2014 International Conference on Indoor Positioning and Indoor Navigation, 27th-30th October 2014

Foot-mounted inertial navigation made easy

John-Olof Nilsson[1], *Amit K Gupta*[2], *and Peter Händel*[1]
[1]Dept. Signal Processing, KTH Royal Institute of Technology, Stockholm, Sweden
[2]GT Silicon Pvt Ltd, Kanpur, India

***Abstract*—Despite being around for almost two decades, foot-mounted inertial navigation only has gotten a limited spread. Contributing factors to this are lack of suitable hardware platforms and difficult system integration. As a solution to this, we present an open-source wireless foot-mounted inertial navigation module with an intuitive and significantly simplified dead reckoning interface. The interface is motivated from statistical properties of the underlying aided inertial navigation and argued to give negligible information loss. The module consists of both a hardware platform and embedded software. Details of the platform and the software are described, and a summarizing description of how to reproduce the module are given. System integration of the module is outlined and finally, we provide a basic performance assessment of the module. In summary, the module provides a modularization of the foot-mounted inertial navigation and makes the technology significantly easier to use.**

I. Introduction

The idea of foot-mounted inertial navigation has been around for almost two decades [1] and offers significantly better performance over other pedestrian inertial tracking techniques [2–4]. Despite this, the usage outside technical research groups has been limited, and few products and applied systems using foot-mounted inertial navigation have been presented. Hardware limitations during the first decade, patent protection, and user group resistance to foot-mounted equipment can partially be attributed to this. However, other significant contributing factors are probably the difficulty of its integration and the lack of suitable hardware platforms, making the technology hard to apply for a wider user group. The difficulty of integration primarily comes from the sensor placement and the lack of a high-level statistical interface. The inertial navigation is “blind” and has to be combined with other information sources. The fusion is normally done by simply running the aided inertial navigation at some (non-foot-mounted) application platform, meaning that the full system complexity is exposed and that the high rate measurements have to be transferred from the foot. Wireless links may limit the rate, and they will be power hungry and sensitive due to high strain. The alternative is cabled connections, which have their obvious drawbacks. Unfortunately, without any pre-processing, there is no way around it. However, what pre-processing to do is not clear and the number of platforms, including necessary sensor, processing, and communication hardware and software support, which are suitable for foot-mounting, is limited.

The problems with the exposed complexity, the high data rates, and the hardware boil down to poor modularization, which makes application development challenging. As a solution, in this paper we present a modularization of the technology in the shape of a complete open-source wireless foot-mounted inertial navigation module. The module exposes a dead reckoning interface justified from statistical properties of the underlying system. In essence, the inertial navigation is implemented in the foot-mounted module and the user is provided with low rate dead reckoning updates over a Bluetooth link. The dead reckoning can be shown to reproduce the statistics of the foot-mounted inertial navigation and is therefore, equivalent at an application level [5]. The system perspective of the modules is illustrated in Fig. 1. Resources for reproducing the modules are available at www.openshoe.org. *This paper describes the module in detail, but the point is that the user do not have to care about most of the details.*

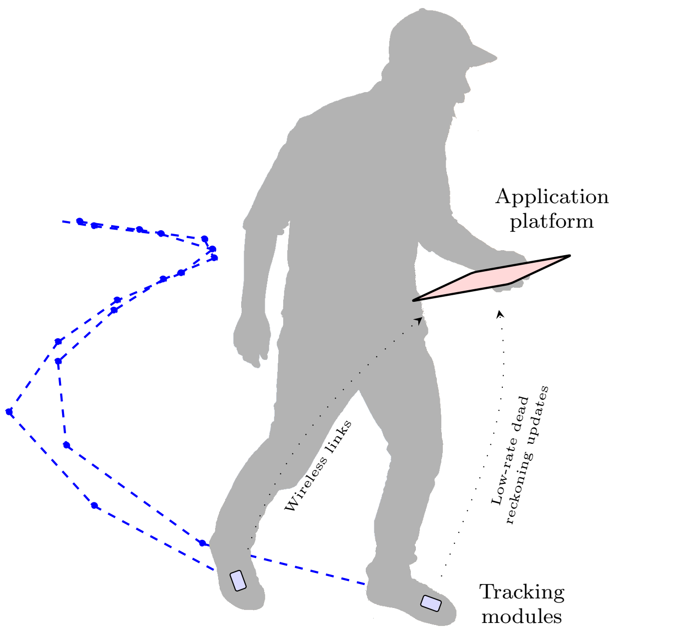


Fig. 1: We present a modularization of foot-mounted inertial navigation providing the accuracy of the inertial navigation at the simplicity of traditional dead reckoning. The modularization is supported by open-source wireless foot-mounted inertial navigation tracking modules. An application platform receives a low-rate (∼1 [Hz]) stream of displacement and heading changes, which it sums up to track the carrier. This is a significant simplification compared with handling and processing high-rate raw inertial measurements.

The remainder of the paper is organized as follows. In Section II, the basis for the dead reckoning interface is explained. Subsequently, in Section III, a detailed description of the tracking module is given. In Section IV, the application system integration and the necessary processing are outlined. Finally, in Section V, a performance evaluation is presented, and in Section VI conclusions and final remarks are given.

II. THE DEAD RECKONING INTERFACE

Foot-mounted inertial navigation is typically implemented as a zero-velocity aided inertial navigation system. The inertial navigation essentially consists of the inertial sensors combined with mechanization equations. In the simplest form, the mechanization equations are

$$\begin{bmatrix}\mathbf{p}_k\\ \mathbf{v}_k\\ \mathbf{q}_k\end{bmatrix} = \begin{bmatrix}\mathbf{p}_{k-1}+\mathbf{v}_{k-1}dt\\ \mathbf{v}_{k-1}+(\mathbf{q}_{k-1}\mathbf{f}_k\mathbf{q}^\star_{k-1}-\mathbf{g})dt\\ \mathbf{\Omega}(\boldsymbol{\omega}_k dt)\mathbf{q}_{k-1}\end{bmatrix} \tag{1}$$

where k is a time index, dt is the time differential, $\mathbf{p}_k$ is the position, $\mathbf{v}_k$ is the velocity, $\mathbf{f}_k$ is the specific force, $\mathbf{g}=[0,0,g]$ is the gravity, and $\boldsymbol{\omega}_k$ is the angular rate (all in $\mathbb{R}^3$). Further, $\mathbf{q}_k$ is the quaternion describing the orientation of the system, the triple product $\mathbf{q}_{k-1}\mathbf{f}_k\mathbf{q}^\star_{k-1}$ denotes the rotation of $\mathbf{f}_k$ by $\mathbf{q}_k$, and $\mathbf{\Omega}(\cdot)$ is the quaternion update matrix. Refer to [6,7] for further details of inertial navigation. For analytical convenience, the orientation $\mathbf{q}_k$ may interchangeably be represented by the equivalent Euler angles (roll,pitch,yaw) $\boldsymbol{\theta}_k=[\phi_k,\theta_k,\psi_k]$. Note that $[\cdot,\dots]$ is used to denote a column vector.

Statistical models for the measurement errors in $\mathbf{f}_k$ and $\boldsymbol{\omega}_k$, and an error model of (1) are used to propagate statistics of the errors of the states. To estimate the error states, stationary time instances are detected based on the condition $Z(\{\mathbf{f}_\kappa,\boldsymbol{\omega}_\kappa\}_{W_k})<\gamma_{\text{Z}}$, where $Z(\cdot)$ is some zero-velocity test statistic, $\{\mathbf{f}_\kappa,\boldsymbol{\omega}_\kappa\}_{W_k}$ is the inertial measurements over some time window W_k, and γ_{Z} is a zero-velocity detection threshold. Refer to [8,9] for further details. The implied zero-velocities are used as pseudo-measurements, so called zero-velocity updates (ZUPTs), with the measurement matrix $\mathbf{H}=[\mathbf{0}\ \mathbf{I}\ \mathbf{0}]$: where $\mathbf{I}$ and $\mathbf{0}$ are 3×3 identity and zero matrices, respectively. Refer to [10] for a detailed treatment of aided navigation.

Unfortunately, all states are not observable based on the ZUPTs. During periods of consecutive ZUPTs, the system (1) becomes essentially linear and time invariant. Zero-velocity for consecutive time instances means no acceleration and ideally $\mathbf{f}_k=\mathbf{q}_k^\star\mathbf{g}\mathbf{q}_k$, giving the system and observability matrices

$$\mathbf{F}=\begin{bmatrix}\mathbf{I} & \mathbf{I}dt & \mathbf{0}\\ \mathbf{0} & \mathbf{I} & [\mathbf{g}]_\times dt\\ \mathbf{0} & \mathbf{0} & \mathbf{I}\end{bmatrix} \text{ and } \begin{bmatrix}\mathbf{H}\\ \mathbf{HF}\\ \mathbf{HF}^2\end{bmatrix}=\begin{bmatrix}\mathbf{0} & \mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{I} & [\mathbf{g}]_\times dt\\ \mathbf{0} & \mathbf{I} & 2[\mathbf{g}]_\times dt\end{bmatrix}.$$

where $[\cdot]_\times$ is the cross-product matrix. Obviously, the position and heading (errors) are not observable, whereas the velocity, roll and pitch are. This implies that the covariances of the observable states decay as one over the number of consecutive ZUPTs. Consequently, during stand-still, after a reasonable number of ZUPTs, the state estimate covariance becomes

$$\text{cov}\left((\mathbf{p}_k,\mathbf{v}_k,\boldsymbol{\theta}_k)\right)\approx\begin{bmatrix}\mathbf{P}_{\mathbf{p}_k} & \mathbf{0}_{3\times5} & \mathbf{P}_{\mathbf{p}_k,\psi_k}\\ \mathbf{0}_{5\times3} & \mathbf{0}_{5\times5} & \mathbf{0}_{5\times1}\\ \mathbf{P}^\top_{\mathbf{p}_k,\psi_k} & \mathbf{0}_{1\times5} & P_{\psi_k,\psi_k}\end{bmatrix} \tag{2}$$

where $\mathbf{P}_{x,y}=\text{cov}(x,y)$, $\mathbf{P}_x=\text{cov}(x)=\text{cov}(x,x)$, $(\cdot)^\top$ denotes the transpose, and $\mathbf{0}_{n\times m}$ denotes a zero matrix of size $n\times m$.

The covariance matrix (2) tells us that the errors of $\mathbf{p}_k$ and ψ_k are uncorrelated with those of $\mathbf{v}_k$ and $[\phi_k,\theta_k]$. Together with the Markovian assumption of the state space models and the translational and in-plane rotation invariance of (1), this implies that errors of future states $\mathbf{v}_{k+n}$ and $[\phi_{k+n},\theta_{k+n}]$ $(n>0)$ are uncorrelated with those of $\mathbf{p}_k$ and ψ_k. Consequently, future ZUPTs cannot be used to deduce information about the current position and heading errors. In turn, this means that, considering only the ZUPTs, it makes no difference if we reset the system and add the new relative position and heading to those before the reset. However, for other information sources, we must keep track of the global (total) error covariance of the position and heading estimates.

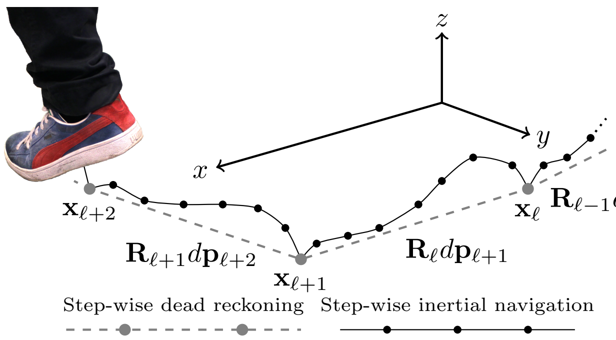

z
y
x
$\mathbf{x}_{\ell+2}$
$\mathbf{x}_\ell$
$\mathbf{R}_{\ell-1}d\mathbf{p}_\ell$
$\mathbf{R}_{\ell+1}d\mathbf{p}_{\ell+2}$
$\mathbf{R}_\ell d\mathbf{p}_{\ell+1}$
$\mathbf{x}_{\ell+1}$
Step-wise dead reckoning
Step-wise inertial navigation

Fig. 2: Illustration of the step-wise inertial navigation and the step-wise dead reckoning. The displacement and heading change over a step given by the inertial navigation are used to perform the step-wise dead-reckoning.

Resetting the system means setting position $\mathbf{p}_k$ and heading ψ_k and corresponding covariances to zero. Denote the position and heading estimates at a reset ℓ by $d\mathbf{p}_\ell$ and $d\psi_\ell$. These values can be used to drive the step-wise dead reckoning

$$\begin{bmatrix}\mathbf{x}_\ell\\ \chi_\ell\end{bmatrix}=\begin{bmatrix}\mathbf{x}_{\ell-1}\\ \chi_{\ell-1}\end{bmatrix}+\begin{bmatrix}\mathbf{R}_{\ell-1}d\mathbf{p}_\ell\\ d\psi_\ell\end{bmatrix}+\mathbf{w}_\ell \tag{3}$$

where $\mathbf{x}_\ell$ and χ_ℓ are the position and heading of the inertial navigation system relative to the navigation frame,

$$\mathbf{R}_\ell=\begin{bmatrix}\cos(\chi_\ell) & -\sin(\chi_\ell) & 0\\ \sin(\chi_\ell) & \cos(\chi_\ell) & 0\\ 0 & 0 & 1\end{bmatrix}$$

is the in-plane rotation matrix from the local coordinate frame of the last reset to the navigation frame, and $\mathbf{w}_\ell$ is a (by assumption) white noise with covariance,

$$\begin{aligned}\text{cov}(\mathbf{w}_\ell)&=\text{cov}\big([\mathbf{R}_{\ell-1}d\mathbf{p}_\ell,d\psi_\ell]\big)\\ &=\begin{bmatrix}\mathbf{R}_{\ell-1}\mathbf{P}_{\mathbf{p}_\ell}\mathbf{R}^\top_{\ell-1} & \mathbf{R}_{\ell-1}\mathbf{P}_{\mathbf{p}_\ell,\psi_\ell}\\ \mathbf{P}^\top_{\mathbf{p}_\ell,\psi_\ell}\mathbf{R}^\top_{\ell-1} & P_{\psi_\ell,\psi_\ell}\end{bmatrix}.\end{aligned} \tag{4}$$

The noise $\mathbf{w}_\ell$ in (3) represents the accumulated uncertainty in position and heading since the last reset, i.e. the essentially non-zero elements in (2) transformed to the navigation frame. As described in Section IV, the dead reckoning (3) is easily used to estimate $\mathbf{x}_\ell$ and χ_ℓ and their error covariances from $d\mathbf{p}_\ell$ and $d\psi_\ell$ and related covariances. The relation between the step-wise inertial navigation and dead reckoning is illustrated in Fig. 2. Refer to [5] for further details.

The dead reckoning statistical interface (3)-(4) separates the tracking from the high rate inertial navigation. If the ZUPT-aided inertial navigation is implemented with the inertial sensors on the foot, then only the low rate steps $d\mathbf{p}_\ell$ and $d\psi_\ell$ and the related uncertainties $\mathbf{P}_{\mathbf{p}_\ell}$, $\mathbf{P}_{\mathbf{p}_\ell,\psi_\ell}$, and P_{ψ_ℓ,ψ_ℓ} have to be transferred to the user. Due to the decreased rate and communication bandwidth requirements, a wireless communication link can easily be used while retaining the high rate inertial measurements. However, this requires support of a suitable hardware platform and dedicated embedded software.

III. The tracking modules

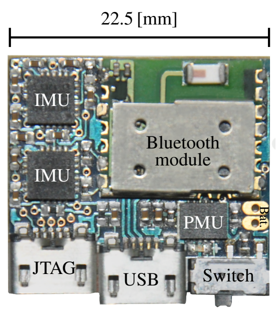
A handful of inertial platforms can be found in the literature, e.g. [11–16], and on the market, e.g. [17–20]. Unfortunately, none of them fulfill all the requirements to implement the inertial navigation and the dead reckoning interface. For the processing, the computational power must be sufficiently high, and it is highly desirable to have a floating point core or coprocessor. To get satisfactory performance, the inertial sensors must have sufficient stability, dynamic range, sampling rate, and analog bandwidth. Further, the platform should be open (for reprogramming) and reasonably well documented. Finally, the platform should have a suitable wireless link and be sufficiently small for convenient foot-mounting. In order to provide these capabilities, we have designed our own hardware platform and developed necessary embedded software, together making up a complete tracking module.

The module is a development and merger of two of our previous platforms: the first generation of the OpenShoe module [11] and the Massive-MIMU platform [21]. Note that the module may also be used as a normal (wireless) IMU.

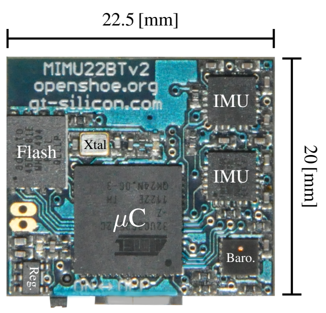
A. Hardware platform

The hardware platform and its components are shown in Fig. 3. A block diagram of all non-passive components of the PCB is shown in Fig. 4. The key components of the PCB are the hardware floating-point capable AT32UC3C2512 microcontroller (μC) from Atmel, the four MPU9150 combined IMU and magnetometer chips from InvenSense (the magnetometers are currently not used), the SPBT2632C2A Bluetooth v3.0 module from STMicroelectronics, and the LTC4067 power manager unit (PMU) from Linear Technology. The PCB also contains a FLASH memory and a barometer, which are not used for the foot-mounted inertial navigation. The size of the 4-layer PCB is 22.5×20 [mm].

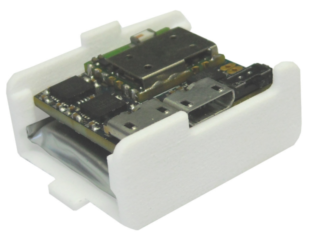
The μC is accessible via a built-in USB (slave) interface and a JTAG interface, both provided through micro-USB connectors. The μC is clocked by a 16 [MHz] crystal. Multiple (four) IMUs are used to boost performance. See [21,22] for Allan variance plots, calibration procedures, and details of the multi-IMU (MIMU) sensory subsystem. The scale range and analogue bandwidth of the accelerometers are ± 16 [g] and 260 [Hz], respectively, and of the gyroscopes ± 2000 [°/s] and 256 [Hz], respectively. The μC communicates with the Bluetooth module over a UART, and the Bluetooth module exposes an SPP profile. The range of the Bluetooth module is ~ 10 [m]. The platform is seamlessly powered from the $4 \times 20 \times 25$ [mm] 150 [mAh] Li-Ion battery or from the USB connector, which also serves as a charging port for the battery.



(a) Top side of the PCB.



(b) Bottom side of the PCB.



(c) PCB, casing bottom, battery.

(d) The complete tracking module.

Fig. 3: PCB and module assemblies. The PCB sits in the casing with the battery below. The casing lid snaps around the lower piece. The USB and the on/off switch are accessible via a slit in the casing. The complete $23.2 \times 31 \times 13.5$ [mm] module is above displayed in approximately natural size.

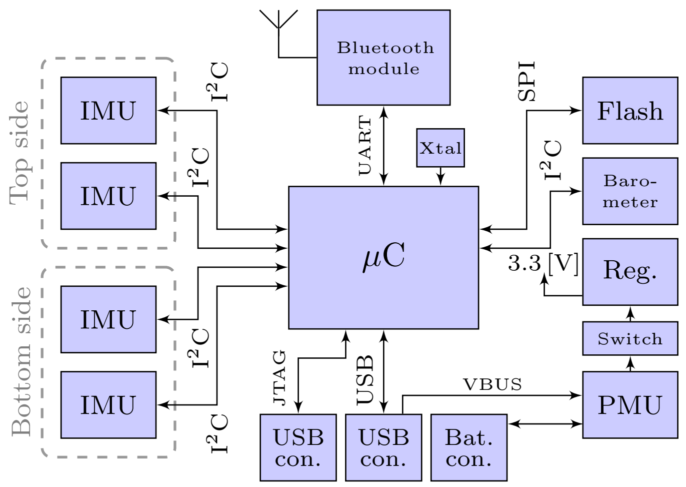


Fig. 4: Block diagram of tracking module hardware. For comparison, see Fig. 3.(a)-(b). The I^{2}C busses of the IMUs are handled in parallel in software. Refer to [21] for details.

At maximum sampling and clock frequency, the battery life is ~ 1.5 [h]. The board runs on regulated 3.3 [V]. The module is turned on/off by a slide switch controlling the regulator. The battery, with in-built protection circuitry, can be charged irrespective of the on/off power status. A fully discharged battery takes ~ 2 [h] to charge. The module casing is designed to be printed with a 3D-printer and features a snap-fit. The complete module dimensions are $23.2 \times 31 \times 13.5$ [mm].

B. Embedded software

The embedded software comprises the actual zero-velocity aided inertial navigation filtering implementations, as well as supportive components such as runtime framework, communication logics and defines, and device drivers. The code is written in C (C99) and uses components from Atmel Software Framework (ASF) for many device drives and low level functions. The code has been developed in Atmel Studio 6 IDE and compiled with avr32gcc. The code runs directly on the μC without any operating system (OS).

The runtime framework provides the high level execution logics. The architecture is of simple loop type, with a main loop essentially calling four subroutines: read data from the IMUs, receive commands, run a user manipulatable process sequence, and transmit data to the user. The loop is paced by a timer interrupt. The flow is illustrated in Fig 5. The serial commands from the user (consisting of a 1 byte command header, a command specific command argument payload, and a 2 byte checksum) can essentially do two things: request a state to be output at a certain frequency and manipulate the process sequence. The content of the process sequence will determine the function of the module. In the process sequence, the algorithm implementations, and other auxiliary functions, are inserted. These functions will in turn change the states. The functions may also manipulate the process sequence itself. This can be used to change the module behavior due to some condition, e.g. initialization has ended. States may also be output at some process function defined condition. The state is transmitted over the interface (USB/Bluetooth), over which the corresponding command was sent, i.e. a request over USB will result in data back over USB. For the Bluetooth interface, there is the option of lossy or lossless communication. In the lossy mode, the module will fire and forget. In the lossless communication mode, the module will buffer new outputs (until the buffer is full, after which it will still lose data) and keep sending the oldest output until it gets a numbered acknowledgement back. However, this will limit the achievable data rate. The timer interrupt of the main loop determines the read out frequency of the IMUs. The internal sampling frequency of the IMUs are 1000 [Hz], and the maximum read out frequency is also 1000 [Hz]. A lower clock frequency, resulting in lower read out frequency, may be set to extend the battery life.

The algorithm implementations are that of [8,23] and a basic aided inertial navigation without any sensor error states, e.g. described in [10]. The code is written in plain C without any linked matrix libraries. The aided inertial navigation arithmetics are in single-precision floating point (since it is running on a 32-bit μC), while the zero-velocity test statistics calculations are implemented in manual fixed point (32-bit integer arithmetics).

The process sequence may run arbitrary linked functions. Consequently, the modules can easily be used to run user implemented functions. New supportive states and commands can freely be defined.

C. Reproducing the platform

The complete module including the hardware platform and the embedded software are released under the permissive Creative Commons Attribution 4.0 International Public License. The modules can freely be reproduced, modified, integrated, or built upon, as long as the original work is appropriately credited. Source material and documentation are provided at www.openshoe.org and linked sources.

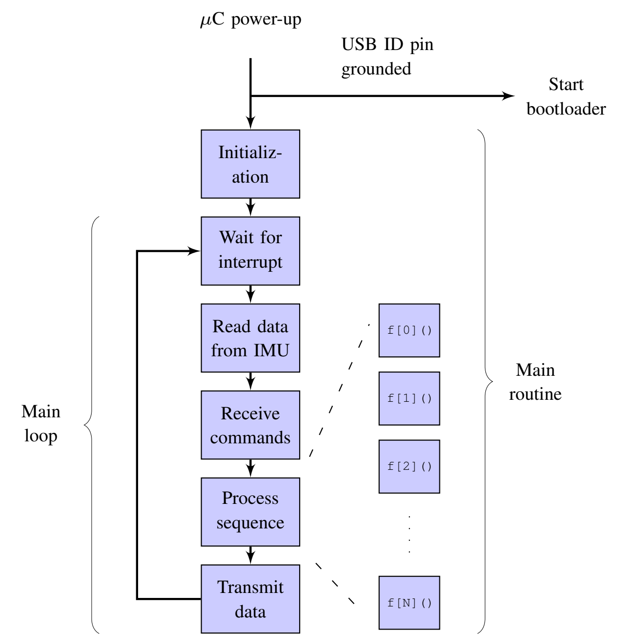


Fig. 5: Flow chart of the runtime framework. The runtime framework runs in an infinite loop that reads data from the IMU, receives commands, cycles through the process sequence, and transmits data to the user.

Reproducing the tracking module entails

- Printing and populating the PCB
- Printing the casing
- Assembling the PCB, the casing, and the battery
- Loading the embedded software onto the platform

For the PCBs, a complete EAGLE 6.5.0 project, Gerber files, and a bill of material (BOM) are provided. The minimum design dimensions are: track size 0.15 [mm], gap size 0.15 [mm], edge clearance 0.25 [mm], drill diameter 0.3 [mm], and annular ring 0.2 [mm]. For the casing, STL-files are provided. For the software, complete Atmel Studio 6 projects are provided.

The AT32UC3C μCs (C2 series) are preloaded with a bootloader such that the modules can be programmed over the USB, using an USB OTG cable (grounding the ID pin on power-up will cause the bootloader to start), without a dedicated JTAG programmer. However, a JTAG programmer may also be used, and for debugging a JTAG debugger is required.

IV. System integration

To use a tracking module, it has to be attached to the foot, powered on, and paired with an application platform as a Bluetooth device. Following a command, it will start providing dead reckoning updates $\{d\mathbf{p}_\ell, d\psi_\ell, \mathbf{P}_{\mathbf{p}_\ell}, \mathbf{P}_{\mathbf{p}_\ell,\psi_\ell}, P_{\psi_\ell,\psi_\ell}\}$. To avoid dropping updates, one reasonably uses the lossless mode of communication.

To track the user and fuse the data with other information sources, the updates have to be "summed up". This is done by

$$\begin{bmatrix}\hat{\mathbf{x}}_\ell\\ \hat{\chi}_\ell\end{bmatrix}=\begin{bmatrix}\hat{\mathbf{x}}_{\ell-1}\\ \hat{\chi}_{\ell-1}\end{bmatrix}+\begin{bmatrix}\hat{\mathbf{R}}_{\ell-1}d\mathbf{p}_\ell\\ d\psi_\ell\end{bmatrix}$$

$$\mathbf{P}_\ell=\mathbf{F}(\hat{\theta}_{\ell-1},dx_\ell,dy_\ell)\mathbf{P}_{\ell-1}\mathbf{F}^\top(\hat{\theta}_{\ell-1},dx_\ell,dy_\ell)+\begin{bmatrix}\hat{\mathbf{R}}_{\ell-1}\mathbf{P}_{\mathbf{p}_\ell}\hat{\mathbf{R}}_{\ell-1}^\top & \hat{\mathbf{R}}_{\ell-1}\mathbf{P}_{\mathbf{p}_\ell,\psi_\ell}\\ \mathbf{P}_{\mathbf{p}_\ell,\psi_\ell}^\top\hat{\mathbf{R}}_{\ell-1}^\top & P_{\psi_\ell,\psi_\ell}\end{bmatrix}$$

where the system matrix is

$$\mathbf{F}(\hat{\theta}_{k\text{-}1},dx_\ell,dy_\ell)=\begin{bmatrix}1&0&0&-\sin(\hat{\theta}_{\ell\text{-}1})dx_\ell-\cos(\hat{\theta}_{\ell\text{-}1})dy_\ell\\0&1&0&\cos(\hat{\theta}_{\ell\text{-}1})dx_\ell-\sin(\hat{\theta}_{\ell\text{-}1})dy_\ell\\0&0&1&0\\0&0&0&1\end{bmatrix}.$$

With the dead reckoning interface, this is the only foot-mounted inertial navigation specific processing which has to be performed. For complexity comparison and a detailed description of the complete foot-mounted inertial navigation processing, see [8,24]. The estimates $[\hat{\mathbf{x}}_\ell,\hat{\chi}_\ell]$ and $\mathbf{P}_\ell$ provide the mean and covariance of the state (position and heading) of the user and may be appended to other states and covariances of the system. Combined with standard Bayesian state estimation methods, see e.g. [25], this may be used to fuse the inertial tracking with other information sources. For further details and examples of how to fuse the tracking with other information sources, see [5,26–28].

V. Performance

The tracking performance of foot-mounted inertial navigation systems is highly dependent on the true trajectory. Consequently, the performance needs to be quantified with respect to a reasonably well defined trajectory. For longer trajectories, the position error, induced by the growing heading error, will dominate. The worst case is just a straight line which gives the largest leverage for the heading error. Consequently, in this basic performance assessment, we use a straight line trajectory to quantify the performance of the modules.

Fifty 100 [m] straight lines (100 [m], plus 10 [m] for alignment) were walked on a hard leveled concrete surface at normal gait speed by a single subject. The modules were firmly attached on top of the forefoot. A module was worn on each forefoot and 25 trajectories were walked with each pair, totaling 100 trajectories. Plates with imprints of the shoes were placed at 0 [m], 10 [m], and 110 [m]. The first plate provided the start position. The initial headings were set such that all trajectories were aligned at the second plate. Finally, the last plate provided the stop position. Plates were placed in both directions such that half of the trajectories were walked in one direction of the line, and half in the other direction. To enable online gyro bias calibration and for convenient alignment, the subject stopped for 7 [s] at each plate. The modules were calibrated as described in [22] 5 days before the data were recorded.

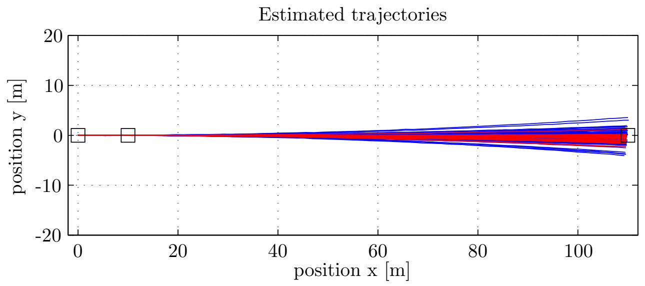


Fig. 6: Tracking results from fifty 100 [m] straight-line trajectories. The results from single modules are shown in blue and dual modules in red. The size of the reference plates have been exaggerated.

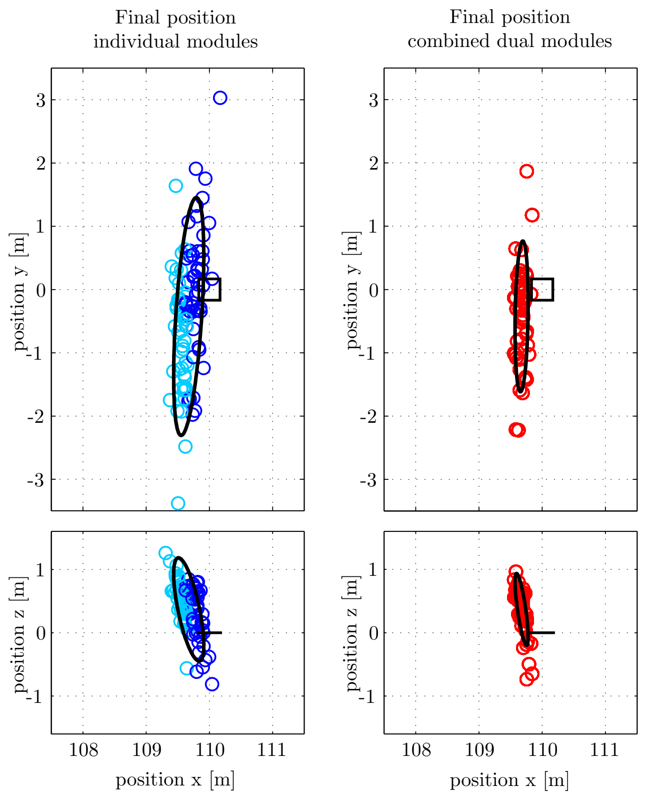


Fig. 7: Scatter plots of the final positions given by individual modules (left) and dual modules (right). Dark and light blue indicate module worn on left and right foot, respectively. The covariances are indicated by 1σ confidence ellipsoids in black. The true final positions are indicated by the plates.

The tracking results from the 100 trajectories are shown in blue in Fig. 6. Since dual modules are preferably used [5], the combined tracking results from the modules on opposite feet are shown in red. Close-ups of the final positions are also shown in Fig. 7. Comparative tracking data can be found in [11,29–32]. Algorithms for combining the tracking can be found in [5,31,33]. The mean systematic errors in length and height estimates are seen to be below 0.5%. A couple of decimeters of the lateral errors (both mean and spread) are most likely due to alignment errors. The alignment plates were placed by sighting, and the repeatability of the foot-placement is only accurate to a centimeter level. Further discussions about the observed systematic errors can be found in [30].

Note that the shown tracking is performed with only inertial measurements and no magnetic measurements were used. Also, no attempt has been made to remove the systematic length and height errors. *The displayed estimates are the raw results of the measurements and the sensor error, the low level mechanization, and the zero-velocity hypothesis assumptions (models).* Heuristics could potentially be added to mitigate the systematic errors.

VI. Conclusion and final remarks

The application layer integration of foot-mounted inertial navigation is demanding, due to exposed complexity, sensor placement, and high data rates. The difficulties are greatly reduced by the dead reckoning interface supported by wireless communication links. Consequently, the presented open-source wireless tracking modules, implementing the foot-mounted inertial navigation with the dead reckoning interface, simplifies the user of the tracking technology significantly, and thereby facilitates the application development in general.

The improved performance that the foot-mounted inertial navigation can offer, in comparison with heuristic inertial pedestrian tracking techniques, and its simplified use provide an easily reachable enhanced tracking performance for a range of applications, as well as the potential for many new applications, for which the tracking performance of easily available (heuristic) pedestrian tracking technologies have been insufficient. The potential use scenarios are numerous with a range of security, first responder, military, social, management, retail, entertainment, sports, and medical applications already being investigated. The hope of the authors is that this work will support and encourage the introduction of the foot-mounted inertial navigation tracking technology over the whole application range.

References

[1] L. Hutchings, B. Way, and C. Valley, "Systems and method for measuring movement of objects," US Patent 5,724,265, 1998 (filed 1995).

[2] R. Harle, "A survey of indoor inertial positioning systems for pedestrians," *Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1281–1293, 2013.

[3] E. Foxlin, "Pedestrian tracking with shoe-mounted inertial sensors," *IEEE Comput. Graph. Appl.*, vol. 25, pp. 38 –46, 2005.

[4] M. Laverne, M. George, D. Lord, A. Kelly, and T. Mukherjee, "Experimental validation of foot to foot range measurements in pedestrian tracking," in *ION GNSS*, (Portland, OR, US), 19-23 Sept. 2011.

[5] J.-O. Nilsson, D. Zachariah, I. Skog, and P. Händel, "Cooperative localization by dual foot-mounted inertial sensors and inter-agent ranging," *EURASIP J. Adv. Sig. Pr.*, vol. 2013:164, 2013.

[6] K. R. Britting, *Inertial Navigation Systems Analysis*. John Wiley & Sons, Inc., 1971.

[7] C. Jekeli, *Inertial Navigation Systems with Geodetic Applications*. de Gruyter, 2001.

[8] I. Skog, P. Händel, J.-O. Nilsson, and J. Rantakokko, "Zero-velocity detection: An algorithm evaluation," *IEEE Trans. Biomed. Eng*, vol. 57, pp. 2657 –2666, nov. 2010.

[9] I. Skog, J.-O. Nilsson, and P. Händel, "Evaluation of zero-velocity detectors for foot-mounted inertial navigation systems," in *Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on*, (Zürich, Switzerland), 15-17 Sept. 2010.

[10] J. A. Farrell, *Aided Navigation*. Mc Graw Hill, 2008.

[11] J.-O. Nilsson, I. Skog, P. Händel, and K. Hari, "Foot-mounted INS for everybody – An open-source embedded implementation," in *Position, Location and Navigation Symposium (PLANS), 2012 IEEE/ION*, (Myrtle Beach, SC, USA), 23-26 Apr. 2012.

[12] F. Varesano, "FreeIMU: An open hardware framework for orientation and motion sensing," *CoRR*, vol. abs/1303.4949, 2013.

[13] D. Comotti, M. Galizzi, and A. Vitali, "neMEMSi: One step forward in wireless attitude and heading reference systems," in *Inertial Sensors and Systems (ISISS), 2014 International Symposium on*, (Laguna Beach, CA, USA), 25-26 Feb. 2014.

[14] M. Benocci, L. Rocchi, E. Farella, L. Chiari, and L. Benini, "A wireless system for gait and posture analysis based on pressure insoles and inertial measurement units," in *Pervasive Computing Technologies for Healthcare, 2009. PervasiveHealth 2009. 3rd International Conference on*, (London, UK), 1-3 Apr. 2009.

[15] R. Ashkar, M. Romanovas, V. Goridko, M. Schwaab, M. Traechtler, and Y. Manoli, "A low-cost shoe-mounted inertial navigation system with magnetic disturbance compensation," in *Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on*, (Montbéliard - Belfort, France), 28-31 Oct. 2013.

[16] F. Höflinger, J. Müller, R. Zhang, L. Reindl, and W. Burgard, "A wireless micro inertial measurement unit (IMU)," *IEEE Trans. Instrum. Meas.*, vol. 62, pp. 2583–2595, Sept 2013.

[17] "3-Space Product Family, YEI Technology (www.yeitechnology.com/3-space-product-family)." (Retrieved 30 Jul. 2014).

[18] "x-BIMU/x-OSC, X-IO Technologies (www.x-io.co.uk/products/)." (Retrieved 30 Jul. 2014).

[19] "Opal, APDM Inc. (adpm.com/Wearable-Sensors/Opal)." (Retrieved 30 Jul. 2014).

[20] "MotionNode, Motion Workshop (www.motionnode.com)." (Retrieved 30 Jul. 2014).

[21] I. Skog, J.-O. Nilsson, and P. Händel, "An open-source multi inertial measurement unit (MIMU) platform," in *Inertial Sensors and Systems (ISISS), 2014 International Symposium on*, (Laguna Beach, CA, USA), 25-26 Feb. 2014.

[22] J.-O. Nilsson, I. Skog, and P. Händel, "Aligning the forces – Eliminating the misalignments in IMU arrays," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 10, pp. 2498–2500, 2014.

[23] J.-O. Nilsson and P. Händel, "Standing still with inertial navigation," in *Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on*, (Montbéliard - Belfort, France), 28-31 Oct. 2013.

[24] A. Jiménez, F. Seco, J. Prieto, and J. Guevara, "Indoor pedestrian navigation using an INS/EKF framework for yaw drift reduction and a foot-mounted IMU," in *Positioning Navigation and Communication (WPNC), 2010 7th Workshop on*, (Dresden, Germany), 11-12 Mar. 2010.

[25] D. Simon, *Optimal state estimation*. John Wiley & Sons, Inc., 2006.

[26] J.-O. Nilsson and P. Händel, "Recursive Bayesian initialization of localization based on ranging and dead reckoning," in *Intelligent Robots and Systems, 2013. IROS 2013. IEEE/RSJ International Conference on*, (Tokyo, Japan), 3-7 Nov. 2013.

[27] B. Krach and P. Robertson, "Integration of foot-mounted inertial sensors into a Bayesian location estimation framework," in *Positioning, Navigation and Communication, 2008. WPNC 2008. 5th Workshop on*, (Hannover, Germany), 27 Mar. 2008.

[28] J.-O. Nilsson, C. Schüldt, and P. Händel, "Voice radio communication, 3D audio and the tactical use of pedestrian localization," in *Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on*, (Montbéliard - Belfort, France), 28-31 Oct. 2013.

[29] J. Rantakokko, P. Strömbäck, E. Emilsson, and J. Rydell, "Soldier positioning in GNSS-denied operations," in *Navigation Sensors and Systems in GNSS Denied Environments (SET-168)*, (Izmir, Turkey), 8-9 Oct. 2012.

[30] J.-O. Nilsson, I. Skog, and P. Händel, "A note on the limitations of ZUPTs and the implications on sensor error modeling," in *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on*, (Sydney, Australia), 13-15 Nov. 2012.

[31] I. Skog, J.-O. Nilsson, D. Zachariah, and P. Händel, "Fusing information from two navigation system using an upper bound on their maximum spatial separation," in *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on*, (Sydney, Australia), 13-15 Nov. 2012.

[32] I. Skog, J.-O. Nilsson, and P. Händel, "Pedestrian tracking using an IMU array," in *IEEE CONECCT*, (Bangalore, India), 6-7 Jan. 2014.

[33] G. V. Prateek, R. Girisha, K. Hari, and P. Händel, "Data fusion of dual foot-mounted INS to reduce the systematic heading drift," in *Proc. 4th International Conference on Intelligent Systems, Modelling and Simulation*, (Bangkok, Thailand), 29-31 Jan. 2013.