

COMPUTER-IMPLEMENTED METHOD AND NON-TRANSITORY COMPUTER- READABLE MEMORY FOR OBJECT LOCALISATION

Field of the Invention

The present invention relates to the field of object localisation and more
5 particularly to a computer-implemented method and a non-transitory computer-readable
memory for object localisation.

Background of the Invention

Classical extended Kalman filter-inertial measurement unit (EKF-IMU) fusion
frameworks include body-frame and navigation-frame accelerometer, gyroscope and/or
10 magnetometer measurements as part of the EKF state to be estimated. This results in
an inherent coupling of individual sensors needed for position or velocity estimation of
an intended target. Also, the entire navigation frame is considered to be non-rigid,
allowing for dynamic updates to maintain true heading and speed information.

A disadvantage of the classical EKF-IMU fusion framework arises from the
15 coupling of individual IMU sensors. As a result of the coupling, when one of the sensors
is faulty or has lower performance accuracy than any of the other sensors, the overall
estimation is affected because of the coupled nature of the state matrix in such a
framework.

Another disadvantage of classical EKF-IMU implementations arises from use of
20 a non-rigid navigation frame with continuous rotation and translation in every state. This
poses a problem because an accelerometer cannot distinguish between accelerations
from a body being tracked by an inertial measurement unit (IMU) and gravity.
Consequently, with gravitational changes, small errors are accumulated in the
accelerometer and these are then translated into incorrect rotation and translation
25 measurements of the navigation frame. The errors are propagated over time, resulting
in system divergence. These errors become even more conspicuous when consumer-
grade IMUs are employed.

Although use of the IMU for outdoor localization is known, magnetometer
measurements from the IMU, which are often used for orientation or heading

observations, are unusable for indoor localization due to ferro-magnetic interference from objects in indoor spaces.

Furthermore, commercial IMU fusion solutions are currently applicable to localization at metre-level accuracy. Even in the field of autonomous vehicles, industrial-grade IMUs are required for higher stability and performance for lane-level accuracy of between 1 to 3 metres.

With emerging ultra-wideband (UWB) markets and demand for UWB solutions, integrating consumer-grade IMUs for non-line-of-sight (NLOS) or outage scenarios is becoming increasingly important.

There is therefore a need for a reliable computer-implemented method and a non-transitory computer-readable memory for object localisation that is able to utilize noisy and poor quality IMU sensor data, whilst maintaining centimetre-level positioning accuracy.

Summary of the Invention

Accordingly, in a first aspect, the present invention provides a computer-implemented method for object localisation. The method includes executing on one or more processors the steps of: predicting a state vector x_k of an object at time k using Equation (1):

$$x_k = F_k x_{k-1} \quad (1)$$

where F_k represents a state-transition matrix, and x_{k-1} represents the state vector at time $k - 1$; predicting a covariance matrix P_k of the state vector x_k of the object using Equation (2):

$$P_k = F_k P_{k-1} F_k^T + Q_k \quad (2)$$

where P_{k-1} represents an estimated covariance matrix at time $k - 1$, F_k^T represents a transpose of the state-transition matrix F_k , and Q_k represents a process noise

covariance; receiving inertial measurement unit (IMU) data; receiving ultra-wideband (UWB) data; determining observation residual y_k using Equation (3):

$$y_k = z_k - H_k x_k \quad (3)$$

where z_k represents an observation of a true state of the object based on the IMU data
 5 and the UWB data, and H_k represents a sensor observation matrix; determining an optimal Kalman gain using Equation (4):

$$K_k = P_k H_k^T (H_k P_k H_k^T + R_k) \quad (4)$$

where H_k^T represents a transpose of the sensor observation matrix H_k , and R_k
 represents an observation or sensor noise covariance; and updating the predicted state
 10 vector x_k using Equation (5) to generate a localization solution L_{est} :

$$x_k = x_k + K_k y_k \quad (5).$$

In a second aspect, the present invention provides a non-transitory computer-readable memory storing computer program instructions executable by a computer processor to perform operations for object localisation. The operations include:
 15 predicting a state vector x_k of an object at time k using Equation (1):

$$x_k = F_k x_{k-1} \quad (1)$$

where F_k represents a state-transition matrix, and x_{k-1} represents the state vector at time $k - 1$; predicting a covariance matrix P_k of the state vector x_k of the object using Equation (2):

$$P_k = F_k P_{k-1} F_k^T + Q_k \quad (2)$$

where P_{k-1} represents an estimated covariance matrix at time $k - 1$, F_k^T represents a transpose of the state-transition matrix F_k , and Q_k represents a process noise

covariance; receiving inertial measurement unit (IMU) data; receiving ultra-wideband (UWB) data; determining observation residual y_k using Equation (3):

$$y_k = z_k - H_k x_k \quad (3)$$

where z_k represents an observation of a true state of the object based on the IMU data
 5 and the UWB data, and H_k represents an observation model; determining an optimal Kalman gain using Equation (4):

$$K_k = P_k H_k^T (H_k P_k H_k^T + R_k) \quad (4)$$

where H_k^T represents a transpose of the sensor observation matrix H_k , and R_k
 represents an observation or sensor noise covariance; and updating the predicted state
 10 vector x_k using Equation (5) to generate a localization solution L_{est} :

$$x_k = x_k + K_k y_k \quad (5).$$

Other aspects and advantages of the invention will become apparent from the following detailed description, taken in conjunction with the accompanying drawings, illustrating by way of example the principles of the invention.

15 Brief Description of the Drawings

Embodiments of the invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

FIG. 1 is a schematic flow diagram illustrating a computer-implemented method for object localisation in accordance with an embodiment of the present invention;

20 FIG. 2A is a boxplot of estimation errors of a calibrated accelerometer of a consumer-grade inertial measurement unit (IMU);

FIG. 2B is a boxplot of estimation errors of a calibrated gyroscope of the consumer-grade IMU;

FIG. 3 is a schematic block diagram illustrating a computer system suitable for implementing the method for object localisation disclosed herein;

FIGS. 4A is a graph of indoor localisation estimates from a classical extended Kalman filter-inertial measurement unit (EKF-IMU) fusion framework;

5 FIG. 4B is a graph of indoor localisation estimates from an auxiliary EKF-IMU fusion framework in accordance with an embodiment of the present invention;

FIG. 5A is a graph of tracked extended Kalman filter (EKF) velocities obtained from a classical EKF-IMU fusion framework;

10 FIG. 5B is a graph of tracked EKF velocities obtained from an auxiliary EKF-IMU fusion framework in accordance with an embodiment of the present invention;

FIG. 6 is a photograph of a non-line-of-sight (NLOS) simulated environment using foil blockage;

FIG. 7A is a graph of indoor localisation estimates from a classical EKF-IMU fusion framework in the NLOS simulated environment of FIG. 6; and

15 FIG. 7B is a graph of indoor localisation estimates from an auxiliary EKF-IMU fusion framework in accordance with an embodiment of the present invention in the NLOS simulated environment of FIG. 6.

Detailed Description of Exemplary Embodiments

20 The detailed description set forth below in connection with the appended drawings is intended as a description of presently preferred embodiments of the invention, and is not intended to represent the only forms in which the present invention may be practiced. It is to be understood that the same or equivalent functions may be accomplished by different embodiments that are intended to be encompassed within the scope of the invention.

25 Referring now to FIG. 1, a computer-implemented method 10 for object localisation is shown. More particularly, a flowchart of sensor fusion and update is

shown. The method 10 for object localisation may be executed on one or more processors.

The method 10 begins at step 12 by predicting a state vector x_k of an object at time k using Equation (1):

$$x_k = F_k x_{k-1} \quad (1)$$

where F_k represents a state-transition matrix, and x_{k-1} represents the state vector at time $k - 1$. The state-transition matrix F_k may be a 9-state motion model. The state vector x_k may be defined by Equation (6):

$$x_k = [p_k, v_k, hd_k, a_k] \quad (6)$$

where p_k represents a three-dimensional (3D) position of the object at time k , v_k represents velocity states of the object at time k , hd_k represents an absolute navigational heading of the object in a single dimension (1D) at time k , and a_k represents an acceleration of the object at time k .

Referring now to FIGS. 2A and 2B, stationary android IMU data from a Huawei-P30 phone placed on a flat surface was collected and boxplots of estimation errors of a calibrated accelerometer and a calibrated gyroscope of the consumer-grade inertial measurement unit (IMU) are shown in FIGS. 2A and 2B, respectively. It can be observed from FIGS. 2A and 2B that while the gyroscope readings are quite stable, the calibrated accelerometer readings register standard deviations in the order of 10-20 millimetres per second squared (mm/s^2). This is more obvious in the z-plane where gravitational disturbance can cause fluctuations in observed body accelerations.

Referring again to FIG. 1, in one or more embodiments, a_k may represent the acceleration of the object in two dimensions (2D), excluding gravitational acceleration. Advantageously, by ignoring accelerations along the gravity plane (z-axis), error propagation due to indistinguishability of body accelerations and gravitational pull of earth may be limited.

With absolute acceleration and heading information from a rigid navigation frame, each element of the state vector x_k may be modelled by Equations (7) to (12):

$$p_k = p_{k-1} + T_s v_{k-1} \quad (7)$$

$$v_k^x = \left(v_{k-1}^x \cos(hd_{k-1}) + T_s \sqrt{a_{k-1}^{x^2} + a_{k-1}^{y^2}} \right) \cos(hd_{k-1}) \quad (8)$$

$$5 \quad v_k^y = \left(v_{k-1}^y \sin(hd_{k-1}) + T_s \sqrt{a_{k-1}^{x^2} + a_{k-1}^{y^2}} \right) \sin(hd_{k-1}) \quad (9)$$

$$v_k^z = v_{k-1}^z \quad (10)$$

$$hd_k = hd_{k-1} \quad (11)$$

$$a_k = a_{k-1} \quad (12)$$

where p_{k-1} represents the three-dimensional (3D) position of the object at time $k - 1$,
10 v_{k-1} represents the velocity states of the object at time $k - 1$, T_s represents a time step, v_k^x , v_k^y and v_k^z represent the velocity states of the object at time k in x , y and z dimensions, respectively, v_{k-1}^x , v_{k-1}^y and v_{k-1}^z represent the velocity states of the object at time $k - 1$ in x , y and z dimensions, respectively, hd_{k-1} represents the absolute navigational heading of the object in 1D at time $k - 1$, a_{k-1} represents the
15 acceleration of the object at time $k - 1$, and a_{k-1}^x and a_{k-1}^y represent navigational accelerations of the object at time $k - 1$ in x and y dimensions, respectively. These set of equations form the state transition matrix F_k of the extended Kalman filter (EKF) and a state prediction for a time-step T_s may thus be derived using Equations (7) to (12).

After state prediction at step 12, state equations of UWB ranging, accelerometer
20 and gyroscope may be updated as follows:

$$\text{UWB: } h_k = \|p_k - c_{m,k}\| \quad (21)$$

Accelerometer:
$$h_k = a_k \quad (22)$$

Gyroscope:
$$h_k = h d_k \quad (23)$$

where h_k represents a sensor state observation, and $c_{m,k}$ represents m^{th} UWB anchor coordinates at state k . For example, h_k for accelerometer may be $h_k = x_k[7:9]$, representing last two (2) positions of the state vector corresponding to navigational acceleration in x and y axis.

At step 14, a covariance matrix P_k of the state vector x_k of the object is predicted using Equation (2):

$$P_k = F_k P_{k-1} F_k^T + Q_k \quad (2)$$

where P_{k-1} represents an estimated covariance matrix at time $k - 1$, F_k^T represents a transpose of the state-transition matrix F_k , and Q_k represents a process noise covariance.

Inertial measurement unit (IMU) data is received at step 16 and ultra-wideband (UWB) data is received at step 18. The computer-implemented method 10 for object localisation may take as measurement inputs 6D IMU data (3-axis accelerometer and 3-axis gyroscope) and UWB two-way ranging (TWR) data. These sensor inputs may have individual data rates, statistical standard deviation and sensor noise covariances, known or tuned for the purposes of fusion.

At step 20, observation residual y_k is determined using Equation (3):

$$y_k = z_k - H_k x_k \quad (3)$$

where z_k represents an observation of a true state of the object based on the IMU data and the UWB data, and H_k represents a sensor observation matrix, that is, a Jacobian matrix of h_k or partial derivatives of the vector h_k with respect to x_k . Sensor residual

y_k is the residual or difference between a true observation Z_k and an observed state $H_k x_k$.

Z_k may be the observation of sensor data from IMU and UWB ranging. Acceleration, heading and UWB ranging may all be treated as independent sensors that have individual and sequential extended Kalman filter (EKF) updates. The sensor readings have different output rates and when received, are assigned to Z_k for further processing. Accordingly, Z_k may be IMU accelerometer data $[a_n^i[0:1]]$, a_n^i representing an instantaneous navigational acceleration of the object in a rigid navigation frame and using only x and y dimensions (2D) of navigational acceleration, IMU gyroscope data $[h_n^i[1]]$, h_n^i representing an instantaneous navigational heading of the object in the rigid navigation frame and using only y dimension (1D) of gyroscope heading, and/or UWB ranging data $[d_m]$, d_m representing a distance between an m -th anchor and the object computed using a UWB two-way ranging (TWR) protocol.

The calibration process shown in FIG. 1 considers N static data points where accelerometer readings may be stored and the navigational frame adjusted (optimal roll, pitch, and yaw) to minimize bias in navigational acceleration. This calibrated bias and fixed navigational frame may subsequently be used to compute instantaneous navigational accelerations and heading for extended Kalman filter (EKF) update.

The step 20 of determining the observation residual y_k may include a step 22 of calibrating an accelerometer measurement bias b_{a_n} in the rigid navigation frame using Equation (13):

$$b_{a_n} = \frac{1}{N} \sum_{j=1}^N C_{bn}(\phi^*, \theta^*, \psi^*) a_b^j \quad (13)$$

where N represents a number of static data points where measurements are stored, C_{bn} represents a rotation matrix defined by Equation (14) describing orientation of a body frame with respect to the rigid navigation frame:

$$C_{bn} = \begin{bmatrix} \cos(\theta_k) \cos(\psi_k) & \sin(\phi_k) \sin(\theta_k) \cos(\psi_k) - \cos(\phi_k) \sin(\psi_k) \\ \cos(\theta_k) \sin(\psi_k) & \sin(\phi_k) \sin(\theta_k) \sin(\psi_k) + \cos(\phi_k) \cos(\psi_k) \\ -\sin(\theta_k) & \sin(\phi_k) \cos(\theta_k) \end{bmatrix}$$

$$\begin{bmatrix} \cos(\phi_k) \sin(\theta_k) \cos(\psi_k) + \sin(\phi_k) \sin(\psi_k) \\ \cos(\phi_k) \sin(\theta_k) \sin(\psi_k) - \sin(\phi_k) \cos(\psi_k) \\ \cos(\phi_k) \cos(\theta_k) \end{bmatrix} \quad (14)$$

where θ_k represents a pitch angle of the object, ψ_k represents a yaw angle of the object, and ϕ_k represents a roll angle of the object, a_b^j represents a three-dimensional (3D) body-frame acceleration for a j^{th} stored data point, and ϕ^*, θ^*, ψ^* represents an optimal roll ϕ^* , pitch θ^* and yaw ψ^* of the object in the rigid navigation frame as defined by Equation (15):

$$\phi^*, \theta^*, \psi^* = \min_{-\pi < \phi, \theta, \psi < \pi} \sum_{j=1}^N C_{bn}(\phi, \theta, \psi) a_b^j \quad (15).$$

The step 20 of determining the observation residual y_k may further include a step 24 of computing the instantaneous navigational acceleration a_n^i of the object in the rigid navigation frame using Equation (16):

$$a_n^i = C_{bn}(\phi^*, \theta^*, \psi^*) a_b^i - b_{a_n} \quad (16)$$

where a_b^i represents instantaneous 3D body-frame accelerations.

The step 20 of determining the observation residual y_k may include a step 26 of calibrating a gyroscope measurement bias b_{w_n} in the rigid navigation frame using Equation (17)

$$b_{w_n} = \frac{1}{N} \sum_{j=1}^N A_{bn}(\phi^*, \theta^*, \psi^*) w_b^j \quad (17)$$

where A_{bn} represents a body to navigation frame matrix for angular velocity as defined by Equation (18):

$$A_{bn} = \begin{bmatrix} 1 & \sin(\phi_k) \tan(\theta_k) & \cos(\phi_k) \tan(\theta_k) \\ 0 & \cos(\phi_k) & -\sin(\phi_k) \\ 0 & \frac{\sin(\phi_k)}{\cos(\theta_k)} & \frac{\cos(\phi_k)}{\cos(\theta_k)} \end{bmatrix} \quad (18),$$

and w_b^j represents a 3D body-frame angular velocity for the j^{th} stored data point.

- 5 As can be seen from steps 22 and 26, the IMU data is pre-calibrated and its frame orientation optimized to minimize bias during a stationary-phase. In this manner, the computer-implemented method 10 for object localisation involves a de-coupled implementation with each sensor of the IMU having its own tuning control that allows for independent reliance of the fusion framework on each sensor. Consequently, any
- 10 erroneous measurements from one sensor will not affect accuracy of and corresponding positioning update by another sensor. Advantageously, this makes the framework impervious to particularly high bias in individual sensors.

The step 20 of determining the observation residual y_k may also include a step 28 of computing an instantaneous navigational angular velocity w_n^i of the object in the

15 rigid navigation frame using Equation (19):

$$w_n^i = \frac{1}{T_s} A_{bn}(\phi^*, \theta^*, \psi^*) w_b^i - b_{w_n} \quad (19)$$

where w_b^i represents instantaneous 3D body-frame angular velocities.

The step 20 of determining the observation residual y_k may further include a step 30 of computing the instantaneous navigational heading h_n^i of the object in the rigid

20 navigation frame using Equation (20):

$$h_n^i = w_n^i + w_n^{i-1} \quad (20)$$

where w_n^{i-1} represents a previous instantaneous navigation-frame angular velocity. In essence, h_n^i represents an absolute navigational heading or sum of instantaneous navigation-frame angular velocities since a start at time 0.

Advantageously, pre-processing of the IMU data in steps 24, 28 and 30 with a fixed body and navigation frame directly aids velocity updating of the filtering process through speed and heading information derived from the instantaneous IMU data.

Further advantageously, use of the rigid navigation frame eliminates propagation of errors due to incorrect or noisy accelerometer readings, allowing fusion with UWB measurements to reach centimetre-level positioning accuracy using low-cost commercially available IMUs.

An optimal Kalman gain is determined at step 32 using Equation (4):

$$K_k = P_k H_k^T (H_k P_k H_k^T + R_k) \quad (4)$$

where H_k^T represents a transpose of the sensor observation matrix H_k , and R_k represents an observation or sensor noise covariance. As can be seen from step 32, accelerometer and gyroscope measurements from the IMU have individual tuning parameters R_k that decide a degree to which the extended Kalman filter (EKF) update depends on each of the sensors. The observation noise or sensor covariance R_k may be IMU accelerometer sensor noise covariance R_{acc} , IMU gyroscope data R_{gyro} , and/or UWB sensor or observation noise covariance R_{UWB} .

At step 34, the predicted state vector x_k is updated using Equation (5) to generate a localization solution L_{est} :

$$x_k = x_k + K_k y_k \quad (5).$$

The sensor residual y_k is weighed by $K_k y_k$ before being applied to the state vector x_k . The Kalman gain K_k is affected by the sensor or observation covariance R_k , in particular, the larger the observation covariance R_k , the larger the Kalman gain K_k . The

reverse applies as well. Therefore, the sensor or observation covariance R_k has a weighted impact on application of the sensor residual y_k to update the state vector update. The updated state vector x_k informs the localization solution L_{est} of the output 3D position coordinates at each state. The localization solution L_{est} may be generated

5 from position estimates of the state vector x_k , $L_{est} = x_k[0:2]$.

The update stage is an iterative process for each of the sensor information available to the EKF. z_k is equivalent to each of a_n^i , h_n^i , and d_m and R_k is correspondingly equivalent to each of R_{acc} , R_{gyro} and R_{UWB} . When a sensor (UWB/accelerometer/gyroscope) receives data, it does so at different data rates. For

10 example, UWB sensors are capable of 2-10 Hertz (Hz) outputs, while consumer-grade IMUs (accelerometer/gyroscopes) can work at 50-200 Hz. When, for example, a UWB chip receives ranging data, $z_k = [d_m]$, where d_m is the ranging data received, and $R_k = R_{UWB}$, where R_{UWB} is the UWB sensor or observation covariance, and then the update steps 20, 32 and 34 are carried out for this sensor. After this, the accelerometer,

15 for example, receives data, $z_k = [a_n^i[0:1]]$, that is, the x and y axis of navigational acceleration, and $R_k = R_{acc}$, where R_{acc} is the accelerometer sensor noise covariance, and then the update steps 20, 32 and 34 are carried out again. Likewise, when the gyroscope receives data, $z_k = [h_n^i[1]]$, that is, the y axis of the heading, and $R_k = R_{gyro}$, where R_{gyro} is the gyroscope sensor noise covariance, and then

20 the steps 20, 32 and 34 are carried out for this sensor again. This process is repeated as and when new sensor data is received from the UWB, accelerometer and gyroscope at their individual data rates. On the other hand, the prediction steps 12 and 14 are performed only every $(1/T_s)$ Hz, which is the time-step chosen as the maximum sensor data rate. In one embodiment, $1/T_s = 50$ Hz to match the data rate of the IMU datarate

25 or the highest sensor data rate.

It follows from the foregoing that the computer-implemented method 10 for object localisation provides a unique way of filtering and fusing noisy consumer-grade inertial

measurement unit (IMU) data with UWB ranging data in a robust and de-coupled extended Kalman filter (EKF) for accurate centimetre-level localization at low complexity. The computer-implemented method 10 for object localisation provides a de-coupled solution to utilizing accelerometer and gyroscope data with separate EKF tuning parameters. The robust implementation does not allow accumulation of frame orientation error, is impervious to bias in accelerometer or gyroscope readings and also provides lower complexity filtering and higher process update rate than traditional IMU-EKF fusion frameworks.

Computer program instructions executable by one or more computer processors to perform operations for object localisation described above may be stored in non-transitory computer-readable memory.

Referring now to FIG. 3, a computer system 100 suitable for implementing the method 10 for object localisation is shown. The computer system 100 includes a processor 102 (which may be referred to as a central processor unit or CPU) that is in communication with memory devices including secondary storage 104, read only memory (ROM) 106, random access memory (RAM) 108, input/output (I/O) devices 110, and network connectivity devices 112. The processor 102 may be implemented as one or more CPU chips.

It is understood that by programming and/or loading executable instructions onto the computer system 100, at least one of the CPU 102, the RAM 108, and the ROM 106 are changed, transforming the computer system 100 in part into a particular machine or apparatus having the novel functionality taught by the present disclosure. It is fundamental to the electrical engineering and software engineering arts that functionality that can be implemented by loading executable software into a computer can be converted to a hardware implementation by well-known design rules. Decisions between implementing a concept in software versus hardware typically hinge on considerations of stability of the design and numbers of units to be produced rather than any issues involved in translating from the software domain to the hardware domain. Generally, a design that is still subject to frequent change may be preferred to be implemented in software, because re-spinning a hardware implementation is more expensive than re-spinning a software design. Generally, a design that is stable that will be produced in large volume may be preferred to be implemented in hardware, for example in an

application specific integrated circuit (ASIC), because for large production runs the hardware implementation may be less expensive than the software implementation. Often a design may be developed and tested in a software form and later transformed, by well-known design rules, to an equivalent hardware implementation in an application specific integrated circuit that hardwires the instructions of the software. In the same manner as a machine controlled by a new ASIC is a particular machine or apparatus, likewise a computer that has been programmed and/or loaded with executable instructions may be viewed as a particular machine or apparatus.

Additionally, after the system 100 is turned on or booted, the CPU 102 may execute a computer program or application. For example, the CPU 102 may execute software or firmware stored in the ROM 106 or stored in the RAM 108. In some cases, on boot and/or when the application is initiated, the CPU 102 may copy the application or portions of the application from the secondary storage 104 to the RAM 108 or to memory space within the CPU 102 itself, and the CPU 102 may then execute instructions that the application is comprised of. In some cases, the CPU 102 may copy the application or portions of the application from memory accessed via the network connectivity devices 112 or via the I/O devices 110 to the RAM 108 or to memory space within the CPU 102, and the CPU 102 may then execute instructions that the application is comprised of. During execution, an application may load instructions into the CPU 102, for example load some of the instructions of the application into a cache of the CPU 102. In some contexts, an application that is executed may be said to configure the CPU 102 to do something, for example, to configure the CPU 102 to perform the function or functions promoted by the subject application. When the CPU 102 is configured in this way by the application, the CPU 102 becomes a specific purpose computer or a specific purpose machine.

The secondary storage 104 is typically comprised of one or more disk drives or tape drives and is used for non-volatile storage of data and as an over-flow data storage device if RAM 108 is not large enough to hold all working data. Secondary storage 104 may be used to store programs which are loaded into RAM 108 when such programs are selected for execution. The ROM 106 is used to store instructions and perhaps data which are read during program execution. ROM 106 is a non-volatile memory device which typically has a small memory capacity relative to the larger memory capacity of

secondary storage 104. The RAM 108 is used to store volatile data and perhaps to store instructions. Access to both ROM 106 and RAM 108 is typically faster than to secondary storage 104. The secondary storage 104, the RAM 108, and/or the ROM 106 may be referred to in some contexts as computer readable storage media and/or non-transitory
5 computer readable media.

I/O devices 110 may include cameras, printers, video monitors, liquid crystal displays (LCDs), plasma displays, touch screen displays, keyboards, keypads, switches, dials, mice, track balls, voice recognizers, card readers, paper tape readers, or other well-known input devices. The network connectivity devices 112 may take the form of
10 modems, modem banks, Ethernet cards, universal serial bus (USB) interface cards, serial interfaces, token ring cards, fiber distributed data interface (FDDI) cards, wireless local area network (WLAN) cards, radio transceiver cards that promote radio communications using protocols such as code division multiple access (CDMA), global system for mobile communications (GSM), long-term evolution (LTE), worldwide
15 interoperability for microwave access (WiMAX), near field communications (NFC), radio frequency identity (RFID), and/or other air interface protocol radio transceiver cards, and other well-known network devices. These network connectivity devices 112 may enable the processor 102 to communicate with the Internet or one or more intranets. With such a network connection, it is contemplated that the processor 102 might receive information
20 from the network, or might output information to the network in the course of performing the above-described method steps. Such information, which is often represented as a sequence of instructions to be executed using processor 102, may be received from and outputted to the network, for example, in the form of a computer data signal embodied in a carrier wave. Such information, which may include data or instructions to be executed
25 using processor 102 for example, may be received from and outputted to the network, for example, in the form of a computer data baseband signal or signal embodied in a carrier wave. The baseband signal or signal embedded in the carrier wave, or other types of signals currently used or hereafter developed, may be generated according to several methods well-known to one skilled in the art. The baseband signal and/or signal
30 embedded in the carrier wave may be referred to in some contexts as a transitory signal.

The processor 102 executes instructions, codes, computer programs, scripts which it accesses from hard disk, floppy disk, optical disk (these various disk based

systems may all be considered secondary storage 104), flash drive, ROM 106, RAM 108, or the network connectivity devices 112. While only one processor 102 is shown, multiple processors may be present. Thus, while instructions may be discussed as executed by a processor, the instructions may be executed simultaneously, serially, or otherwise
5 executed by one or multiple processors. Instructions, codes, computer programs, scripts, and/or data that may be accessed from the secondary storage 104, for example, hard drives, floppy disks, optical disks, and/or other device, the ROM 106, and/or the RAM 108 may be referred to in some contexts as non-transitory instructions and/or non-transitory information.

10 In an embodiment, the computer system 100 may comprise two or more computers in communication with each other that collaborate to perform a task. For example, but not by way of limitation, an application may be partitioned in such a way as to permit concurrent and/or parallel processing of the instructions of the application. Alternatively, the data processed by the application may be partitioned in such a way as
15 to permit concurrent and/or parallel processing of different portions of a data set by the two or more computers. In an embodiment, virtualization software may be employed by the computer system 100 to provide the functionality of a number of servers that is not directly bound to the number of computers in the computer system 100. For example, virtualization software may provide twenty virtual servers on four physical computers. In
20 an embodiment, the functionality disclosed above may be provided by executing the application and/or applications in a cloud computing environment. Cloud computing may comprise providing computing services via a network connection using dynamically scalable computing resources. Cloud computing may be supported, at least in part, by virtualization software. A cloud computing environment may be established by an
25 enterprise and/or may be hired on an as-needed basis from a third party provider. Some cloud computing environments may comprise cloud computing resources owned and operated by the enterprise as well as cloud computing resources hired and/or leased from a third party provider.

In an embodiment, some or all of the functionality disclosed above may be
30 provided as a computer program product. The computer program product may comprise one or more computer readable storage medium having computer usable program code embodied therein to implement the functionality disclosed above. The computer program

product may comprise data structures, executable instructions, and other computer usable program code. The computer program product may be embodied in removable computer storage media and/or non-removable computer storage media. The removable computer readable storage medium may comprise, without limitation, a paper tape, a
5 magnetic tape, magnetic disk, an optical disk, a solid state memory chip, for example analog magnetic tape, compact disk read only memory (CD-ROM) disks, floppy disks, jump drives, digital cards, multimedia cards, and others. The computer program product may be suitable for loading, by the computer system 100, at least portions of the contents of the computer program product to the secondary storage 104, to the ROM 106, to the
10 RAM 108, and/or to other non-volatile memory and volatile memory of the computer system 100. The processor 102 may process the executable instructions and/or data structures in part by directly accessing the computer program product, for example by reading from a CD-ROM disk inserted into a disk drive peripheral of the computer system 100. Alternatively, the processor 102 may process the executable instructions and/or
15 data structures by remotely accessing the computer program product, for example by downloading the executable instructions and/or data structures from a remote server through the network connectivity devices 112. The computer program product may comprise instructions that promote the loading and/or copying of data, data structures, files, and/or executable instructions to the secondary storage 104, to the ROM 106, to
20 the RAM 108, and/or to other non-volatile memory and volatile memory of the computer system 100.

In some contexts, the secondary storage 104, the ROM 106, and the RAM 108 may be referred to as a non-transitory computer readable medium or a computer readable storage media. A dynamic RAM embodiment of the RAM 108, likewise, may be
25 referred to as a non-transitory computer readable medium in that while the dynamic RAM receives electrical power and is operated in accordance with its design, for example during a period of time during which the computer system 100 is turned on and operational, the dynamic RAM stores information that is written to it. Similarly, the processor 102 may comprise an internal RAM, an internal ROM, a cache memory, and/or
30 other internal non-transitory storage blocks, sections, or components that may be referred to in some contexts as non-transitory computer readable media or computer readable storage media.

Experimental Results

Referring now to FIGS. 4A and 4B, an experiment was performed with six (6) ultra-wideband (UWB) anchors (marked by circles in FIGS. 4A and 4B) that carry out two-way ranging (TWR) with a UWB tag (or object of interest) to be tracked. The object
5 surface was provided with an android inertial measurement device (IMU) device that provided heading and acceleration data for fusion.

It can be seen from FIGS. 4A and 4B that whilst UWB with a least square solution provides very good positioning accuracy, overall positioning quality can be improved even with low-cost IMU data and an extended Kalman filter (EKF) motion model.
10 Additionally, the difference in performance when a classical EKF-IMU fusion framework is used as compared to an auxiliary framework in accordance with an embodiment of the present invention can be observed from FIGS. 4A and 4B.

Referring now to FIGS. 5A and 5B, the noisy acceleration data that updates the EKF state velocity is shown. With the auxiliary framework, the rigid navigation frame
15 allows for a smoother velocity update and a correspondingly more precise position update for the tag. A clear distinction can also be observed in FIGS. 5A and 5B in the reduction in velocity along each axis to 0 in the auxiliary framework compared to the classical IMU implementation.

Referring now to FIG. 6, an outage scenario where the surface of the tag was
20 blocked with foil and one-half of the total available anchors experience non-line-of-sight (NLOS) with the tag was simulated for comparison.

Referring now to FIGS. 7A and 7B, the tracking results for both frameworks are shown. From FIGS. 7A and 7B, the outage use-case for integrating IMU with UWB is clear. Even with limited or no ranging, the EKF+UWB+IMU auxiliary framework
25 demonstrates the possibility of maintaining indoor tracking with desirable centimetre-level accuracy. However, it was also observed that in the absence of UWB ranging data to guide the classical framework, the solution diverges from the intended trajectory.

From extensive testing, and as evidenced from the results presented, it is clearly demonstrated that the described EKF+UWB+IMU tracking solution is able to deliver an
30 accuracy within 10 centimetres (cm) for line-of-sight (LOS) environments and 20 cm for

non-line-of-sight (NLOS) environments, both of which are within acceptable margins for most emerging UWB applications.

The experimental results highlight the need for a new approach to treat consumer-grade IMU data and fuse it with UWB ranging to maintain accurate centimetre-scale tracking. From testing performed, it was noted that UWB fusion solutions are sensitive to drift errors in motion sensors which can be propagated during the trajectory. Additionally, given mass-scale production of UWB chipsets, integrating consumer-grade IMUs is more feasible. With the performance of the individual sensors demonstrated above and the scale of localization accuracy needed for UWB applications, the auxiliary framework in accordance with an embodiment of the invention considers a pre-calibrated rigid navigation frame to limit the error propagation due to individual IMU sensors.

As is evident from the foregoing discussion, the present invention provides a computer-implemented method and non-transitory computer-readable memory storing computer program instructions for object localisation that is able to utilize noisy and poor quality IMU sensor data, whilst maintaining centimetre-level positioning accuracy. Advantageously, the EKF-IMU auxiliary framework of the present invention provides a platform to fuse a motion model with consumer-grade IMU and UWB ranging data to track a position of an object at centimetre-level accuracy. The EKF framework of the present invention is able to utilize noisy consumer-grade IMU sensors for fusion with UWB ranging data for centimetre-level positioning accuracy. The computer-implemented method and non-transitory computer-readable memory storing computer program instructions for object localisation of the present invention also provide a decoupled solution to utilizing accelerometer and gyroscope data with separate EKF tuning parameters. Further advantageously, the computer-implemented method and non-transitory computer-readable memory storing computer program instructions for object localisation of the present invention also affords lower complexity filtering and higher process update rate than traditional IMU-EKF fusion frameworks. Furthermore, as the present invention is able to avoid accumulation of frame orientation error, a robust implementation is provided that is impervious to bias in accelerometer or gyroscope readings.

While preferred embodiments of the invention have been illustrated and described, it will be clear that the invention is not limited to the described embodiments

only. Numerous modifications, changes, variations, substitutions and equivalents will be apparent to those skilled in the art without departing from the scope of the invention as described in the claims.

Further, unless the context clearly requires otherwise, throughout the description
5 and the claims, the words "comprise", "comprising" and the like are to be construed in an inclusive as opposed to an exclusive or exhaustive sense; that is to say, in the sense of "including, but not limited to".

CLAIMS

1. A computer-implemented method for object localisation, comprising executing on one or more processors the steps of:

5 predicting a state vector x_k of an object at time k using Equation (1):

$$x_k = F_k x_{k-1} \quad (1)$$

wherein

F_k represents a state-transition matrix, and

x_{k-1} represents the state vector at time $k - 1$;

10 predicting a covariance matrix P_k of the state vector x_k of the object using Equation (2):

$$P_k = F_k P_{k-1} F_k^T + Q_k \quad (2)$$

wherein

P_{k-1} represents an estimated covariance matrix at time $k - 1$,

15 F_k^T represents a transpose of the state-transition matrix F_k , and

Q_k represents a process noise covariance;

receiving inertial measurement unit (IMU) data;

receiving ultra-wideband (UWB) data;

determining observation residual y_k using Equation (3):

20
$$y_k = z_k - H_k x_k \quad (3)$$

wherein

z_k represents an observation of a true state of the object based on the IMU data and the UWB data, and

H_k represents a sensor observation matrix;

25 determining an optimal Kalman gain using Equation (4):

$$K_k = P_k H_k^T (H_k P_k H_k^T + R_k) \quad (4)$$

wherein

H_k^T represents a transpose of the sensor observation matrix H_k , and

R_k represents an observation or sensor noise covariance; and
 updating the predicted state vector x_k using Equation (5) to generate a
 localization solution L_{est} :

$$x_k = x_k + K_k y_k \quad (5).$$

5

2. The computer-implemented method for object localisation according to claim 1,
 wherein the state vector x_k is defined by Equation (6):

$$x_k = [p_k, v_k, hd_k, a_k] \quad (6)$$

wherein

10 p_k represents a three-dimensional (3D) position of the object at time k ,
 v_k represents velocity states of the object at time k ,
 hd_k represents an absolute navigational heading of the object in a single
 dimension (1D) at time k , and
 a_k represents an acceleration of the object at time k .

15

3. The computer-implemented method for object localisation according to claim 2,
 wherein a_k represents the acceleration of the object in two dimensions (2D), excluding
 gravitational acceleration.

20 4. The computer-implemented method for object localisation according to claim 3,
 wherein each element of the state vector x_k is modelled by Equations (7) to (12):

$$p_k = p_{k-1} + T_s v_{k-1} \quad (7)$$

$$v_k^x = \left(v_{k-1}^x \cos(hd_{k-1}) + T_s \sqrt{a_{k-1}^{x^2} + a_{k-1}^{y^2}} \right) \cos(hd_{k-1}) \quad (8)$$

$$v_k^y = \left(v_{k-1}^y \sin(hd_{k-1}) + T_s \sqrt{a_{k-1}^{x^2} + a_{k-1}^{y^2}} \right) \sin(hd_{k-1}) \quad (9)$$

25

$$v_k^z = v_{k-1}^z \quad (10)$$

$$hd_k = hd_{k-1} \quad (11)$$

$$a_k = a_{k-1} \quad (12)$$

wherein

p_{k-1} represents the three-dimensional (3D) position of the object at time $k - 1$,

5 v_{k-1} represents the velocity states of the object at time $k - 1$,

T_s represents a time step,

v_k^x , v_k^y and v_k^z represent the velocity states of the object at time k in x , y and z dimensions, respectively,

10 v_{k-1}^x , v_{k-1}^y and v_{k-1}^z represent the velocity states of the object at time $k - 1$ in x , y and z dimensions, respectively,

hd_{k-1} represents the absolute navigational heading of the object in 1D at time $k - 1$,

a_{k-1} represents the acceleration of the object at time $k - 1$, and

15 a_{k-1}^x and a_{k-1}^y represent navigational accelerations of the object at time $k - 1$ in x and y dimensions, respectively.,

5. The computer-implemented method for object localisation according to any one of the preceding claims, wherein Z_k is one of IMU accelerometer data $[a_n^i[0:1]]$, a_n^i representing an instantaneous navigational acceleration of the object in a rigid navigation frame, IMU gyroscope data $[h_n^i[1]]$, h_n^i representing an instantaneous navigational heading of the object in the rigid navigation frame, and UWB ranging data $[d_m]$, d_m representing a distance between an m -th anchor and the object computed using a UWB two-way ranging (TWR) protocol.

25 6. The computer-implemented method for object localisation according to claim 5, wherein the step of determining the observation residual y_k comprises:

calibrating an accelerometer measurement bias b_{a_n} in the rigid navigation frame using Equation (13):

$$b_{a_n} = \frac{1}{N} \sum_{j=1}^N C_{bn}(\phi^*, \theta^*, \psi^*) a_b^j \quad (13)$$

wherein

- 5 N represents a number of static data points where measurements are stored,
 C_{bn} represents a rotation matrix defined by Equation (14) describing orientation of a body frame with respect to the rigid navigation frame:

$$C_{bn} = \begin{bmatrix} \cos(\theta_k) \cos(\psi_k) & \sin(\phi_k) \sin(\theta_k) \cos(\psi_k) - \cos(\phi_k) \sin(\psi_k) \\ \cos(\theta_k) \sin(\psi_k) & \sin(\phi_k) \sin(\theta_k) \sin(\psi_k) + \cos(\phi_k) \cos(\psi_k) \\ -\sin(\theta_k) & \sin(\phi_k) \cos(\theta_k) \\ \cos(\phi_k) \sin(\theta_k) \cos(\psi_k) + \sin(\phi_k) \sin(\psi_k) \\ \cos(\phi_k) \sin(\theta_k) \sin(\psi_k) - \sin(\phi_k) \cos(\psi_k) \\ \cos(\phi_k) \cos(\theta_k) \end{bmatrix} \quad (14)$$

- 10 wherein

θ_k represents a pitch angle of the object,

ψ_k represents a yaw angle of the object, and

ϕ_k represents a roll angle of the object,

- a_b^j represents a three-dimensional (3D) body-frame acceleration for a j^{th} stored
 15 data point, and

ϕ^*, θ^*, ψ^* represents an optimal roll ϕ^* , pitch θ^* and yaw ψ^* of the object in the rigid navigation frame as defined by Equation (15):

$$\phi^*, \theta^*, \psi^* = \min_{-\pi < \phi, \theta, \psi < \pi} \sum_{j=1}^N C_{bn}(\phi, \theta, \psi) a_b^j \quad (15).$$

- 20 7. The computer-implemented method for object localisation according to claim 6, wherein the step of determining the observation residual y_k further comprises:

computing the instantaneous navigational acceleration a_n^i of the object in the rigid navigation frame using Equation (16):

$$a_n^i = C_{bn}(\phi^*, \theta^*, \psi^*) a_b^i - b_{a_n} \quad (16)$$

wherein

a_b^i represents instantaneous 3D body-frame accelerations.

8. The computer-implemented method for object localisation according to claim 6 or
5 7, wherein the step of determining the observation residual y_k further comprises:

calibrating a gyroscope measurement bias b_{w_n} in the rigid navigation frame
using Equation (17)

$$b_{w_n} = \frac{1}{N} \sum_{j=1}^N A_{bn}(\phi^*, \theta^*, \psi^*) w_b^j \quad (17)$$

wherein

- 10 A_{bn} represents a body to navigation frame matrix for angular velocity as defined
by Equation (18):

$$A_{bn} = \begin{bmatrix} 1 & \sin(\phi_k) \tan(\theta_k) & \cos(\phi_k) \tan(\theta_k) \\ 0 & \cos(\phi_k) & -\sin(\phi_k) \\ 0 & \frac{\sin(\phi_k)}{\cos(\theta_k)} & \frac{\cos(\phi_k)}{\cos(\theta_k)} \end{bmatrix} \quad (18), \text{ and}$$

w_b^j represents a 3D body-frame angular velocity for the j^{th} stored data point.

- 15 9. The computer-implemented method for object localisation according to claim 8,
wherein the step of determining the observation residual y_k further comprises:

computing an instantaneous navigational angular velocity w_n^i of the object in the
rigid navigation frame using Equation (19):

$$w_n^i = \frac{1}{T_s} A_{bn}(\phi^*, \theta^*, \psi^*) w_b^i - b_{w_n} \quad (19)$$

- 20 wherein

w_b^i represents instantaneous 3D body-frame angular velocities.

10. The computer-implemented method for object localisation according to claim 9,
wherein the step of determining the observation residual y_k further comprises:

- 25 computing the instantaneous navigational heading h_n^i of the object in the rigid
navigation frame using Equation (20):

$$h_n^i = w_n^i + w_n^{i-1} \quad (20)$$

wherein

w_n^{i-1} represents a previous instantaneous navigation-frame angular velocity.

11. The computer-implemented method for object localisation according to any one of the preceding claims, wherein the observation or sensor noise covariance R_k is one of IMU accelerometer sensor noise covariance R_{acc} , IMU gyroscope data R_{gyro} , and UWB sensor or observation noise covariance R_{UWB} .

12. A non-transitory computer-readable memory storing computer program instructions executable by one or more computer processors to perform operations for object localisation, the operations comprising:

predicting a state vector x_k of an object at time k using Equation (1):

$$x_k = F_k x_{k-1} \quad (1)$$

wherein

F_k represents a state-transition matrix, and

x_{k-1} represents the state vector at time $k - 1$;

predicting a covariance matrix P_k of the state vector x_k of the object using Equation (2):

$$P_k = F_k P_{k-1} F_k^T + Q_k \quad (2)$$

wherein

P_{k-1} represents an estimated covariance matrix at time $k - 1$,

F_k^T represents a transpose of the state-transition matrix F_k , and

Q_k represents a process noise covariance;

receiving inertial measurement unit (IMU) data;

receiving ultra-wideband (UWB) data;

determining observation residual y_k using Equation (3):

$$y_k = z_k - H_k x_k \quad (3)$$

wherein

Z_k represents an observation of a true state of the object based on the IMU data and the UWB data, and

H_k represents an observation model;

determining an optimal Kalman gain using Equation (4):

$$K_k = P_k H_k^T (H_k P_k H_k^T + R_k) \quad (4)$$

wherein

H_k^T represents a transpose of the sensor observation matrix H_k , and

R_k represents an observation or sensor noise covariance; and

updating the predicted state vector x_k using Equation (5) to generate a
10 localization solution L_{est} :

$$x_k = x_k + K_k y_k \quad (5).$$

13. The non-transitory computer-readable memory according to claim 12, wherein the state vector x_k is defined by Equation (6):

$$x_k = [p_k, v_k, hd_k, a_k] \quad (6)$$

wherein

p_k represents a three-dimensional (3D) position of the object at time k ,

v_k represents velocity states of the object at time k ,

hd_k represents an absolute navigational heading of the object in a single
20 dimension (1D) at time k , and

a_k represents an acceleration of the object at time k .

14. The non-transitory computer-readable memory according to claim 13, wherein a_k represents the acceleration of the object in two dimensions (2D), excluding
25 gravitational acceleration.

15. The non-transitory computer-readable memory according to claim 13 or 14, wherein each element of the state vector x_k is modelled by Equations (7) to (12):

$$p_k = p_{k-1} + T_s v_{k-1} \quad (7)$$

$$v_k^x = \left(v_{k-1}^x \cos(hd_{k-1}) + T_s \sqrt{a_{k-1}^x{}^2 + a_{k-1}^y{}^2} \right) \cos(hd_{k-1}) \quad (8)$$

$$v_k^y = \left(v_{k-1}^y \sin(hd_{k-1}) + T_s \sqrt{a_{k-1}^x{}^2 + a_{k-1}^y{}^2} \right) \sin(hd_{k-1}) \quad (9)$$

$$v_k^z = v_{k-1}^z \quad (10)$$

$$hd_k = hd_{k-1} \quad (11)$$

$$a_k = a_{k-1} \quad (12)$$

wherein

p_{k-1} represents the three-dimensional (3D) position of the object at time $k - 1$,

v_{k-1} represents the velocity states of the object at time $k - 1$,

T_s represents a time step,

v_k^x , v_k^y and v_k^z represent the velocity states of the object at time k in x , y and z dimensions, respectively,

v_{k-1}^x , v_{k-1}^y and v_{k-1}^z represent the velocity states of the object at time $k - 1$ in x , y and z dimensions, respectively,

hd_{k-1} represents the absolute navigational heading of the object in 1D at time $k - 1$,

a_{k-1} represents the acceleration of the object at time $k - 1$, and

a_{k-1}^x and a_{k-1}^y represent navigational accelerations of the object at time $k - 1$ in x and y dimensions, respectively.

16. The non-transitory computer-readable memory according to any one of claims 12 to 15, wherein Z_k is one of IMU accelerometer data $[a_n^i[0:1]]$, a_n^i representing an instantaneous navigational acceleration of the object in a rigid navigation frame, IMU

gyroscope data $\left[h_n^i[1]\right]$, h_n^i representing an instantaneous navigational heading of the object in the rigid navigation frame, and UWB ranging data $[d_m]$, d_m representing a distance between an m -th anchor and the object computed using a UWB two-way ranging (TWR) protocol.

5

17. The non-transitory computer-readable memory according to claim 16, wherein the operation of determining the observation residual y_k comprises:

calibrating an accelerometer measurement bias b_{a_n} in the rigid navigation frame using Equation (13):

$$10 \quad b_{a_n} = \frac{1}{N} \sum_{j=1}^N C_{bn}(\phi^*, \theta^*, \psi^*) a_b^j \quad (13)$$

wherein

N represents a number of static data points where measurements are stored,

C_{bn} represents a rotation matrix defined by Equation (14) describing orientation of a body frame with respect to the rigid navigation frame:

$$15 \quad C_{bn} = \begin{bmatrix} \cos(\theta_k) \cos(\psi_k) & \sin(\phi_k) \sin(\theta_k) \cos(\psi_k) - \cos(\phi_k) \sin(\psi_k) \\ \cos(\theta_k) \sin(\psi_k) & \sin(\phi_k) \sin(\theta_k) \sin(\psi_k) + \cos(\phi_k) \cos(\psi_k) \\ -\sin(\theta_k) & \sin(\phi_k) \cos(\theta_k) \\ \cos(\phi_k) \sin(\theta_k) \cos(\psi_k) + \sin(\phi_k) \sin(\psi_k) \\ \cos(\phi_k) \sin(\theta_k) \sin(\psi_k) - \sin(\phi_k) \cos(\psi_k) \\ \cos(\phi_k) \cos(\theta_k) \end{bmatrix} \quad (14)$$

wherein

θ_k represents a pitch angle of the object,

ψ_k represents a yaw angle of the object, and

20 ϕ_k represents a roll angle of the object,

a_b^j represents a three-dimensional (3D) body-frame acceleration for a j^{th} stored data point, and

ϕ^*, θ^*, ψ^* represents an optimal roll ϕ^* , pitch θ^* and yaw ψ^* of the object in the rigid navigation frame as defined by Equation (15):

$$\emptyset^*, \theta^*, \psi^* = \min_{-\pi < \emptyset, \theta, \psi < \pi} \sum_{j=1}^N C_{bn}(\emptyset, \theta, \psi) a_b^j \quad (15).$$

18. The non-transitory computer-readable memory according to claim 17, wherein the operation of determining the observation residual y_k further comprises:

5 computing the instantaneous navigational acceleration a_n^i of the object in the rigid navigation frame using Equation (16):

$$a_n^i = C_{bn}(\emptyset^*, \theta^*, \psi^*) a_b^i - b_{a_n} \quad (16)$$

wherein

a_b^i represents instantaneous 3D body-frame accelerations.

10

19. The non-transitory computer-readable memory according to claim 17 or 18, wherein the operation of determining the observation residual y_k further comprises:

calibrating a gyroscope measurement bias b_{w_n} in the rigid navigation frame using Equation (17)

$$15 \quad b_{w_n} = \frac{1}{N} \sum_{j=1}^N A_{bn}(\emptyset^*, \theta^*, \psi^*) w_b^j \quad (17)$$

wherein

A_{bn} represents a body to navigation frame matrix for angular velocity as defined by Equation (18):

$$A_{bn} = \begin{bmatrix} 1 & \sin(\phi_k) \tan(\theta_k) & \cos(\phi_k) \tan(\theta_k) \\ 0 & \cos(\phi_k) & -\sin(\phi_k) \\ 0 & \frac{\sin(\phi_k)}{\cos(\theta_k)} & \frac{\cos(\phi_k)}{\cos(\theta_k)} \end{bmatrix} \quad (18), \text{ and}$$

20 w_b^j represents a 3D body-frame angular velocity for the j^{th} stored data point.

20. The non-transitory computer-readable memory according to claim 19, wherein the operation of determining the observation residual y_k further comprises:

computing an instantaneous navigational angular velocity w_n^i of the object in the
25 rigid navigation frame using Equation (19):

$$w_n^i = \frac{1}{T_s} A_{bn}(\phi^*, \theta^*, \psi^*) w_b^i - b_{w_n} \quad (19)$$

wherein

w_b^i represents instantaneous 3D body-frame angular velocities.

- 5 21. The non-transitory computer-readable memory according to claim 20, wherein the operation of determining the observation residual y_k further comprises:

computing the instantaneous navigational heading h_n^i of the object in the rigid navigation frame using Equation (20):

$$h_n^i = w_n^i + w_n^{i-1} \quad (20)$$

- 10 wherein

w_n^{i-1} represents a previous instantaneous navigation-frame angular velocity.

22. The non-transitory computer-readable memory according to any one of claims 12
15 to 21, wherein the observation or sensor noise covariance R_k is one of IMU accelerometer sensor noise covariance R_{acc} , IMU gyroscope data R_{gyro} , and UWB sensor or observation noise covariance R_{UWB} .