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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
26 December 2024 (26.12.2024)

WIPO | PCT

(10) International Publication Number
WO 2024/259614 A1

(51) International Patent Classification:
G06T 17/00 (2006.01) *G06T 9/00* (2006.01)

(21) International Application Number:
PCT/CN2023/101585

(22) International Filing Date:
21 June 2023 (21.06.2023)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HUAWEI TECHNOLOGIES CO., LTD.** [CN/CN]; Huawei Administration Building, Bantian, Longgang District, Shenzhen, Guangdong 518129 (CN).

(72) Inventors: **REN, Yuan**; Suite 400, 303 Terry Fox Drive, Kanata, Ontario 231 (CA). **LIU, Yibo**; Suite 400, 303 Terry Fox Drive, Kanata, Ontario 231 (CA). **ZHU, Kelly**; Suite 400, 303 Terry Fox Drive, Kanata, Ontario 231 (CA). **LIU, Bingbing**; Huawei Administration Building, Bantian, Longgang District, Shenzhen, Guangdong 518129 (CN).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, MG, MK, MN, MU, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, CV,

(54) Title: METHODS AND PROCESSORS FOR IMPLICIT MODELING WITH MULTI-SWEEP POINT CLOUDS FOR 3D VEHICLE RECONSTRUCTION IN AUTONOMOUS DRIVING

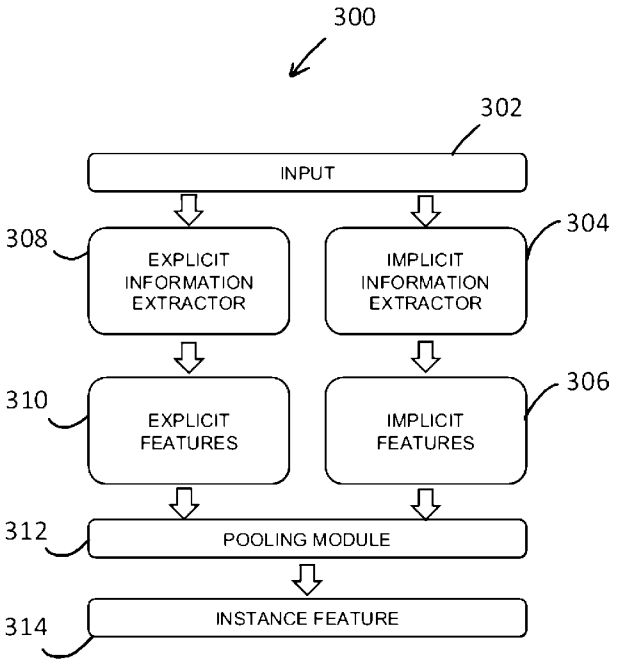

FIG. 3

(57) Abstract: Methods and electronic devices for performing 3D reconstruction of an object are disclosed. The method includes acquiring a first point cloud and a second point cloud, the first and second point clouds being multi-sweep point clouds representing at least the object, extracting, using a first network, explicit features about a shape of the object based on the first and second point clouds, the first network having been trained to explicitly model the shape of the object, extracting, using a second network, implicit features about the shape of the object based on the first and second point clouds, the second network having been trained to implicitly model the shape of the object, generating, using a third network, a latent code by projecting the implicit features and the explicit features into a latent space, and generating a 3D shape representative of the object using the latent code.

WO 2024/259614 A1

[Continued on next page]

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

Published:

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

METHODS AND PROCESSORS FOR IMPLICIT MODELING WITH MULTI-SWEEP POINT CLOUDS FOR 3D VEHICLE RECONSTRUCTION IN AUTONOMOUS DRIVING

CROSS-REFERENCE TO RELATED APPLICATIONS

[01] This is the first application related to the present technology.

FIELD

[02] The present technology relates generally to 3D object reconstruction; and in particular, to electronic devices and methods for executing 3D reconstruction of objects.

BACKGROUND

[03] Autonomous vehicles, also known as self-driving cars, have gained attention in recent years due to their potential to revolutionize transportation and enhance road safety. These vehicles rely on a combination of sensors, algorithms, and artificial intelligence to navigate through complex environments without human intervention.

[04] LiDAR technology has emerged as an important component for autonomous vehicles, offering a robust and accurate perception system. LiDAR systems employ laser beams to measure distances and create detailed three-dimensional representations of the surrounding environment. For example, by emitting laser pulses and measuring the time it takes for the pulses to bounce back after hitting objects, LiDAR sensors can precisely calculate the distance and location of obstacles, pedestrians, vehicles, and other elements in the vehicle's vicinity.

[05] LIDAR sensors provide a rich and high-resolution point cloud representation of the environment, enabling precise detection and localization of objects. This detailed perception allows autonomous vehicles to make informed decisions in real-time, enhancing their ability to navigate complex scenarios and avoid potential hazards. Also, LiDAR sensors are capable of operating in various environmental conditions, including low-light situations, rain, fog, and snow. This versatility ensures that autonomous vehicles equipped with LiDAR can operate reliably and safely in different weather conditions, enhancing their overall robustness and dependability. Moreover, LiDAR sensors can complement other sensor modalities used in autonomous vehicles, such as cameras and radar. While cameras provide visual information and radar can detect objects based on their reflective properties, LiDAR sensors offer precise depth information that is independent of lighting conditions, making it valuable for object detection and tracking.

[06] However, there are certain challenges and limitations associated with the use of LiDAR sensors in autonomous vehicles. LiDAR sensors may experience difficulties in accurately detecting certain objects with low reflectivity, such as non-metallic or transparent materials. Also, 3D vehicle reconstruction from sparse and partial point clouds may be required for autonomous driving, where sensor simulators require 3D foreground object models as their input.

[07] Therefore, there is a need for improved LiDAR-based autonomous vehicle systems that address the limitations of existing approaches, while providing reliable and cost-effective solutions for safe navigation and obstacle detection.

SUMMARY

[08] Developers have devised methods and devices for overcoming at least some drawbacks present in prior art solutions.

[09] In the autonomous driving industry, a sensor simulator usually requires 3D foreground object models as input. In order to guarantee the diversity of foreground objects, a large number of models may be needed. Developers have realized that generating an object library is an expensive task and using 3D reconstruction to that end can greatly save costs. Using partial LiDAR sweeps of vehicles to reconstruct an object library can reduce domain differences and therefore maintain the original distributions of the object size, type and pose, for example.

[10] Conventional 3D reconstruction approaches can be classified as either explicit modeling-based or implicit modeling-based.

Explicit modeling

[11] Explicit modeling-based approaches can be further classified into point-based, voxel-based, and mesh-based and directly represent 3D object shape structure with points, voxels, and meshes respectively. Point-based 3D reconstruction methods, such as Point Completion Network (PCN), GRNet, and PoinTr, output a point cloud as a shape representation of the 3D reconstruction. Voxel-based modeling described volumes by subdividing the space into a 3D grip and mesh-based approaches generate meshes using a fixed reference template from a given object class. Conventional explicit reconstruction approaches, including point-based, voxel-based, and mesh-based, suffer from problems such as low resolution and high memory costs.

[12] In some embodiments, "explicit" features may refer to features generated using one or more explicit modeling techniques. It can be said that explicit features represent features of a first type and which are extracted from an explicitly modeled object. In some embodiments, intermediate features of PCN models, GRNet models, and PoinTr models may be used as features of the first type. In other embodiments, features of the first type can be generated using (i) a voxel-based technique for shape completion using 3D-Encoder-Predictor CNNs and shape synthesis, and (ii) a mesh-based technique for multi-chart generative surface modeling.

[13] Broadly, the PoinTr model, short for "Point Transformer," is a deep learning model designed for image recognition and object detection tasks. The PoinTr model uses convolutional neural networks (CNNs) and transformers to effectively capture spatial relationships and contextual information within images. It uses self-attention mechanisms to model long-range dependencies between image regions, allowing it to attend to relevant image features and objects. Unlike traditional CNN-based models that rely solely on convolutional layers, the PoinTr model replaces the convolutional layers with transformer layers, enabling it to perform both local and global image analysis. By considering the entire image as a sequence of patches, the model can learn patterns and relationships between different parts of the image.

Implicit modeling

[14] Contrary to explicit modeling methods, implicit modeling uses function-based decision boundaries to implicitly define surfaces for 3D representations. For example, DeepSDF approach disclosed in an article entitled "*DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation*", authored by Jeong Joon Park et al., and published in 2019 (the content of which is incorporated herein by reference in its entirety), is an implicit approach that maps the 3D shape to a low-dimensional latent space and learns the projection from the latent space to a continuous signed distance function (SDF) that describes that 3D shape. Broadly speaking, SDF is an implicit function that specifies whether a querying point is inside, or outside, the surface of an object. The value of an SDF represents the distance of a querying point

from the surface of the object and its sign, either positive or negative, indicates whether the querying point lies on the exterior or interior of the surface, respectively.

[15] In some embodiments, "implicit" features may refer to features generated using one or more implicit modeling techniques. It can be said that implicit features represent features of a second type and which are extracted from an implicitly modeled object. In some embodiments, features of the second type may be extracted using occupancy networks by learning 3D reconstruction in function space, NeuS models by learning neural implicit surfaces by volume rendering for multi-view reconstruction, NeAT models by learning neural implicit surfaces with arbitrary topologies from multi-view Images, and the like.

[16] Broadly, Neuro-Evolution of Augmenting Topologies (NeAT) models combine evolutionary algorithms with neural network training. They evolve both the structure and weights of neural networks through mutation and crossover operations. By evaluating fitness and selecting networks with higher scores, NeAT models can evolve architectures well-suited to specific tasks. NeuS can be implemented as a combination of a DeepSDF model and a NeRF model. It can be said that a DeepSDF model may be a MLP portion which is used to represent the 3D shape of the object, and NeRF model may be an other MLP portion which represents the texture of the object.

[17] Thus, the 3D shape can be stored as a one-dimensional memory-saving latent code and the continuous SDF supports 3D mesh extraction at different resolutions. For example, the input of a DeepSDF network is a latent code and a query point. The output is a signed distance of this query point. The parameters in the DeepSDF network are fixed after training, so the 3D shape is represented by the latent code.

[18] Developers of the present technology have realized that conventional 3D explicit modeling approaches suffer from many problems due to explicit representation of objects. For example, point-based approaches can only output point clouds with limited resolution since a number of points is fixed and not suitable for generating watertight surfaces due to lack of topology description in the data. In another example, voxel-based modeling approaches are computationally-intensive and require considerable memory resources, leading to slow training times and low-resolution representations. In a further example, mesh-based 3D reconstruction approaches can only generate meshes with a simple topology given by a fixed reference template from a common object class and cannot guarantee watertight surfaces.

[19] Developers of the present technology have also realized that conventional implicit methods also have drawbacks. For example, the DeepSDF method mainly focuses on recovering 3D shapes from a single-view partial point cloud and fails to leverage the multi-sweep information of vehicles. As a result, when noises or annotation errors exist in an individual sweep, single-view methods usually produce low-fidelity results. Additionally, the DeepSDF method only takes the given surface points of a partial point cloud into consideration when generating 3D shapes. This can lead to suboptimal reconstruction results in areas not captured in the partial point cloud, since the DeepSDF method will perform reconstruction arbitrarily based on prior knowledge learned during training. Let it be assumed that a point cloud only captures a front of a pickup truck. In this case, a DeepSDF method may be used to reconstruct the object as a pickup truck, or alternatively as an SUV, since the front of the two object models is similar. In this case, the process of reconstructing the rear of the object can be said to be "arbitrary" and/or involve uncertainty.

[20] With reference to FIG. 2A, PC1 and PC2 are point clouds of a same object from different viewpoints. The DeepSDF method reconstruction results, Mesh 1 and Mesh 2 are reconstruction outputs of the DeepSDF method applied onto PC1 and PC2, respectively. It should be noted that although Mesh 1 and Mesh 2 are reconstruction outputs of the same

object, they are notably different. Moreover, due to the noise and registration error, surface reconstructed by the stacked multi-sweep point cloud is usually not smooth, as seen in FIG. 2B.

[21] In some embodiments, conventional geometry methods (e.g. Iterative closest point (ICP)) can be used to generate stacked multi-sweep point clouds. In other embodiments, a stacking operation may be performed based on the annotations (e.g., bounding box). In further embodiments, a neural network can be trained for performing the stacking operation. For example, a processor may use a neural network to learn the relative poses among point clouds (e.g., weakly-supervised 3D shape completion in the wild).

[22] Developers of the present technology have realized that 3D vehicle reconstruction from sparse and partial point clouds may be used in variety of applications in the autonomous driving industry. In some embodiments of the present technology, there is provided a processor configured to infer from sparse and/or partial point clouds 3D structure of one or more vehicles at a variety of resolutions during in-use. This information may be further employed in one or more downstream tasks executed by the processor during operation the autonomous vehicle.

[23] Developers of the present technology have realized that conventional point cloud-based implicit modeling methods approaches only use a single-view partial point cloud for 3D reconstruction. In some embodiments of the present technology, there is provided a processor configured to perform an implicit modeling technique using multi-sweep point clouds.

[24] Broadly, multi-sweep point clouds refer to a collection of point clouds obtained from multiple scanning or sensing passes over the same area or object. Each individual point cloud in the multi-sweep dataset represents a separate scan or sweep taken at a different time or from a different perspective. The point cloud data is typically acquired using technologies such as LIDAR, for example. By combining multiple sweeps or scans, the multi-sweep point clouds provide a detailed representation of the scanned area. It can capture different viewpoints, occluded areas, or changes that may occur over time.

[25] Broadly, a stacked point cloud refers to a combination or fusion of multiple point clouds into a single composite point cloud. Unlike multi-sweep point clouds, where each individual scan or sweep is preserved separately, a stacked point cloud merges the data from different sources or scans into a unified representation. The process of creating a stacked point cloud involves aligning and combining the individual point clouds, sometimes obtained from different sensors, scanning devices, or viewpoints. The alignment is necessary to ensure that all the points are registered correctly in a common coordinate system. In some embodiments, multi-sweep point clouds may be stacked into a stacked multi-sweep point cloud.

[26] Developers of the present technology have realized that at least some conventional methods only extract global features from explicit representations, such as point clouds, voxels, and meshes. In some embodiments of the present technology, there is provided a processor configured to use multi-sweep point cloud input to (i) extract implicit features in the latent space, and (ii) extract global features, and (iii) combine the extracted implicit features and extracted explicit features into a common output.

[27] In further embodiments of the present technology, there is a processor configured to run a model architecture for leveraging multi-sweep LiDAR point cloud data as input, and which is trained on both explicit global features and implicit features in the latent space. Such a model may allow generating robust and higher fidelity reconstruction results on real-world autonomous driving datasets.

[28] In some embodiments, a processor may be configured to perform a stacking operation onto one or more input point clouds (e.g., the first and second point clouds), and thereby generate a stacked point cloud. Broadly, a stacked point cloud refers to a combination or fusion of multiple point clouds into a single point cloud. Unlike multi-sweep point clouds, where

each individual scan or sweep is preserved separately, a stacked point cloud is generated from merging the data from different sources or scans into a unified representation. The process of creating a stacked point cloud involves aligning and combining the individual point clouds, sometimes obtained from different sensors, scanning devices, and/or viewpoints. The alignment may be necessary to ensure that all the points are registered correctly in a common coordinate system. In some embodiments, multi-sweep point clouds may be stacked into a stacked multi-sweep point cloud.

[29] In a first broad aspect of the present technology, there is provided a method of performing 3D reconstruction of an object. The method executable by a processor. The method comprises acquiring a first point cloud and a second point cloud, the first and second point clouds being multi-sweep point clouds representing at least the object. The method comprises extracting, using a first network, explicit features about a shape of the object based on the first and second point clouds, the first network having been trained to explicitly model the shape of the object. The method comprises extracting, using a second network, implicit features about the shape of the object based on the first and second point clouds, the second network having been trained to implicitly model the shape of the object. The method comprises generating, using a third network, a latent code by projecting the implicit features and the explicit features into a latent space. The method comprises generating a 3D shape representative of the object using the latent code.

[30] In some embodiments of the method, the first network is at least one of a PCN encoder, and a PointNet encoder.

[31] In some embodiments of the method, the second network is at least one of a DeepSDF encoder, and a ONet encoder.

[32] In some embodiments of the method, the method further comprises generating a first modified point cloud and a second modified point cloud by applying a Farthest Point Sampling (FPS) function onto the first point cloud and the second point cloud, respectively. The first modified point cloud and the second modified point cloud have N respective number of points, N being an integer. The extracting the explicit features comprises extracting the explicit features based on the first modified point cloud and the second modified point cloud.

[33] In some embodiments of the method, the third network comprises a pooling layer for combining the implicit features and the explicit features and a fully-connected layer for outputting the latent code.

[34] In some embodiments of the method, the input further comprises images, voxels, meshes, and/or any combination thereof.

[35] In some embodiments of the method, the method further comprises, generating a database including the latent code.

[36] In a second broad aspect of the present technology, there is provided an electronic device for performing 3D reconstruction of an object. The electronic device comprises a processor and a memory, the memory storing instructions that when executed by the processor cause the electronic device to acquire a first point cloud and a second point cloud, the first and second point clouds being multi-sweep point clouds representing at least the object, extract, using a first network, explicit features about a shape of the object based on the first and second point clouds, the first network having been trained to explicitly model the shape of the object, extract, using a second network, implicit features about the shape of the object based on the first and second point clouds, the second network having been trained to implicitly model the shape of the object, generate, using a third network, a latent code by projecting the implicit features and the explicit features into a latent space, and generate a 3D shape representative of the object using the latent code.

[37] In some embodiments of the electronic device, the first network is at least one of a PCN encoder, and a PointNet encoder.

[38] In some embodiments of the electronic device, the second network is at least one of a DeepSDF encoder, and a ONet encoder.

[39] In some embodiments of the electronic device, the electronic device further generates a first modified point cloud and a second modified point cloud by applying a Farthest Point Sampling (FPS) function onto the first point cloud and the second point cloud, respectively, the first modified point cloud and the second modified point cloud have N respective number of points, N being an integer. The electronic device extracting the explicit features comprises the extracting the explicit features based on the first modified point cloud and the second modified point cloud.

[40] In some embodiments of the electronic device, the third network comprises a pooling layer for combining the implicit features and the explicit features and a fully-connected layer for outputting the latent code.

[41] In some embodiments of the electronic device, the input further comprises images, voxels, meshes, and/or any combination thereof.

[42] In some embodiments of the electronic device, the electronic device further generates a database including the latent code.

[43] In the context of the present specification, a "server" is a computer program that is running on appropriate hardware and is capable of receiving requests (e.g., from devices) over a network, and carrying out those requests, or causing those requests to be carried out. The hardware may be one physical computer or one physical computer system, but neither is required to be the case with respect to the present technology. In the present context, the use of the expression a "server" is not intended to mean that every task (e.g., received instructions or requests) or any particular task will have been received, carried out, or caused to be carried out, by the same server (i.e., the same software and/or hardware); it is intended to mean that any number of software elements or hardware devices may be involved in receiving/sending, carrying out or causing to be carried out any task or request, or the consequences of any task or request; and all of this software and hardware may be one server or multiple servers, both of which are included within the expression "at least one server".

[44] In the context of the present specification, "device" is any computer hardware that is capable of running software appropriate to the relevant task at hand. Thus, some (non-limiting) examples of devices include personal computers (desktops, laptops, netbooks, etc.), smartphones, and tablets, as well as network equipment such as routers, switches, and gateways. It should be noted that a device acting as a device in the present context is not precluded from acting as a server to other devices. The use of the expression "a device" does not preclude multiple devices being used in receiving/sending, carrying out or causing to be carried out any task or request, or the consequences of any task or request, or steps of any method described herein.

[45] In the context of the present specification, a "database" is any structured collection of data, irrespective of its particular structure, the database management software, or the computer hardware on which the data is stored, implemented or otherwise rendered available for use. A database may reside on the same hardware as the process that stores or makes use of the information stored in the database or it may reside on separate hardware, such as a dedicated server or plurality of servers. It can be said that a database is a logically ordered collection of structured data kept electronically in a computer system

[46] In the context of the present specification, the expression "information" includes information of any nature or kind whatsoever capable of being stored in a database. Thus information includes, but is not limited to audiovisual works (images, movies, sound records, presentations etc.), data (location data, numerical data, etc.), text (opinions, comments, questions, messages, etc.), documents, spreadsheets, lists of words, etc.

[47] In the context of the present specification, the expression "component" is meant to include software (appropriate to a particular hardware context) that is both necessary and sufficient to achieve the specific function(s) being referenced.

[48] In the context of the present specification, the expression "computer usable information storage medium" is intended to include media of any nature and kind whatsoever, including RAM, ROM, disks (CD-ROMs, DVDs, floppy disks, hard drivers, etc.), USB keys, solid state-drives, tape drives, etc.

[49] In the context of the present specification, the words "first", "second", "third", etc. have been used as adjectives only for the purpose of allowing for distinction between the nouns that they modify from one another, and not for the purpose of describing any particular relationship between those nouns. Thus, for example, it should be understood that, the use of the terms "first server" and "third server" is not intended to imply any particular order, type, chronology, hierarchy or ranking (for example) of/between the server, nor is their use (by itself) intended imply that any "second server" must necessarily exist in any given situation. Further, as is discussed herein in other contexts, reference to a "first" element and a "second" element does not preclude the two elements from being the same actual real-world element. Thus, for example, in some instances, a "first" server and a "second" server may be the same software and/or hardware, in other cases they may be different software and/or hardware.

[50] Implementations of the present technology each have at least one of the above-mentioned object and/or aspects, but do not necessarily have all of them. It should be understood that some aspects of the present technology that have resulted from attempting to attain the above-mentioned object may not satisfy this object and/or may satisfy other objects not specifically recited herein.

[51] Additional and/or alternative features, aspects and advantages of implementations of the present technology will become apparent from the following description, the accompanying drawings and the appended claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[52] For a better understanding of the present technology, as well as other aspects and further features thereof, reference is made to the following description which is to be used in conjunction with the accompanying drawings, where:

[53] FIG. 1 illustrates an example of a computing device that may be used to implement any of the methods described herein.

[54] FIG. 2A shows point clouds of a same ground-truth object from different viewpoints and reconstruction results using a conventional reconstruction technique.

[55] FIG. 2B shows a stacked point cloud and a mesh obtained directly from the stack point cloud using a conventional reconstruction technique.

[56] FIG. 3 illustrates a schematic representation of a model configured to extract both implicit and explicit features for generating a representation in a latent space, in accordance with at least some non-limiting embodiments of the present technology.

[57] FIG. 4 illustrates a non-limiting example of a neural architecture of a model in accordance with at least some non-limiting embodiments of the present technology.

[58] FIG. 5 illustrates a comparison of different methods using the Waymo dataset for 3D object reconstruction.

[59] FIG. 6 illustrates a comparison of different methods using the KITTI dataset for 3D object reconstruction.

[60] FIG. 7 is a scheme-block illustration of a method executed by a processor of the computing device of FIG. 1, in accordance with at least some non-limiting embodiments of the present technology.

DETAILED DESCRIPTION

[61] The examples and conditional language recited herein are principally intended to aid the reader in understanding the principles of the present technology and not to limit its scope to such specifically recited examples and conditions. It will be appreciated that those skilled in the art may devise various arrangements which, although not explicitly described or shown herein, nonetheless embody the principles of the present technology and are included within its spirit and scope.

[62] Furthermore, as an aid to understanding, the following description may describe relatively simplified implementations of the present technology. As persons skilled in the art would understand, various implementations of the present technology may be of a greater complexity.

[63] In some cases, what are believed to be helpful examples of modifications to the present technology may also be set forth. This is done merely as an aid to understanding, and, again, not to define the scope or set forth the bounds of the present technology. These modifications are not an exhaustive list, and a person skilled in the art may make other modifications while nonetheless remaining within the scope of the present technology. Further, where no examples of modifications have been set forth, it should not be interpreted that no modifications are possible and/or that what is described is the sole manner of implementing that element of the present technology.

[64] Moreover, all statements herein reciting principles, aspects, and implementations of the present technology, as well as specific examples thereof, are intended to encompass both structural and functional equivalents thereof, whether they are currently known or developed in the future. Thus, for example, it will be appreciated by those skilled in the art that any block diagrams herein represent conceptual views of illustrative circuitry embodying the principles of the present technology. Similarly, it will be appreciated that any flowcharts, flow diagrams, state transition diagrams, pseudo-code, and the like represent various processes which may be substantially represented in computer-readable media and so executed by a computer or processor, whether or not such computer or processor is explicitly shown.

[65] The functions of the various elements shown in the figures, including any functional block labeled as a "processor", may be provided through the use of dedicated hardware as well as hardware capable of executing software in association with appropriate software. When provided by a processor, the functions may be provided by a single dedicated processor, by a single shared processor, or by a plurality of individual processors, some of which may be shared. In some embodiments of the present technology, the processor may be a general purpose processor, such as a central processing unit (CPU) or a processor dedicated to a specific purpose, such as a digital signal processor (DSP). Moreover, explicit use of the term a

"processor" should not be construed to refer exclusively to hardware capable of executing software, and may implicitly include, without limitation, application specific integrated circuit (ASIC), field programmable gate array (FPGA), read-only memory (ROM) for storing software, random access memory (RAM), and non-volatile storage. Other hardware, conventional and/or custom, may also be included.

[66] Software modules, or simply modules which are implied to be software, may be represented herein as any combination of flowchart elements or other elements indicating performance of process steps and/or textual description. Such modules may be executed by hardware that is expressly or implicitly shown. Moreover, it should be understood that module may include for example, but without being limitative, computer program logic, computer program instructions, software, stack, firmware, hardware circuitry or a combination thereof which provides the required capabilities.

[67] With these fundamentals in place, we will now consider some non-limiting examples to illustrate various implementations of aspects of the present technology.

[68] FIG. 1 illustrates a diagram of a computing environment 100 in accordance with an embodiment of the present technology is shown. In some embodiments, the computing environment 100 may be implemented by any of a conventional personal computer, a computer dedicated to operating and/or monitoring systems relating to a data center, a controller and/or an electronic device (such as, but not limited to, a mobile device, a tablet device, a server, a controller unit, a control device, a monitoring device etc.) and/or any combination thereof appropriate to the relevant task at hand. In some embodiments, the computing environment 100 comprises various hardware components including one or more single or multi-core processors collectively represented by a processor 110, a solid-state drive 120, a random access memory 130 and an input/output interface 150.

[69] In some embodiments, the computing environment 100 may also be a sub-system of one of the above-listed systems. In some other embodiments, the computing environment 100 may be an "off the shelf" generic computer system. In some embodiments, the computing environment 100 may also be distributed amongst multiple systems. The computing environment 100 may also be specifically dedicated to the implementation of the present technology. As a person in the art of the present technology may appreciate, multiple variations as to how the computing environment 100 is implemented may be envisioned without departing from the scope of the present technology.

[70] Communication between the various components of the computing environment 100 may be enabled by one or more internal and/or external buses 160 (e.g. a PCI bus, universal serial bus, IEEE 1394 "Firewire" bus, SCSI bus, Serial-ATA bus, ARINC bus, etc.), to which the various hardware components are electronically coupled.

[71] The input/output interface 150 may allow enabling networking capabilities such as wire or wireless access. As an example, the input/output interface 150 may comprise a networking interface such as, but not limited to, a network port, a network socket, a network interface controller and the like. Multiple examples of how the networking interface may be implemented will become apparent to the person skilled in the art of the present technology. For example, but without being limitative, the networking interface may implement specific physical layer and data link layer standard such as Ethernet, Fibre Channel, Wi-Fi or Token Ring. The specific physical layer and the data link layer may provide a base for a full network protocol stack, allowing communication among small groups of computers on the same local area network (LAN) and large-scale network communications through routable protocols, such as Internet Protocol (IP).

[72] According to implementations of the present technology, the solid-state drive 120 stores program instructions suitable for being loaded into the random access memory 130 and executed by the processor 110 for executing operating

data centers based on a generated machine learning pipeline. For example, the program instructions may be part of a library or an application.

[73] In some embodiments of the present technology, the computing environment 100 may be implemented as part of a cloud computing environment. Broadly, a cloud computing environment is a type of computing that relies on a network of remote servers hosted on the internet, for example, to store, manage, and process data, rather than a local server or personal computer. This type of computing allows users to access data and applications from remote locations, and provides a scalable, flexible, and cost-effective solution for data storage and computing. Cloud computing environments can be divided into three main categories: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). In an IaaS environment, users can rent virtual servers, storage, and other computing resources from a third-party provider, for example. In a PaaS environment, users have access to a platform for developing, running, and managing applications without having to manage the underlying infrastructure. In a SaaS environment, users can access pre-built software applications that are hosted by a third-party provider, for example. In summary, cloud computing environments offer a range of benefits, including cost savings, scalability, increased agility, and the ability to quickly deploy and manage applications.

[74] With reference to FIG. 3, there is depicted a schematic pipeline 300 of the processor 110 generating instance features 314. In this embodiment, the processor 110 is configured to provide input 302 to an implicit information extractor 308 and to an implicit information extractor 304.

[75] The content of the input 302 may vary depending on *inter alia* various implementations of the present technology. The input 302 may comprise point clouds, voxels, meshes, images or any combination thereof. For example, the input 302 may comprise point clouds and images of a given scene.

[76] The explicit information extractor 308 is configured to extract explicit features 310 from an explicit shape of an object in the input 302 (e.g., point clouds, meshes, voxels, and images). How the explicit information extractor 308 can be implemented is not particularly limiting. A choice of machine learning model for implement the explicit information extractor 308 may be made based on *inter alia* a format of the input 302. For example, if the input 302 comprises multiple point clouds, the explicit information extractor 308 can be implemented as PointNet encoder, Point Completion Network (PCN) encoder, and the like.

[77] Broadly, a PointNet encoder is a deep learning architecture designed for processing point cloud data, which represents the geometric structure of objects or environments. The PointNet encoder can employ shared Multi-Layer Perceptrons (MLPs) to capture local geometric information, while max-pooling aggregates the local features to generate a global feature vector. The PointNett encoder can also comprise a transformation network to handle rotation and translation invariance. The PointNet encoder may process unstructured point clouds.

[78] Broadly, a PCN encoder is a deep learning architecture designed for point cloud completion tasks. Point cloud completion aims to reconstruct missing or occluded parts of a 3D object based on the available partial point cloud data. The PCN encoder addresses the challenge of completing partial point clouds by effectively capturing the underlying structure and geometry. It takes in the incomplete point cloud as input and generates a complete and coherent representation of the object. The PCN encoder employs a hierarchical feature extraction process to capture local and global contextual information. It consists of multiple layers that progressively refine the feature representation of the input point cloud. This hierarchical structure allows the network to capture both fine-grained details and higher-level semantics. To generate the completed point cloud, the encoder may comprise MLPs and convolutional layers, for example. These layers analyze the available point cloud data and learn to fill in the missing regions based on the observed context. The PCN encoder may be trained using a

combination of supervised and unsupervised learning approaches. Supervised learning utilizes paired datasets, where complete and partial point clouds are available, to provide direct supervision for training. Unsupervised learning techniques, such as adversarial training or reconstruction loss, are also employed to encourage the model to generate plausible and realistic completions.

[79] The implicit information extractor 304 is configured to extract implicit features 306 from an implicit shape of the object in the input 302, such as *via* SDFs, Neural Radiance Field (NeRF) model, and occupancy functions, for example.

[80] Broadly, NeRF model comprises a deep learning architecture that enables synthesis of detailed 3D scenes from a sparse set of 2D images or 2D-to-3D correspondences. NeRF models are configured to model the volumetric scene as a continuous function that predicts radiance values for different 3D points within the scene. By training a neural network on a set of observed images, the NeRF model can learn to infer the volumetric density and color properties of the scene. This allows for the generation of novel views and renderings from different viewpoints. The NeRF model can capture scene details, including reflections, refractions, and complex lighting effects, during 3D reconstruction. The NeRF model can operate by sampling rays from the camera through the scene and using a deep neural network to estimate the radiance along each ray. The NeRF model may enable synthesis of novel views by integrating information from multiple rays and/or viewpoints.

[81] Broadly, occupancy functions are configured to model an occupancy status of space in a 3D environment. These functions may be used to infer the presence, or absence, of object(s) and/or structure(s) within a given volume. The occupancy function may be used to discretize the 3D space into voxels or cells and assign a binary value to each voxel indicating whether it is occupied or unoccupied. This binary representation forms a voxel grid or occupancy grid, which serves as a compact and structured representation of the 3D scene. Occupancy functions can be estimated from sensor data, such as point clouds, depth maps, or volumetric data obtained from techniques like LiDAR or structured light scanning. Various algorithms and techniques can be employed to process the sensor data and determine the occupancy status of each voxel. In some cases, a probabilistic occupancy estimation may be performed where the occupancy function assigns a probability value to each voxel, representing the likelihood of it being occupied. This probabilistic representation may be selected to facilitate fusion of multiple sensor modalities or data sources.

[82] How the implicit information extractor 304 can be implemented is not particularly limiting. A choice of machine learning model for implement the implicit information extractor 304 may be made based on *inter alia* a format of the input 302. For example, if the input is multiple point clouds, the implicit information extractor 304 can be implemented as a DeepSDF model, ONet model, and the like.

[83] Broadly, DeepSDF model comprises a deep learning architecture that models the surface of 3D objects as a continuous implicit function and estimates the signed distance to the object surface from any given point in space. DeepSDF is a neural network trained to learn the implicit representation of 3D shapes from a set of example shapes. The network is provided with 3D coordinates as input and predicts the signed distance to the surface. By training on a large dataset of shapes, the DeepSDF model learns to capture the underlying shape structure and generalizes to reconstruct new shapes. The DeepSDF model may comprise one or more deep neural networks, such as MLPs and/or convolutional networks for learning shape representation. During training, an optimization process can be used to minimize the difference between the predicted signed distances and ground truth distances from the training shapes.

[84] Broadly, the ONet model comprises a deep learning architecture configured to model the occupancy status of 3D space. It provides a compact and structured representation of the 3D scene by estimating the occupancy probabilities of a grid of 3D voxels. The ONet model comprises neural networks that are used to model the 3D occupancy of space based on observed data, such as point clouds, depth maps, or images. It estimates the probability of each voxel being occupied, which indicates whether an object or structure exists at that location in space. The ONet model comprises of an encoder-decoder architecture where the encoder processes the input data and extracts features, while the decoder predicts the occupancy probabilities for each voxel in the 3D grid. The network is trained on labeled data, typically using supervised learning techniques, to learn the relationship between the observed data and the corresponding occupancy probabilities.

[85] In this embodiment, the processor 110 is configured to provide the explicit features 310 and the implicit features 306 are provided to a pooling module 312. Broadly speaking, the pooling module 312 is configured to pool explicit and implicit information and project the instance features 314 to a space and/or size. In one specific implementation, the output may be in a form of a 1 1x256 vector, called a latent code. In one specific implementation, the pooling module 312 may be a neural architecture with a max pooling layer and two fully-connected layers. In other embodiments, if the input format is meshes and/or images, neural architecture of the pooling module 312 may be implemented differently to be suitable for use with the input format.

[86] With reference to FIG. 4, there is depicted a neural architecture 400 implemented on the processor 110 for generating a 3D mesh 422 based on input 402 in accordance with at least one non-limiting embodiment of the present technology. The input 402 comprises multi-view LiDAR sweeps of a given object. In this example, the object is a same vehicle shown in the multi-view LiDAR sweep data. The processor 110 provides the input 402 to a Farthest Point Sampling (FPS) function. The FPS function is configured to provide a network 410 with processed point clouds, all including N respective number of points. For example, the FPS function may be executed with a number of centroids parameter set to "256".

[87] The processor 110 is configured to perform explicit information extraction *via* the network 410. In this example, the network 410 is a shared PCN model employed to extract global features from respective processed point clouds. The processor 110 is configured to perform implicit information extraction from the input 402 using a network 406. In this example, the network 406 may be a DeepSDF model for generate latent code(s) 408 for respective partial point cloud(s) from the input 402. Outputs of the network 410 and of the network 406 (e.g., tensors of latent codes and global features) are concatenated and then provided to a network 414. The network 414 transforms the concatenated data into instance features via a max pooling operation(s). The network 414 then projects the instance features to the size of a latent code 416 by two fully-connected layers.

[88] In some embodiments, post-FPS point clouds are used as input into the network 410 to extract explicit global features for each post-FPS point cloud. In parallel, the original partial point clouds from the input 402 are used as input into the network 406 to extract latent codes for each partial point cloud. The processor 110 may be configured execute a combination operation 412 to combine outputs of the networks 410 and 406. The combination operation 412 may be a concatenation operation performed by the processor 110 on the output data of the networks 410 and 406. A combined output is the provided to an average pooling operation to aggregate the information into a single instance tensor. Finally, this instance tensor is transformed into the predicted latent code 416 by one or more fully-connected layers.

[89] The processor 110 is configured to use a pre-trained DeepSDF decoder 418 to map the latent code 416 to the SDF for the given object (vehicle). The processor 110 may be configured to mesh extrication function to extract a 3D mesh at a variety of different resolutions.

[90] In some embodiments, a predicted latent code may be converted to an SDF using a pre-trained DeepSDF decoder. The iso-surface composed of querying points with zero SDF values represents the surface of the given object and the implicit surface can be rasterized to a 3D mesh using Marching Cubes, for example. Since the SDF is a continuous function, 3D mesh extraction can be supported at a variety of different resolutions.

[91] How one or more networks of the neural architecture 400 can be trained in some embodiments of the present technology will now be described. In at least some embodiments of the present technology, the neural architecture 400 can be trained by the processor 110 in a two-stage process.

[92] During a first training stage, the processor 110 is configured to train the networks 406 and 418. For example, the processor 110 may be configured to train a DeepSDF decoder using watertight CAD models. In some implementations, the processor 110 may be configured to obtain a DeepSDF decoder pre-trained based on a ShapeNetV2 dataset.

[93] In one non-limiting example, a CAD model may be stored by the processor 110 in memory as an STL file. Broadly, an STL file describes a raw, unstructured triangulated surface by the unit normal and vertices (ordered by the right-hand rule) of the triangles using a 3D cartesian system. It should be noted that in some cases, to perform a 3D analysis on an STL model, the model may need to be watertight before being imported.

[94] Broadly, a watertight model may be defined as a model with at least some of the following features: (i) every triangle edge in the model has exactly two neighbors, thereby there are no holes or non-manifold edges, (ii) every node in the triangle is connected to only one fan of triangles around it- that is, for a given node, every triangle that shares that node must be accessible from any other triangle that shares the same node by moving across triangle edges (this may also be a condition on the domain being a proper manifold), (iii) there are no geometric overlaps or intersections in the model, where the model may be properly manifold but may still have overlaps and/or intersections due to triangles intersecting or overlapping with each other geometrically, and (iv) there are no geometric errors that produce unrealistically thin areas, where although similar to an overlap, may not be severe enough to be picked up as an overlap.

[95] In some embodiments, a partial point cloud generation method can be used for generating the training set from the CAD models. For example, at least some partial point cloud generation methods allow to position a virtual LiDAR with real-world parameters (resolution and sampling pattern) about the object and simulates the point cloud captured by the virtual LiDAR from that position. These methods differ from a method typically used for DeepSDF decoder training, where a simulated depth camera is used as the virtual sensor instead of a LiDAR. The processor 110 can use the pre-trained DeepSDF decoder to generate a latent code for a corresponding input partial point cloud.

[96] During a second training stage, the processor 110 is configured to train the networks 410 and 414 to reconstruct 3D objects based on partial multi-sweep point clouds and their corresponding latent codes.

[97] Let it be assumed that processing is performed for a c^{th} vehicle instance. Let it be assumed that $P_{B,c}$ represents a set of partial multi-sweep point clouds, $Z_{B,c}$ represents a set of the latent codes corresponding to the partial multi-view point clouds, and $z_{gt,c}$ represents the ground truth latent code obtained during the first training stage. It should be noted that multi-sweep point clouds refer to data representing an object that has been scanned by a LiDAR sensor, for example, in several frames. However, the point of view of these scans may or may not be the same. It should be noted that multi-view point clouds are a subset of multi-sweep clouds where the point clouds are taken from different points of view. However, a point of view may be fixed and then the object may be scanned multiple times. Furthermore, let it be assumed that $g(\cdot)$ represents

the function of the implicit shape prediction network 406 and α represents the parameters of the networks 419 and 414. The training objective of second training stage can be defined as:

$$\arg\min_{\alpha} \sum_{c=1}^{C} \mathcal{L}_2\left(f_\alpha\left(Z_{\beta,c}, \mathbf{P}_{\beta,c}\right), z_{gt,c}\right), \tag{1}$$

where C represents the number of instances employed for training and $\mathcal{L}_2$ is the Mean Squared Error (MSE) loss. In some implementations, an Adam optimizer may be employed with a learning rate of $1e^{-5}$ and network training includes 20 epochs. Broadly, an Adam optimizer is an optimization algorithm used in machine learning for updating the parameters of a model during the training process. It is an extension of the stochastic gradient descent (SGD) optimization method that incorporates adaptive learning rates.

[98] With reference to FIG. 5, there is depicted experimental results comparing different 3D reconstruction methods including some 3D reconstructions methods contemplated in the context of the present technology. For example, there is depicted a set of input sweeps 502, a set of ground-truth shapes 504 associated with the set of input sweeps 502, outputs 506 of a plurality of conventional methods, and outputs 508 in accordance with some implementations of the present technology. In FIG. 5, experimentation has been performed on Waymo dataset which comprises the set of input sweeps 502 and ground-truth shapes 504.

[99] With reference to FIG. 6, there is depicted experimental results comparing different 3D reconstruction methods including some 3D reconstructions methods contemplated in the context of the present technology. For example, there is depicted a set of input sweeps 602, a set of ground-truth shapes 604 associated with the set of input sweeps 602, outputs 606 of a plurality of conventional methods, and outputs 608 in accordance with some implementations of the present technology. In FIG. 6, experimentation has been performed on KITTI dataset which comprises the set of input sweeps 602 and ground-truth shapes 604.

[100] It is contemplated that at least some networks from the network architecture 400 can be replaced or swapped for other networks. For example, the network 410 with a PCN-based architecture, can be replaced with PointNet, PointNet++, Dynamic Graph Convolutional Neural Network (DGCNN), or Frustum PointNets to perform global feature extraction. The network 406 with a DeepSDF architecture, can be replaced by other frameworks that extract features from an implicit shape representation, such as ONet, NeRF, and the like. The network 414 may be configured to combine the outputs of the networks 410 and 414 and my perform concatenations, average pooling, and mapping. However, the network 414 may be replaced a network with other operations that also merge/combine outputs of the network 410 and 414 and changes tensor dimensions, such as max pooling and bit-wise multiplication or addition operations, for example.

[101] Broadly, PointNet++ is a hierarchical architecture involving multiple stages of sampling, grouping, and transforming operations. A point cloud is downsampled, and local features are extracted at each level. The features are then upsampled and combined with higher-level features to capture more detailed information. This process allows PointNet++ to capture both local and global context in a scalable manner. The PointNet++ architecture performs feature learning hierarchically, enabling it to capture fine-grained geometric patterns and achieve improved performance on tasks like semantic segmentation and object classification compared to the original PointNet.

[102] Broadly, DGCNN is a deep learning architecture designed for processing point cloud data and exploits inherent spatial relationships between points in a point cloud by constructing a dynamic graph representation. The dynamic graph is

built by connecting each point to its k nearest neighbors based on Euclidean distance. This graph structure captures the local geometric context of each point. The DGCNN architecture can comprise several components. Local feature extraction can be performed where per-point features are computed by passing each point's position information through shared MLPs. Edge convolution can be performed where a dynamic graph is used to perform graph convolutions on the local feature representations. This operation aggregates the information from neighboring points and captures the local geometric relationships. Global feature extraction can be performed where max pooling or permutation invariant functions are applied to the output of the edge convolution to generate global feature representations. Classification or segmentation can be performed where global features are fed into fully connected layers for the final classification or segmentation predictions. DGCNN leverages the graph convolutional neural network (GCN) framework, which enables it to capture and process the local and global geometric features of point cloud data. The dynamic graph construction allows the network to adaptively learn and update the graph structure during training.

[103] Broadly speaking, Frustum PointNets are deep learning architectures designed for 3D object detection in point cloud data by leveraging both 2D image data and 3D point cloud information. Frustum PointNets use 2D image-based object proposals (such as bounding boxes) generated by a 2D object detector as a starting point. These proposals are then transformed into 3D frustums (conical frustum shapes) in the point cloud data corresponding to the visible region in the image. The frustums serve as the regions of interest for further analysis. The Frustum PointNets architecture can comprises of different components. Frustum Extraction can be performed where the 2D object proposals are projected into the 3D point cloud, and the corresponding points within the frustum are extracted. PointNet is then applied to these frustums to learn local geometric features. A PointNet can also be employed to extract per-point features within the frustum. This helps capture fine-grained details and local object characteristics. Voting and object detection can be performed where a voting mechanism is utilized to estimate the 3D bounding box parameters for each object within the frustum. These estimated boxes are refined using a fully connected network, and a final classification and localization step is performed to detect and localize objects in 3D space.

[104] It should be noted that by replacing/swapping components of network architecture 400, the network architecture 400 can be tuned to work with point-based, mesh-based, or voxel-based input and output formats. It should also be noted that the network architecture 400 can also be used for other taxonomies in a variety of industries that benefit from 3D reconstruction. This can be achieved by using other application-specific datasets on which one or more networks of the network architecture 400 are trained.

[105] In some embodiments of the present technology, the processor 110 is configured to execute a method 700 a scheme-block illustration of which is depicted in FIG.7. It is contemplated that the method 700 can be executed by an electronic device implemented similarly to what has been described above with reference to FIG.1. In some embodiments, one or more steps of the method 700 may be executed by more than one physical processors. For example, more than one physical processors may be communicatively coupled over a network for performing one or more steps in a distributed manner. It is therefore contemplated that one or more steps from the method 700 may be executed by distinct electronic devices, without departing from the scope of the present technology.

STEP 702: acquiring a first point cloud and a second point cloud, the first and second point clouds being multi-sweep point clouds representing at least the object

[106] The method 700 begins with a processor acquiring a first point cloud and a second point cloud. The first and second point clouds are multi-sweep point clouds representing an object.

[107] Broadly, "multi-sweep" point clouds refer to one or more point clouds obtained from multiple scanning or sensing passes over a same area and/or object. In some embodiments, a processor may be configured to acquire a multi-sweep dataset comprising point clouds each of which represent a separate scan or sweep taken at a different time and/or from a different perspective. The point cloud data can be acquired from a LIDAR system, for example. By combining multiple sweeps or scans, the multi-sweep point clouds provide a detailed representation of the scanned area and/or object.

STEP 704: extracting, using a first network, explicit features about a shape of the object based on the first and second point clouds, the first network having been trained to explicitly model the shape of the object

[108] The method 700 continues to step 704 with extracting, using a first network, explicit features about a shape of the object based on the first and second point clouds. The first network has been trained to explicitly model the shape of the object.

[109] In some embodiments, "explicit" features may refer to features generated using one or more explicit modeling techniques. Explicit modeling techniques comprise point-based, voxel-based, and mesh-based techniques and are configured to directly represent 3D object shape structure with points, voxels, and meshes, respectively. A point-based 3D reconstruction model, such as a PCN model, a GRNet model, and a PoinTr model, for example, can output a point cloud as a shape representation of the 3D reconstruction. A voxel-based model can model volumes by subdividing the space into a 3D grid. Mesh-based models can generate meshes using a fixed reference template from a given object class. Conventional explicit reconstruction approaches, including point-based, voxel-based, and mesh-based models, suffer from problems such as low resolution and high memory costs.

[110] It can be said that explicit features represent features of a first type and which are extracted from an explicitly modeled object as opposed to an implicitly modeled object. In some embodiments, it is contemplated that explicit features may include one or more intermediate features of at least one of a PCN model, GRNet model, and PoinTr model, and the like.

[111] In some embodiments, the first network may be configured to extract explicit features from an explicitly modeled shape of an object in the input (e.g., point clouds, meshes, voxels, and images). A choice of machine learning model for implement the explicit information extractor 308 may be made based on *inter alia* a format of the input 302.

[112] In some embodiments, the method 700 may further comprise a step of generating a first modified point cloud and a second modified point cloud by applying an FPS function onto the first point cloud and the second point cloud, respectively. This may allow generating modified points clouds having a same number N of points. In these embodiments, the step 704 may be performed on the first modified and second modified point clouds, as opposed to the first and second point clouds.

STEP 706: extracting, using a second network, implicit features about the shape of the object based on the first and second point clouds, the second network having been trained to implicitly model the shape of the object

[113] The method 700 continues to step 706 with extracting, using a second network, implicit features about the shape of the object based on the first and second point clouds. The second network has been trained to implicitly model the shape of the object.

[114] In some embodiments, "implicit" features may refer to features generated using one or more implicit modeling techniques. Contrary to explicit modeling techniques, implicit modeling uses function-based decision boundaries to implicitly define surfaces for 3D representations. For example, DeepSDF model trained to map the 3D shape to a low-

dimensional latent space and learn the projection from the latent space to an SDF that describes that 3D shape. Broadly speaking, SDF is an implicit function that specifies whether a querying point is inside, or outside, the surface of an object. The value of an SDF represents the distance of a querying point from the surface of the object and its sign, either positive or negative, indicates whether the querying point lies on the exterior or interior of the surface, respectively.

[115] Developers of the present technology have also realized that conventional implicit techniques have drawbacks. For example, the conventional DeepSDF model mainly focuses on recovering 3D shapes from a single-view partial point cloud and fails to leverage the multi-sweep information of vehicles. As a result, when noises or annotation errors exist in an individual sweep, single-view techniques (e.g., conventional implicit techniques) usually produce low-fidelity results. Additionally, the conventional DeepSDF model only takes the given surface points of a partial point cloud into consideration when generating 3D shapes. This can lead to suboptimal reconstruction results in areas not captured in the partial point cloud, since the DeepSDF method will perform reconstruction arbitrarily based on prior knowledge learned during training.

[116] It can be said that implicit features represent features of a second type and which are extracted from an implicitly modeled object. In some embodiments, implicit features may comprise one or more features extracted using occupancy networks, NeuS models, NeAT models, and the like. A given machine learning model for implementing the second network may be made based on *inter alia* a format of the input (e.g., point clouds, meshes, voxels, and images).

STEP 708: generating, using a third network, a latent code by projecting the implicit features and the explicit features into a latent space

[117] The method 700 continues to step 708 with generating, using a third network, a lateener code by projecting the implicit features of the explicit features into a latent space.

[118] Broadly speaking, the third network is configured to combine (e.g., pool) explicit and implicit information and project instance features to a latent space. In one specific implementation, the output of the third network may be in a form of a given vector, called a latent code. In one specific implementation, the third network may be a neural network with at least one max pooling layer and at least two fully-connected layers. In other embodiments, if the input format is meshes and/or images, the neural architecture used to implement the third neural network may be selected based on *inter alia* the format of the input (e.g., point clouds, meshes, voxels, and images).

[119] It should be noted that various steps of the method 700 may be performed by respective portions of a larger neural network. For example, a combined neural network may comprise one or more of the first network, the second network, and the third network for performing one or more steps of the method 700. The combined neural network may be a distributed network, without departing from the scope of the present technology.

[120] Developers of the present technology have realized that different feature extractors may have different domains. Even for the same type of feature extractor (e.g., for two explicit extractors), different extractors may not share the feature expression. As a result, a specific design of a system contemplated in the context of the present technology may depending on *inter alia* input sizes and/or input type may need to be considered during design of a system. Furthermore, developers have realized that combining explicit and implicit methodologies is a technically challenging task. It is contemplated that in at least some systems contemplated in the context of the present technology, in order to use in combination both implicit and explicit features from the input, features may need to be mapped/expressed in the same domain.

STEP 710: generating a 3D shape representative of the object using the latent code

[121] The method 700 continues to step 710 with generating a 3D shape representative of the object using the latent code.

[122] In some embodiments, a processor may use a pre-trained DeepSDF decoder to map the latent code to the SDF for the given object (e.g., vehicle). It is further contemplated that a processor may be configured to execute a mesh extrication function to extract a 3D mesh at one or more resolutions representing the given object.

[123] In some embodiments, a predicted latent code may be converted to an SDF using a pre-trained DeepSDF decoder. The iso-surface composed of querying points with zero SDF values represents the surface of the given object and the implicit surface can be rasterized to a 3D mesh using Marching Cubes, for example. Since the SDF is a continuous function, 3D mesh extraction can be supported at a variety of different resolutions.

[124] Modifications and improvements to the above-described implementations of the present technology may become apparent to those skilled in the art. The foregoing description is intended to be exemplary rather than limiting. The scope of the present technology is therefore intended to be limited solely by the scope of the appended claims.

CLAIMS

1. A method of performing 3D reconstruction of an object, the method executable by a processor, the method comprising:

acquiring a first point cloud and a second point cloud, the first and second point clouds being multi-sweep point clouds representing at least the object;

extracting, using a first network, explicit features about a shape of the object based on the first and second point clouds, the first network having been trained to explicitly model the shape of the object;

extracting, using a second network, implicit features about the shape of the object based on the first and second point clouds, the second network having been trained to implicitly model the shape of the object;

generating, using a third network, a latent code by projecting the implicit features and the explicit features into a latent space; and

generating a 3D shape representative of the object using the latent code.

2. The method of claim 1, wherein the first network is at least one of a PCN encoder, and a PointNet encoder.

3. The method of claim 1, wherein the second network is at least one of a DeepSDF encoder, and a ONet encoder.

4. The method of claim 1, wherein the method further comprises generating a first modified point cloud and a second modified point cloud by applying a Farthest Point Sampling (FPS) function onto the first point cloud and the second point cloud, respectively, the first modified point cloud and the second modified point cloud have N respective number of points, N being an integer, and wherein the extracting the explicit features comprises extracting the explicit features based on the first modified point cloud and the second modified point cloud.

5. The method of claim 1, wherein the third network comprises a pooling layer for combining the implicit features and the explicit features and a fully-connected layer for outputting the latent code.

6. The method of claim 1, wherein the input further comprises images, voxels, meshes, and/or any combination thereof.

7. The method of claim 1, wherein the method further comprises, generating a database including the latent code.

8. An electronic device for performing 3D reconstruction of an object, the electronic deivce comprising a processor and a memory, the memory storing instructions that when executed by the processor cause the electronic device to:

acquire a first point cloud and a second point cloud, the first and second point clouds being multi-sweep point clouds representing at least the object;

extract, using a first network, explicit features about a shape of the object based on the first and second point clouds, the first network having been trained to explicitly model the shape of the object;

extract, using a second network, implicit features about the shape of the object based on the first and second point clouds, the second network having been trained to implicitly model the shape of the object;

generate, using a third network, a latent code by projecting the implicit features and the explicit features into a latent space; and

generate a 3D shape representative of the object using the latent code.

9. The electronic device of claim 8, wherein the first network is at least one of a PCN encoder, and a PointNet encoder.

10. The electronic device of claim 8, wherein the second network is at least one of a DeepSDF encoder, and a ONet encoder.

11. The electronic device of claim 8, wherein the electronic device further generates a first modified point cloud and a second modified point cloud by applying a Farthest Point Sampling (FPS) function onto the first point cloud and the second point cloud, respectively, the first modified point cloud and the second modified point cloud have N respective number of points, N being an integer, and wherein the electronic device extracting the explicit features comprises the extracting the explicit features based on the first modified point cloud and the second modified point cloud.

12. The electronic device of claim 8, wherein the third network comprises a pooling layer for combining the implicit features and the explicit features and a fully-connected layer for outputting the latent code.

13. The electronic device of claim 8, wherein the input further comprises images, voxels, meshes, and/or any combination thereof.

14. The electronic device of claim 8, wherein the electronic device further generates a database including the latent code.

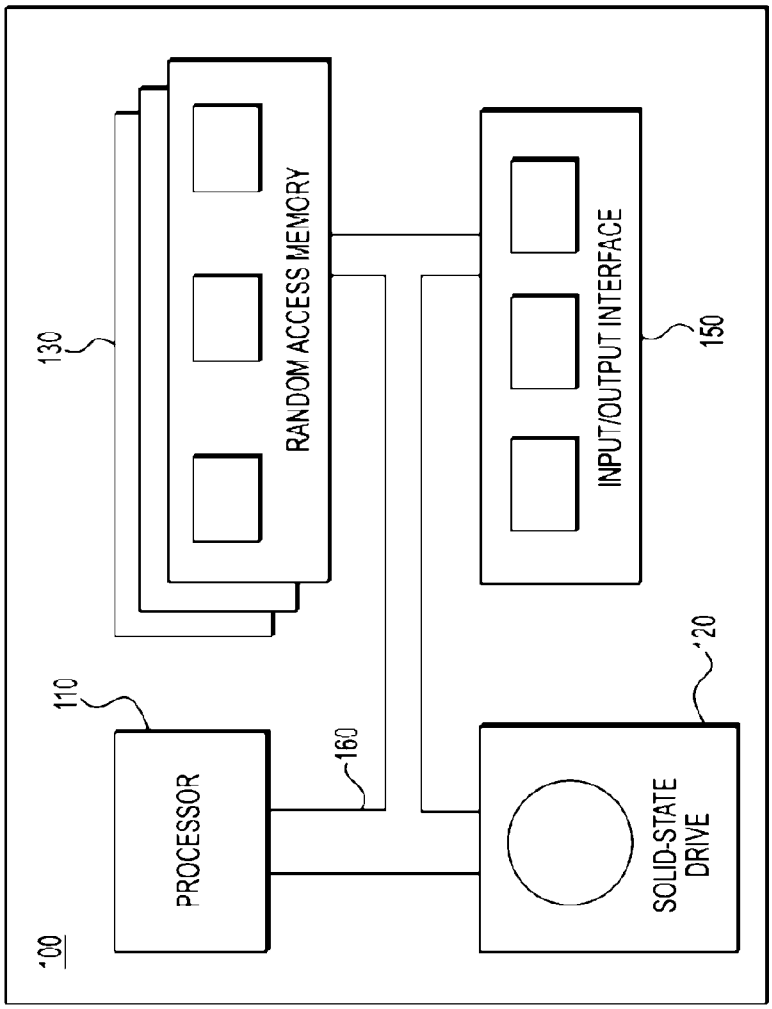
100
PROCESSOR
110
160
SOLID-STATE DRIVE
120
INPUT/OUTPUT INTERFACE
150
RANDOM ACCESS MEMORY
130

FIG. 1

FIG. 2A (PRIOR ART)

FIG. 2B (PRIOR ART)

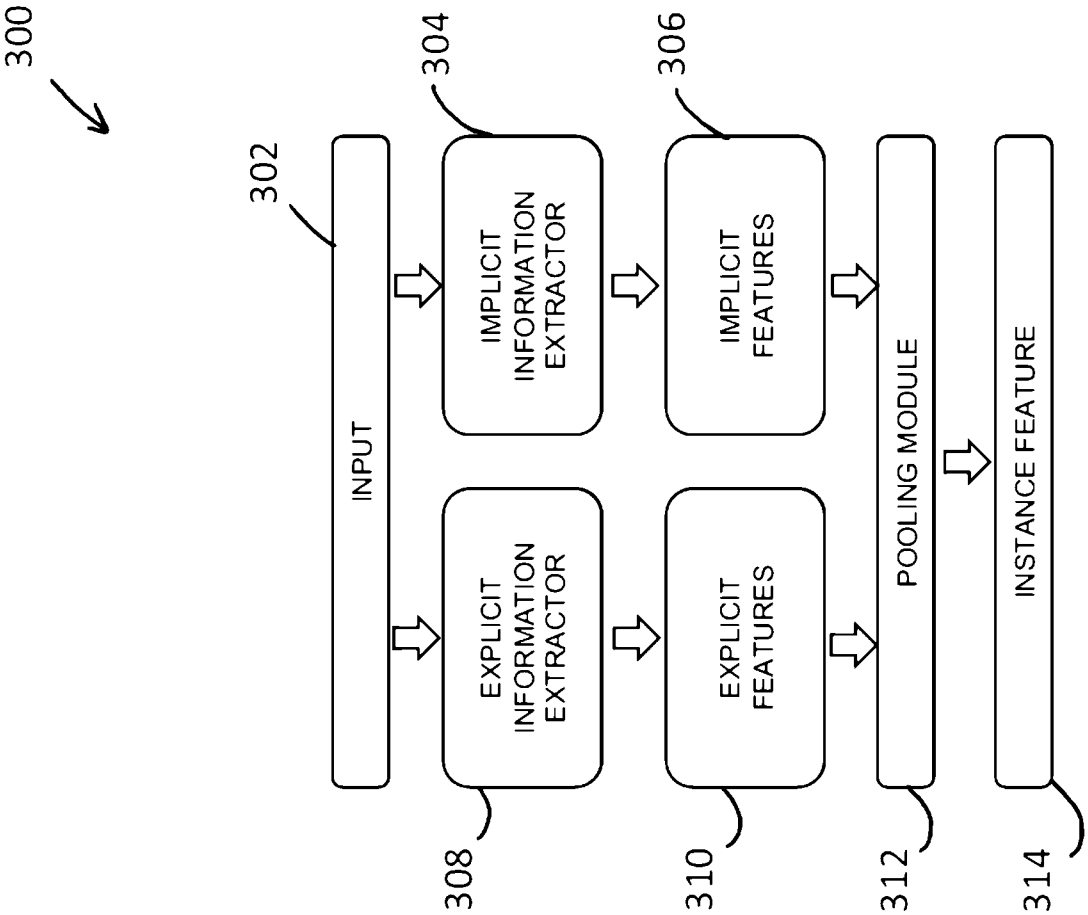
300
302
INPUT
304
IMPLICIT INFORMATION EXTRACTOR
306
IMPLICIT FEATURES
308
EXPLICIT INFORMATION EXTRACTOR
310
EXPLICIT FEATURES
312
POOLING MODULE
314
INSTANCE FEATURE

FIG. 3

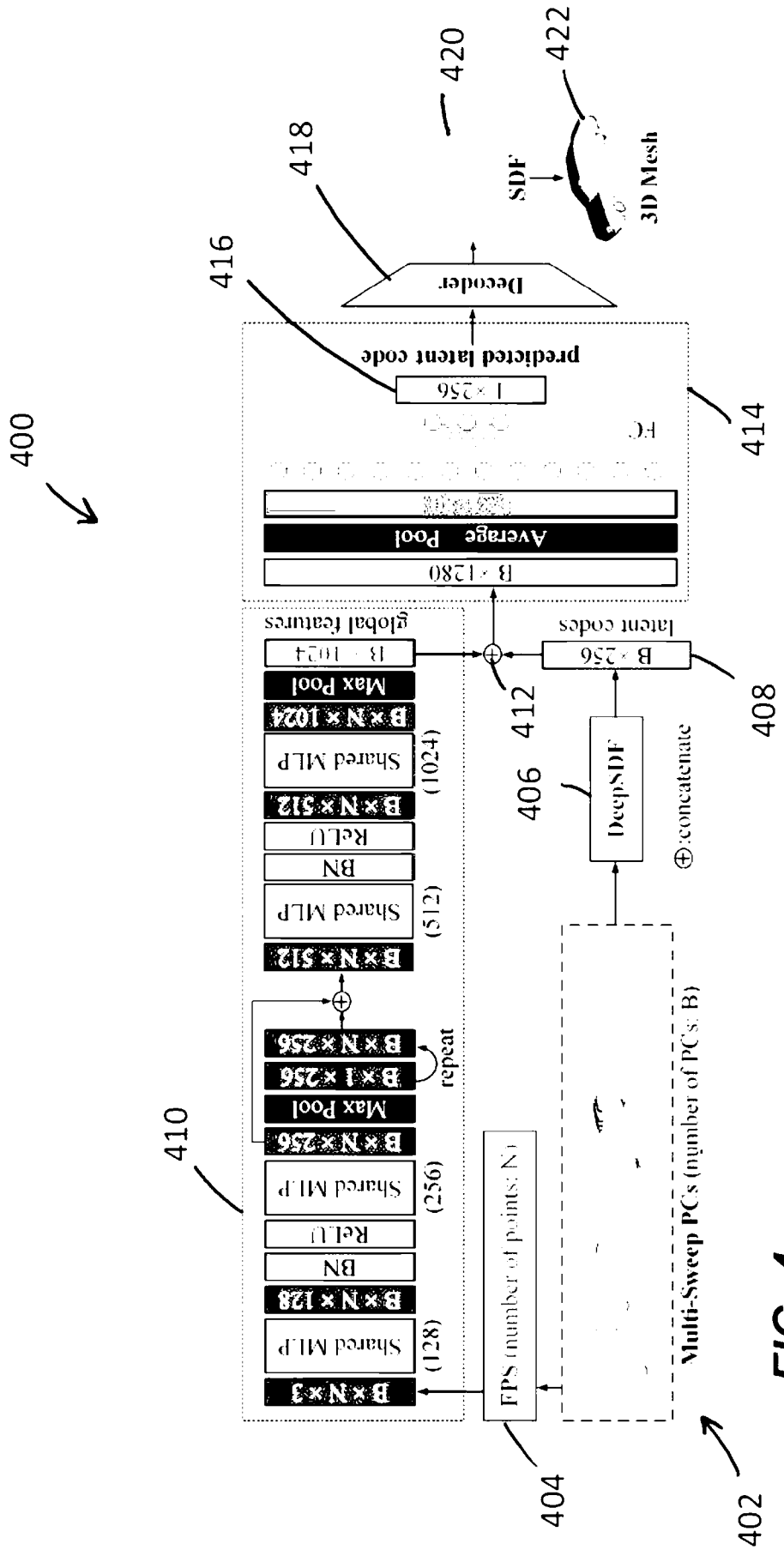
400
Multi-Sweep PCs (number of PCs: B)
FPS (number of points: N)
B × N × 3
Shared MLP
(128)
B × N × 128
BN
ReLU
Shared MLP
(256)
B × N × 256
Max Pool
B × 1 × 256
repeat
B × N × 256
Shared MLP
(512)
B × N × 512
BN
ReLU
Shared MLP
(1024)
B × N × 512
B × N × 1024
Max Pool
B × 1024
global features
B × 256
latent codes
DeepSDF
⊕:concatenate
B × 1280
Average Pool
FC
1 × 256
predicted latent code
Decoder
SDF
3D Mesh
402
404
406
408
410
412
414
416
418
420
422

FIG. 4

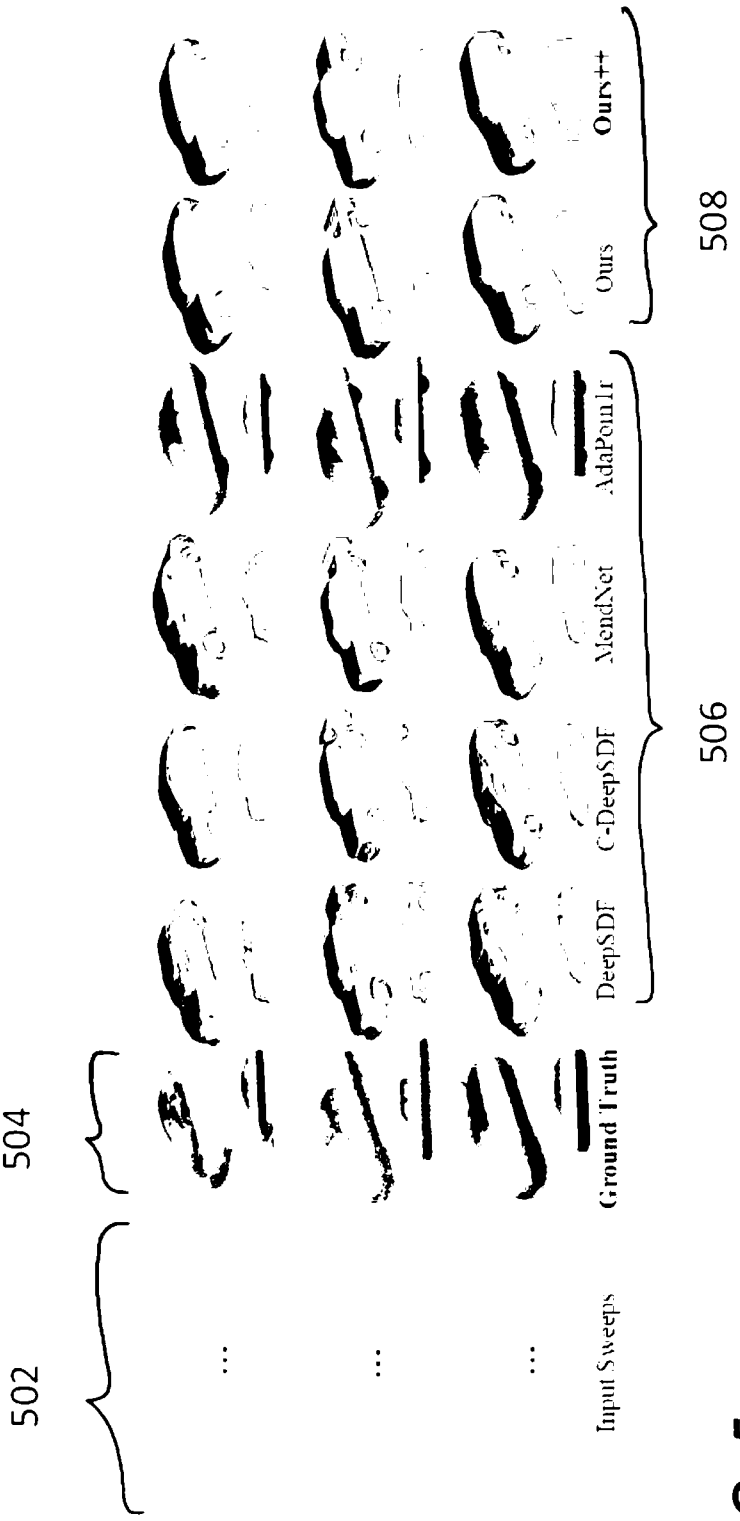
502
504
506
508
Input Sweeps
Ground Truth
DeepSDF
C-DeepSDF
MendNet
AdaPoinTr
Ours
Ours++

FIG. 5

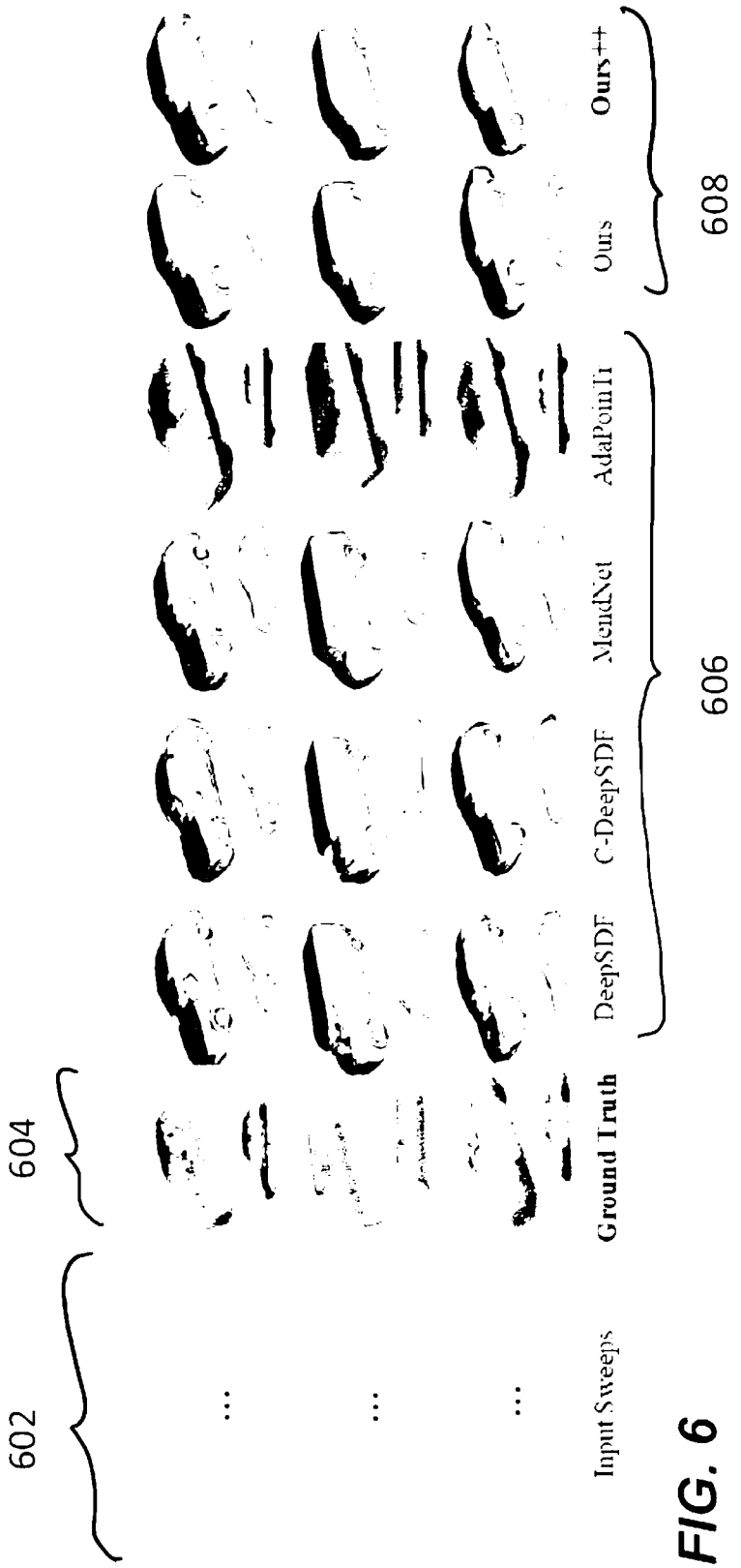
602
604
606
608
Input Sweeps
Ground Truth
DeepSDF
C-DeepSDF
MeshNet
AdaPoinTr
Ours
Ours++

FIG. 6

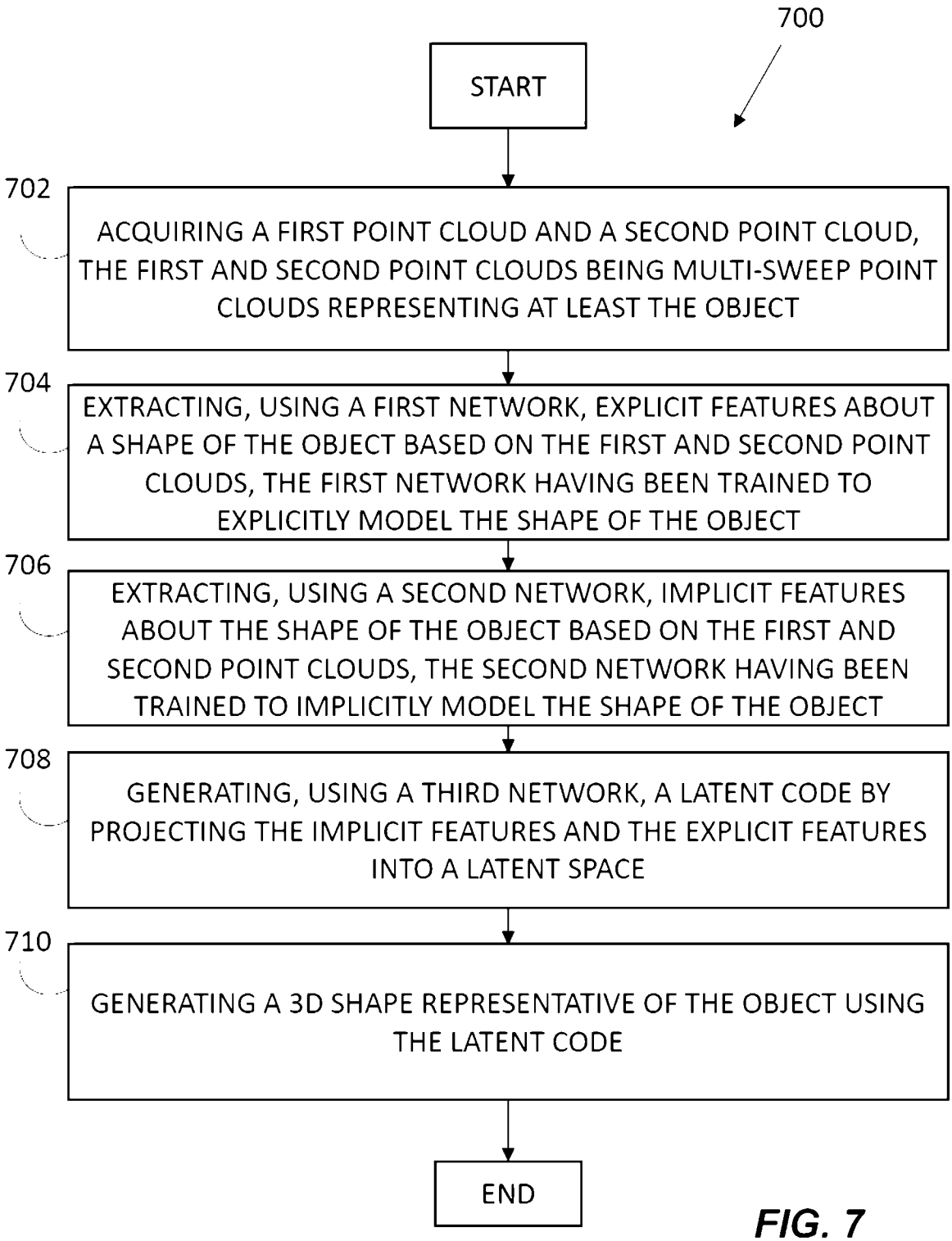
700
START
702
ACQUIRING A FIRST POINT CLOUD AND A SECOND POINT CLOUD, THE FIRST AND SECOND POINT CLOUDS BEING MULTI-SWEEP POINT CLOUDS REPRESENTING AT LEAST THE OBJECT
704
EXTRACTING, USING A FIRST NETWORK, EXPLICIT FEATURES ABOUT A SHAPE OF THE OBJECT BASED ON THE FIRST AND SECOND POINT CLOUDS, THE FIRST NETWORK HAVING BEEN TRAINED TO EXPLICITLY MODEL THE SHAPE OF THE OBJECT
706
EXTRACTING, USING A SECOND NETWORK, IMPLICIT FEATURES ABOUT THE SHAPE OF THE OBJECT BASED ON THE FIRST AND SECOND POINT CLOUDS, THE SECOND NETWORK HAVING BEEN TRAINED TO IMPLICITLY MODEL THE SHAPE OF THE OBJECT
708
GENERATING, USING A THIRD NETWORK, A LATENT CODE BY PROJECTING THE IMPLICIT FEATURES AND THE EXPLICIT FEATURES INTO A LATENT SPACE
710
GENERATING A 3D SHAPE REPRESENTATIVE OF THE OBJECT USING THE LATENT CODE
END

FIG. 7

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2023/101585

A. CLASSIFICATION OF SUBJECT MATTER

G06T 17/00(2006.01)i; G06T 9/00(2006.01)i

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC: G06T, G06N

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

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

CNTXT, ENTXT, ENTXTC, DWPI, CNKI: IMPLICIT, EXPLICIT, POINT CLOUD, SIGNED DISTANCE FUNCTION, SDF, PCN, POINTNET, ONET, FPS, MULTI-VIEW, MULTI-SWEEP, RECONSTRUCTION, LATENT SPACE

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	CN 111161364 A (UNIV SOUTHEAST) 15 May 2020 (2020-05-15) description, paragraphs [0002], [0047]-[0085], and figures 1-6	1-3, 5-10, 12-14
A	CN 111161364 A (UNIV SOUTHEAST) 15 May 2020 (2020-05-15) description, paragraphs [0002], [0047]-[0085], and figures 1-6	4, 11
Y	CN 115690324 A (GUANGZHOU ZHONGSI ARTIFICIAL INTELLIGENC) 03 February 2023 (2023-02-03) description, paragraphs [0097]-[0173]	1-3, 5-10, 12-14
A	CN 115690324 A (GUANGZHOU ZHONGSI ARTIFICIAL INTELLIGENC) 03 February 2023 (2023-02-03) description, paragraphs [0097]-[0173]	4, 11
A	CN 112184899 A (UNIV SUN YAT-SEN) 05 January 2021 (2021-01-05) the whole document	1-14
A	EP 4152274 A1 (TOYOTA JIDOSHA KK et al.) 22 March 2023 (2023-03-22) the whole document	1-14

☑ Further documents are listed in the continuation of Box C. ☑ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"D" document cited by the applicant in the international application
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
24 October 2023	**02 January 2024**
Name and mailing address of the ISA/CN **CHINA NATIONAL INTELLECTUAL PROPERTY ADMINISTRATION** **6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing 100088, China**	Authorized officer **HU,Yan** Telephone No. **(+86) 62089101**

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2023/101585

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	CN 115840971 A (DASSAULT SYSTEMES et al.) 24 March 2023 (2023-03-24) the whole document	1-14

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2023/101585

Patent document cited in search report	Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
CN 111161364 A	15 May 2020	CN 111161364 B	18 November 2022
CN 115690324 A	03 February 2023	None	
CN 112184899 A	05 January 2021	None	
EP 4152274 A1	22 March 2023	None	
CN 115840971 A	24 March 2023	US 2022405448 A1	22 December 2022
		JP 2022184829 A	13 December 2022
		EP 4099208 A1	07 December 2022