

UNSUPERVISED ANOMALY DETECTION FOR AUTONOMOUS VEHICLES

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application is based on U.S. Application No. 16/887,194, filed May 29, 2020, which claims priority to U.S. Provisional Application No. 62/900,380, filed September 13, 2019, the contents both of which are hereby incorporated by reference for all purposes.

TECHNICAL FIELD

[0002] This disclosure relates generally to detecting anomalies in time series data, and in particular but not exclusively, relates to detecting anomalies in telemetry information from autonomous vehicles.

BACKGROUND INFORMATION

[0003] As aerial robots become increasingly capable of complex navigation, perceptual reasoning and ability to learn from experience, it is expected that a large number of delivery missions will soon be executed by small air-vehicles taking off autonomously, flying far beyond line of sight over densely populated areas, hovering inside a residential zone within touching distance of humans to deliver the package, and returning to their "home" upon mission completion. Ensuring as high degree of operational reliability and safety as passenger airplanes is critical for delivery drones to achieve economies of scale.

SUMMARY OF INVENTION

[0004] In some embodiments, a non-transitory computer-readable medium is provided. The computer-readable medium has logic stored thereon that, in response to execution by one or more processors of a computing system, cause the computing system to perform actions for detecting anomalies in time series data records. The actions comprise receiving, by the computing system, a plurality of time series data records; initializing, by the computing system, a machine learning model having a set of fitting weights and a set of time series data record weights; optimizing, by the computing system, the set of fitting weights of the machine learning model while

holding the set of trip weights of the machine learning model constant; optimizing, by the computing system, the set of time series data record weights while holding the set of fitting weights constant; and storing, by the computing system, the optimized time series data record weights and the optimized fitting weights for use in monitoring for anomalies.

[0005] In some embodiments, a method for detecting anomalies in time series data records is provided. A computing device receives a time series data record from a monitored system. The computing device processes the time series data record using a machine learning model to generate an anomaly score, wherein the machine learning model was trained on a plurality of previous time series data records. The computing device compares the anomaly score to an anomaly threshold value. In response to determining that the anomaly score is greater than the anomaly threshold value the computing device determines an action to be taken to address the anomaly and transmits a command to the monitored system to cause the monitored system to perform the action to address the anomaly.

[0006] In some embodiments, a system is provided. The system comprises at least one computing device that includes at least one processor a non-transitory computer-readable medium. The computer-readable medium has logic stored thereon that, in response to execution by the at least one processor, causes the system to perform actions comprising: receiving a time series data record from a monitored system; processing the time series data record using a machine learning model to generate an anomaly score, wherein the machine learning model was trained on a plurality of previous time series data records; comparing the anomaly score to an anomaly threshold value; and in response to determining that the anomaly score is greater than the anomaly threshold value: determining an action to be taken to address the anomaly; and transmitting a command to the monitored system to cause the monitored system to perform the action to address the anomaly.

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] Non-limiting and non-exhaustive embodiments of the invention are described with reference to the following figures, wherein like reference numerals refer to like parts throughout the various views unless otherwise specified. Not all instances of an element are necessarily labeled so as not to clutter the drawings where

appropriate. The drawings are not necessarily to scale, emphasis instead being placed upon illustrating the principles being described. To easily identify the discussion of any particular element or act, the most significant digit or digits in a reference number refer to the figure number in which that element is first introduced.

5 **[0008]** FIG. 1 is a perspective top view illustration of a non-limiting example embodiment of an autonomous vehicle according to various aspects of the present disclosure.

[0009] FIG. 2 is a bottom side plan view illustration of the autonomous vehicle illustrated in FIG. 1.

10 **[0010]** FIG. 3 is a block diagram that illustrates a non-limiting example embodiment of an autonomous vehicle according to various aspects of the present disclosure.

[0011] FIG. 4 - FIG. 6 include several charts that illustrate non-limiting example embodiments of telemetry information collected by an autonomous vehicle
15 according to various aspects of the present disclosure.

[0012] FIG. 7 is a block diagram that illustrates a non-limiting example embodiment of an anomaly detection system according to various aspects of the present disclosure.

[0013] FIG. 8 and FIG. 9 are a flowchart that illustrates a non-limiting
20 example embodiment of a method of detecting anomalies in telemetry information according to various aspects of the present disclosure.

[0014] FIG. 10 includes charts that illustrate aspects of the results of alternating optimization of the smoothed trimmed loss according to various aspects of the present disclosure.

25 **[0015]** FIG. 11 is a chart that illustrates a comparison of anomaly detectors based on the approach disclosed herein against pure least-squares models and several M -estimators proposed in the Robust Statistics literature.

[0016] FIG. 12 - FIG. 14 includes several charts that illustrate experimental
30 results of using a machine learning model to detect anomalies according to various aspects of the present disclosure.

[0017] FIG. 15 - FIG. 16 include charts that illustrate the performance of various detectors on a test set of 5000 missions according to various aspects of the present disclosure.

5 [0018] FIG. 17 is a chart that illustrates a smoothed distribution of mission weights learned by an experimental example embodiment of the trimming technique.

[0019] FIG. 18 is a block diagram that illustrates a non-limiting example embodiment of a computing device appropriate for use as a computing device with embodiments of the present disclosure.

10

DETAILED DESCRIPTION

[0020] In some embodiments of the present disclosure, techniques are provided for analyzing time series data to detect anomalies. The time series data is processed using a machine learning model. The machine learning model is trained in an unsupervised manner on large amounts of previous time series data, thus allowing
15 highly accurate models to be created from novel data. Because a machine learning model is used, anomalies can be detected within complex systems, including but not limited to autonomous vehicles such as unmanned aerial vehicles. When anomalies are detected, commands can be transmitted to the monitored system (such as an autonomous vehicle) to respond to the anomaly.

20 [0021] While simple statistical thresholds and logical rules can be hand-designed to trigger on frequently occurring problematic events for an autonomous vehicle (e.g., battery too low, or control surface non-functional), they cannot exhaustively cover all potential future failure modes which are unknown apriori, particularly as a fleet of autonomous vehicles grows in mission complexity and
25 vehicle types. With this motivation, embodiments of the present disclosure provide an anomaly detection system based on a machine learning model that is continuously trained on thousands of time series data records such as flight logs that include telemetry information. When this model reports large predictive errors for a new trip, the autonomous vehicle can be flagged for manual inspection and possibly removed
30 from service for safety until the issue is resolved. Importantly, embodiments of the present disclosure are designed to discover normality, and do not require upfront labeling of normal and anomalous trips or time series data records. Indeed, sifting

through thousands of flight logs comprising of dozens of time series looking for subtle abnormalities stretches the limits of what is manually feasible.

[0022] The present disclosure is configured to learn a predictive model of time series data records, such as time series data records that reflect flight dynamics.

5 The linear and angular acceleration of an aircraft depend on the aerodynamic forces it is subject to, which are a function of the vehicle state, control commands, dynamic pressure and other flight condition variables. Simple linear or quadratic models trained on historical flight logs show impressive predictive power. The norm of the predictive residuals at a given time for a given flight, or the mean residual over an
10 entire flight can be used as thresholds for anomaly detection. However, in contrast to prior work that focused on large fixed wing passenger aircrafts and cruising performance only, we are interested in monitoring much smaller delivery drones across an entire flight mission that includes takeoff, package delivery, and landing.

15 **[0023]** FIG. 1 and FIG. 2 illustrate a non-limiting example embodiment an aerial vehicle or UAV 100, in accordance with an embodiment of the present disclosure. The illustrated embodiment of UAV 100 is a vertical takeoff and landing (VTOL) unmanned aerial vehicle (UAV) that includes separate propulsion units 106 and propulsion units 112 for providing horizontal and vertical propulsion,
20 respectively. UAV 100 is a fixed-wing aerial vehicle, which as the name implies, has a wing assembly 102 that can generate lift based on the wing shape and the vehicle's forward airspeed when propelled horizontally by propulsion units 106. FIG. 1 is a perspective top view illustration of UAV 100 while FIG. 2 is a bottom side plan view illustration of UAV 100.

25 **[0024]** The illustrated embodiment of UAV 100 includes a fuselage 104. In one embodiment, fuselage 104 is modular and includes a battery module, an avionics module, and a mission payload module. These modules are detachable from each other and mechanically securable to each other to contiguously form at least a portion of the fuselage 104 or UAV main body.

30 **[0025]** The battery module includes a cavity for housing one or more batteries for powering UAV 100. The avionics module houses flight control circuitry of UAV 100, which may include a processor and memory, communication electronics and antennas (e.g., cellular transceiver, Wi-Fi transceiver, etc.), and various sensors

(e.g., global positioning sensor, an inertial measurement unit (IMU), a magnetic compass, etc.). The mission payload module houses equipment associated with a mission of UAV 100. For example, the mission payload module may include a payload actuator for holding and releasing an externally attached payload. In another embodiment, the mission payload module may include a camera/sensor equipment holder for carrying camera/sensor equipment (e.g., camera, lenses, radar, LIDAR, pollution monitoring sensors, weather monitoring sensors, etc.). Other components that may be carried by some embodiments of the UAV 100 are illustrated in FIG. 3.

[0026] The illustrated embodiment of UAV 100 further includes horizontal propulsion units 106 positioned on wing assembly 102, which can each include a motor, shaft, motor mount, and propeller, for propelling UAV 100. The illustrated embodiment of UAV 100 includes two boom assemblies 110 that secure to wing assembly 102.

[0027] The illustrated embodiments of boom assemblies 110 each include a boom housing 116 in which a boom is disposed, vertical propulsion units 112, printed circuit boards 118, and stabilizers 108. Vertical propulsion units 112 can each include a motor, shaft, motor mounts, and propeller, for providing vertical propulsion. Vertical propulsion units 112 may be used during a hover mode where UAV 100 is descending (e.g., to a delivery location) or ascending (e.g., following a delivery). Stabilizers 108 (or fins) may be included with UAV 100 to stabilize the UAV's yaw (left or right turns) during flight. In some embodiments, UAV 100 may be configured to function as a glider. To do so, UAV 100 may power off its propulsion units and glide for a period of time.

[0028] During flight, UAV 100 may control the direction and/or speed of its movement by controlling its pitch, roll, yaw, and/or altitude. For example, the stabilizers 108 may include one or more rudders 122 for controlling the UAV's yaw, and wing assembly 102 may include elevators for controlling the UAV's pitch and/or ailerons 124 for controlling the UAV's roll. As another example, increasing or decreasing the speed of all the propellers simultaneously can result in UAV 100 increasing or decreasing its altitude, respectively. The UAV 100 may also include components for sensing the environment around the UAV 100, including but not limited to audio sensor 114 and audio sensor 120. Further examples of sensor devices are illustrated in FIG. 3 and described below.

[0029] Many variations on the illustrated fixed-wing aerial vehicle are possible. For instance, aerial vehicles with more wings (e.g., an “x-wing” configuration with four wings), are also possible. Although FIG. 1 and FIG. 2 illustrate one wing assembly 102, two boom assemblies 110, two horizontal
5 propulsion units 106, and six vertical propulsion units 112 per boom assembly 110, it should be appreciated that other variants of UAV 100 may be implemented with more or fewer of these components.

[0030] It should be understood that references herein to an “unmanned” aerial vehicle or UAV can apply equally to autonomous and semi-autonomous aerial
10 vehicles. In a fully autonomous implementation, all functionality of the aerial vehicle is automated; e.g., pre-programmed or controlled via real-time computer functionality that responds to input from various sensors and/or pre-determined information. In a semi-autonomous implementation, some functions of an aerial vehicle may be controlled by a human operator, while other functions are carried out
15 autonomously. Further, in some embodiments, a UAV may be configured to allow a remote operator to take over functions that can otherwise be controlled autonomously by the UAV. Yet further, a given type of function may be controlled remotely at one level of abstraction and performed autonomously at another level of abstraction. For example, a remote operator may control high level navigation decisions for a UAV,
20 such as specifying that the UAV should travel from one location to another (e.g., from a warehouse in a suburban area to a delivery address in a nearby city), while the UAV’s navigation system autonomously controls more fine-grained navigation decisions, such as the specific route to take between the two locations, specific flight controls to achieve the route and avoid obstacles while navigating the route, and so
25 on.

[0031] Some embodiments of the present disclosure are designed to work with the hybrid small aerial vehicle illustrated in FIG. 1 and FIG. 2. In the UAV 100, an array of 12 vertical propulsion units 112 provide thrust for hovering flight. Two horizontal propulsion units 106, two ailerons 124, and two rudders 122 are used
30 primarily for cruise flight. This hybrid configuration makes the task of building an accurate model of the system more challenging, as the aerodynamic interactions are more complex than on larger fixed-wing aircraft (e.g. rotor cross-flow, flow around small structures, etc.). As an alternative to pushing the boundary of computational

fluid dynamics tools, or performing complex and expensive measurement campaigns using wind tunnels, learning models from raw flight data as described below turns out to be surprisingly effective. Trained on among the largest scale real-world delivery drone data reported to date, the anomaly detection techniques described herein

5 successfully flag missions with disabled actuators, off-nominal hardware conditions, turbulence and other anomalous events. The approach described herein is based on a combination of non-parametric dynamics modeling and a novel algorithm for robust and scalable least trimmed squares estimation, which may be of independent interest.

[0032] FIG. 3 is a block diagram that illustrates a non-limiting example

10 embodiment of an autonomous vehicle 300 according to various aspects of the present disclosure. In some embodiments, the autonomous vehicle 300 is configured to collect telemetry data and transmit the collected telemetry data to an anomaly detection system. In some embodiments, the autonomous vehicle 300 is configured to receive commands from the anomaly detection system in the event of an anomaly

15 being detected, and to take appropriate action to address the anomaly. In some embodiments, the autonomous vehicle 300 is an aircraft. In other embodiments, any other type of autonomous vehicle 300 capable of navigating along a route, such as a wheeled vehicle, may be used. The UAV 100 illustrated in FIG. 1 and FIG. 2 is a non-limiting example embodiment of an autonomous vehicle 300. In some

20 embodiments, the autonomous vehicle 300 may be a different type of autonomous vehicle.

[0033] As shown, the autonomous vehicle 300 includes a communication interface 302, one or more vehicle state sensor device(s) 304, a power supply 306, one or more processor(s) 308, one or more propulsion device(s) 310, and a computer-

25 readable medium 312.

[0034] In some embodiments, the communication interface 302 includes hardware and software to enable any suitable communication technology for communicating with an anomaly detection system. In some embodiments, the communication interface 302 includes multiple communication interfaces, each for

30 use in appropriate circumstances. For example, the communication interface 302 may include a long-range wireless interface such as a 4G or LTE interface, or any other type of long-range wireless interface (e.g., 2G, 3G, 5G, or WiMAX), to be used to communicate with the anomaly detection system while traversing a route. The

communication interface 302 may also include a medium-range wireless interface such as a Wi-Fi interface to be used when the autonomous vehicle 300 is at an area near a start location or an endpoint where Wi-Fi coverage is available. The communication interface 302 may also include a short-range wireless interface such as a Bluetooth interface to be used when the autonomous vehicle 300 is in a maintenance location or is otherwise stationary and waiting to be assigned a route. The communication interface 302 may also include a wired interface, such as an Ethernet interface or a USB interface, which may also be used when the autonomous vehicle 300 is in a maintenance location or is otherwise stationary and waiting to be assigned a route. In some embodiments, the communication interface 302 may support transfer of a removable computer-readable medium between the autonomous vehicle 300 and the anomaly detection system to provide the transfer of information between the systems.

[0035] In some embodiments, the power supply 306 may be any suitable device or system for storing and/or generating power. Some non-limiting examples of a power supply 306 include one or more batteries, one or more solar panels, a fuel tank, and combinations thereof. In some embodiments, the propulsion device(s) 310 may include any suitable devices for causing the autonomous vehicle 300 to travel along the path. For an aircraft, the propulsion device(s) 310 may include devices such as, but not limited to, horizontal propulsion units 106, vertical propulsion units 112, and/or one or more flight control surfaces such as ailerons 124 and/or rudders 122. For a wheeled vehicle, the propulsion device(s) 310 may include devices such as, but not limited to, one or more motors, one or more wheels, and one or more steering mechanisms.

[0036] In some embodiments, the vehicle state sensor device(s) 304 are configured to detect states of various components of the autonomous vehicle 300, and to transmit signals representing those states to other components of the autonomous vehicle 300. Some non-limiting examples of vehicle state sensor device(s) 304 include a battery state sensor, a sensor that reports positions or states of propulsion device(s) 310, a sensor that reports the state of a servo for moving a control surface, an inertial sensor, an attitude sensor, a velocity sensor, and a positioning sensor (such as a global navigation satellite system (GNSS) sensor).

[0037] In some embodiments, the processor(s) 308 may include any type of computer processor capable of receiving signals from other components of the autonomous vehicle 300 and executing instructions stored on the computer-readable medium 312. In some embodiments, the computer-readable medium 312 may include
5 one or more devices capable of storing information for access by the processor(s) 308. In some embodiments, the computer-readable medium 312 may include one or more of a hard drive, a flash drive, an EEPROM, and combinations thereof.

[0038] As shown, the computer-readable medium 312 has stored thereon a telemetry data store 314, a telemetry collection engine 316, and a telemetry
10 communication engine 318. In some embodiments, the telemetry collection engine 316 is configured to receive information from the components of the autonomous vehicle 300, and to store the information in the telemetry data store 314. In some embodiments, the telemetry communication engine 318 is configured to transmit information from the telemetry data store 314 to an anomaly detection system. In
15 some embodiments, the telemetry communication engine 318 may also be configured to receive notifications of detected anomalies from the anomaly detection system, and may help control the autonomous vehicle 300 to respond to the detected anomalies.

[0039] In some embodiments, the autonomous vehicle 300 may include components that are not illustrated in FIG. 3, but that one would understand are
20 present. For example, the autonomous vehicle 300 may include one or more wired or wireless communication interfaces that allow the illustrated components of the autonomous vehicle 300 to communicate with each other, including but not limited to an Ethernet network, a USB network, a Bluetooth network, or a CANBUS network.

[0040] FIG. 4 - FIG. 6 include several charts that illustrate non-limiting
25 example embodiments of telemetry information collected by an autonomous vehicle according to various aspects of the present disclosure. The charts in FIG. 4 - FIG. 6 illustrate telemetry information collected by vehicle state sensor device(s) 304 of an autonomous vehicle 300. For each line in each chart, the telemetry information may be provided as a time series of values generated for the given characteristic. In some
30 embodiments, a group of multiple time series such as the illustrated time series may be collected to create a time series data record for a given time period.

[0041] On the top of FIG. 4, FIG. 5, and FIG. 6, charts are included that illustrate vertical propulsion unit 112 speed, horizontal propulsion unit 106 speed, and

control surface servo positions, respectively. The charts include separate lines for separate devices (e.g., the left chart for vertical propulsion unit 112 speed includes one line for each of four vertical propulsion units 112, the middle chart for horizontal propulsion unit 106 speed includes one line for each of two horizontal propulsion units 106, and the right chart for control surface servo position includes one line each for servos for a left rudder 122, a right rudder 122, a left aileron 124, and a right aileron 124). The propulsion unit speeds and servo positions may be reported by vehicle state sensor device(s) 304 coupled to the propulsion units and servos, by the propulsion units and servos themselves, or by any other suitable technique.

10 **[0042]** On the bottom of FIG. 4, FIG. 5, and FIG. 6, charts are included that illustrate linear velocity of the autonomous vehicle 300, attitude or orientation of the autonomous vehicle 300, and angular velocity of the autonomous vehicle 300, respectively. These values may be generated via any suitable technique, including but not limited to from a positioning sensor such as a GNSS sensor and/or a motion sensor
15 such as an inertial measurement unit (IMU).

[0043] FIG. 7 is a block diagram that illustrates a non-limiting example embodiment of an anomaly detection system according to various aspects of the present disclosure. The anomaly detection system 702 may be implemented by any suitable collection of one or more computing devices, each of which may be a desktop
20 computing device, a server computing device, a laptop computing device, a tablet computing device, a mobile computing device, a smartphone computing device, or a computing device in a cloud computing system. In some embodiments, the functionality of the anomaly detection system 702 may be separated between multiple computing devices. In some embodiments, some components illustrated as being
25 present in the anomaly detection system 702 may be present in the autonomous vehicle 300. For example, in some embodiments, the anomaly detection engine 710 may be present within the autonomous vehicle 300 instead of within a separate anomaly detection system 702.

[0044] As shown, the anomaly detection system 702 includes one or more
30 processor(s) 704, a communication interface 712, and a computer-readable medium 714.

[0045] In some embodiments, the processor(s) 704 In some embodiments, the processor(s) 704 may include one or more commercially available general-purpose

computer processors, each of which may include one or more processing cores. In some embodiments, the processor(s) 704 may also include one or more special-purpose computer processors, including but not limited to one or more processors adapted for efficiently performing machine learning tasks.

5 **[0046]** In some embodiments, the communication interface 712 provides any suitable communication technology for communicating with the communication interface 302 of the autonomous vehicle 300 as described above, including but not limited to wired technologies, wireless technologies, removable media technologies, and/or combinations thereof as described above.

10 **[0047]** As shown, the computer-readable medium 714 includes logic stored thereon that, in response to execution by the processor(s) 704, causes the computer-readable medium 714 to provide a telemetry collection engine 706, a model training engine 708, and an anomaly detection engine 710. In some embodiments, the telemetry collection engine 706 is configured to receive time series data records that
15 include telemetry information from autonomous vehicles and to store the time series data records in the telemetry data store 716. In some embodiments, the model training engine 708 is configured to use the time series data records stored in the telemetry data store 716 to train a machine learning model to determine vehicle flight dynamics, discard anomalous data from a training set, and to detect anomalies based on the
20 determined vehicle flight dynamics. In some embodiments, the anomaly detection engine 710 is configured to use the machine learning model trained by the model training engine 708. Further description of the actions performed by each of these components is provided below.

[0048] As used herein, "engine" refers to logic embodied in hardware or
25 software instructions, which can be written in a programming language, such as C, C++, COBOL, JAVA™, PHP, Perl, HTML, CSS, JavaScript, VBScript, ASPX, Microsoft .NET™, Go, Python, and/or the like. An engine may be compiled into executable programs or written in interpreted programming languages. Software engines may be callable from other engines or from themselves. Generally, the
30 engines described herein refer to logical modules that can be merged with other engines, or can be divided into sub-engines. The engines can be implemented by logic stored in any type of computer-readable medium or computer storage device and be stored on and executed by one or more general purpose computers, thus creating a

special purpose computer configured to provide the engine or the functionality thereof. The engines can be implemented by logic programmed into an application-specific integrated circuit (ASIC), a field-programmable gate array (FPGA), or another hardware device.

5 **[0049]** As shown, the anomaly detection system 702 also includes a telemetry data store 716 and a model data store 718. In some embodiments, the telemetry data store 716 stores time series data records that include telemetry information collected by the telemetry collection engine 706. In some embodiments, the model data store 718 stores machine learning model trained by the model training engine 708.

10 **[0050]** As used herein, “data store” refers to any suitable device configured to store data for access by a computing device. One example of a data store is a highly reliable, high-speed relational database management system (DBMS) executing on one or more computing devices and accessible over a high-speed network. Another example of a data store is a key-value store. However, any other suitable storage
15 technique and/or device capable of quickly and reliably providing the stored data in response to queries may be used, and the computing device may be accessible locally instead of over a network, or may be provided as a cloud-based service. A data store may also include data stored in an organized manner on a computer-readable storage medium, such as a hard disk drive, a flash memory, RAM, ROM, or any other type of
20 computer-readable storage medium. One of ordinary skill in the art will recognize that separate data stores described herein may be combined into a single data store, and/or a single data store described herein may be separated into multiple data stores, without departing from the scope of the present disclosure.

[0051] FIG. 8 and FIG. 9 are a flowchart that illustrates a non-limiting
25 example embodiment of a method of detecting anomalies in telemetry information according to various aspects of the present disclosure. The illustrated method 800 shows both training and using the machine learning model. However, in some embodiments, the method 800 might only train or might only use the machine learning model. Further, the method 800 as illustrated describes time series data records that
30 include telemetry information from trips or missions of a UAV 100. These should not be seen as limiting, and in some embodiments, any other type of information may be stored in time series data records and processed by the method 800 to train a machine learning model and/or detect anomalies using the machine learning model.

[0052] From a start block, the method 800 proceeds to block 802, where, for a plurality of autonomous vehicles, a telemetry collection engine 316 of each autonomous vehicle 300 receives telemetry information during a trip from one or more vehicle state sensor device(s) 304 of the autonomous vehicle 300. A trip (sometimes
5 referred to as a “flight” or “mission”) describes an action taken by the autonomous vehicle 300, and typically includes one or more of a takeoff portion, a package pickup portion, a cruise portion, a package delivery portion, and a landing portion.

[0053] At block 804, for each autonomous vehicle 300 of the plurality of autonomous vehicles, the telemetry collection engine 316 of each respective
10 autonomous vehicle 300 stores the telemetry information for the trip in a telemetry data store 314 of the autonomous vehicle 300. In some embodiments, the telemetry collection engine 316 may constantly store telemetry information in the telemetry data store 314 as it is generated by the vehicle state sensor device(s) 304. In some
15 embodiments, the telemetry collection engine 316 may collect the telemetry information in temporary storage and store it in the telemetry data store 314 once a trip is complete. In some embodiments, the telemetry collection engine 316 stores time series data records that include separate time series entries for each separate vehicle state sensor device in a single time series data record. In some embodiments, the telemetry collection engine 316 may store separate time series data records for
20 each separate vehicle state sensor device.

[0054] At block 806, for the plurality of autonomous vehicles, a telemetry communication engine 318 of each autonomous vehicle 300 transmits a time series data record for each trip in its telemetry data store 314 that includes the telemetry information for the trip to a telemetry collection engine 706 of an anomaly detection
25 system 702. In some embodiments, the autonomous vehicle 300 may transmit multiple time series data records for each trip, particularly if separate time series data records are stored for each vehicle state sensor device. In some embodiments, the telemetry communication engine 318 may continually transmit the time series data record to the telemetry collection engine 706 as long as the autonomous vehicle 300 is
30 communicatively connected to the anomaly detection system 702. In some embodiments, the telemetry communication engine 318 may transmit the time series data record once the trip is completed.

[0055] At block 808, the telemetry collection engine 706 of the anomaly detection system 702 stores the time series data records in a telemetry data store 716 of the anomaly detection system 702. By storing the time series data records in the telemetry data store 716, a large set of time series data records may be collected for model training purposes. It is worth nothing that the time series data records need not be tagged as normal or anomalous, as the remainder of the method 800 will automatically detect the anomalous time series data records in the training data and treat them appropriately.

[0056] At block 810, a model training engine 708 of the anomaly detection system 702 initializes a machine learning model having a set of fitting weights and a set of time series data record weights. The anomaly detection system 702 may be configured to train and use any suitable type of machine learning model for detecting anomalies. A description of a non-limiting example of a machine learning model suitable for use with anomaly detection system 702 that has a set of fitting weights and a set of time series data record weights follows.

[0057] Consider an autonomous vehicle interacting with its environment according to an unknown continuous-time nonlinear dynamical system,

$$\dot{x}(t) = f(x(t), u(t))$$

where states $x(t) \in \mathbb{R}^n$ and controls $u(t) \in \mathbb{R}^m$. Assume that a fleet of such autonomous vehicles collectively and continuously execute missions generating trajectory logs of the form,

$$\tau_i = \{(x_i(t), u_i(t), \dot{x}_i(t))\}_{t=0}^{T_i}$$

where i indexes the mission.

[0058] From N mission logs, one may naturally hope to learn f over a suitable family of function approximators $\mathcal{F}$ by solving a least squares problem,

$$f^* = \arg \min_{f \in \mathcal{F}} \sum_{i=1}^N r(\tau_i, f)$$

where r denotes the predictive residual,

$$r(\tau, f) = \frac{1}{T} \sum_{t=1}^T \|\dot{x}(t) - f(x(t), u(t))\|_2^2$$

[0059] While this is reminiscent of model-based Reinforcement Learning, our interest in the present disclosure is not to learn controllers, but rather to turn the dynamics estimate f^* into a detector that can flag mission abnormalities. For any trajectory τ generated by a new mission, the per time-step residual norm

5
$$\|\dot{x}(t) - f^*(x(t), u(t))\|_2^2$$

is a measure of “instantaneous unexpectedness” and the mean residual across time, $r(\tau, f^*)$, defines an anomaly score for that mission.

[0060] Chu et. al. adopt this approach for predicting linear and angular acceleration of the aircraft. By using linear and quadratic functions, a single pass over

10 the mission logs suffices for least squares estimation. In practice, such an approach to anomaly detection may become fragile in the face of the quality of real world data. When the set of training missions is contaminated with operational failures or carry subtle signatures of future catastrophes (e.g., sensor degradation), the detector may extract a misleading characterization of normal behavior. Unlike model-based RL

15 settings where all collected trajectories may be useful for learning the unknown dynamics, for anomaly detection the learning process has to simultaneously filter out missions for such abnormalities while fitting a model to the data that remains. In the absence of such a filtering mechanism, ordinary least squares estimators and associated anomaly detectors may degrade in quality due to the presence of highly

20 abnormal missions in the training set.

[0061] A measure of robustness of an estimator is the *finite-sample breakdown point* which in the present context is the fraction of mission trajectories that may be arbitrarily corrupted so as to cause the parameters of the estimator to blowup (i.e., become infinite). For least squares estimators or even least absolute

25 deviations (l_1) regressors, the finite sample breakdown point is $\frac{1}{N}$ making them fragile in the presence of heavy outliers in the training set. A more robust alternative is trimmed estimators. For any f , denote the order statistics of the residuals as,

$$r(\tau_{[1]}, f) \leq r(\tau_{[2]}, f) \leq \dots \leq r(\tau_{[N]}, f)$$

[0062] Then we define the *trimmed estimator* as the sum of the smallest k

30 residuals,

$$f^* = \arg \min_{f \in \mathcal{F}} \sum_{i=1}^k r(\tau_{[i]}, f)$$

[0063] The breakdown point of such an estimator is

$$\frac{N - k + 1}{N}$$

where k is the number of missions that should not be trimmed. In practice, k is unknown and is treated as a hyper-parameter. By making k small enough, the
5 breakdown point can even be made larger than 50%.

[0064] The price of strong robustness is computational complexity of least trimmed squares estimation: for an exact solution, the complexity scales as

$$O(N^{d+1})$$

for $d \geq 3$ dimensional regression problems. The optimization task is both non-
10 smooth and non-convex. Due to its combinatorial flavor, it is not amenable to standard gradient techniques or least squares solvers even for linear models. Thus, the development of practical approximate algorithms is of significant interest. The present disclosure provides a novel algorithm for robust learning based on smoothing the trimmed squares loss. The algorithm is inspired by Nesterov's smoothing
15 procedure for minimizing non-smooth objective functions, and is also closely related to Deterministic Annealing methods for combinatorial optimization.

[0065] Consider the function that maps a vector $r \in \mathbb{R}^N$ to the sum of its k smallest elements,

$$h_k(r) = \sum_{i=1}^k r_{[i]}, \quad \text{where } r_{[1]} \leq r_{[2]} \leq \dots \leq r_{[N]}$$

20 [0066] This function admits a smoothing defined as follows,

$$\begin{aligned} h_k^T(r) &= \min_{\alpha \in \mathbb{R}^N} \alpha^T r + T \sum_{i=1}^N H(\alpha_i) \\ \text{s. t. : } &\sum_{i=1}^N \alpha_i = k, \quad 0 \leq \alpha_i \leq 1 \end{aligned}$$

$$\text{where } H(u) = u \log(u) + (1 - u) \log(1 - u)$$

[0067] Above, T is a smoothing parameter also referred to as the
25 "temperature" in the annealing literature. Intuitively, if α_i tends to zero, the corresponding mission is considered too anomalous for training and is trimmed away.

The α 's may also be interpreted as probability distributions over binary indicator variables encoding whether or not to trim a mission. As such, when T is high, the smoothed objective is dominated by the entropy of the α 's and tend to approach the uniform distribution

$$5 \quad \forall i: \alpha_i = \frac{k}{N}$$

[0068] As $T \rightarrow 0$, the weights harden towards binary values. This strategy of starting with a highly smoothed proxy to a non-convex non-smooth function and gradually increasing the degree of convexity is the central idea of homotopy, continuation and graduated non-convexity methods for global optimization. In the
10 ideal case, the highly smoothed function is close to being convex allowing the global minimum to be found efficiently. As smoothing is reduced, one hopes that following the continuous path of the minimizer would lead to the global minimum.

[0069] Spurious local minima can be eliminated due to smoothing, making the optimization task much easier. In particular, the smoothing discussed above has
15 the following properties:

- h_k^T is a concave function.
- h_k^T is continuously differentiable.
- $h_k^T(r) - TR \leq h_k(r) \leq h_k^T(r)$ holds for some fixed constant R .

[0070] With this smoothing of the trimmed loss, for a fixed number of k
20 missions to retain, we consider the following optimization problem,

$$f^* = \arg \min_{f \in \mathcal{F}} \sum_{i=1}^k h_k^T(r(f)), r(f) = [r(\tau_1, f) \dots r(\tau_N, f)]$$

[0071] Equivalently,

$$f^* = \arg \min_{f \in \mathcal{F}, \alpha \in \mathbb{R}^N} \sum_{i=1}^N \alpha_i r(\tau_i, f) + T H(\alpha_i)$$

such that:

$$25 \quad \sum_{i=1}^N \alpha_i = k, \quad 0 \leq \alpha_i \leq 1$$

[0072] The method 800 then advances to a continuation terminal ("terminal A"). From terminal A (FIG. 9), the method 800 proceeds to block 902 and block 904. These blocks represent a fitting phase and a trimming phase, and the method 800 alternates between them until convergence. Both of these phases are fast, efficient, and easily scale to thousands of missions and millions of measurements. The optimization may be initialized with

$$\alpha = \frac{k}{N}$$

which corresponds to the non-robust least squares estimator and the limit of $T \rightarrow \infty$. This may be part of the initialization of the machine learning model performed in block 810.

[0073] At block 902, the model training engine 708 optimizes the set of fitting weights while holding the set of time series data record weights constant. In some embodiments, this optimization may be performed a predetermined number of times before advancing to block 904. In some embodiments, this optimization may be performed until a loss function that measures performance converges to a minimum value before advancing to block 904.

[0074] In designing the machine learning model, we consider linear combinations of fixed nonlinear basis functions,

$$f(x, u) = W\phi(x, u)$$

where

$$\phi: \mathbb{R}^{n+m} \mapsto \mathbb{R}^d$$

is a nonlinear feature map and W is a $n \times d$ parameter matrix.

[0075] For fixed α 's, optimizing W is a weighted least squares problem which admits a fast single pass solution,

$$W = [A + \lambda I_d]^{-1} B$$

where:

$$A = \sum_{i=1}^N \alpha_i \sum_t \phi(x_i(t), u_i(t)) \phi(x_i(t), u_i(t))^T$$

$$B = \sum_{i=1}^N \alpha_i \sum_t \phi(x_i(t), u_i(t)) \dot{x}_i(t)^T$$

[0076] At block 904, the model training engine 708 optimizes the set of time series data record weights while holding the set of fitting weights constant. In some embodiments, this optimization may be performed a predetermined number of times
 5 before advancing to decision block 906. In some embodiments, this optimization may be performed until a loss function that measures performance converges to a minimum value before advancing to decision block 906.

[0077] For fixed W , we compute the vector of N residuals given by $r_i = r(\tau_i, W)$. The α optimization takes the form:

$$10 \quad \alpha_i = \frac{1}{1 + \exp\left(\frac{r_i - v}{T}\right)}$$

[0078] where the scalar v satisfies the nonlinear equation,

$$\psi(v) = \sum_{i=1}^N \frac{1}{1 + \exp\left(\frac{r_i - v}{T}\right)} - k = 0$$

[0079] The root of this equation can be easily solved e.g., via the bisection method noting that $\psi(a) < 0$ for

$$15 \quad a = \min_i r_i - T \log \frac{N - k}{N}$$

and $\psi(b) > 0$ for

$$b = \max_i r_i - T \log \frac{N - k}{N}$$

provides an initial bracketing of the root.

[0080] We experimented with both linear models as well as nonlinear random
 20 basis functions of the form,

$$\phi(x, u) = \sqrt{\frac{2}{d}} \cos(\sigma^{-1} Gx + \sigma^{-1} Hu + b)$$

where

$$G_{ij}, H_{ij} \sim \mathcal{N}(0, 1), b \sim \mathcal{U}(0, 2\pi)$$

and

$$G \in \mathbb{R}^{d \times n}, H \in \mathbb{R}^{d \times m}$$

[0081] Here, the feature map dimensionality d controls the capacity of the dynamics model. In particular, as $d \rightarrow \infty$, inner products in the random feature space
 5 approximate the Gaussian Kernel,

$$\phi(x, u)^T \phi(\bar{x}, \bar{u}) \approx e^{-\frac{\|x - \bar{x}\|_2^2 + \|u - \bar{u}\|_2^2}{2\sigma^2}}$$

[0082] The implication of this approximation is that each component of the learnt dynamics function is a linear combination of similarity to training mission measurements, in the following sense.

$$10 \quad f_j(x, u) = w_j^T \phi(x, u) \approx \sum_{i=1}^N \sum_t \beta_{j,i,t} e^{-\frac{\|x - x_i(t)\|_2^2 + \|u - u_i(t)\|_2^2}{2\sigma^2}}$$

for some coefficients $\beta_{j,i,t}$, and where w_j^T refers to the j -th row of W .

[0083] The random feature method scales linearly in the number of measurements, as opposed to cubically for β when working with exact kernels. At the price of losing this linear training complexity and globally optimal solution, one may
 15 also embrace deep networks for this application to parameterize the dynamics model.

[0084] The method 800 then proceeds to decision block 906, where a determination is made regarding whether optimization of the machine learning model is complete. In some embodiments, the determination may be based on whether a loss function that measures performance of the machine learning model has converged to a
 20 minimum value. In some embodiments, the determination may be made based on whether a predetermined number of iterations has been completed.

[0085] If optimization of the machine learning model is not yet complete, then the result of decision block 906 is NO, and the method 800 returns to block 902 to further iterate on the weights. Otherwise, if optimization of the machine learning
 25 model is complete, then the result of decision block 906 is YES, and the method 800 advances to block 908.

[0086] At block 908, the model training engine 708 determines an anomaly threshold value for the machine learning model and stores the machine learning model and the anomaly threshold value in a model data store 718 of the anomaly detection

system 702. The machine learning model is stored in the model data store 718 so that it can be distributed to other devices to perform anomaly detection without having to newly train a machine learning model. In some embodiments, the anomaly threshold value may be determined by trading off precision and recall. The receiver operating
5 characteristic (ROC) curve may be plotted for the training data, and a desired true positive rate may be used to determine the anomaly threshold value. Having a high true positive rate generally comes at the expense of a higher false negative rate (false alarms), and a good regime is determined using the ROC curve.

[0087] At block 910, an anomaly detection engine 710 of the anomaly
10 detection system 702 loads the machine learning model from the model data store 718. In some embodiments, the anomaly detection system 702 may be a different anomaly detection system 702 than an anomaly detection system 702 that originally trained the machine learning model and stored the machine learning model in the model data store 718.

15 [0088] At block 912, the telemetry collection engine 706 of the anomaly detection system 702 receives a new time series data record containing telemetry information from an autonomous vehicle 300. In some embodiments, the new time series data record may represent an entire trip, or may represent a portion of a trip. In some embodiments, the new time series data record may be transmitted during the trip
20 by the autonomous vehicle 300 so that anomalies can be detected in real-time.

[0089] At block 914, the anomaly detection engine 710 processes the new time series data record using the machine learning model. In some embodiments, the machine learning model takes the new time series data record as input and outputs an anomaly score, which is compared to the anomaly threshold value.

25 [0090] The method 800 then advances to a decision block 916, where a determination is made regarding whether an anomaly has been detected. In some embodiments, the anomaly is detected when the anomaly score is greater than or equal to the anomaly threshold value.

[0091] If an anomaly has not been detected, then the method 800 proceeds to
30 an end block and terminates. If an anomaly has been detected, then the result of decision block 916 is YES, and the method 800 advances to block 918, where the anomaly detection engine 710 transmits a command to address the anomaly to the

autonomous vehicle 300. Any suitable command may be transmitted. For example, in some embodiments, the anomaly detection engine 710 may determine an action to be taken in response to the anomaly, and may transmit a command to the autonomous vehicle 300 to cause the autonomous vehicle 300 to perform the action to address the anomaly. In some embodiments, the action may be at least one of rescheduling a future trip, navigating to an emergency repair location, and immediately performing a landing procedure. In some embodiments, the action may include accepting remote control from a human operator to address the anomaly. The method 800 then proceeds to an end block and terminates. The method 800 is shown as terminating at this point for the sake of clarity. In some embodiments, the method 800 may instead return to block 914 to check further time series data records for anomalies before terminating.

[0092] We generated synthetic 8-dimensional input and 3-dimensional output time series following a linear model as follows. The output time series for normal missions carry moderate Gaussian noise, but anomalous missions are heavily corrupted by non-Gaussian noise sampled uniformly from the interval [0,10]. 200 training and 200 test missions, each with 100 time steps were generated with 50% anomalies following the procedure below. The anomaly labels were discarded for training and only used for evaluation.

$$\begin{aligned}
 x(t) &\in \mathbb{R}^8 = \cos(g * t + b), \quad g, b \in \mathbb{R}^8, g_i, b_i \sim \mathcal{N}(0,1) \\
 y(t) &\in \mathbb{R}^3 = Wx(t) + \epsilon(t), \quad W \in \mathbb{R}^{3 \times 8} \\
 \epsilon(t) &\in \mathbb{R}^3 := \begin{cases} \epsilon_i \sim \mathcal{N}(0,1) & \text{for normal} \\ \epsilon_i \sim \text{Unif}(0,10) & \text{for anomaly} \end{cases}
 \end{aligned}$$

[0093] FIG. 10 includes two charts that illustrate how alternating optimization of the smoothed trimmed loss (with temperature $T=1.0$) leads to monotonic descent in the sum of the smallest $k = 100$ residuals, a consequence of our smoothing formulation. The optimization converges to a set of mission weights (α 's) that clearly trim away nearly all the anomalies present in the training set despite heavy 50% corruption and no explicit anomaly labels provided to the algorithm.

[0094] FIG. 11 is a chart that illustrates a comparison of anomaly detectors based on the approach disclosed herein against pure least-squares models and several M -estimators proposed in the Robust Statistics literature. Due to its non-robustness, the least-squares detector is hurt the most due to corruptions in the training set. By

limiting outlier performance, robust loss functions such as l_1 and *huber* show improved performance. Yet, they are outperformed by the proposed trimming approach which gives perfect detection despite massive corruption of the training data.

[0095] FIG. 12 - FIG. 14 include several charts that illustrate experimental results of using a machine learning model to detect anomalies according to various aspects of the present disclosure. The results relate to data collected from a fleet of delivery drones flying in multiple environments on real delivery flight missions. A typical mission consists of a takeoff, package pickup, a cruise phase to deliver the package, and subsequent landing. To the best of our knowledge, machine learning on real delivery drone data at this scale is unprecedented: 5000 historical missions prior to a cutoff date generating around 80 million measurements are used for training.

[0096] Trained detectors were tested on 5000 outdoor missions after the cutoff date. By contrast, recent papers have reported results on 20 to 50 test missions in controlled lab environments. Our large-scale flight log data covers multiple vehicle types, package configurations and variable temperature and wind conditions. Additionally, the mission logs are mixed with a variety of research and development flights that include flight envelope expansion, prototype hardware and software, and other experiments designed to stress-test the flight system. Flight missions generally last approximately 5 minutes including several kilometers of cruise flight.

[0097] FIG. 4 - FIG. 6, discussed above, show examples for the input signals used for training models to predict linear and angular acceleration of the vehicle. Each input time series is re-scaled so that values lie in the interval $[-1.0, 1.0]$. Training a nonlinear trimmed model with $d = 100$ random Fourier features on 80 million measurements, including data preprocessing, is completed within 1.15 hours on a single CPU. FIG. 12 - FIG. 14 each illustrate prediction results from a normal flight (top) and an anomalous flight (bottom). The predictions show a large spike towards the end of the anomalous test mission. This large spike causes the mean residual error to be large, flagging the flight as anomalous.

[0098] The vehicle's position, velocity, and attitude estimates from an EKF-based state estimator are compared with commands generated by a high-level mission planning system. The controller generates actuator commands to reduce errors between the state estimate and commands. The controller incorporates a real-time

airspeed estimate to properly allocate control between individual hover motors and aerodynamic control surfaces throughout the airspeed envelope.

[0099] FIG. 15 and FIG. 16 include charts that illustrate the performance of various detectors on the test set of 5000 missions. In FIG. 15, from top to bottom, the charts illustrate a linear model and a linear + trimming model. In FIG. 16, from top to bottom, the charts illustrate a non-linear model and a non-linear plus trimming model.

[0100] We report detection rates for multiple anomaly types:

- **are-basic-stats-exceeded**: Basic statistical measures such as velocity command error, error from commanded path, pitch, roll, Root Mean Squared pitch and roll error, pitch and roll torque commands are more than 3 standard deviations from the mean computed over the entire training set.
- **has-flight-dynamics-issue**: The particular flight had an issue where the flight dynamics were off-nominal due to various factors such as an intentionally disabled actuator or other off-nominal airframe modifications to test system robustness.
- **is-high-wind**: The prevailing wind speed is greater than 10m/s, which qualitatively indicates elevated levels of turbulence.

[0101] Approximately 12% of the test set of 5000 missions has these anomalies.

[0102] For the non-linear trimmed detectors ($d = 100$, $T = 1.0$, $k = 0.75N$) on a test set of 5000 missions, the area under the True-Positive-Rate vs. False-Positive-Rate curve exceeds 0.90. The detector coverage goes beyond simple statistical anomaly measures firing reliably across a multitude of factors such as disabled actuators, otherwise off-nominal hardware conditions and the vehicle experiencing turbulent conditions.

[0103] FIG. 17 is a chart that illustrates a smoothed distribution of mission weights learned by an experimental example embodiment of the trimming technique. The distribution of α 's peaks close to 0 in comparison to the mean over the entire training set which is close to 1.0. This confirms the ability of the proposed method to successfully filter out anomalies from the training set, in order to extract normal flight patterns, without requiring any form of supervision. Without nonlinear modeling and

trimming, we observed performance degradation in an analysis across finer anomaly type categories.

[0104] FIG. 18 is a block diagram that illustrates aspects of an exemplary computing device 1800 appropriate for use as a computing device of the present disclosure. While multiple different types of computing devices were discussed above, the exemplary computing device 1800 describes various elements that are common to many different types of computing devices. While FIG. 18 is described with reference to a computing device that is implemented as a device on a network, the description below is applicable to servers, personal computers, mobile phones, smart phones, tablet computers, embedded computing devices, and other devices that may be used to implement portions of embodiments of the present disclosure. Some embodiments of a computing device may be implemented in or may include an application-specific integrated circuit (ASIC), a field-programmable gate array (FPGA), or other customized device. Moreover, those of ordinary skill in the art and others will recognize that the computing device 1800 may be any one of any number of currently available or yet to be developed devices.

[0105] In its most basic configuration, the computing device 1800 includes at least one processor 1802 and a system memory 1804 connected by a communication bus 1806. Depending on the exact configuration and type of device, the system memory 1804 may be volatile or nonvolatile memory, such as read only memory ("ROM"), random access memory ("RAM"), EEPROM, flash memory, or similar memory technology. Those of ordinary skill in the art and others will recognize that system memory 1804 typically stores data and/or program modules that are immediately accessible to and/or currently being operated on by the processor 1802. In this regard, the processor 1802 may serve as a computational center of the computing device 1800 by supporting the execution of instructions.

[0106] As further illustrated in FIG. 18, the computing device 1800 may include a network interface 1810 comprising one or more components for communicating with other devices over a network. Embodiments of the present disclosure may access basic services that utilize the network interface 1810 to perform communications using common network protocols. The network interface 1810 may also include a wireless network interface configured to communicate via one or more wireless communication protocols, such as Wi-Fi, 2G, 3G, LTE, WiMAX, Bluetooth,

Bluetooth low energy, and/or the like. As will be appreciated by one of ordinary skill in the art, the network interface 1810 illustrated in FIG. 18 may represent one or more wireless interfaces or physical communication interfaces described and illustrated above with respect to particular components of the computing device 1800.

- 5 **[0107]** In the exemplary embodiment depicted in FIG. 18, the computing device 1800 also includes a storage medium 1808. However, services may be accessed using a computing device that does not include means for persisting data to a local storage medium. Therefore, the storage medium 1808 depicted in FIG. 18 is represented with a dashed line to indicate that the storage medium 1808 is
- 10 optional. In any event, the storage medium 1808 may be volatile or nonvolatile, removable or nonremovable, implemented using any technology capable of storing information such as, but not limited to, a hard drive, solid state drive, CD ROM, DVD, or other disk storage, magnetic cassettes, magnetic tape, magnetic disk storage, and/or the like.
- 15 **[0108]** Suitable implementations of computing devices that include a processor 1802, system memory 1804, communication bus 1806, storage medium 1808, and network interface 1810 are known and commercially available. For ease of illustration and because it is not important for an understanding of the claimed subject matter, FIG. 18 does not show some of the typical components of many computing
- 20 devices. In this regard, the computing device 1800 may include input devices, such as a keyboard, keypad, mouse, microphone, touch input device, touch screen, tablet, and/or the like. Such input devices may be coupled to the computing device 1800 by wired or wireless connections including RF, infrared, serial, parallel, Bluetooth, Bluetooth low energy, USB, or other suitable connections protocols using wireless or
- 25 physical connections. Similarly, the computing device 1800 may also include output devices such as a display, speakers, printer, etc. Since these devices are well known in the art, they are not illustrated or described further herein.

- [0109]** In the preceding description, numerous specific details are set forth to provide a thorough understanding of various embodiments of the present
- 30 disclosure. One skilled in the relevant art will recognize, however, that the techniques described herein can be practiced without one or more of the specific details, or with other methods, components, materials, etc. In other instances, well-known structures,

materials, or operations are not shown or described in detail to avoid obscuring certain aspects.

5 **[0110]** Reference throughout this specification to “one embodiment” or “an embodiment” means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the present invention. Thus, the appearances of the phrases “in one embodiment” or “in an embodiment” in various places throughout this specification are not necessarily all referring to the same embodiment. Furthermore, the particular features, structures, or characteristics may be combined in any suitable manner in one or more embodiments.

10 **[0111]** The order in which some or all of the blocks appear in each method flowchart should not be deemed limiting. Rather, one of ordinary skill in the art having the benefit of the present disclosure will understand that actions associated with some of the blocks may be executed in a variety of orders not illustrated, or even in parallel.

15 **[0112]** The processes explained above are described in terms of computer software and hardware. The techniques described may constitute machine-executable instructions embodied within a tangible or non-transitory machine (e.g., computer) readable storage medium, that when executed by a machine will cause the machine to perform the operations described. Additionally, the processes may be embodied
20 within hardware, such as an application specific integrated circuit (“ASIC”) or otherwise.

[0113] The above description of illustrated embodiments of the invention, including what is described in the Abstract, is not intended to be exhaustive or to limit the invention to the precise forms disclosed. While specific embodiments of, and
25 examples for, the invention are described herein for illustrative purposes, various modifications are possible within the scope of the invention, as those skilled in the relevant art will recognize.

[0114] These modifications can be made to the invention in light of the above detailed description. The terms used in the following claims should not be construed to
30 limit the invention to the specific embodiments disclosed in the specification. Rather, the scope of the invention is to be determined entirely by the following claims, which are to be construed in accordance with established doctrines of claim interpretation.

CLAIMS

What is claimed is:

1. A non-transitory computer-readable medium having logic stored thereon that, in response to execution by one or more processors of a computing system, cause the computing system to perform actions for detecting anomalies in time series data records, the actions comprising:

receiving, by the computing system, a plurality of time series data records;

initializing, by the computing system, a machine learning model having a set of fitting weights and a set of time series data record weights;

optimizing, by the computing system, the set of fitting weights of the machine learning model while holding the set of trip weights of the machine learning model constant;

optimizing, by the computing system, the set of time series data record weights while holding the set of fitting weights constant; and

storing, by the computing system, the optimized time series data record weights and the optimized fitting weights for use in monitoring for anomalies.

2. The non-transitory computer-readable medium of claim 1, wherein the actions further comprise determining an anomaly threshold value.

3. The non-transitory computer-readable medium of claim 1, wherein the plurality of time series data records include telemetry information from one or more autonomous vehicles.

4. The non-transitory computer-readable medium of claim 3, wherein the one or more autonomous vehicles includes at least one unmanned aerial vehicle (UAV).

5. The non-transitory computer-readable medium of claim 4, wherein the UAV includes at least one horizontal propulsion unit and at least one vertical propulsion unit.

6. The non-transitory computer-readable medium of claim 4, wherein the telemetry information includes information from at least two of a takeoff portion of a trip, a cruise portion of the trip, and a landing portion of the trip.

7. The non-transitory computer-readable medium of claim 4, wherein the time series data records include data that represents at least one of a motor speed, a control surface servo position, a linear velocity, an orientation, and an angular velocity.

8. The non-transitory computer-readable medium of claim 3, wherein the actions further comprise:

receiving a new time series data record from an autonomous vehicle; and

processing the new time series data record with the machine learning model to detect anomalies.

9. The non-transitory computer-readable medium of claim 8, wherein the actions further comprise, in response to detecting an anomaly:

determining an action to be taken to address the anomaly; and

transmitting a command to the autonomous vehicle to cause the autonomous vehicle to perform the action to address the anomaly.

10. The non-transitory computer-readable medium of claim 9, wherein the action to address the anomaly is at least one of rescheduling a future trip, navigating to an emergency repair location, immediately performing a landing procedure, and accepting remote control by a human operator.

11. A method for detecting anomalies in time series data records, the method comprising:

receiving, by a computing device, a time series data record from a monitored system;

processing, by the computing device, the time series data record using a machine learning model to generate an anomaly score, wherein the machine learning model was trained on a plurality of previous time series data records;

comparing, by the computing device, the anomaly score to an anomaly threshold value; and

in response to determining that the anomaly score is greater than the anomaly threshold value:

determining, by the computing device, an action to be taken to address the anomaly; and

transmitting, by the computing device, a command to the monitored system to cause the monitored system to perform the action to address the anomaly.

12. The method of claim 11, wherein the monitored system is an autonomous vehicle.

13. The method of claim 12, wherein the autonomous vehicle is an unmanned aerial vehicle (UAV).

14. The method of claim 13, wherein the UAV includes at least one horizontal propulsion unit and at least one vertical propulsion unit.

15. The method of claim 13, wherein the time series data record includes telemetry information from at least two of a takeoff portion of a trip, a cruise portion of the trip, and a landing portion of the trip.

16. The method of claim 13, wherein the time series data record includes data representing at least one of a motor speed, a control surface servo position, a linear velocity, an orientation, and an angular velocity.

17. The method of claim 13, wherein the action is at least one of rescheduling a future trip, navigating to an emergency repair location, and immediately performing a landing procedure.

18. The method of claim 11, further comprising training the machine learning model by performing actions including:

collecting a plurality of time series data records;

initializing the machine learning model to have a set of fitting weights and a set of time series data record weights;

optimizing the set of fitting weights while holding the set of time series data record weights constant; and

optimizing the set of time series data record weights while holding the set of fitting weights constant.

19. The method of claim 18, wherein training the machine learning model further includes determining the anomaly threshold value using the plurality of time series data records.

20. A system, comprising at least one computing device that includes:

at least one processor; and

a non-transitory computer-readable medium having logic stored thereon that, in response to execution by the at least one processor, causes the system to perform actions comprising:

receiving a time series data record from a monitored system;

processing the time series data record using a machine learning model to generate an anomaly score, wherein the machine learning model was trained on a plurality of previous time series data records;

comparing the anomaly score to an anomaly threshold value; and

in response to determining that the anomaly score is greater than the anomaly threshold value:

determining an action to be taken to address the anomaly; and

transmitting a command to the monitored system to cause the monitored system to perform the action to address the anomaly.