

一种基于智能手机传感器的设备指纹提取方法

技术领域

本发明属于移动终端领域，基于智能手机传感器的设备指纹提取技术，主要涉及了对智能手机的传感器信息收集、推验、校准等操作，具体涉及了对于陀螺仪、加速度计、磁力计三类传感器的偏差增益推算验证以及智能手机设备指纹生成验证。

背景技术

传感器是智能手机的重要组成，每一部智能手机都包含很多传感器原件，从相机、麦克风、光线感应到 GPS、陀螺仪、加速度计等等，这些都是基于 MEMS (Mississippi Emergency Management Agency) 技术。对于一个良好的传感器，精度是至关重要的，因此设备制造商一般会使用工厂校准来补偿制造过程中产生的偏差。对传感器的输出数据进行分析就能推算出每个传感器的校准数据，这个校准数据就可以作为设备的指纹。

目前智能手机主要包括两大操作系统：Android、iOS。传统的 Android 系统设备的指纹提取主要是依靠 IMEI、MAC、ANDROID_ID 等信息来生成，一般采用一个或多个元素来作为设备指纹；传统的 iOS 系统设备指纹的提取主要会依靠 IDFA、IDFV、OpenUDID 等值来生成。随着 Android 系统的升级，Google 对于 IMEI、MAC 等值的限制获取，传统的指纹获取已经不能满足唯一标识设备，且这些信息的获取需要权限申请会明显的影响用户体验，同时在 Android 低版本中这些值也存在被篡改、冒用等风险；当前 iOS 系统在完全还原初始的情况下也是不能得到设备唯一指纹的，这些对于设备归因、追溯都存在的问题。

发明内容

本发明的目的是通过以下技术方案实现的。

为了解决这些问题，本发明提出了一套以智能手机传感器信息为基础的设备指纹提取技术。为达上述目的，本发明第一方面实施例提出了一种基于智能手机传感器的设备指纹提取方法，包括如下步骤：

步骤一：数据采集；
步骤二：数据预处理；
步骤三：ADC 值恢复；
步骤四：增益矩阵的估算；
步骤五：有效性检查；
步骤六：设备指纹 ID 生成。

为达上述目的，本发明第二方面实施例提出了一种电子设备，包括：存储器、处理器及存储在所述存储器上并可在所述处理器上运行的计算机程序，所述处理器运行所述计算机程序时执行以实现所述的方法。

为达上述目的，本发明第三方面实施例提出了一种计算机可读存储介质，其上存储有计算机程序，所述程序被处理器执行时实现所述的方法。

本发明的优点在于：能够准确的提取设备指纹，在抽样对一些设备进行恢复出厂设置、刷机升级、时间打点测试时，设备指纹 ID 都没有发生变化，能够唯一标识设备。

附图说明

通过阅读下文优选实施方式的详细描述，各种其他的优点和益处对于本领域普通技术人员将变得清楚明了。附图仅用于示出优选实施方式的目的，而并不认为是对本发明的限制。而且在整个附图中，用相同的参考符号表示相同的部件。在附图中：

图 1 示出了根据本发明实施方式的一种基于智能手机传感器的设备指纹提取方法流程图；

图 2 示出了根据本发明实施方式的传感器采集数据示例图；

图 3 示出了本发明一实施例所提供的一种电子设备的结构示意图；

图 4 示出了本发明一实施例所提供的一种计算机介质的示意图。

具体实施方式

下面将参照附图更详细地描述本发明的示例性实施方式。虽然附图中显示了本发明的示例性实施方式，然而应当理解，可以以各种形式实现本发明而不

应被这里阐述的实施方式所限制。相反，提供这些实施方式是为了能够更透彻地理解本发明，并且能够将本发明的范围完整的传达给本领域的技术人员。

需要注意的是，除非另有说明，本发明使用的技术术语或者科学术语应当为本发明所属领域技术人员所理解的通常意义。

另外，术语“第一”和“第二”等是用于区别不同对象，而不是用于描述特定顺序。此外，术语“包括”和“具有”以及它们任何变形，意图在于覆盖不排他的包含。例如包含了一系列步骤或单元的过程、方法、系统、产品或设备没有限定于已列出的步骤或单元，而是可选地还包括没有列出的步骤或单元，或可选地还包括对于这些过程、方法、产品或设备固有的其它步骤或单元。

在本发明的工作中，本发明是通过陀螺仪、加速度计、磁力计等传感器的输出来推断出每个传感器的出厂校准数据，用这些校准数据来构建出一个有效的设备指纹，得到的这个设备指纹能够有效的标识设备的唯一性。本发明选择使用传感器信息是因为这些数据的获取不需要任何的权限申请、Android 和 iOS 系统都能有效获取具有共通性，且有效有值率比较高，样本数据的采集可以在一秒内完成，提取的指纹信息不会随着时间的推移和恢复出厂设置、更新升级等操作而发生变化。本发明证明，本发明的方法可以为设备生成唯一指纹。

本发明采用的方法是根据对陀螺仪、加速度计和磁力计等传感器输出去推断出每个设备的出厂校准数据。MEMS 传感器通常是通过模数转换器（ADC）将模拟量转换为数字量并存储在寄存器内。对于三轴运动传感器，ADC 的输出一般表示为 $A = [A_x, A_y, A_z]^T$ ，根据标准传感器校准模型，传感器标准输出可以表示为：

$$\begin{pmatrix} O_x \\ O_y \\ O_z \end{pmatrix} = \begin{bmatrix} P_x & 0 & 0 \\ 0 & P_y & 0 \\ 0 & 0 & P_z \end{bmatrix} \begin{bmatrix} N_{xx} & N_{xy} & N_{xz} \\ N_{yx} & N_{yy} & N_{yz} \\ N_{zx} & N_{zy} & N_{zz} \end{bmatrix} \begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix} + \begin{bmatrix} D_x \\ D_y \\ D_z \end{bmatrix}$$

在这里， $P_i \in P$ 是指比例因子； $N_{ij} \in N$ 是指轴 i 和 j 之间的非正交性；

$B_i \in B$ 是指标准偏差；A 是指传感器的灵敏度或增益，定义为输出信号与测量属性之间的比率，传感器的标称增益是传感器的预期工作灵敏度。在本发明中使用 M 来表示传感器的标称增益。如果传感器是理想的，则其标度矩阵 P 和非正交性矩阵 N 应分别为 $M \cdot I$ 和 I，其中 I 是单位矩阵。所以上面的等式可以

进一步简化为:

$$O = HA + D$$

这里 $H = PN$ 表示标称增益。传感器的校准标准有很多纬度, 包括有高精度设备, 多位置, 卡尔曼滤波器和基于视觉等, 厂商一般会选择只校准偏移矢量来降低成本。在工厂校准完成后会将这些校准数据写入到硬件存储中, 这些数据不会随时间变化, 也不会轻易被改写。

在设备经过工厂校准后, 增益矩阵 H 和偏置矩阵 D 都应该是唯一的, 所以可以恢复这两个数据的任何一个都可以当做设备的指纹。本发明具体实施例讲述增益矩阵 H 的恢复。为恢复增益矩阵本发明需要知道每个传感器件的标称增益, 而设备的标称增益一般是在厂商的传感器件规范中都能找到的, 如 HUAWEI Mate 30 的陀螺仪标称增益为 60 mdps、iPhone X 的陀螺仪标称增益为 61 mdps、Google Pixel3XL 的陀螺仪标称增益为 64 mdps 等等。通过两个传感器的差值可以消除偏置矩阵 D , 实际增益矩阵和理想增益矩阵很相近, 但不相等。

本发明生成传感器指纹基本方法包括六个步骤: 数据采集、数据预处理、ADC 值恢复、增益矩阵估算、有效性检查、设备指纹 ID 生成。当设备处于静止或者缓慢移动的状态时获取传感器指纹是能够比较有效的生成。下面本发明以陀螺仪数据生成指纹为例。

实施例 1

步骤一: 数据采集

首先编写应用程序 app 在设备上采集陀螺仪传感器信息, 经过本发明测试在 Android 设备上不同的机型采集效率不同, 在 HUAWEI Mate 30 一秒钟能采集约 400 组数据, 在 GooglePixel3XL 一秒钟大约能采集 100 组数据, 而在 iOS 设备上一秒钟大约能采集到 110 组数据。这里采集 100 组数据就足够了, 本发明用 $O = [O_0, O_1, \dots, O_{n-1}]$ 来标识单组组数数据, 用 $O_i = [O_x, O_y, O_z]^T$ 来表示不同轴的获取。

步骤二: 数据预处理

在得到数据后, 首先对数据进行差分计算, 以消除偏置矩阵误差 D , 用以

下等式得到 ΔO ：

$$\Delta O = [O_1 - O_0, O_2 - O_1, \dots, O_{n-1} - O_{n-2}]$$

步骤三：ADC 值恢复

这一步本发明主要是恢复 ΔA ，它是连续 ADC 之间的差值， ΔA 的获取可以从 $\Delta A = H^{-1}\Delta O$ 算式得出。 H^{-1} 是实际增益矩阵 H 的逆矩阵，但是实际增益矩阵这里是未知的，所以本发明可以用标称增益先临时代替实际增益值：

$H_0 = M_H \cdot I$ ， M_H 是陀螺仪的标称增益，这个值本发明可以从厂商获取到。在

这里本发明只取 ΔA 的整数值，通过以下等式估算出：

$$\tilde{\Delta A} = \text{round}(H_0^{-1}\Delta O) \quad (1)$$

此时的 H_0 还不等于 H ，这只是个估计值，所以 $\tilde{\Delta A}$ 还不是真正要得到的数据，本发明还需要通过以下等式剔除无效值：

$$E_{\Delta A}^r = \left| \tilde{\Delta A} - H_0^{-1}\Delta O \right| \quad (2)$$

为确保数据的准确性，需要将超出阈值的数据剔除，此时可以设置阈值为 0.1。

根据 ADC 的值来剔除，最后保留的 ADC 值可以当做真实的数据。此时 ΔA 的绝对值很小，只适用于静止或缓慢移动的状态。

步骤四：增益矩阵的估算

在获取了 ADC 值 ΔA 后，本发明可以用以下等式估算出标称增益 $\tilde{H}$ ：

$$\tilde{H} = \Delta O / \Delta A$$

此时矩阵右除运算返回 H 的最小二乘解 $\tilde{H}\Delta A = \Delta O$ 。

步骤五：有效性检查

为了量化 $\tilde{H}$ 和真实 H 的偏差，本发明用以下等式来计算：

$$E^e = \text{std}(\Delta O - \tilde{H}\Delta A)$$

这里的 std 是标准偏差函数，如果误差很小，则此时的 $\tilde{H}$ 就很接近真实的 H 。

这里以 HUAWEI Mate 30 为例, $\tilde{H}$ 此时得到的值如下:

$$\begin{bmatrix} 0.001065797050075 & -0.000010120011969 & -0.000004793689880 \\ 0.000002929477149 & 0.001071656004373 & 0.000003994741566 \\ 0.000007989483133 & 0.000006924218716 & 0.001070058107746 \end{bmatrix}$$

Mate30 设备的 ADC 分辨率是 16 位的, 本发明可将 rds 转为 dps 并估算出整数值:

$$\tilde{H} \cdot 2^{16} \cdot 180/\pi = \begin{bmatrix} 4002.000000021879 & -37.99999999747552 & -17.999999998606565 \\ 10.99999999956567 & 4024.000000021010 & 14.999999996335514 \\ 29.99999999642596 & 25.9999999965612 & 4018.0000000202235 \end{bmatrix}$$

可以看到在这种情况下, 这些数非常接近整数, 因此可以舍入以获得真正的增益矩阵 H。为了确保可靠, 本发明需要进行误差计算:

$$E_H^r = \left| \tilde{H}^d \cdot 2^{16} - \text{round}(\tilde{H}^d \cdot 2^{16}) \right|$$

这里的 $\tilde{H}^d$ 是指 $\tilde{H}$ 单位为 dps 时的值。此时得到的 E 误差范围可以是 0.01, 当误差大于这个值的时候就说明该组数据不可用, 当前设备可能是在移动的状态, 当出现这种情况的时候本发明需要重新采集数据去计算, 直到误差小到一定程度。

步骤六: 设备指纹 ID 生成

指纹 ID 的生成可以按照如下公式得出:

$$DeviceID = \text{round}(\tilde{H}^d \cdot 2^{16}) - \text{round}(H_0^d \cdot 2^{16})$$

可以说 DeviceID 就是以 dps 为单位提取出标称增益之后的增益矩阵 H, HUAWEI Mate 30 的 DeviceID 示例如下:

$$DeviceID = \begin{bmatrix} 4 & -38 & -18 \\ 11 & 27 & 15 \\ 30 & 26 & 22 \end{bmatrix}$$

这里基本说明了生成陀螺仪设备指纹 ID 的思路和过程, 总的来说这些步骤易于实现, 但是这个要求设备测量的时候是处于静止或者缓慢移动的状态。为了解决这个问题本发明还研究出了一种改进方案, 这个方案解决了设备在移动下的设备指纹提取。见以下实施例 2。

实施例 2

在步骤三内的公式一中，本发明是使用 $H_0 = M_H \cdot I$ 来估算 ΔA 的，但是由于 H_0 不等于 H ，因此舍入值 ΔA 还不同于实际值 ΔA ，通常随着 H_0 中的元素值变大， ΔO 和 $H_0 \cdot \Delta A$ 之间的差异就会增大，从而导致不正确的舍入。在改进方法中，除了数据预处理、ADC 值恢复和增益矩阵估计外，本发明使用与基础步骤相同的方案。这里本发明不直接使用 ΔO 输入算法，而是使用不同范围内的数据更新迭代 H 。具体的包括如下步骤：

步骤一：数据采集

首先编写应用程序 app 在设备上采集陀螺仪传感器信息，经过本发明测试在 Android 设备上不同的机型采集效率不同，在 HUAWEI Mate 30 一秒钟能采集约 400 组数据，在 GooglePixel3XL 一秒钟大约能采集 100 组数据，而在 iOS 设备上一秒钟大约能采集到 110 组数据。这里采集 100 组数据就足够了，本发明用 $O = [O_0, O_1, \dots, O_{n-1}]$ 来标识单组组数数据，用 $O_i = [O_x, O_y, O_z]^T$ 来表示不同轴的获取。

步骤二：数据预处理

这里本发明首先从 ΔO 生成具有较小值的数据，因为较小值不太可能会在舍入误差之外，所以本发明可以对 ΔO 中的元素进行排序，然后去相邻元素之间的差，这样得到的结果值会很小。得到的数据表示为 $O = [O_0, O_1, \dots, O_{n-1}]$ 其中 $O_i = [O_x, O_y, O_z]^T$ 表示单轴数据，首先要对 ΔO_x 的值进行升序排序处理， $[\Delta O]_x$ 表示排序后的数组。类似的本发明对 ΔO_y 、 ΔO_z 的值进行排序，并输出结果 $[\Delta O]_y$ 、 $[\Delta O]_z$ ，然后计算如下：

$$\Delta \Delta O = [\text{diff}([\Delta O]_x) \text{diff}([\Delta O]_y) \text{diff}([O]_z)] \quad (3)$$

其中 diff 函数使用矩阵中的连续向量不同，例如 $\text{diff}(\Delta O)$ 的计算公式为：

$$\text{diff}(\Delta O) = [\Delta O_1 - \Delta O_0, \dots, \Delta O_{n-1} - \Delta O_{n-2}]$$

通过减去相似的向量， $\Delta \Delta O$ 包含了更多较小值的数据，根据等式 1 本发明可

以得到:

$$H^{-1}diff(\Delta O) = diff(\Delta A)$$

这里 $diff(\Delta A)$ 的值都是整数, 结合公式 3 本发明可以看到 $H^{-1}\Delta\Delta O$ 的值应该仅包含整数, 因此本发明可以直接将 $\Delta\Delta O$ 添加到 ΔO 内, 可以多次执行此拓展, 以生成更多的较小值数据。然后本发明根据值的范围从拓展数据中生成多批数据, 并进行更新, 所有元素的绝对值都低于编程增益的乘积, 也就是:

$$\Delta O^{\epsilon_i} = \{\Delta O_j \in \Delta O \mid \max(|\Delta O_j|) < (\epsilon_i + 0.5)M_H\}$$

以此批次进行计算, 然后每个批次的值加倍, 直至该批次与 ΔO 相同为止。然后逐步将 ΔO^{ϵ_i} 进行计算, 并将 H 的值往后更新。

步骤三: ADC 值恢复

这里本发明不是使用 H_0 来估计 ΔA 并舍入误差 $E_{\Delta A}^r$, 而是使用初始值的 H_0 来替换 $\tilde{H}$, 因此上面的公式 1、2 应该更新为:

$$\begin{aligned}\tilde{\Delta A}^{\epsilon_i} &= round(\tilde{H}^{-1} \Delta O^{\epsilon_i}) \\ E_{\Delta A^{\epsilon_i}}^r &= \left| \tilde{\Delta A}^{\epsilon_i} - \tilde{H}^{-1} \Delta O^{\epsilon_i} \right|\end{aligned}$$

步骤四: 增益矩阵的估算

这一步主要是从每批数据 ΔO^{ϵ_i} 区更新 $\tilde{H}$ 。改进方法与基本方法之间的主要区别是改进方法可以在增益矩阵估计之后返回到数据预处理的步骤。在每次更新 $\tilde{H}$ 后, 算法都会去检查 $\Delta O^{\epsilon_i} = \Delta O$ 是否会成立, 如果成立则表示本发明已经处理了所有的输出数据, 并将估计的 $\tilde{H}$ 传递到有效性检查过程, 否则该算法将使用更新的 $\tilde{H}$ 返回到预处理阶段, 并将过滤序数乘 2 进行下一批数据处理。

改进方法是从小范围的数据到整个数据集的更新迭代, 并通过估计的 $\tilde{H}$ 的值来估计 ADC 的输出值, 这样减少了每个估计的误差, 因此即使设备是在移动的状态也能够生成 DeviceID。基本方法可以说明传感器指纹的可行性, 改进

方法可以适用于实际场景并且更加可靠,因此这是实际情况下设备指纹获取的首选方式。

步骤五: 有效性检查

为了量化 $\tilde{H}$ 和真实 H 的偏差,本发明用以下等式来计算:

$$E^e = \text{std}(\Delta O - \tilde{H} \Delta A)$$

这里的 std 是标准偏差函数,如果误差很小,则此时的 $\tilde{H}$ 就很接近真实的 H 。

这里以 HUAWEI Mate 30 为例, $\tilde{H}$ 此时得到的值如下:

$$\begin{bmatrix} 0.001065797050075 & -0.000010120011969 & -0.000004793689880 \\ 0.000002929477149 & 0.001071656004373 & 0.000003994741566 \\ 0.000007989483133 & 0.000006924218716 & 0.001070058107746 \end{bmatrix}$$

Mate30 设备的 ADC 分辨率是 16 位的,本发明可将 rds 转为 dps 并估算出整数值:

$$\tilde{H} \cdot 2^{16} \cdot 180/\pi = \begin{bmatrix} 4002.000000021879 & -37.99999999747552 & -17.999999998606565 \\ 10.99999999956567 & 4024.000000021010 & 14.999999996335514 \\ 29.99999999642596 & 25.9999999965612 & 4018.0000000202235 \end{bmatrix}$$

可以看到在这种情况下,这些数非常接近整数,因此可以舍入以获得真正的增益矩阵 H 。为了确保可靠,本发明需要进行误差计算:

$$E_H^r = \left| \tilde{H}^d \cdot 2^{16} - \text{round}(\tilde{H}^d \cdot 2^{16}) \right|$$

这里的 $\tilde{H}^d$ 是指 $\tilde{H}$ 单位为 dps 时的值。此时得到的 E 误差范围可以是 0.01,当误差大于这个值的时候就说明该组数据不可用,当前设备可能是在移动的状态,当出现这种情况的时候本发明需要重新采集数据去计算,直到误差小到一定程度。

步骤六: 设备指纹 ID 生成

指纹 ID 的生成可以按照如下公式得出:

$$\text{DeviceID} = \text{round}(\tilde{H}^d \cdot 2^{16}) - \text{round}(H_0^d \cdot 2^{16})$$

可以说 DeviceID 就是以 dps 为单位提取出标称增益之后的增益矩阵 H , HUAWEI Mate 30 的 DeviceID 示例如下:

$$\text{DeviceID} = \begin{bmatrix} 4 & -38 & -18 \\ 11 & 27 & 15 \\ 30 & 26 & 22 \end{bmatrix}$$

测试结果

本发明经过真实设备测试，不同设备厂家对于传感器的校准有缺失，对于廉价低端的智能手机厂商一般不会做 MEMS 的校准验证，而有些 Android 厂商仅是对部分机型设备内的器件做了校准，如 HUAWEI Mate 系列、Google Pixel 系列等等，而 iOS 设备早在 iPhone4、iPad3 起就已经对传感器进行工厂校准。对于为传感器原件进行过工厂校准的设备，本发明能准确的提取出设备指纹，根据发明的统计，在测试 5680 部 Android 设备和 4675 部 iOS 设备后，都能准确的提取到设备指纹 ID，准确率是 100%，并抽样对一些设备进行恢复出厂设置、刷机升级、时间打点测试等，设备指纹 ID 都没有发生变化，能够唯一标识设备。

本发明实施方式还提供一种与前述实施方式所提供的基于智能手机传感器的设备指纹提取方法对应的电子设备，以执行上述基于智能手机传感器的设备指纹提取方法，所述电子设备可以是手机、平板电脑、摄像机等，本发明实施例不做限定。

请参考图 3，其示出了本发明的一些实施方式所提供的一种电子设备的示意图。如图 3 所示，所述电子设备 2 包括：处理器 200，存储器 201，总线 202 和通信接口 203，所述处理器 200、通信接口 203 和存储器 201 通过总线 202 连接；所述存储器 201 中存储有可在所述处理器 200 上运行的计算机程序，所述处理器 200 运行所述计算机程序时执行本发明前述任一实施方式所提供的基于智能手机传感器的设备指纹提取方法。

其中，存储器 201 可能包含高速随机存取存储器（RAM：Random Access Memory），也可能还包括非不稳定的存储器（non-volatile memory），例如至少一个磁盘存储器。通过至少一个通信接口 203（可以是有线或者无线）实现该系统网元与至少一个其他网元之间的通信连接，可以使用互联网、广域网、本地网、城域网等。

总线 202 可以是 ISA 总线、PCI 总线或 EISA 总线等。所述总线可以分为地址总线、数据总线、控制总线等。其中，存储器 201 用于存储程序，所述处理器 200 在接收到执行指令后，执行所述程序，前述本发明实施例任一实施方式揭示的所述基于智能手机传感器的设备指纹提取方法可以应用于处理器 200 中，或者由处理器 200 实现。

处理器 200 可能是一种集成电路芯片，具有信号的处理能力。在实现过程中，上述方法的各步骤可以通过处理器 200 中的硬件的集成逻辑电路或者软件形式的指令完成。上述的处理器 200 可以是通用处理器，包括中央处理器 (Central Processing Unit，简称 CPU)、网络处理器 (Network Processor，简称 NP) 等；还可以是数字信号处理器 (DSP)、专用集成电路 (ASIC)、现成可编程门阵列 (FPGA) 或者其他可编程逻辑器件、分立门或者晶体管逻辑器件、分立硬件组件。可以实现或者执行本发明实施例中的公开的各方法、步骤及逻辑框图。通用处理器可以是微处理器或者该处理器也可以是任何常规的处理器等。结合本发明实施例所公开的方法的步骤可以直接体现为硬件译码处理器执行完成，或者用译码处理器中的硬件及软件模块组合执行完成。软件模块可以位于随机存储器，闪存、只读存储器，可编程只读存储器或者电可擦写可编程存储器、寄存器等本领域成熟的存储介质中。该存储介质位于存储器 201，处理器 200 读取存储器 201 中的信息，结合其硬件完成上述方法的步骤。

本发明实施例提供的电子设备与本发明实施例提供的基于智能手机传感器的设备指纹提取方法出于相同的发明构思，具有与其采用、运行或实现的方法相同的有益效果。

本发明实施方式还提供一种与前述实施方式所提供的基于智能手机传感器的设备指纹提取方法对应的计算机可读介质，请参考图 4，其示出的计算机可读存储介质为光盘 30，其上存储有计算机程序（即程序产品），所述计算机程序在被处理器运行时，会执行前述任意实施方式所提供的基于智能手机传感器的设备指纹提取方法。

需要说明的是，所述计算机可读存储介质的例子还可以包括，但不限于相变内存 (PRAM)、静态随机存取存储器 (SRAM)、动态随机存取存储器 (DRAM)、其他类型的随机存取存储器 (RAM)、只读存储器 (ROM)、电可擦除可编程只读存储器 (EEPROM)、快闪记忆体或其他光学、磁性存储介质，在此不再一一赘述。

本发明的上述实施例提供的计算机可读存储介质与本发明实施例提供的基于智能手机传感器的设备指纹提取方法出于相同的发明构思，具有与其存储的应用程序所采用、运行或实现的方法相同的有益效果。

在本说明书的描述中，参考术语“一个实施例”、“一些实施例”、“示例”、“具体示例”、或“一些示例”等的描述意指结合该实施例或示例描述的具体特征、结构、材料或者特点包含于本发明的至少一个实施例或示例中。在本说明

书中，对上述术语的示意性表述不必须针对的是相同的实施例或示例。而且，描述的具体特征、结构、材料或者特点可以在任一个或多个实施例或示例中以合适的方式结合。此外，在不相互矛盾的情况下，本领域的技术人员可以将本说明书中描述的不同实施例或示例以及不同实施例或示例的特征进行结合和组合。

此外，术语“第一”、“第二”仅用于描述目的，而不能理解为指示或暗示相对重要性或者隐含指明所指示的技术特征的数量。由此，限定有“第一”、“第二”的特征可以明示或者隐含地包括至少一个该特征。在本发明的描述中，“多个”的含义是至少两个，例如两个，三个等，除非另有明确具体的限定。

流程图中或在此以其他方式描述的任何过程或方法描述可以被理解为，表示包括一个或更多个用于实现定制逻辑功能或过程的步骤的可执行指令的代码的模块、片段或部分，并且本发明的优选实施方式的范围包括另外的实现，其中可以不按所示出或讨论的顺序，包括根据所涉及的功能按基本同时的方式或按相反的顺序，来执行功能，这应被本发明的实施例所属技术领域的技术人员所理解。

在流程图中表示或在此以其他方式描述的逻辑和/或步骤，例如，可以被认为用于实现逻辑功能的可执行指令的定序列表，可以具体实现在任何计算机可读介质中，以供指令执行系统、装置或设备(如基于计算机的系统、包括处理器的系统或其他可以从指令执行系统、装置或设备取指令并执行指令的系统)使用，或结合这些指令执行系统、装置或设备而使用。就本说明书而言，“计算机可读介质”可以是任何可以包含、存储、通信、传播或传输程序以供指令执行系统、装置或设备或结合这些指令执行系统、装置或设备而使用的装置。计算机可读介质的更具体的示例(非穷尽性列表)包括以下：具有一个或多个布线的电连接部(电子装置)，便携式计算机盘盒(磁装置)，随机存取存储器(RAM)，只读存储器(ROM)，可擦除可编程只读存储器(EPROM 或闪速存储器)，光纤装置，以及便携式光盘只读存储器(CDROM)。另外，计算机可读介质甚至可以是可在其上打印所述程序的纸或其他合适的介质，因为可以例如通过对纸或其他介质进行光学扫描，接着进行编辑、解译或必要时以其他合适方式进行处理来以电子方式获得所述程序，然后将其存储在计算机存储器中。

应当理解，本发明的各部分可以用硬件、软件、固件或它们的组合来实现。在上述实施方式中，多个步骤或方法可以用存储在存储器中且由合适的指令执

行系统执行的软件或固件来实现。如，如果用硬件来实现和在另一实施方式中一样，可用本领域公知的下列技术中的任一项或他们的组合来实现：具有用于对数据信号实现逻辑功能的逻辑门电路的离散逻辑电路，具有合适的组合逻辑门电路的专用集成电路，可编程门阵列(PGA)，现场可编程门阵列(FPGA)等。

本技术领域的普通技术人员可以理解实现上述实施例方法携带的全部或部分步骤是可以通过程序来指令相关的硬件完成，所述的程序可以存储于一种计算机可读存储介质中，该程序在执行时，包括方法实施例的步骤之一或其组合。

此外，在本发明各个实施例中的各功能单元可以集成在一个处理模块中，也可以是各个单元单独物理存在，也可以两个或两个以上单元集成在一个模块中。上述集成的模块既可以采用硬件的形式实现，也可以采用软件功能模块的形式实现。所述集成的模块如果以软件功能模块的形式实现并作为独立的产品销售或使用，也可以存储在一个计算机可读存储介质中。

上述提到的存储介质可以是只读存储器，磁盘或光盘等。尽管上面已经示出和描述了本发明的实施例，可以理解的是，上述实施例是示例性的，不能理解为对本发明的限制，本领域的普通技术人员在本发明的范围内可以对上述实施例进行变化、修改、替换和变型。

以上所述，仅为本发明较佳的具体实施方式，但本发明的保护范围并不局限于此，任何熟悉本技术领域的技术人员在本发明揭露的技术范围内，可轻易想到的变化或替换，都应涵盖在本发明的保护范围之内。因此，本发明的保护范围应以所述权利要求的保护范围为准。