s) **1702**.

[0167] Embodiments presented herein can provide for the automated matching of landmarks in tracks of sensor data.

Autonomous Vehicle

[0168] FIG. **18**A illustrates an example of an autonomous vehicle **1800**, according to at least one embodiment. In at least one embodiment, autonomous vehicle **1800** (alternatively referred to herein as "vehicle **1800**") may be, without limitation, a passenger vehicle, such as a car, a truck, a bus, and/or another type of vehicle that accommodates one or more passengers. In at least one embodiment, vehicle **1800** may be a semi-tractor-trailer truck used for hauling cargo. In at least one embodiment, vehicle **1800** may be an airplane, robotic vehicle, or other kind of vehicle.

[0169] Autonomous vehicles may be described in terms of automation levels, defined by National Highway Traffic Safety Administration ("NHTSA"), a division of US Department of Transportation, and Society of Automotive Engineers ("SAE") "Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles" (e.g., Standard No. J3016-201806, published on Jun. 15, 2018, Standard No. J3016-201609, published on Sep. 30, 2016, and previous and future versions of this standard). In at least one embodiment, vehicle **1800** may be capable of functionality in accordance with one or more of Level 1 through Level 5 of autonomous driving levels. For example, in at least one embodiment, vehicle **1800** may be capable of conditional automation (Level 3), high automation (Level 4), and/or full automation (Level 5), depending on embodiment.

[0170] In at least one embodiment, vehicle **1800** may include, without limitation, components such as a chassis, a vehicle body, wheels (e.g., 2, 4, 6, 8, 18, etc.), tires, axles, and other components of a vehicle. In at least one embodiment, vehicle **1800** may include, without limitation, a propulsion system **1850**, such as an internal combustion engine, hybrid electric power plant, an all-electric

engine, and/or another propulsion system type. In at least one embodiment, propulsion system **1850** may be connected to a drive train of vehicle **1800**, which may include, without limitation, a transmission, to enable propulsion of vehicle **1800**. In at least one embodiment, propulsion system **1850** may be controlled in response to receiving signals from a throttle/accelerator(s) **1852**.
[0171] In at least one embodiment, a steering system **1854**, which may include, without limitation, a steering wheel, is used to steer vehicle **1800** (e.g., along a desired path or route) when propulsion system **1850** is operating (e.g., when vehicle **1800** is in motion). In at least one embodiment, steering system **1854** may receive signals from steering actuator(s) **1856**. In at least one embodiment, a steering wheel may be optional for full automation (Level 5) functionality. In at least one embodiment, a brake sensor system **1846** may be used to operate vehicle brakes in response to receiving signals from brake actuator(s) **1848** and/or brake sensors.
[0172] In at least one embodiment, controller(s) **1836**, which may include, without limitation, one or more system on chips ("SoCs") (not shown in FIG. **18**A) and/or graphics processing unit(s) ("GPU(s)"), provide signals (e.g., representative of commands) to one or more components and/or systems of vehicle **1800**. For instance, in at least one embodiment, controller(s) **1836** may send signals to operate vehicle brakes via brake actuator(s) **1848**, to operate steering system **1854** via steering actuator(s) **1856**, to operate propulsion system **1850** via throttle/accelerator(s) **1852**. In at least one embodiment, controller(s) **1836** may include one or more onboard (e.g., integrated) computing devices that process sensor signals, and output operation commands (e.g., signals representing commands) to enable autonomous driving and/or to assist a human driver in driving vehicle **1800**. In at least one embodiment, controller(s) **1836** may include a first controller for autonomous driving functions, a second controller for functional safety functions, a third controller for artificial intelligence functionality (e.g., computer vision), a fourth controller for infotainment functionality, a fifth controller for redundancy in emergency conditions, and/or other controllers. In at least one embodiment, a single controller may handle two or more of above functionalities, two or more controllers may handle a single functionality, and/or any combination thereof.
[0173] In at least one embodiment, controller(s) **1836** provide signals for controlling one or more components and/or systems of vehicle **1800** in response to sensor data received from one or more sensors (e.g., sensor inputs). In at least one embodiment, sensor data may be received from, for example and without limitation, global navigation satellite systems ("GNSS") sensor(s) **1858** (e.g., Global Positioning System sensor(s)), RADAR sensor(s) **1860**, ultrasonic sensor(s) **1862**, LiDAR sensor(s) **1864**, inertial measurement unit ("IMU") sensor(s) **1866** (e.g., accelerometer(s), gyroscope(s), a magnetic compass or magnetic compasses, magnetometer(s), etc.), microphone(s) **1896**, stereo camera(s) **1868**, wide-view camera(s) **1870** (e.g., fisheye cameras), infrared camera(s) **1872**, surround camera(s) **1874** (e.g., 360 degree cameras), long-range cameras (not shown in FIG. **18**A), mid-range camera(s) (not shown in FIG. **18**A), speed sensor(s) **1844** (e.g., for measuring speed of vehicle **1800**), vibration sensor(s) **1842**, steering sensor(s) **1840**, brake sensor(s) (e.g., as part of brake sensor system **1846**), and/or other sensor types.
[0174] In at least one embodiment, one or more of controller(s) **1836** may receive inputs (e.g., represented by input data) from an instrument cluster **1832** of vehicle **1800** and provide outputs (e.g., represented by output data, display data, etc.) via a human-machine interface ("HMI") display **1834**, an audible annunciator, a loudspeaker, and/or via other components of vehicle **1800**. In at least one embodiment, outputs may include information such as vehicle velocity, speed, time, map data (e.g., a High Definition map (not shown in FIG. **18**A)), location data (e.g., vehicle's **1800** location, such as on a map), direction, location of other vehicles (e.g., an occupancy grid), information about objects and status of objects as perceived by controller(s) **1836**, etc. For example, in at least one embodiment, HMI display **1834** may display information about presence of one or more objects (e.g., a street sign, caution sign, traffic light changing, etc.), and/or information about driving maneuvers vehicle has made, is making, or will make (e.g., changing lanes now, taking exit **34**B in two miles, etc.).

[0175] In at least one embodiment, vehicle **1800** further includes a network interface **1824** which may use wireless antenna(s) **1826** and/or modem(s) to communicate over one or more networks. For example, in at least one embodiment, network interface **1824** may be capable of communication over Long-Term Evolution ("LTE"), Wideband Code Division Multiple Access ("WCDMA"), Universal Mobile Telecommunications System ("UMTS"), Global System for Mobile communication ("GSM"), IMT-CDMA Multi-Carrier ("CDMA2000") networks, etc. In at least one embodiment, wireless antenna(s) **1826** may also enable communication between objects in environment (e.g., vehicles, mobile devices, etc.), using local area network(s), such as Bluetooth, Bluetooth Low Energy ("LE"), Z-Wave, ZigBee, etc., and/or low power wide-area network(s) ("LPWANs"), such as LoRaWAN, SigFox, etc. protocols.
[0176] Inference and/or training logic **815** are used to perform inferencing and/or training operations associated with one or more embodiments. In at least one embodiment, inference and/or training logic **815** may be used in system FIG. **18**A for inferencing or predicting operations based, at least in part, on weight parameters calculated using neural network training operations, neural network functions and/or architectures, or neural network use cases described herein.
[0177] Such components can be used to generate a single, consistent tokenized description of at least a portion of a physical environment based in part on a set of observations and aligned map data.
[0178] FIG. **18**B illustrates an example of camera locations and fields of view for autonomous vehicle **1800** of FIG. **18**A, according to at least one embodiment. In at least one embodiment, cameras and respective fields of view are one example embodiment and are not intended to be limiting. For instance, in at least one embodiment, additional and/or alternative cameras may be included and/or cameras may be located at different locations on vehicle **1800**.
[0179] In at least one embodiment, camera types for cameras may include, but are not limited to, digital cameras that may be adapted for use with components and/or systems of vehicle **1800**. In at least one embodiment, camera(s) may operate at automotive safety integrity level ("ASIL") B and/or at another ASIL. In at least one embodiment, camera types may be capable of any image capture rate, such as 60 frames per second (fps), 1220 fps, 240 fps, etc., depending on embodiment. In at least one embodiment, cameras may be capable of using rolling shutters, global shutters, another type of shutter, or a combination thereof. In at least one embodiment, color filter array may include a red clear clear clear ("RCCC") color filter array, a red clear clear blue ("RCCB") color filter array, a red blue green clear ("RBGC") color filter array, a Foveon X3 color filter array, a Bayer sensors ("RGGB") color filter array, a monochrome sensor color filter array, and/or another type of color filter array. In at least one embodiment, clear pixel cameras, such as cameras with an RCCC, an RCCB, and/or an RBGC color filter array, may be used in an effort to increase light sensitivity.
[0180] In at least one embodiment, one or more of camera(s) may be used to perform advanced driver assistance systems ("ADAS") functions (e.g., as part of a redundant or fail-safe design). For example, in at least one embodiment, a Multi-Function Mono Camera may be installed to provide functions including lane departure warning, traffic sign assist and intelligent headlamp control. In at least one embodiment, one or more of camera(s) (e.g., all cameras) may record and provide image data (e.g., video) simultaneously.
[0181] In at least one embodiment, one or more camera may be mounted in a mounting assembly, such as a custom designed (three-dimensional ("3D") printed) assembly, in order to cut out stray light and reflections from within vehicle **1800** (e.g., reflections from dashboard reflected in windshield mirrors) which may interfere with camera image data capture abilities. With reference to wing-mirror mounting assemblies, in at least one embodiment, wing-mirror assemblies may be custom 3D printed so that a camera mounting plate matches a shape of a wing-mirror. In at least one embodiment, camera(s) may be integrated into wing-mirrors. In at least one embodiment, for side-view cameras, camera(s) may also be integrated within four pillars at each corner of a cabin.

[0182] In at least one embodiment, cameras with a field of view that include portions of an environment in front of vehicle **1800** (e.g., front-facing cameras) may be used for surround view, to help identify forward facing paths and obstacles, as well as aid in, with help of one or more of controller(s) **1836** and/or control SoCs, providing information critical to generating an occupancy grid and/or determining preferred vehicle paths. In at least one embodiment, front-facing cameras may be used to perform many similar ADAS functions as LiDAR, including, without limitation, emergency braking, pedestrian detection, and collision avoidance. In at least one embodiment, front-facing cameras may also be used for ADAS functions and systems including, without limitation, Lane Departure Warnings ("LDW"), Autonomous Cruise Control ("ACC"), and/or other functions such as traffic sign recognition.
[0183] In at least one embodiment, a variety of cameras may be used in a front-facing configuration, including, for example, a monocular camera platform that includes a CMOS ("complementary metal oxide semiconductor") color imager. In at least one embodiment, a wide-view camera **1870** may be used to perceive objects coming into view from a periphery (e.g., pedestrians, crossing traffic or bicycles). Although only one wide-view camera **1870** is illustrated in FIG. **18**B, in other embodiments, there may be any number (including zero) wide-view cameras on vehicle **1800**. In at least one embodiment, any number of long-range camera(s) **1898** (e.g., a long-view stereo camera pair) may be used for depth-based object detection, especially for objects for which a neural network has not yet been trained. In at least one embodiment, long-range camera(s) **1898** may also be used for object detection and classification, as well as basic object tracking.
[0184] In at least one embodiment, any number of stereo camera(s) **1868** may also be included in a front-facing configuration. In at least one embodiment, one or more of stereo camera(s) **1868** may include an integrated control unit comprising a scalable processing unit, which may provide a programmable logic ("FPGA") and a multi-core micro-processor with an integrated Controller Area Network ("CAN") or Ethernet interface on a single chip. In at least one embodiment, such a unit may be used to generate a 3D map of an environment of vehicle **1800**, including a distance estimate for all points in an image. In at least one embodiment, one or more of stereo camera(s) **1868** may include, without limitation, compact stereo vision sensor(s) that may include, without limitation, two camera lenses (one each on left and right) and an image processing chip that may measure distance from vehicle **1800** to target object and use generated information (e.g., metadata) to activate autonomous emergency braking and lane departure warning functions. In at least one embodiment, other types of stereo camera(s) **1868** may be used in addition to, or alternatively from, those described herein.
[0185] In at least one embodiment, cameras with a field of view that include portions of environment to sides of vehicle **1800** (e.g., side-view cameras) may be used for surround view, providing information used to create and update an occupancy grid, as well as to generate side impact collision warnings. For example, in at least one embodiment, surround camera(s) **1874** (e.g., four surround cameras as illustrated in FIG. **18**B) could be positioned on vehicle **1800**. In at least one embodiment, surround camera(s) **1874** may include, without limitation, any number and combination of wide-view cameras, fisheye camera(s), 360 degree camera(s), and/or similar cameras. For instance, in at least one embodiment, four fisheye cameras may be positioned on a front, a rear, and sides of vehicle **1800**. In at least one embodiment, vehicle **1800** may use three surround camera(s) **1874** (e.g., left, right, and rear), and may leverage one or more other camera(s) (e.g., a forward-facing camera) as a fourth surround-view camera.
[0186] In at least one embodiment, cameras with a field of view that include portions of an environment behind vehicle **1800** (e.g., rear-view cameras) may be used for parking assistance, surround view, rear collision warnings, and creating and updating an occupancy grid. In at least one embodiment, a wide variety of cameras may be used including, but not limited to, cameras that are also suitable as a front-facing camera(s) (e.g., long-range camera(s) **1898** and/or mid-range

camera(s) **1876**, stereo camera(s) **1868**, infrared camera(s) **1872**, etc.,) as described herein.
[0187] Inference and/or training logic **815** are used to perform inferencing and/or training operations associated with one or more embodiments. In at least one embodiment, inference and/or training logic **815** may be used in system FIG. **18**B for inferencing or predicting operations based, at least in part, on weight parameters calculated using neural network training operations, neural network functions and/or architectures, or neural network use cases described herein.
[0188] Such components can be used to generate a single, consistent tokenized description of at least a portion of a physical environment based in part on a set of observations and aligned map data.
[0189] FIG. **18**C is a block diagram illustrating an example system architecture for autonomous vehicle **1800** of FIG. **18**A, according to at least one embodiment. In at least one embodiment, each of components, features, and systems of vehicle **1800** in FIG. **18**C is illustrated as being connected via a bus **1802**. In at least one embodiment, bus **1802** may include, without limitation, a CAN data interface (alternatively referred to herein as a "CAN bus"). In at least one embodiment, a CAN may be a network inside vehicle **1800** used to aid in control of various features and functionality of vehicle **1800**, such as actuation of brakes, acceleration, braking, steering, windshield wipers, etc. In at least one embodiment, bus **1802** may be configured to have dozens or even hundreds of nodes, each with its own unique identifier (e.g., a CAN ID). In at least one embodiment, bus **1802** may be read to find steering wheel angle, ground speed, engine revolutions per minute ("RPMs"), button positions, and/or other vehicle status indicators. In at least one embodiment, bus **1802** may be a CAN bus that is ASIL B compliant.
[0190] In at least one embodiment, in addition to, or alternatively from CAN, FlexRay and/or Ethernet protocols may be used. In at least one embodiment, there may be any number of busses forming bus **1802**, which may include, without limitation, zero or more CAN busses, zero or more FlexRay busses, zero or more Ethernet busses, and/or zero or more other types of busses using different protocols. In at least one embodiment, two or more busses may be used to perform different functions, and/or may be used for redundancy. For example, a first bus may be used for collision avoidance functionality and a second bus may be used for actuation control. In at least one embodiment, each bus of bus **1802** may communicate with any of components of vehicle **1800**, and two or more busses of bus **1802** may communicate with corresponding components. In at least one embodiment, each of any number of system(s) on chip(s) ("SoC(s)") **1804** (such as SoC **1804**(A) and SoC **1804**(B)), each of controller(s) **1836**, and/or each computer within vehicle may have access to same input data (e.g., inputs from sensors of vehicle **1800**), and may be connected to a common bus, such CAN bus.
[0191] In at least one embodiment, vehicle **1800** may include one or more controller(s) **1836**, such as those described herein with respect to FIG. **18**A. In at least one embodiment, controller(s) **1836** may be used for a variety of functions. In at least one embodiment, controller(s) **1836** may be coupled to any of various other components and systems of vehicle **1800**, and may be used for control of vehicle **1800**, artificial intelligence of vehicle **1800**, infotainment for vehicle **1800**, and/or other functions.
[0192] In at least one embodiment, vehicle **1800** may include any number of SoCs **1804**. In at least one embodiment, each of SoCs **1804** may include, without limitation, central processing units ("CPU(s)") **1806**, graphics processing units ("GPU(s)") **1808**, processor(s) **1810**, cache(s) **1812**, accelerator(s) **1814**, data store(s) **1816**, and/or other components and features not illustrated. In at least one embodiment, SoC(s) **1804** may be used to control vehicle **1800** in a variety of platforms and systems. For example, in at least one embodiment, SoC(s) **1804** may be combined in a system (e.g., system of vehicle **1800**) with a High Definition ("HD") map **1822** which may obtain map refreshes and/or updates via network interface **1824** from one or more servers (not shown in FIG. **18**C).
[0193] In at least one embodiment, CPU(s) **1806** may include a CPU cluster or CPU complex

(alternatively referred to herein as a “CCPLEX”). In at least one embodiment, CPU(s) **1806** may include multiple cores and/or level two (“L2”) caches. For instance, in at least one embodiment, CPU(s) **1806** may include eight cores in a coherent multi-processor configuration. In at least one embodiment, CPU(s) **1806** may include four dual-core clusters where each cluster has a dedicated L2 cache (e.g., a 2 megabyte (MB) L2 cache). In at least one embodiment, CPU(s) **1806** (e.g., CCPLEX) may be configured to support simultaneous cluster operations enabling any combination of clusters of CPU(s) **1806** to be active at any given time.

[0194] In at least one embodiment, one or more of CPU(s) **1806** may implement power management capabilities that include, without limitation, one or more of following features: individual hardware blocks may be clock-gated automatically when idle to save dynamic power; each core clock may be gated when such core is not actively executing instructions due to execution of Wait for Interrupt (“WFI”)/Wait for Event (“WFE”) instructions; each core may be independently power-gated; each core cluster may be independently clock-gated when all cores are clock-gated or power-gated; and/or each core cluster may be independently power-gated when all cores are power-gated. In at least one embodiment, CPU(s) **1806** may further implement an enhanced algorithm for managing power states, where allowed power states and expected wakeup times are specified, and hardware/microcode determines which best power state to enter for core, cluster, and CCPLEX. In at least one embodiment, processing cores may support simplified power state entry sequences in software with work offloaded to microcode.

[0195] In at least one embodiment, GPU(s) **1808** may include an integrated GPU (alternatively referred to herein as an “iGPU”). In at least one embodiment, GPU(s) **1808** may be programmable and may be efficient for parallel workloads. In at least one embodiment, GPU(s) **1808** may use an enhanced tensor instruction set. In at least one embodiment, GPU(s) **1808** may include one or more streaming microprocessors, where each streaming microprocessor may include a level one (“L1”) cache (e.g., an L1 cache with at least 96 KB storage capacity), and two or more streaming microprocessors may share an L2 cache (e.g., an L2 cache with a 512 KB storage capacity). In at least one embodiment, GPU(s) **1808** may include at least eight streaming microprocessors. In at least one embodiment, GPU(s) **1808** may use compute application programming interface(s) (API(s)). In at least one embodiment, GPU(s) **1808** may use one or more parallel computing platforms and/or programming models (e.g., NVIDIA's CUDA model).

[0196] In at least one embodiment, one or more of GPU(s) **1808** may be power-optimized for best performance in automotive and embedded use cases. For example, in at least one embodiment, GPU(s) **1808** could be fabricated on Fin field-effect transistor (“FinFET”) circuitry. In at least one embodiment, each streaming microprocessor may incorporate a number of mixed-precision processing cores partitioned into multiple blocks. For example, and without limitation, 64 PF32 cores and 32 PF64 cores could be partitioned into four processing blocks. In at least one embodiment, each processing block could be allocated 16 FP32 cores, 8 FP64 cores, 16 INT32 cores, two mixed-precision NVIDIA Tensor cores for deep learning matrix arithmetic, a level zero (“L0”) instruction cache, a scheduler (e.g., warp scheduler) or sequencer, a dispatch unit, and/or a 64 KB register file. In at least one embodiment, streaming microprocessors may include independent parallel integer and floating-point data paths to provide for efficient execution of workloads with a mix of computation and addressing calculations. In at least one embodiment, streaming microprocessors may include independent thread scheduling capability to enable finer-grain synchronization and cooperation between parallel threads. In at least one embodiment, streaming microprocessors may include a combined L1 data cache and shared memory unit in order to improve performance while simplifying programming.

[0197] In at least one embodiment, one or more of GPU(s) **1808** may include a high bandwidth memory (“HBM”) and/or a 16 GB HBM2 memory subsystem to provide, in some examples, about 900 GB/second peak memory bandwidth. In at least one embodiment, in addition to, or alternatively from, HBM memory, a synchronous graphics random-access memory (“SGRAM”)

may be used, such as a graphics double data rate type five synchronous random-access memory ("GDDR5").

[0198] In at least one embodiment, GPU(s) **1808** may include unified memory technology. In at least one embodiment, address translation services ("ATS") support may be used to allow GPU(s) **1808** to access CPU(s) **1806** page tables directly. In at least one embodiment, embodiment, when a GPU of GPU(s) **1808** memory management unit ("MMU") experiences a miss, an address translation request may be transmitted to CPU(s) **1806**. In response, 2 CPU of CPU(s) **1806** may look in its page tables for a virtual-to-physical mapping for an address and transmit translation back to GPU(s) **1808**, in at least one embodiment. In at least one embodiment, unified memory technology may allow a single unified virtual address space for memory of both CPU(s) **1806** and GPU(s) **1808**, thereby simplifying GPU(s) **1808** programming and porting of applications to GPU(s) **1808**.

[0199] In at least one embodiment, GPU(s) **1808** may include any number of access counters that may keep track of frequency of access of GPU(s) **1808** to memory of other processors. In at least one embodiment, access counter(s) may help ensure that memory pages are moved to physical memory of a processor that is accessing pages most frequently, thereby improving efficiency for memory ranges shared between processors.

[0200] In at least one embodiment, one or more of SoC(s) **1804** may include any number of cache(s) **1812**, including those described herein. For example, in at least one embodiment, cache(s) **1812** could include a level three ("L3") cache that is available to both CPU(s) **1806** and GPU(s) **1808** (e.g., that is connected to CPU(s) **1806** and GPU(s) **1808**). In at least one embodiment, cache(s) **1812** may include a write-back cache that may keep track of states of lines, such as by using a cache coherence protocol (e.g., MEI, MESI, MSI, etc.). In at least one embodiment, a L3 cache may include 4 MB of memory or more, depending on embodiment, although smaller cache sizes may be used.

[0201] In at least one embodiment, one or more of SoC(s) **1804** may include one or more accelerator(s) **1814** (e.g., hardware accelerators, software accelerators, or a combination thereof). In at least one embodiment, SoC(s) **1804** may include a hardware acceleration cluster that may include optimized hardware accelerators and/or large on-chip memory. In at least one embodiment, large on-chip memory (e.g., 4 MB of SRAM), may enable a hardware acceleration cluster to accelerate neural networks and other calculations. In at least one embodiment, a hardware acceleration cluster may be used to complement GPU(s) **1808** and to off-load some of tasks of GPU(s) **1808** (e.g., to free up more cycles of GPU(s) **1808** for performing other tasks). In at least one embodiment, accelerator(s) **1814** could be used for targeted workloads (e.g., perception, convolutional neural networks ("CNNs"), recurrent neural networks ("RNNs"), etc.) that are stable enough to be amenable to acceleration. In at least one embodiment, a CNN may include a region-based or regional convolutional neural networks ("RCNNs") and Fast RCNNs (e.g., as used for object detection) or other type of CNN.

[0202] In at least one embodiment, accelerator(s) **1814** (e.g., hardware acceleration cluster) may include one or more deep learning accelerator ("DLA"). In at least one embodiment, DLA(s) may include, without limitation, one or more Tensor processing units ("TPUs") that may be configured to provide an additional ten trillion operations per second for deep learning applications and inferencing. In at least one embodiment, TPUs may be accelerators configured to, and optimized for, performing image processing functions (e.g., for CNNs, RCNNs, etc.). In at least one embodiment, DLA(s) may further be optimized for a specific set of neural network types and floating point operations, as well as inferencing. In at least one embodiment, design of DLA(s) may provide more performance per millimeter than a typical general-purpose GPU, and typically vastly exceeds performance of a CPU. In at least one embodiment, TPU(s) may perform several functions, including a single-instance convolution function, supporting, for example, INT8, INT16, and FP16 data types for both features and weights, as well as post-processor functions. In at least

one embodiment, DLA(s) may quickly and efficiently execute neural networks, especially CNNs, on processed or unprocessed data for any of a variety of functions, including, for example and without limitation: a CNN for object identification and detection using data from camera sensors; a CNN for distance estimation using data from camera sensors; a CNN for emergency vehicle detection and identification and detection using data from microphones; a CNN for facial recognition and vehicle owner identification using data from camera sensors; and/or a CNN for security and/or safety related events.

[0203] In at least one embodiment, DLA(s) may perform any function of GPU(s) **1808**, and by using an inference accelerator, for example, a designer may target either DLA(s) or GPU(s) **1808** for any function. For example, in at least one embodiment, a designer may focus processing of CNNs and floating point operations on DLA(s) and leave other functions to GPU(s) **1808** and/or accelerator(s) **1814**.

[0204] In at least one embodiment, accelerator(s) **1814** may include programmable vision accelerator ("PVA"), which may alternatively be referred to herein as a computer vision accelerator. In at least one embodiment, PVA may be designed and configured to accelerate computer vision algorithms for advanced driver assistance system ("ADAS") **1838**, autonomous driving, augmented reality ("AR") applications, and/or virtual reality ("VR") applications. In at least one embodiment, PVA may provide a balance between performance and flexibility. For example, in at least one embodiment, each PVA may include, for example and without limitation, any number of reduced instruction set computer ("RISC") cores, direct memory access ("DMA"), and/or any number of vector processors.

[0205] In at least one embodiment, RISC cores may interact with image sensors (e.g., image sensors of any cameras described herein), image signal processor(s), etc. In at least one embodiment, each RISC core may include any amount of memory. In at least one embodiment, RISC cores may use any of a number of protocols, depending on embodiment. In at least one embodiment, RISC cores may execute a real-time operating system ("RTOS"). In at least one embodiment, RISC cores may be implemented using one or more integrated circuit devices, application specific integrated circuits ("ASICs"), and/or memory devices. For example, in at least one embodiment, RISC cores could include an instruction cache and/or a tightly coupled RAM.

[0206] In at least one embodiment, DMA may enable components of PVA to access system memory independently of CPU(s) **1806**. In at least one embodiment, DMA may support any number of features used to provide optimization to a PVA including, but not limited to, supporting multi-dimensional addressing and/or circular addressing. In at least one embodiment, DMA may support up to six or more dimensions of addressing, which may include, without limitation, block width, block height, block depth, horizontal block stepping, vertical block stepping, and/or depth stepping.

[0207] In at least one embodiment, vector processors may be programmable processors that may be designed to efficiently and flexibly execute programming for computer vision algorithms and provide signal processing capabilities. In at least one embodiment, a PVA may include a PVA core and two vector processing subsystem partitions. In at least one embodiment, a PVA core may include a processor subsystem, DMA engine(s) (e.g., two DMA engines), and/or other peripherals. In at least one embodiment, a vector processing subsystem may operate as a primary processing engine of a PVA, and may include a vector processing unit ("VPU"), an instruction cache, and/or vector memory (e.g., "VMEM"). In at least one embodiment, VPU core may include a digital signal processor such as, for example, a single instruction, multiple data ("SIMD"), very long instruction word ("VLIW") digital signal processor. In at least one embodiment, a combination of SIMD and VLIW may enhance throughput and speed.

[0208] In at least one embodiment, each of vector processors may include an instruction cache and may be coupled to dedicated memory. As a result, in at least one embodiment, each of vector processors may be configured to execute independently of other vector processors. In at least one

embodiment, vector processors that are included in a particular PVA may be configured to employ data parallelism. For instance, in at least one embodiment, plurality of vector processors included in a single PVA may execute a common computer vision algorithm, but on different regions of an image. In at least one embodiment, vector processors included in a particular PVA may simultaneously execute different computer vision algorithms, on one image, or even execute different algorithms on sequential images or portions of an image. In at least one embodiment, among other things, any number of PVAs may be included in hardware acceleration cluster and any number of vector processors may be included in each PVA. In at least one embodiment, PVA may include additional error correcting code ("ECC") memory, to enhance overall system safety.

[0209] In at least one embodiment, accelerator(s) **1814** may include a computer vision network on-chip and static random-access memory ("SRAM"), for providing a high-bandwidth, low latency SRAM for accelerator(s) **1814**. In at least one embodiment, on-chip memory may include at least 4 MB SRAM, comprising, for example and without limitation, eight field-configurable memory blocks, that may be accessible by both a PVA and a DLA. In at least one embodiment, each pair of memory blocks may include an advanced peripheral bus ("APB") interface, configuration circuitry, a controller, and a multiplexer. In at least one embodiment, any type of memory may be used. In at least one embodiment, a PVA and a DLA may access memory via a backbone that provides a PVA and a DLA with high-speed access to memory. In at least one embodiment, a backbone may include a computer vision network on-chip that interconnects a PVA and a DLA to memory (e.g., using APB).

[0210] In at least one embodiment, a computer vision network on-chip may include an interface that determines, before transmission of any control signal/address/data, that both a PVA and a DLA provide ready and valid signals. In at least one embodiment, an interface may provide for separate phases and separate channels for transmitting control signals/addresses/data, as well as burst-type communications for continuous data transfer. In at least one embodiment, an interface may comply with International Organization for Standardization ("ISO") 26262 or International Electrotechnical Commission ("IEC") 61508 standards, although other standards and protocols may be used.

[0211] In at least one embodiment, one or more of SoC(s) **1804** may include a real-time ray-tracing hardware accelerator. In at least one embodiment, real-time ray-tracing hardware accelerator may be used to quickly and efficiently determine positions and extents of objects (e.g., within a world model), to generate real-time visualization simulations, for RADAR signal interpretation, for sound propagation synthesis and/or analysis, for simulation of SONAR systems, for general wave propagation simulation, for comparison to LiDAR data for purposes of localization and/or other functions, and/or for other uses.

[0212] In at least one embodiment, accelerator(s) **1814** can have a wide array of uses for autonomous driving. In at least one embodiment, a PVA may be used for key processing stages in ADAS and autonomous vehicles. In at least one embodiment, a PVA's capabilities are a good match for algorithmic domains needing predictable processing, at low power and low latency. In other words, a PVA performs well on semi-dense or dense regular computation, even on small data sets, which might require predictable run-times with low latency and low power. In at least one embodiment, such as in vehicle **1800**, PVAs might be designed to run classic computer vision algorithms, as they can be efficient at object detection and operating on integer math.

[0213] For example, according to at least one embodiment of technology, a PVA is used to perform computer stereo vision. In at least one embodiment, a semi-global matching-based algorithm may be used in some examples, although this is not intended to be limiting. In at least one embodiment, applications for Level 3-5 autonomous driving use motion estimation/stereo matching on-the-fly (e.g., structure from motion, pedestrian recognition, lane detection, etc.). In at least one embodiment, a PVA may perform computer stereo vision functions on inputs from two monocular cameras.

[0214] In at least one embodiment, a PVA may be used to perform dense optical flow. For example,

in at least one embodiment, a PVA could process raw RADAR data (e.g., using a 4D Fast Fourier Transform) to provide processed RADAR data. In at least one embodiment, a PVA is used for time of flight depth processing, by processing raw time of flight data to provide processed time of flight data, for example.

[0215] In at least one embodiment, a DLA may be used to run any type of network to enhance control and driving safety, including for example and without limitation, a neural network that outputs a measure of confidence for each object detection. In at least one embodiment, confidence may be represented or interpreted as a probability, or as providing a relative "weight" of each detection compared to other detections. In at least one embodiment, a confidence measure enables a system to make further decisions regarding which detections should be considered as true positive detections rather than false positive detections. In at least one embodiment, a system may set a threshold value for confidence and consider only detections exceeding threshold value as true positive detections. In an embodiment in which an automatic emergency braking ("AEB") system is used, false positive detections would cause vehicle to automatically perform emergency braking, which is obviously undesirable. In at least one embodiment, highly confident detections may be considered as triggers for AEB. In at least one embodiment, a DLA may run a neural network for regressing confidence value. In at least one embodiment, neural network may take as its input at least some subset of parameters, such as bounding box dimensions, ground plane estimate obtained (e.g., from another subsystem), output from IMU sensor(s) **1866** that correlates with vehicle **1800** orientation, distance, 3D location estimates of object obtained from neural network and/or other sensors (e.g., LiDAR sensor(s) **1864** or RADAR sensor(s) **1860**), among others.

[0216] In at least one embodiment, one or more of SoC(s) **1804** may include data store(s) **1816** (e.g., memory). In at least one embodiment, data store(s) **1816** may be on-chip memory of SoC(s) **1804**, which may store neural networks to be executed on GPU(s) **1808** and/or a DLA. In at least one embodiment, data store(s) **1816** may be large enough in capacity to store multiple instances of neural networks for redundancy and safety. In at least one embodiment, data store(s) **1816** may comprise L2 or L3 cache(s).

[0217] In at least one embodiment, one or more of SoC(s) **1804** may include any number of processor(s) **1810** (e.g., embedded processors). In at least one embodiment, processor(s) **1810** may include a boot and power management processor that may be a dedicated processor and subsystem to handle boot power and management functions and related security enforcement. In at least one embodiment, a boot and power management processor may be a part of a boot sequence of SoC(s) **1804** and may provide runtime power management services. In at least one embodiment, a boot power and management processor may provide clock and voltage programming, assistance in system low power state transitions, management of SoC(s) **1804** thermals and temperature sensors, and/or management of SoC(s) **1804** power states. In at least one embodiment, each temperature sensor may be implemented as a ring-oscillator whose output frequency is proportional to temperature, and SoC(s) **1804** may use ring-oscillators to detect temperatures of CPU(s) **1806**, GPU(s) **1808**, and/or accelerator(s) **1814**. In at least one embodiment, if temperatures are determined to exceed a threshold, then a boot and power management processor may enter a temperature fault routine and put SoC(s) **1804** into a lower power state and/or put vehicle **1800** into a chauffeur to safe stop mode (e.g., bring vehicle **1800** to a safe stop).

[0218] In at least one embodiment, processor(s) **1810** may further include a set of embedded processors that may serve as an audio processing engine which may be an audio subsystem that enables full hardware support for multi-channel audio over multiple interfaces, and a broad and flexible range of audio I/O interfaces. In at least one embodiment, an audio processing engine is a dedicated processor core with a digital signal processor with dedicated RAM.

[0219] In at least one embodiment, processor(s) **1810** may further include an alwayson processor engine that may provide necessary hardware features to support low power sensor management and wake use cases. In at least one embodiment, an alwayson processor engine may include, without

limitation, a processor core, a tightly coupled RAM, supporting peripherals (e.g., timers and interrupt controllers), various I/O controller peripherals, and routing logic.
[0220] In at least one embodiment, processor(s) **1810** may further include a safety cluster engine that includes, without limitation, a dedicated processor subsystem to handle safety management for automotive applications. In at least one embodiment, a safety cluster engine may include, without limitation, two or more processor cores, a tightly coupled RAM, support peripherals (e.g., timers, an interrupt controller, etc.), and/or routing logic. In a safety mode, two or more cores may operate, in at least one embodiment, in a lockstep mode and function as a single core with comparison logic to detect any differences between their operations. In at least one embodiment, processor(s) **1810** may further include a real-time camera engine that may include, without limitation, a dedicated processor subsystem for handling real-time camera management. In at least one embodiment, processor(s) **1810** may further include a high-dynamic range signal processor that may include, without limitation, an image signal processor that is a hardware engine that is part of a camera processing pipeline.
[0221] In at least one embodiment, processor(s) **1810** may include a video image compositor that may be a processing block (e.g., implemented on a microprocessor) that implements video post-processing functions needed by a video playback application to produce a final image for a player window. In at least one embodiment, a video image compositor may perform lens distortion correction on wide-view camera(s) **1870**, surround camera(s) **1874**, and/or on in-cabin monitoring camera sensor(s). In at least one embodiment, in-cabin monitoring camera sensor(s) are preferably monitored by a neural network running on another instance of SoC **1804**, configured to identify in cabin events and respond accordingly. In at least one embodiment, an in-cabin system may perform, without limitation, lip reading to activate cellular service and place a phone call, dictate emails, change a vehicle's destination, activate or change a vehicle's infotainment system and settings, or provide voice-activated web surfing. In at least one embodiment, certain functions are available to a driver when a vehicle is operating in an autonomous mode and are disabled otherwise.
[0222] In at least one embodiment, a video image compositor may include enhanced temporal noise reduction for both spatial and temporal noise reduction. For example, in at least one embodiment, where motion occurs in a video, noise reduction weights spatial information appropriately, decreasing weights of information provided by adjacent frames. In at least one embodiment, where an image or portion of an image does not include motion, temporal noise reduction performed by video image compositor may use information from a previous image to reduce noise in a current image.
[0223] In at least one embodiment, a video image compositor may also be configured to perform stereo rectification on input stereo lens frames. In at least one embodiment, a video image compositor may further be used for user interface composition when an operating system desktop is in use, and GPU(s) **1808** are not required to continuously render new surfaces. In at least one embodiment, when GPU(s) **1808** are powered on and active doing 3D rendering, a video image compositor may be used to offload GPU(s) **1808** to improve performance and responsiveness.
[0224] In at least one embodiment, one or more SoC of SoC(s) **1804** may further include a mobile industry processor interface (“MIPI”) camera serial interface for receiving video and input from cameras, a high-speed interface, and/or a video input block that may be used for a camera and related pixel input functions. In at least one embodiment, one or more of SoC(s) **1804** may further include an input/output controller(s) that may be controlled by software and may be used for receiving I/O signals that are uncommitted to a specific role.
[0225] In at least one embodiment, one or more Soc of SoC(s) **1804** may further include a broad range of peripheral interfaces to enable communication with peripherals, audio encoders/decoders (“codecs”), power management, and/or other devices. In at least one embodiment, SoC(s) **1804** may be used to process data from cameras (e.g., connected over Gigabit Multimedia Serial Link

and Ethernet channels), sensors (e.g., LiDAR sensor(s) **1864**, RADAR sensor(s) **1860**, etc. that may be connected over Ethernet channels), data from bus **1802** (e.g., speed of vehicle **1800**, steering wheel position, etc.), data from GNSS sensor(s) **1858** (e.g., connected over a Ethernet bus or a CAN bus), etc. In at least one embodiment, one or more SoC of SoC(s) **1804** may further include dedicated high-performance mass storage controllers that may include their own DMA engines, and that may be used to free CPU(s) **1806** from routine data management tasks.

[0226] In at least one embodiment, SoC(s) **1804** may be an end-to-end platform with a flexible architecture that spans automation Levels 3-5, thereby providing a comprehensive functional safety architecture that leverages and makes efficient use of computer vision and ADAS techniques for diversity and redundancy, and provides a platform for a flexible, reliable driving software stack, along with deep learning tools. In at least one embodiment, SoC(s) **1804** may be faster, more reliable, and even more energy-efficient and space-efficient than conventional systems. For example, in at least one embodiment, accelerator(s) **1814**, when combined with CPU(s) **1806**, GPU(s) **1808**, and data store(s) **1816**, may provide for a fast, efficient platform for Level 3-5 autonomous vehicles.

[0227] In at least one embodiment, computer vision algorithms may be executed on CPUs, which may be configured using a high-level programming language, such as C, to execute a wide variety of processing algorithms across a wide variety of visual data. However, in at least one embodiment, CPUs are oftentimes unable to meet performance requirements of many computer vision applications, such as those related to execution time and power consumption, for example. In at least one embodiment, many CPUs are unable to execute complex object detection algorithms in real-time, which is used in in-vehicle ADAS applications and in practical Level 3-5 autonomous vehicles.

[0228] Embodiments described herein allow for multiple neural networks to be performed simultaneously and/or sequentially, and for results to be combined together to enable Level 3-5 autonomous driving functionality. For example, in at least one embodiment, a CNN executing on a DLA or a discrete GPU (e.g., GPU(s) **1820**) may include text and word recognition, allowing reading and understanding of traffic signs, including signs for which a neural network has not been specifically trained. In at least one embodiment, a DLA may further include a

[0229] neural network that is able to identify, interpret, and provide semantic understanding of a sign, and to pass that semantic understanding to path planning modules running on a CPU Complex.

[0230] In at least one embodiment, multiple neural networks may be run simultaneously, as for Level 3, 4, or 5 driving. For example, in at least one embodiment, a warning sign stating “Caution: flashing lights indicate icy conditions,” along with an electric light, may be independently or collectively interpreted by several neural networks. In at least one embodiment, such warning sign itself may be identified as a traffic sign by a first deployed neural network (e.g., a neural network that has been trained), text “flashing lights indicate icy conditions” may be interpreted by a second deployed neural network, which informs a vehicle's path planning software (preferably executing on a CPU Complex) that when flashing lights are detected, icy conditions exist. In at least one embodiment, a flashing light may be identified by operating a third deployed neural network over multiple frames, informing a vehicle's path-planning software of a presence (or an absence) of flashing lights. In at least one embodiment, all three neural networks may run simultaneously, such as within a DLA and/or on GPU(s) **1808**.

[0231] In at least one embodiment, a CNN for facial recognition and vehicle owner identification may use data from camera sensors to identify presence of an authorized driver and/or owner of vehicle **1800**. In at least one embodiment, an alwayson sensor processing engine may be used to unlock a vehicle when an owner approaches a driver door and turns on lights, and, in a security mode, to disable such vehicle when an owner leaves such vehicle. In this way, SoC(s) **1804** provide for security against theft and/or carjacking.

[0232] In at least one embodiment, a CNN for emergency vehicle detection and identification may use data from microphones **1896** to detect and identify emergency vehicle sirens. In at least one embodiment, SoC(s) **1804** use a CNN for classifying environmental and urban sounds, as well as classifying visual data. In at least one embodiment, a CNN running on a DLA is trained to identify a relative closing speed of an emergency vehicle (e.g., by using a Doppler effect). In at least one embodiment, a CNN may also be trained to identify emergency vehicles specific to a local area in which a vehicle is operating, as identified by GNSS sensor(s) **1858**. In at least one embodiment, when operating in Europe, a CNN will seek to detect European sirens, and when in North America, a CNN will seek to identify only North American sirens. In at least one embodiment, once an emergency vehicle is detected, a control program may be used to execute an emergency vehicle safety routine, slowing a vehicle, pulling over to a side of a road, parking a vehicle, and/or idling a vehicle, with assistance of ultrasonic sensor(s) **1862**, until emergency vehicles pass.
[0233] In at least one embodiment, vehicle **1800** may include CPU(s) **1818** (e.g., discrete CPU(s), or dCPU(s)), that may be coupled to SoC(s) **1804** via a high-speed interconnect (e.g., PCIe). In at least one embodiment, CPU(s) **1818** may include an X86 processor, for example. CPU(s) **1818** may be used to perform any of a variety of functions, including arbitrating potentially inconsistent results between ADAS sensors and SoC(s) **1804**, and/or monitoring status and health of controller(s) **1836** and/or an infotainment system on a chip ("infotainment SoC") **1830**, for example. In at least one embodiment, SoC(s) **1804** includes one or more interconnects, and an interconnect can include a peripheral component interconnect express (PCIe).
[0234] In at least one embodiment, vehicle **1800** may include GPU(s) **1820** (e.g., discrete GPU(s), or dGPU(s)), that may be coupled to SoC(s) **1804** via a high-speed interconnect (e.g., NVIDIA's NVLINK channel). In at least one embodiment, GPU(s) **1820** may provide additional artificial intelligence functionality, such as by executing redundant and/or different neural networks, and may be used to train and/or update neural networks based at least in part on input (e.g., sensor data) from sensors of a vehicle **1800**.
[0235] In at least one embodiment, vehicle **1800** may further include network interface **1824** which may include, without limitation, wireless antenna(s) **1826** (e.g., one or more wireless antennas for different communication protocols, such as a cellular antenna, a Bluetooth antenna, etc.). In at least one embodiment, network interface **1824** may be used to enable wireless connectivity to Internet cloud services (e.g., with server(s) and/or other network devices), with other vehicles, and/or with computing devices (e.g., client devices of passengers). In at least one embodiment, to communicate with other vehicles, a direct link may be established between vehicle **1800** and another vehicle and/or an indirect link may be established (e.g., across networks and over the Internet). In at least one embodiment, direct links may be provided using a vehicle-to-vehicle communication link. In at least one embodiment, a vehicle-to-vehicle communication link may provide vehicle **1800** information about vehicles in proximity to vehicle **1800** (e.g., vehicles in front of, on a side of, and/or behind vehicle **1800**). In at least one embodiment, such aforementioned functionality may be part of a cooperative adaptive cruise control functionality of vehicle **1800**.
[0236] In at least one embodiment, network interface **1824** may include an SoC that provides modulation and demodulation functionality and enables controller(s) **1836** to communicate over wireless networks. In at least one embodiment, network interface **1824** may include a radio frequency front-end for up-conversion from baseband to radio frequency, and down conversion from radio frequency to baseband. In at least one embodiment, frequency conversions may be performed in any technically feasible fashion. For example, frequency conversions could be performed through well-known processes, and/or using super-heterodyne processes. In at least one embodiment, radio frequency front end functionality may be provided by a separate chip. In at least one embodiment, network interfaces may include wireless functionality for communicating over LTE, WCDMA, UMTS, GSM, CDMA2000, Bluetooth, Bluetooth LE, Wi-Fi, Z-Wave, ZigBee, LoRaWAN, and/or other wireless protocols.

[0237] In at least one embodiment, vehicle **1800** may further include data store(s) **1828** which may include, without limitation, off-chip (e.g., off SoC(s) **1804**) storage. In at least one embodiment, data store(s) **1828** may include, without limitation, one or more storage elements including RAM, SRAM, dynamic random-access memory ("DRAM"), video random-access memory ("VRAM"), flash memory, hard disks, and/or other components and/or devices that may store at least one bit of data.
[0238] In at least one embodiment, vehicle **1800** may further include GNSS sensor(s) **1858** (e.g., GPS and/or assisted GPS sensors), to assist in mapping, perception, occupancy grid generation, and/or path planning functions. In at least one embodiment, any number of GNSS sensor(s) **1858** may be used, including, for example and without limitation, a GPS using a USB connector with an Ethernet-to-Serial (e.g., RS-232) bridge.
[0239] In at least one embodiment, vehicle **1800** may further include RADAR sensor(s) **1860**. In at least one embodiment, RADAR sensor(s) **1860** may be used by vehicle **1800** for long-range vehicle detection, even in darkness and/or severe weather conditions. In at least one embodiment, RADAR functional safety levels may be ASIL B. In at least one embodiment, RADAR sensor(s) **1860** may use a CAN bus and/or bus **1802** (e.g., to transmit data generated by RADAR sensor(s) **1860**) for control and to access object tracking data, with access to Ethernet channels to access raw data in some examples. In at least one embodiment, a wide variety of RADAR sensor types may be used. For example, and without limitation, RADAR sensor(s) **1860** may be suitable for front, rear, and side RADAR use. In at least one embodiment, one or more sensor of RADAR sensors(s) **1860** is a Pulse Doppler RADAR sensor.
[0240] In at least one embodiment, RADAR sensor(s) **1860** may include different configurations, such as long-range with narrow field of view, short-range with wide field of view, short-range side coverage, etc. In at least one embodiment, long-range RADAR may be used for adaptive cruise control functionality. In at least one embodiment, long-range RADAR systems may provide a broad field of view realized by two or more independent scans, such as within a 250 m (meter) range. In at least one embodiment, RADAR sensor(s) **1860** may help in distinguishing between static and moving objects, and may be used by ADAS system **1838** for emergency brake assist and forward collision warning. In at least one embodiment, sensors **1860** (*s*) included in a long-range RADAR system may include, without limitation, monostatic multimodal RADAR with multiple (e.g., six or more) fixed RADAR antennae and a high-speed CAN and FlexRay interface. In at least one embodiment, with six antennae, a central four antennae may create a focused beam pattern, designed to record vehicle's **1800** surroundings at higher speeds with minimal interference from traffic in adjacent lanes. In at least one embodiment, another two antennae may expand field of view, making it possible to quickly detect vehicles entering or leaving a lane of vehicle **1800**.
[0241] In at least one embodiment, mid-range RADAR systems may include, as an example, a range of up to 160 m (front) or 80 m (rear), and a field of view of up to 42 degrees (front) or 150 degrees (rear). In at least one embodiment, short-range RADAR systems may include, without limitation, any number of RADAR sensor(s) **1860** designed to be installed at both ends of a rear bumper. When installed at both ends of a rear bumper, in at least one embodiment, a RADAR sensor system may create two beams that constantly monitor blind spots in a rear direction and next to a vehicle. In at least one embodiment, short-range RADAR systems may be used in ADAS system **1838** for blind spot detection and/or lane change assist.
[0242] In at least one embodiment, vehicle **1800** may further include ultrasonic sensor(s) **1862**. In at least one embodiment, ultrasonic sensor(s) **1862**, which may be positioned at a front, a back, and/or side location of vehicle **1800**, may be used for parking assist and/or to create and update an occupancy grid. In at least one embodiment, a wide variety of ultrasonic sensor(s) **1862** may be used, and different ultrasonic sensor(s) **1862** may be used for different ranges of detection (e.g., 2.5 m, 4 m). In at least one embodiment, ultrasonic sensor(s) **1862** may operate at functional safety levels of ASIL B.

[0243] In at least one embodiment, vehicle **1800** may include LiDAR sensor(s) **1864**. In at least one embodiment, LiDAR sensor(s) **1864** may be used for object and pedestrian detection, emergency braking, collision avoidance, and/or other functions. In at least one embodiment, LiDAR sensor(s) **1864** may operate at functional safety level ASIL B. In at least one embodiment, vehicle **1800** may include multiple LiDAR sensors **1864** (e.g., two, four, six, etc.) that may use an Ethernet channel (e.g., to provide data to a Gigabit Ethernet switch).
[0244] In at least one embodiment, LiDAR sensor(s) **1864** may be capable of providing a list of objects and their distances for a 360-degree field of view. In at least one embodiment, commercially available LiDAR sensor(s) **1864** may have an advertised range of approximately 100 m, with an accuracy of 2 cm to 3 cm, and with support for a 100 Mbps Ethernet connection, for example. In at least one embodiment, one or more non-protruding LiDAR sensors may be used. In such an embodiment, LiDAR sensor(s) **1864** may include a small device that may be embedded into a front, a rear, a side, and/or a corner location of vehicle **1800**. In at least one embodiment, LiDAR sensor(s) **1864**, in such an embodiment, may provide up to a 120-degree horizontal and 35-degree vertical field-of-view, with a 200 m range even for low-reflectivity objects. In at least one embodiment, front-mounted LiDAR sensor(s) **1864** may be configured for a horizontal field of view between 45 degrees and 135 degrees.
[0245] In at least one embodiment, LiDAR technologies, such as 3D flash LiDAR, may also be used. In at least one embodiment, 3D flash LiDAR uses a flash of a laser as a transmission source, to illuminate surroundings of vehicle **1800** up to approximately 200 m. In at least one embodiment, a flash LiDAR unit includes, without limitation, a receptor, which records laser pulse transit time and reflected light on each pixel, which in turn corresponds to a range from vehicle **1800** to objects. In at least one embodiment, flash LiDAR may allow for highly accurate and distortion-free images of surroundings to be generated with every laser flash. In at least one embodiment, four flash LiDAR sensors may be deployed, one at each side of vehicle **1800**. In at least one embodiment, 3D flash LiDAR systems include, without limitation, a solid-state 3D staring array LiDAR camera with no moving parts other than a fan (e.g., a non-scanning LiDAR device). In at least one embodiment, flash LiDAR device may use a 5 nanosecond class I (eye-safe) laser pulse per frame and may capture reflected laser light as a 3D range point cloud and co-registered intensity data.
[0246] In at least one embodiment, vehicle **1800** may further include IMU sensor(s) **1866**. In at least one embodiment, IMU sensor(s) **1866** may be located at a center of a rear axle of vehicle **1800**. In at least one embodiment, IMU sensor(s) **1866** may include, for example and without limitation, accelerometer(s), magnetometer(s), gyroscope(s), a magnetic compass, magnetic compasses, and/or other sensor types. In at least one embodiment, such as in six-axis applications, IMU sensor(s) **1866** may include, without limitation, accelerometers and gyroscopes. In at least one embodiment, such as in nine-axis applications, IMU sensor(s) **1866** may include, without limitation, accelerometers, gyroscopes, and magnetometers.
[0247] In at least one embodiment, IMU sensor(s) **1866** may be implemented as a miniature, high performance GPS-Aided Inertial Navigation System ("GPS/INS") that combines micro-electro-mechanical systems ("MEMS") inertial sensors, a high-sensitivity GPS receiver, and advanced Kalman filtering algorithms to provide estimates of position, velocity, and attitude. In at least one embodiment, IMU sensor(s) **1866** may enable vehicle **1800** to estimate its heading without requiring input from a magnetic sensor by directly observing and correlating changes in velocity from a GPS to IMU sensor(s) **1866**. In at least one embodiment, IMU sensor(s) **1866** and GNSS sensor(s) **1858** may be combined in a single integrated unit.
[0248] In at least one embodiment, vehicle **1800** may include microphone(s) **1896** placed in and/or around vehicle **1800**. In at least one embodiment, microphone(s) **1896** may be used for emergency vehicle detection and identification, among other things.
[0249] In at least one embodiment, vehicle **1800** may further include any number of camera types, including stereo camera(s) **1868**, wide-view camera(s) **1870**, infrared camera(s) **1872**, surround

camera(s) **1874**, long-range camera(s) **1898**, mid-range camera(s) **1876**, and/or other camera types. In at least one embodiment, cameras may be used to capture image data around an entire periphery of vehicle **1800**. In at least one embodiment, which types of cameras used depends on vehicle **1800**. In at least one embodiment, any combination of camera types may be used to provide necessary coverage around vehicle **1800**. In at least one embodiment, a number of cameras deployed may differ depending on embodiment. For example, in at least one embodiment, vehicle **1800** could include six cameras, seven cameras, ten cameras, twelve cameras, or another number of cameras. In at least one embodiment, cameras may support, as an example and without limitation, Gigabit Multimedia Serial Link ("GMSL") and/or Gigabit Ethernet communications. In at least one embodiment, each camera might be as described with more detail previously herein with respect to FIG. **18**A and FIG. **18**B.

[0250] In at least one embodiment, vehicle **1800** may further include vibration sensor(s) **1842**. In at least one embodiment, vibration sensor(s) **1842** may measure vibrations of components of vehicle **1800**, such as axle(s). For example, in at least one embodiment, changes in vibrations may indicate a change in road surfaces. In at least one embodiment, when two or more vibration sensors **1842** are used, differences between vibrations may be used to determine friction or slippage of road surface (e.g., when a difference in vibration is between a power-driven axle and a freely rotating axle).

[0251] In at least one embodiment, vehicle **1800** may include ADAS system **1838**. In at least one embodiment, ADAS system **1838** may include, without limitation, an SoC, in some examples. In at least one embodiment, ADAS system **1838** may include, without limitation, any number and combination of an autonomous/adaptive/automatic cruise control ("ACC") system, a cooperative adaptive cruise control ("CACC") system, a forward crash warning ("FCW") system, an automatic emergency braking ("AEB") system, a lane departure warning ("LDW)" system, a lane keep assist ("LKA") system, a blind spot warning ("BSW") system, a rear cross-traffic warning ("RCTW") system, a collision warning ("CW") system, a lane centering ("LC") system, and/or other systems, features, and/or functionality.

[0252] In at least one embodiment, ACC system may use RADAR sensor(s) **1860**, LiDAR sensor(s) **1864**, and/or any number of camera(s). In at least one embodiment, ACC system may include a longitudinal ACC system and/or a lateral ACC system. In at least one embodiment, a longitudinal ACC system monitors and controls distance to another vehicle immediately ahead of vehicle **1800** and automatically adjusts speed of vehicle **1800** to maintain a safe distance from vehicles ahead. In at least one embodiment, a lateral ACC system performs distance keeping, and advises vehicle **1800** to change lanes when necessary. In at least one embodiment, a lateral ACC is related to other ADAS applications, such as LC and CW.

[0253] In at least one embodiment, a CACC system uses information from other vehicles that may be received via network interface **1824** and/or wireless antenna(s) **1826** from other vehicles via a wireless link, or indirectly, over a network connection (e.g., over the Internet). In at least one embodiment, direct links may be provided by a vehicle-to-vehicle ("V2V") communication link, while indirect links may be provided by an infrastructure-to-vehicle ("I2V") communication link. In general, V2V communication provides information about immediately preceding vehicles (e.g., vehicles immediately ahead of and in same lane as vehicle **1800**), while I2V communication provides information about traffic further ahead. In at least one embodiment, a CACC system may include either or both I2V and V2V information sources. In at least one embodiment, given information of vehicles ahead of vehicle **1800**, a CACC system may be more reliable and it has potential to improve traffic flow smoothness and reduce congestion on road.

[0254] In at least one embodiment, an FCW system is designed to alert a driver to a hazard, so that such driver may take corrective action. In at least one embodiment, an FCW system uses a front-facing camera and/or RADAR sensor(s) **1860**, coupled to a dedicated processor, DSP, FPGA, and/or ASIC, that is electrically coupled to provide driver feedback, such as a display, speaker,

and/or vibrating component. In at least one embodiment, an FCW system may provide a warning, such as in form of a sound, visual warning, vibration and/or a quick brake pulse.
[0255] In at least one embodiment, an AEB system detects an impending forward collision with another vehicle or other object, and may automatically apply brakes if a driver does not take corrective action within a specified time or distance parameter. In at least one embodiment, AEB system may use front-facing camera(s) and/or RADAR sensor(s) **1860**, coupled to a dedicated processor, DSP, FPGA, and/or ASIC. In at least one embodiment, when an AEB system detects a hazard, it will typically first alert a driver to take corrective action to avoid collision and, if that driver does not take corrective action, that AEB system may automatically apply brakes in an effort to prevent, or at least mitigate, an impact of a predicted collision. In at least one embodiment, an AEB system may include techniques such as dynamic brake support and/or crash imminent braking.
[0256] In at least one embodiment, an LDW system provides visual, audible, and/or tactile warnings, such as steering wheel or seat vibrations, to alert driver when vehicle **1800** crosses lane markings. In at least one embodiment, an LDW system does not activate when a driver indicates an intentional lane departure, such as by activating a turn signal. In at least one embodiment, an LDW system may use front-side facing cameras, coupled to a dedicated processor, DSP, FPGA, and/or ASIC, that is electrically coupled to provide driver feedback, such as a display, speaker, and/or vibrating component. In at least one embodiment, an LKA system is a variation of an LDW system. In at least one embodiment, an LKA system provides steering input or braking to correct vehicle **1800** if vehicle **1800** starts to exit its lane.
[0257] In at least one embodiment, a BSW system detects and warns a driver of vehicles in an automobile's blind spot. In at least one embodiment, a BSW system may provide a visual, audible, and/or tactile alert to indicate that merging or changing lanes is unsafe. In at least one embodiment, a BSW system may provide an additional warning when a driver uses a turn signal. In at least one embodiment, a BSW system may use rear-side facing camera(s) and/or RADAR sensor(s) **1860**, coupled to a dedicated processor, DSP, FPGA, and/or ASIC, that is electrically coupled to driver feedback, such as a display, speaker, and/or vibrating component.
[0258] In at least one embodiment, an RCTW system may provide visual, audible, and/or tactile notification when an object is detected outside a rear-camera range when vehicle **1800** is backing up. In at least one embodiment, an RCTW system includes an AEB system to ensure that vehicle brakes are applied to avoid a crash. In at least one embodiment, an RCTW system may use one or more rear-facing RADAR sensor(s) **1860**, coupled to a dedicated processor,
[0259] DSP, FPGA, and/or ASIC, that is electrically coupled to provide driver feedback, such as a display, speaker, and/or vibrating component.
[0260] In at least one embodiment, conventional ADAS systems may be prone to false positive results which may be annoying and distracting to a driver, but typically are not catastrophic, because conventional ADAS systems alert a driver and allow that driver to decide whether a safety condition truly exists and act accordingly. In at least one embodiment, vehicle **1800** itself decides, in case of conflicting results, whether to heed result from a primary computer or a secondary computer (e.g., a first controller or a second controller of controllers **1836**). For example, in at least one embodiment, ADAS system **1838** may be a backup and/or secondary computer for providing perception information to a backup computer rationality module. In at least one embodiment, a backup computer rationality monitor may run redundant diverse software on hardware components to detect faults in perception and dynamic driving tasks. In at least one embodiment, outputs from ADAS system **1838** may be provided to a supervisory MCU. In at least one embodiment, if outputs from a primary computer and outputs from a secondary computer conflict, a supervisory MCU determines how to reconcile conflict to ensure safe operation.
[0261] In at least one embodiment, a primary computer may be configured to provide a supervisory MCU with a confidence score, indicating that primary computer's confidence in a chosen result. In

at least one embodiment, if that confidence score exceeds a threshold, that supervisory MCU may follow that primary computer's direction, regardless of whether that secondary computer provides a conflicting or inconsistent result. In at least one embodiment, where a confidence score does not meet a threshold, and where primary and secondary computers indicate different results (e.g., a conflict), a supervisory MCU may arbitrate between computers to determine an appropriate outcome.

[0262] In at least one embodiment, a supervisory MCU may be configured to run a neural network(s) that is trained and configured to determine, based at least in part on outputs from a primary computer and outputs from a secondary computer, conditions under which that secondary computer provides false alarms. In at least one embodiment, neural network(s) in a supervisory MCU may learn when a secondary computer's output may be trusted, and when it cannot. For example, in at least one embodiment, when that secondary computer is a RADAR-based FCW system, a neural network(s) in that supervisory MCU may learn when an FCW system is identifying metallic objects that are not, in fact, hazards, such as a drainage grate or manhole cover that triggers an alarm. In at least one embodiment, when a secondary computer is a camera-based LDW system, a neural network in a supervisory MCU may learn to override LDW when bicyclists or pedestrians are present and a lane departure is, in fact, a safest maneuver. In at least one embodiment, a supervisory MCU may include at least one of a DLA or a GPU suitable for running neural network(s) with associated memory. In at least one embodiment, a supervisory MCU may comprise and/or be included as a component of SoC(s) **1804**.

[0263] In at least one embodiment, ADAS system **1838** may include a secondary computer that performs ADAS functionality using traditional rules of computer vision. In at least one embodiment, that secondary computer may use classic computer vision rules (if-then), and presence of a neural network(s) in a supervisory MCU may improve reliability, safety and performance. For example, in at least one embodiment, diverse implementation and intentional non-identity makes an overall system more fault-tolerant, especially to faults caused by software (or software-hardware interface) functionality. For example, in at least one embodiment, if there is a software bug or error in software running on a primary computer, and non-identical software code running on a secondary computer provides a consistent overall result, then a supervisory MCU may have greater confidence that an overall result is correct, and a bug in software or hardware on that primary computer is not causing a material error.

[0264] In at least one embodiment, an output of ADAS system **1838** may be fed into a primary computer's perception block and/or a primary computer's dynamic driving task block. For example, in at least one embodiment, if ADAS system **1838** indicates a forward crash warning due to an object immediately ahead, a perception block may use this information when identifying objects. In at least one embodiment, a secondary computer may have its own neural network that is trained and thus reduces a risk of false positives, as described herein.

[0265] In at least one embodiment, vehicle **1800** may further include infotainment SoC **1830** (e.g., an in-vehicle infotainment system (IVI)). Although illustrated and described as an SoC, infotainment system SoC **1830**, in at least one embodiment, may not be an SoC, and may include, without limitation, two or more discrete components. In at least one embodiment, infotainment SoC **1830** may include, without limitation, a combination of hardware and software that may be used to provide audio (e.g., music, a personal digital assistant, navigational instructions, news, radio, etc.), video (e.g., TV, movies, streaming, etc.), phone (e.g., hands-free calling), network connectivity (e.g., LTE, WiFi, etc.), and/or information services (e.g., navigation systems, rear-parking assistance, a radio data system, vehicle related information such as fuel level, total distance covered, brake fuel level, oil level, door open/close, air filter information, etc.) to vehicle **1800**. For example, infotainment SoC **1830** could include radios, disk players, navigation systems, video players, USB and Bluetooth connectivity, carputers, in-car entertainment, WiFi, steering wheel audio controls, hands free voice control, a heads-up display (“HUD”), HMI display **1834**, a

telematics device, a control panel (e.g., for controlling and/or interacting with various components, features, and/or systems), and/or other components. In at least one embodiment, infotainment SoC **1830** may further be used to provide information (e.g., visual and/or audible) to user(s) of vehicle **1800**, such as information from ADAS system **1838**, autonomous driving information such as planned vehicle maneuvers, trajectories, surrounding environment information (e.g., intersection information, vehicle information, road information, etc.), and/or other information.

[0266] In at least one embodiment, infotainment SoC **1830** may include any amount and type of GPU functionality. In at least one embodiment, infotainment SoC **1830** may communicate over bus **1802** with other devices, systems, and/or components of vehicle **1800**. In at least one embodiment, infotainment SoC **1830** may be coupled to a supervisory MCU such that a GPU of an infotainment system may perform some self-driving functions in event that primary controller(s) **1836** (e.g., primary and/or backup computers of vehicle **1800**) fail. In at least one embodiment, infotainment SoC **1830** may put vehicle **1800** into a chauffeur to safe stop mode, as described herein.

[0267] In at least one embodiment, vehicle **1800** may further include instrument cluster **1832** (e.g., a digital dash, an electronic instrument cluster, a digital instrument panel, etc.). In at least one embodiment, instrument cluster **1832** may include, without limitation, a controller and/or supercomputer (e.g., a discrete controller or supercomputer). In at least one embodiment, instrument cluster **1832** may include, without limitation, any number and combination of a set of instrumentation such as a speedometer, fuel level, oil pressure, tachometer, odometer, turn indicators, gearshift position indicator, seat belt warning light(s), parking-brake warning light(s), engine-malfunction light(s), supplemental restraint system (e.g., airbag) information, lighting controls, safety system controls, navigation information, etc. In some examples, information may be displayed and/or shared among infotainment SoC **1830** and instrument cluster **1832**. In at least one embodiment, instrument cluster **1832** may be included as part of infotainment SoC **1830**, or vice versa.

[0268] Inference and/or training logic **815** are used to perform inferencing and/or training operations associated with one or more embodiments. In at least one embodiment, inference and/or training logic **815** may be used in system FIG. **18**C for inferencing or predicting operations based, at least in part, on weight parameters calculated using neural network training operations, neural network functions and/or architectures, or neural network use cases described herein.

[0269] Such components can be used to generate a single, consistent tokenized description of at least a portion of a physical environment based in part on a set of observations and aligned map data.

[0270] FIG. **18**D is a diagram of a system **1877** for communication between cloud-based server(s) and autonomous vehicle **1800** of FIG. **18**A, according to at least one embodiment. In at least one embodiment, system may include, without limitation, server(s) **1878**, network(s) **1890**, and any number and type of vehicles, including vehicle **1800**. In at least one embodiment, server(s) **1878** may include, without limitation, a plurality of GPUs **1884**(A)-**1884**(H) (collectively referred to herein as GPUs **1884**), PCIe switches **1882**(A)-**1882**(D) (collectively referred to herein as PCIe switches **1882**), and/or CPUs **1880**(A)-**1880**(B) (collectively referred to herein as CPUs **1880**). In at least one embodiment, GPUs **1884**, CPUs **1880**, and PCIe switches **1882** may be interconnected with high-speed interconnects such as, for example and without limitation, NVLink interfaces **1888** developed by NVIDIA and/or PCIe connections **1886**. In at least one embodiment, GPUs **1884** are connected via an NVLink and/or NVSwitch SoC and GPUs **1884** and PCIe switches **1882** are connected via PCIe interconnects. Although eight GPUs **1884**, two CPUs **1880**, and four PCIe switches **1882** are illustrated, this is not intended to be limiting. In at least one embodiment, each of server(s) **1878** may include, without limitation, any number of GPUs **1884**, CPUs **1880**, and/or PCIe switches **1882**, in any combination. For example, in at least one embodiment, server(s) **1878** could each include eight, sixteen, thirty-two, and/or more GPUs **1884**.

[0271] In at least one embodiment, server(s) **1878** may receive, over network(s) **1890** and from

vehicles, image data representative of images showing unexpected or changed road conditions, such as recently commenced road-work. In at least one embodiment, server(s) **1878** may transmit, over network(s) **1890** and to vehicles, neural networks **1892**, updated or otherwise, and/or map information **1894**, including, without limitation, information regarding traffic and road conditions. In at least one embodiment, updates to map information **1894** may include, without limitation, updates for HD map **1822**, such as information regarding construction sites, potholes, detours, flooding, and/or other obstructions. In at least one embodiment, neural networks **1892**, and/or map information **1894** may have resulted from new training and/or experiences represented in data received from any number of vehicles in an environment, and/or based at least in part on training performed at a data center (e.g., using server(s) **1878** and/or other servers).

[0272] In at least one embodiment, server(s) **1878** may be used to train machine learning models (e.g., neural networks) based at least in part on training data. In at least one embodiment, training data may be generated by vehicles, and/or may be generated in a simulation (e.g., using a game engine). In at least one embodiment, any amount of training data is tagged (e.g., where associated neural network benefits from supervised learning) and/or undergoes other preprocessing. In at least one embodiment, any amount of training data is not tagged and/or pre-processed (e.g., where associated neural network does not require supervised learning). In at least one embodiment, once machine learning models are trained, machine learning models may be used by vehicles (e.g., transmitted to vehicles over network(s) **1890**), and/or machine learning models may be used by server(s) **1878** to remotely monitor vehicles.

[0273] In at least one embodiment, server(s) **1878** may receive data from vehicles and apply data to up-to-date real-time neural networks for real-time intelligent inferencing. In at least one embodiment, server(s) **1878** may include deep-learning supercomputers and/or dedicated AI computers powered by GPU(s) **1884**, such as a DGX and DGX Station machines developed by NVIDIA. However, in at least one embodiment, server(s) **1878** may include deep learning infrastructure that uses CPU-powered data centers.

[0274] In at least one embodiment, deep-learning infrastructure of server(s) **1878** may be capable of fast, real-time inferencing, and may use that capability to evaluate and verify health of processors, software, and/or associated hardware in vehicle **1800**. For example, in at least one embodiment, deep-learning infrastructure may receive periodic updates from vehicle **1800**, such as a sequence of images and/or objects that vehicle **1800** has located in that sequence of images (e.g., via computer vision and/or other machine learning object classification techniques). In at least one embodiment, deep-learning infrastructure may run its own neural network to identify objects and compare them with objects identified by vehicle **1800** and, if results do not match and deep-learning infrastructure concludes that AI in vehicle **1800** is malfunctioning, then server(s) **1878** may transmit a signal to vehicle **1800** instructing a fail-safe computer of vehicle **1800** to assume control, notify passengers, and complete a safe parking maneuver.

[0275] In at least one embodiment, server(s) **1878** may include GPU(s) **1884** and one or more programmable inference accelerators (e.g., NVIDIA's TensorRT 3 devices). In at least one embodiment, a combination of GPU-powered servers and inference acceleration may make real-time responsiveness possible. In at least one embodiment, such as where performance is less critical, servers powered by CPUs, FPGAs, and other processors may be used for inferencing.

[0276] Various embodiments can be described by the following clauses:

[0277] 1. A computer-implemented method, comprising: [0278] selecting a landmark associated with one or more first landmark pairs, within a set of first track data, and one or more second landmark pairs, associated with a set of second track data for a region; [0279] comparing at least a subset of the one or more first landmark pairs and the one or more second landmark pairs associated with the landmark; [0280] determining that a number of corresponding first landmark pairs and second landmark pairs exceeds a correspondence threshold; [0281] identifying the landmark as corresponding to both the first set of first track data and the second set of second track

data; and [0282] providing the identified landmark for use in one or more operations relating to an environment in which an object corresponding to the landmark is located.
[0283] 2. The computer-implemented method of clause 1, further comprising: [0284] representing a landmark pair segment as an edge between nodes in a landmark graph representing the landmark in the first set of track data and the landmark in the second set of track data; and [0285] determining a common frame of reference between the first set of track data and the second set of track data based, at least, on the edge.
[0286] 3. The computer-implemented method of clause 1, wherein the first set of track data is acquired by a first sensor-equipped machine and the second set of track data is acquired by a second sensor-equipped machine operating in the environment.
[0287] 4. The computer-implemented method of clause 1, wherein the first set of track data is at least partially overlapping the second set of track data, and wherein the first set of track data is able to be captured in a same or opposite direction of motion in the environment.
[0288] 5. The computer-implemented method of clause 1, further comprising: [0289] determining a first length for a selected first landmark pair; [0290] determining a first direction for the selected first landmark pair; [0291] determining a second length for a selected second landmark pair; [0292] determining a second direction for the selected second landmark pair; [0293] comparing the first length to the second length and the first direction to the second direction; and [0294] determining the first landmark pair corresponds to the second landmark pair based, at least, on the comparing.
[0295] 6. The computer-implemented method of clause 1, wherein the threshold is at least one of a percentage of total landmark pairs associated with the landmark or a specified value.
[0296] 7. The computer-implemented method of clause 1, wherein other landmarks are considered for inclusion in the one or more first landmark pairs or the one or more second landmark pairs if the other landmarks are within a determined distance range from the selected landmark, the determined distance range determined by a minimum distance and a maximum distance from the selected landmark.
[0297] 8. The computer-implemented method of clause 1, wherein the one or more first landmark pairs and the one or more second landmark pairs each include up to a maximum number of pairs, the maximum number of pairs determined based in part upon a desired level of performance or a maximum amount of latency.
[0298] 9. The computer-implemented method of clause 1, wherein the one or more operations include at least one of generating map data corresponding to the environment, updating map data corresponding to the environment, or automatically labeling one or more landmarks associated with the environment.
[0299] 10. A processor, comprising: [0300] one or more circuits to: [0301] generate a first landmark graph based, at least, on track data for a first sensor-equipped machine within a region; [0302] generate a second landmark graph based, at least, on track data for a second sensor-equipped machine within the region; [0303] generate a combined landmark graph including corresponding landmark pairs between the first landmark graph and the second landmark graph; [0304] determine, for each landmark in the combined landmark graph, a respective number of corresponding landmark edges between the corresponding landmark pairs; and [0305] determine one or more landmark matches between the first landmark graph and the second landmark graph responsive to a determination that, for each landmark in the combined landmark graph, that the respective number of landmark edges exceeds a threshold.
[0306] 11. The processor of clause 10, wherein the one or more circuits are further to: [0307] perform a transform operation to cause the one or more landmark matches to be aligned to a common frame of reference.
[0308] 12. The processor of clause 10, wherein other landmarks are considered for inclusion in landmark pairs if the other landmarks are within a determined distance range from a selected landmark, the determined distance range determined by a minimum distance and a maximum

distance from the selected landmark.
[0309] 13. The processor of clause 10, wherein the landmark pairs each include up to a maximum number of pairs, the maximum number of pairs determined based in part upon a desired level of performance or a maximum amount of latency.
[0310] 14. The processor of clause 10, wherein the track data for the first sensor-equipped machine within the region is at least partially overlapping the track data for the second sensor-equipped machine within the region and able to be captured in a same or opposite direction of motion in the region.
[0311] 15. The processor of clause 10, wherein the processor is comprised in at least one of: [0312] a system for performing simulation operations; [0313] a system for performing simulation operations to test or validate autonomous machine applications; [0314] a system for performing digital twin operations; [0315] a system for performing light transport simulation; [0316] a system for rendering graphical output; [0317] a system for performing deep learning operations; [0318] a system for performing generative AI operations using a large language model (LLM); [0319] a system implemented using an edge device; [0320] a system for generating or presenting virtual reality (VR) content; [0321] a system for generating or presenting augmented reality (AR) content; [0322] a system for generating or presenting mixed reality (MR) content; [0323] a system incorporating one or more Virtual Machines (VMs); [0324] a system implemented at least partially in a data center; [0325] a system for performing hardware testing using simulation; [0326] a system for performing generative operations using a language model (LM); [0327] a system for synthetic data generation; [0328] a collaborative content creation platform for 3D assets; or [0329] a system implemented at least partially using cloud computing resources.
[0330] 16. A system comprising: [0331] one or more processors to determine and align corresponding landmarks between a plurality of landmark graphs based, at least, on a threshold number of correlated landmark pair segments, extending from a selected landmark within the plurality of landmark graphs, exceeding a correspondence threshold.
[0332] 17. The system of clause 16, wherein the plurality of landmark graphs correspond to a plurality of tracks of sensor data captured in a region of an environment.
[0333] 18. The system of clause 16, wherein the corresponding landmarks are transformed to align to a common frame of reference.
[0334] 19. The system of clause 16, wherein other landmarks are considered for inclusion in landmark pairs if the other landmarks are within a determined distance range from a selected landmark, the determined distance range determined by a minimum distance and a maximum distance from the selected landmark.
[0335] 20. The system of clause 16, wherein the system comprises at least one of: [0336] a system for performing simulation operations; [0337] a system for performing simulation operations to test or validate autonomous machine applications; [0338] a system for performing digital twin operations; [0339] a system for performing light transport simulation; [0340] a system for rendering graphical output; [0341] a system for performing deep learning operations; [0342] a system for performing generative AI operations using a large language model (LLM); [0343] a system implemented using an edge device; [0344] a system for generating or presenting virtual reality (VR) content; [0345] a system for generating or presenting augmented reality (AR) content; [0346] a system for generating or presenting mixed reality (MR) content; [0347] a system incorporating one or more Virtual Machines (VMs); [0348] a system implemented at least partially in a data center; [0349] a system for performing hardware testing using simulation; [0350] a system for performing generative operations using a language model (LM); [0351] a system for synthetic data generation; [0352] a collaborative content creation platform for 3D assets; or [0353] a system implemented at least partially using cloud computing resources.
[0354] In at least one embodiment, a single semiconductor platform may refer to a sole unitary semiconductor-based integrated circuit or chip. In at least one embodiment, multi-chip modules

may be used with increased connectivity which simulate on-chip operation, and make substantial improvements over utilizing a conventional central processing unit ("CPU") and bus implementation. In at least one embodiment, various modules may also be situated separately or in various combinations of semiconductor platforms per desires of user.

[0355] In at least one embodiment, referring back to FIG. **11**, computer programs in form of machine-readable executable code or computer control logic algorithms are stored in main memory **1104** and/or secondary storage. Computer programs, if executed by one or more processors, enable computer system **1100** to perform various functions in accordance with at least one embodiment. In at least one embodiment, main memory **1104**, storage, and/or any other storage are possible examples of computer-readable media. In at least one embodiment, secondary storage may refer to any suitable storage device or system such as a hard disk drive and/or a removable storage drive, representing a floppy disk drive, a magnetic tape drive, a compact disk drive, digital versatile disk ("DVD") drive, recording device, universal serial bus ("USB") flash memory, etc. In at least one embodiment, architecture and/or functionality of various previous FIGS. **1-7** are implemented in context of CPU **1102**, parallel processing system **1112**, an integrated circuit capable of at least a portion of capabilities of both CPU **1102**, parallel processing system **1112**, a chipset (e.g., a group of integrated circuits designed to work and sold as a unit for performing related functions, etc.), and/or any suitable combination of integrated circuit(s).

[0356] In at least one embodiment, architecture and/or functionality of various previous FIGS. **1-7** are implemented in context of a general computer system, a circuit board system, a game console system dedicated for entertainment purposes, an application-specific system, and more. In at least one embodiment, computer system **1100** may take form of a desktop computer, a laptop computer, a tablet computer, servers, supercomputers, a smart-phone (e.g., a wireless, hand-held device), personal digital assistant ("PDA"), a digital camera, a vehicle, a head mounted display, a hand-held electronic device, a mobile phone device, a television, workstation, game consoles, embedded system, and/or any other type of logic.

[0357] In at least one embodiment, parallel processing system **1112** includes, without limitation, a plurality of parallel processing units ("PPUs") **1114** and associated memories **1116**. In at least one embodiment, PPUs **1114** are connected to a host processor or other peripheral devices via an interconnect **1118** and a switch **1120** or multiplexer. In at least one embodiment, parallel processing system **1112** distributes computational tasks across PPUs **1114** which can be parallelizable—for example, as part of distribution of computational tasks across multiple graphics processing unit ("GPU") thread blocks. In at least one embodiment, memory is shared and accessible (e.g., for read and/or write access) across some or all of PPUs **1114**, although such shared memory may incur performance penalties relative to use of local memory and registers resident to a PPU **1114**. In at least one embodiment, operation of PPUs **1114** is synchronized through use of a command such as _syncthreads( ) wherein all threads in a block (e.g., executed across multiple PPUs **1114**) to reach a certain point of execution of code before proceeding.

[0358] In at least one embodiment, one or more techniques described herein utilize a oneAPI programming model. In at least one embodiment, a oneAPI programming model refers to a programming model for interacting with various compute accelerator architectures. In at least one embodiment, oneAPI refers to an application programming interface (API) designed to interact with various compute accelerator architectures. In at least one embodiment, a oneAPI programming model utilizes a DPC++ programming language. In at least one embodiment, a DPC++ programming language refers to a high-level language for data parallel programming productivity. In at least one embodiment, a DPC++ programming language is based at least in part on C and/or C++ programming languages. In at least one embodiment, a oneAPI programming model is a programming model such as those developed by Intel Corporation of Santa Clara, CA.

[0359] In at least one embodiment, oneAPI and/or oneAPI programming model is utilized to interact with various accelerator, GPU, processor, and/or variations thereof, architectures. In at least

one embodiment, oneAPI includes a set of libraries that implement various functionalities. In at least one embodiment, oneAPI includes at least a oneAPI DPC++ library, a oneAPI math kernel library, a oneAPI data analytics library, a oneAPI deep neural network library, a oneAPI collective communications library, a oneAPI threading building blocks library, a oneAPI video processing library, and/or variations thereof.

[0360] In at least one embodiment, a oneAPI DPC++ library, also referred to as oneDPL, is a library that implements algorithms and functions to accelerate DPC++ kernel programming. In at least one embodiment, oneDPL implements one or more standard template library (STL) functions. In at least one embodiment, oneDPL implements one or more parallel STL functions. In at least one embodiment, oneDPL provides a set of library classes and functions such as parallel algorithms, iterators, function object classes, range-based API, and/or variations thereof. In at least one embodiment, oneDPL implements one or more classes and/or functions of a C++standard library. In at least one embodiment, oneDPL implements one or more random number generator functions.

[0361] In at least one embodiment, a oneAPI math kernel library, also referred to as oneMKL, is a library that implements various optimized and parallelized routines for various mathematical functions and/or operations. In at least one embodiment, oneMKL implements one or more basic linear algebra subprograms (BLAS) and/or linear algebra package (LAPACK) dense linear algebra routines. In at least one embodiment, oneMKL implements one or more sparse BLAS linear algebra routines. In at least one embodiment, oneMKL implements one or more random number generators (RNGs). In at least one embodiment, oneMKL implements one or more vector mathematics (VM) routines for mathematical operations on vectors. In at least one embodiment, oneMKL implements one or more Fast Fourier Transform (FFT) functions.

[0362] In at least one embodiment, a oneAPI data analytics library, also referred to as oneDAL, is a library that implements various data analysis applications and distributed computations. In at least one embodiment, oneDAL implements various algorithms for preprocessing, transformation, analysis, modeling, validation, and decision making for data analytics, in batch, online, and distributed processing modes of computation. In at least one embodiment, oneDAL implements various C++ and/or Java APIs and various connectors to one or more data sources. In at least one embodiment, oneDAL implements DPC++ API extensions to a traditional C++ interface and enables GPU usage for various algorithms.

[0363] In at least one embodiment, a oneAPI deep neural network library, also referred to as oneDNN, is a library that implements various deep learning functions. In at least one embodiment, oneDNN implements various neural network, machine learning, and deep learning functions, algorithms, and/or variations thereof.

[0364] In at least one embodiment, a oneAPI collective communications library, also referred to as oneCCL, is a library that implements various applications for deep learning and machine learning workloads. In at least one embodiment, oneCCL is built upon lower-level communication middleware, such as message passing interface (MPI) and libfabrics. In at least one embodiment, oneCCL enables a set of deep learning specific optimizations, such as prioritization, persistent operations, out of order executions, and/or variations thereof. In at least one embodiment, oneCCL implements various CPU and GPU functions.

[0365] In at least one embodiment, a oneAPI threading building blocks library, also referred to as oneTBB, is a library that implements various parallelized processes for various applications. In at least one embodiment, oneTBB is utilized for task-based, shared parallel programming on a host. In at least one embodiment, oneTBB implements generic parallel algorithms. In at least one embodiment, oneTBB implements concurrent containers. In at least one embodiment, oneTBB implements a scalable memory allocator. In at least one embodiment, oneTBB implements a work-stealing task scheduler. In at least one embodiment, oneTBB implements low-level synchronization primitives. In at least one embodiment, oneTBB is compiler-independent and usable on various processors, such as GPUs, PPUs, CPUs, and/or variations thereof.

[0366] In at least one embodiment, a oneAPI video processing library, also referred to as oneVPL, is a library that is utilized for accelerating video processing in one or more applications. In at least one embodiment, oneVPL implements various video decoding, encoding, and processing functions. In at least one embodiment, oneVPL implements various functions for media pipelines on CPUs, GPUs, and other accelerators. In at least one embodiment, oneVPL implements device discovery and selection in media centric and video analytics workloads. In at least one embodiment, oneVPL implements API primitives for zero-copy buffer sharing.
[0367] In at least one embodiment, a oneAPI programming model utilizes a DPC++ programming language. In at least one embodiment, a DPC++ programming language is a programming language that includes, without limitation, functionally similar versions of CUDA mechanisms to define device code and distinguish between device code and host code. In at least one embodiment, a DPC++ programming language may include a subset of functionality of a CUDA programming language. In at least one embodiment, one or more CUDA programming model operations are performed using a oneAPI programming model using a DPC++ programming language.
[0368] In at least one embodiment, any application programming interface (API) described herein is compiled into one or more instructions, operations, or any other signal by a compiler, interpreter, or other software tool. In at least one embodiment, compilation comprises generating one or more machine-executable instructions, operations, or other signals from source code. In at least one embodiment, an API compiled into one or more instructions, operations, or other signals, when performed, causes one or more processors such as graphics processor **1410**, graphics processor **1440**, graphics core **1500**, parallel processor **1700**, graphics processor **1900**, or any other logic circuit further described herein to perform one or more computing operations.
[0369] It should be noted that, while example embodiments described herein may relate to a CUDA programming model, techniques described herein can be utilized with any suitable programming model, such HIP, oneAPI, and/or variations thereof.
[0370] Other variations are within spirit of present disclosure. Thus, while disclosed techniques are susceptible to various modifications and alternative constructions, certain illustrated embodiments thereof are shown in drawings and have been described above in detail. It should be understood, however, that there is no intention to limit disclosure to specific form or forms disclosed, but on contrary, intention is to cover all modifications, alternative constructions, and equivalents falling within spirit and scope of disclosure, as defined in appended claims.
[0371] Use of terms “a” and “an” and “the” and similar referents in context of describing disclosed embodiments (especially in context of following claims) are to be construed to cover both singular and plural, unless otherwise indicated herein or clearly contradicted by context, and not as a definition of a term. Terms “comprising,” “having,” “including,” and “containing” are to be construed as open-ended terms (meaning “including, but not limited to,”) unless otherwise noted. “Connected,” when unmodified and referring to physical connections, is to be construed as partly or wholly contained within, attached to, or joined together, even if there is something intervening. Recitation of ranges of values herein are merely intended to serve as a shorthand method of referring individually to each separate value falling within range, unless otherwise indicated herein and each separate value is incorporated into specification as if it were individually recited herein. In at least one embodiment, use of term “set” (e.g., “a set of items”) or “subset” unless otherwise noted or contradicted by context, is to be construed as a nonempty collection comprising one or more members. Further, unless otherwise noted or contradicted by context, term “subset” of a corresponding set does not necessarily denote a proper subset of corresponding set, but subset and corresponding set may be equal.
[0372] Conjunctive language, such as phrases of form “at least one of A, B, and C,” or “at least one of A, B and C,” unless specifically stated otherwise or otherwise clearly contradicted by context, is otherwise understood with context as used in general to present that an item, term, etc., may be either A or B or C, or any nonempty subset of set of A and B and C. For instance, in illustrative

example of a set having three members, conjunctive phrases “at least one of A, B, and C” and “at least one of A, B and C” refer to any of following sets: {A}, {B}, {C}, {A, B}, {A, C}, {B, C}, {A, B, C}. Thus, such conjunctive language is not generally intended to imply that certain embodiments require at least one of A, at least one of B and at least one of C each to be present. In addition, unless otherwise noted or contradicted by context, term “plurality” indicates a state of being plural (e.g., “a plurality of items” indicates multiple items). In at least one embodiment, number of items in a plurality is at least two, but can be more when so indicated either explicitly or by context. Further, unless stated otherwise or otherwise clear from context, phrase “based on” means “based at least in part on” and not “based solely on.”

[0373] Operations of processes described herein can be performed in any suitable order unless otherwise indicated herein or otherwise clearly contradicted by context. In at least one embodiment, a process such as those processes described herein (or variations and/or combinations thereof) is performed under control of one or more computer systems configured with executable instructions and is implemented as code (e.g., executable instructions, one or more computer programs or one or more applications) executing collectively on one or more processors, by hardware or combinations thereof. In at least one embodiment, code is stored on a computer-readable storage medium, for example, in form of a computer program comprising a plurality of instructions executable by one or more processors. In at least one embodiment, a computer-readable storage medium is a non-transitory computer-readable storage medium that excludes transitory signals (e.g., a propagating transient electric or electromagnetic transmission) but includes non-transitory data storage circuitry (e.g., buffers, cache, and queues) within transceivers of transitory signals. In at least one embodiment, code (e.g., executable code or source code) is stored on a set of one or more non-transitory computer-readable storage media having stored thereon executable instructions (or other memory to store executable instructions) that, when executed (i.e., as a result of being executed) by one or more processors of a computer system, cause computer system to perform operations described herein. In at least one embodiment, set of non-transitory computer-readable storage media comprises multiple non-transitory computer-readable storage media and one or more of individual non-transitory storage media of multiple non-transitory computer-readable storage media lack all of code while multiple non-transitory computer-readable storage media collectively store all of code. In at least one embodiment, executable instructions are executed such that different instructions are executed by different processors—for example, a non-transitory computer-readable storage medium store instructions and a main central processing unit (“CPU”) executes some of instructions while a graphics processing unit (“GPU”) executes other instructions. In at least one embodiment, different components of a computer system have separate processors and different processors execute different subsets of instructions.

[0374] In at least one embodiment, an arithmetic logic unit is a set of combinational logic circuitry that takes one or more inputs to produce a result. In at least one embodiment, an arithmetic logic unit is used by a processor to implement mathematical operation such as addition, subtraction, or multiplication. In at least one embodiment, an arithmetic logic unit is used to implement logical operations such as logical AND/OR or XOR. In at least one embodiment, an arithmetic logic unit is stateless, and made from physical switching components such as semiconductor transistors arranged to form logical gates. In at least one embodiment, an arithmetic logic unit may operate internally as a stateful logic circuit with an associated clock. In at least one embodiment, an arithmetic logic unit may be constructed as an asynchronous logic circuit with an internal state not maintained in an associated register set. In at least one embodiment, an arithmetic logic unit is used by a processor to combine operands stored in one or more registers of the processor and produce an output that can be stored by the processor in another register or a memory location.

[0375] In at least one embodiment, as a result of processing an instruction retrieved by the processor, the processor presents one or more inputs or operands to an arithmetic logic unit,

causing the arithmetic logic unit to produce a result based at least in part on an instruction code provided to inputs of the arithmetic logic unit. In at least one embodiment, the instruction codes provided by the processor to the ΔLU are based at least in part on the instruction executed by the processor. In at least one embodiment combinational logic in the ΔLU processes the inputs and produces an output which is placed on a bus within the processor. In at least one embodiment, the processor selects a destination register, memory location, output device, or output storage location on the output bus so that clocking the processor causes the results produced by the ΔLU to be sent to the desired location.

[0376] In the scope of this application, the term arithmetic logic unit, or ΔLU, is used to refer to any computational logic circuit that processes operands to produce a result. For example, in the present document, the term ΔLU can refer to a floating point unit, a DSP, a tensor core, a shader core, a coprocessor, or a CPU.

[0377] Accordingly, in at least one embodiment, computer systems are configured to implement one or more services that singly or collectively perform operations of processes described herein and such computer systems are configured with applicable hardware and/or software that enable performance of operations. Further, a computer system that implements at least one embodiment of present disclosure is a single device and, in another embodiment, is a distributed computer system comprising multiple devices that operate differently such that distributed computer system performs operations described herein and such that a single device does not perform all operations.

[0378] Use of any and all examples, or exemplary language (e.g., “such as”) provided herein, is intended merely to better illuminate embodiments of disclosure and does not pose a limitation on scope of disclosure unless otherwise claimed. No language in specification should be construed as indicating any non-claimed element as essential to practice of disclosure.

[0379] All references, including publications, patent applications, and patents, cited herein are hereby incorporated by reference to same extent as if each reference were individually and specifically indicated to be incorporated by reference and were set forth in its entirety herein.

[0380] In description and claims, terms “coupled” and “connected,” along with their derivatives, may be used. It should be understood that these terms may be not intended as synonyms for each other. Rather, in particular examples, “connected” or “coupled” may be used to indicate that two or more elements are in direct or indirect physical or electrical contact with each other. “Coupled” may also mean that two or more elements are not in direct contact with each other, but yet still co-operate or interact with each other.

[0381] Unless specifically stated otherwise, it may be appreciated that throughout specification terms such as “processing,” “computing,” “calculating,” “determining,” or like, refer to action and/or processes of a computer or computing system, or similar electronic computing device, that manipulate and/or transform data represented as physical, such as electronic, quantities within computing system's registers and/or memories into other data similarly represented as physical quantities within computing system's memories, registers or other such information storage, transmission or display devices.

[0382] In a similar manner, term “processor” may refer to any device or portion of a device that processes electronic data from registers and/or memory and transform that electronic data into other electronic data that may be stored in registers and/or memory. As non-limiting examples, “processor” may be a CPU or a GPU. A “computing platform” may comprise one or more processors. As used herein, “software” processes may include, for example, software and/or hardware entities that perform work over time, such as tasks, threads, and intelligent agents. Also, each process may refer to multiple processes, for carrying out instructions in sequence or in parallel, continuously or intermittently. In at least one embodiment, terms “system” and “method” are used herein interchangeably insofar as system may embody one or more methods and methods may be considered a system.

[0383] In present document, references may be made to obtaining, acquiring, receiving, or

inputting analog or digital data into a subsystem, computer system, or computer-implemented machine. In at least one embodiment, process of obtaining, acquiring, receiving, or inputting analog and digital data can be accomplished in a variety of ways such as by receiving data as a parameter of a function call or a call to an application programming interface. In at least one embodiment, processes of obtaining, acquiring, receiving, or inputting analog or digital data can be accomplished by transferring data via a serial or parallel interface. In at least one embodiment, processes of obtaining, acquiring, receiving, or inputting analog or digital data can be accomplished by transferring data via a computer network from providing entity to acquiring entity. In at least one embodiment, references may also be made to providing, outputting, transmitting, sending, or presenting analog or digital data. In various examples, processes of providing, outputting, transmitting, sending, or presenting analog or digital data can be accomplished by transferring data as an input or output parameter of a function call, a parameter of an application programming interface or interprocess communication mechanism.

[0384] Although descriptions herein set forth example implementations of described techniques, other architectures may be used to implement described functionality, and are intended to be within scope of this disclosure. Furthermore, although specific distributions of responsibilities may be defined above for purposes of description, various functions and responsibilities might be distributed and divided in different ways, depending on circumstances.

[0385] Furthermore, although subject matter has been described in language specific to structural features and/or methodological acts, it is to be understood that subject matter claimed in appended claims is not necessarily limited to specific features or acts described. Rather, specific features and acts are disclosed as exemplary forms of implementing the claims.

Claims

1. A computer-implemented method, comprising: selecting a landmark associated with one or more first landmark pairs, within a set of first track data, and one or more second landmark pairs, associated with a set of second track data for a region; comparing at least a subset of the one or more first landmark pairs and the one or more second landmark pairs associated with the landmark; determining that a number of corresponding first landmark pairs and second landmark pairs exceeds a correspondence threshold; identifying the landmark as corresponding to both the first set of first track data and the second set of second track data; and providing the identified landmark for use in one or more operations relating to an environment in which an object corresponding to the landmark is located.

2. The computer-implemented method of claim 1, further comprising: representing a landmark pair segment as an edge between nodes in a landmark graph representing the landmark in the first set of track data and the landmark in the second set of track data; and determining a common frame of reference between the first set of track data and the second set of track data based, at least, on the edge.

3. The computer-implemented method of claim 1, wherein the first set of track data is acquired by a first sensor-equipped machine and the second set of track data is acquired by a second sensor-equipped machine operating in the environment.

4. The computer-implemented method of claim 1, wherein the first set of track data is at least partially overlapping the second set of track data, and wherein the first set of track data is able to be captured in a same or opposite direction of motion in the environment.

5. The computer-implemented method of claim 1, further comprising: determining a first length for a selected first landmark pair; determining a first direction for the selected first landmark pair; determining a second length for a selected second landmark pair; determining a second direction for the selected second landmark pair; comparing the first length to the second length and the first direction to the second direction; and determining the first landmark pair corresponds to the second

landmark pair based, at least, on the comparing.
6. The computer-implemented method of claim 1, wherein the threshold is at least one of a percentage of total landmark pairs associated with the landmark or a specified value.
7. The computer-implemented method of claim 1, wherein other landmarks are considered for inclusion in the one or more first landmark pairs or the one or more second landmark pairs if the other landmarks are within a determined distance range from the selected landmark, the determined distance range determined by a minimum distance and a maximum distance from the selected landmark.
8. The computer-implemented method of claim 1, wherein the one or more first landmark pairs and the one or more second landmark pairs each include up to a maximum number of pairs, the maximum number of pairs determined based in part upon a desired level of performance or a maximum amount of latency.
9. The computer-implemented method of claim 1, wherein the one or more operations include at least one of generating map data corresponding to the environment, updating map data corresponding to the environment, or automatically labeling one or more landmarks associated with the environment.
10. A processor, comprising: one or more circuits to: generate a first landmark graph based, at least, on track data for a first sensor-equipped machine within a region; generate a second landmark graph based, at least, on track data for a second sensor-equipped machine within the region; generate a combined landmark graph including corresponding landmark pairs between the first landmark graph and the second landmark graph; determine, for each landmark in the combined landmark graph, a respective number of corresponding landmark edges between the corresponding landmark pairs; and determine one or more landmark matches between the first landmark graph and the second landmark graph responsive to a determination that, for each landmark in the combined landmark graph, that the respective number of landmark edges exceeds a threshold.
11. The processor of claim 10, wherein the one or more circuits are further to: perform a transform operation to cause the one or more landmark matches to be aligned to a common frame of reference.
12. The processor of claim 10, wherein other landmarks are considered for inclusion in landmark pairs if the other landmarks are within a determined distance range from a selected landmark, the determined distance range determined by a minimum distance and a maximum distance from the selected landmark.
13. The processor of claim 10, wherein the landmark pairs each include up to a maximum number of pairs, the maximum number of pairs determined based in part upon a desired level of performance or a maximum amount of latency.
14. The processor of claim 10, wherein the track data for the first sensor-equipped machine within the region is at least partially overlapping the track data for the second sensor-equipped machine within the region and able to be captured in a same or opposite direction of motion in the region.
15. The processor of claim 10, wherein the processor is comprised in at least one of: a system for performing simulation operations; a system for performing simulation operations to test or validate autonomous machine applications; a system for performing digital twin operations; a system for performing light transport simulation; a system for rendering graphical output; a system for performing deep learning operations; a system for performing generative AI operations using a large language model (LLM); a system implemented using an edge device; a system for generating or presenting virtual reality (VR) content; a system for generating or presenting augmented reality (AR) content; a system for generating or presenting mixed reality (MR) content; a system incorporating one or more Virtual Machines (VMs); a system implemented at least partially in a data center; a system for performing hardware testing using simulation; a system for performing generative operations using a language model (LM); a system for synthetic data generation; a collaborative content creation platform for 3D assets; or a system implemented at least partially

using cloud computing resources.

16. A system comprising: one or more processors to determine and align corresponding landmarks between a plurality of landmark graphs based, at least, on a threshold number of correlated landmark pair segments, extending from a selected landmark within the plurality of landmark graphs, exceeding a correspondence threshold.

17. The system of claim 16, wherein the plurality of landmark graphs correspond to a plurality of tracks of sensor data captured in a region of an environment.

18. The system of claim 16, wherein the corresponding landmarks are transformed to align to a common frame of reference.

19. The system of claim 16, wherein other landmarks are considered for inclusion in landmark pairs if the other landmarks are within a determined distance range from a selected landmark, the determined distance range determined by a minimum distance and a maximum distance from the selected landmark.

20. The system of claim 16, wherein the system comprises at least one of: a system for performing simulation operations; a system for performing simulation operations to test or validate autonomous machine applications; a system for performing digital twin operations; a system for performing light transport simulation; a system for rendering graphical output; a system for performing deep learning operations; a system for performing generative AI operations using a large language model (LLM); a system implemented using an edge device; a system for generating or presenting virtual reality (VR) content; a system for generating or presenting augmented reality (AR) content; a system for generating or presenting mixed reality (MR) content; a system incorporating one or more Virtual Machines (VMs); a system implemented at least partially in a data center; a system for performing hardware testing using simulation; a system for performing generative operations using a language model (LM); a system for synthetic data generation; a collaborative content creation platform for 3D assets; or a system implemented at least partially using cloud computing resources.