arXiv:2303.01641v1 [cs.RO] 3 Mar 2023

RIOT：基于低成本IMU测量值进行定位的递归惯性里程计Transformer

预印本

James Brotchie*
皇家墨尔本理工大学科学学院，
澳大利亚维多利亚州墨尔本 3000，
s3238455@student.rmit.edu.au

Wenchao Li皇家墨尔本理工大学科学学院，澳大利亚维多利亚州墨尔本 3000，
wenchao.li@rmit.edu.au

Andrew D.Greentree皇家墨尔本理工大学科学学院，澳大利亚维多利亚州墨尔本 3000，
andrew.greentree@rmit.edu.au

Allison Kealy维多利亚州环境、土地、水与规划部 皇家墨尔本理工大学 墨尔本，维多利亚州 3000，澳大利亚
allison.kealy@rmit.edu.au

2023年3月6日

摘要

惯性定位是一项重要技术，可在无法获得外部观测的情况下实现自运动估计。然而，低成本惯性传感器不可避免地受到偏置和噪声的影响，导致误差无限累积，使得通过直接积分求取位置变得不可行。传统数学方法依赖先验系统知识和几何理论，并受预定义动力学的限制。深度学习的最新进展得益于不断增长的数据量和计算能力，使数据驱动的解决方案成为可能，并提供更全面的理解。现有的深度惯性里程计方法依赖于对速度等隐状态的估计，或依赖固定的传感器位置和周期性运动模式。在这项工作中，我们提出将传统的递归状态估计方法应用于深度学习领域。我们的方法将真实位置先验纳入训练过程，利用惯性测量和真实位移数据进行训练，从而实现递归，并学习运动特征以及系统误差中的偏置和漂移。我们提出了两种端到端的姿态不变深度惯性里程计框架，利用自注意力捕捉惯性数据中的空间特征和长程依赖。我们将所提出的方法与一个定制的两层门控循环单元进行比较，该单元以相同方式在相同数据上训练；并在多位用户、不同设备和多种活动上对各方法进行测试。每个网络的序列长度加权相对轨迹误差均值为 ≤ 0.4594 米，凸显了我们在开发这些模型时所采用的学习过程的有效性。

关键词 惯性导航 · 深度学习 · 传感器融合 · 里程计 · 姿态估计 · 轨迹估计 ·自注意力 · 惯性测量单元 (IMU)

1 引言

惯性里程计对移动智能体至关重要，因为它能支持自动驾驶、健康/活动监测、室内导航、人机交互和增强现实/虚拟现实等多种应用中的自运动估计。惯性测量单元（IMU）功耗低、隐私性高，并且能够适应各种环境。因此，它们提供了一种廉价且完全以自我为中心的定位方式。IMU 通常由三维加速度计、三维陀螺仪

*通讯作者。

和三维磁力计组成。通过准确融合来自 IMU 和其他传感器的数据，可以构建可靠的系统，用于估计自主系统和行人导航中的运动与位置。然而，低成本惯性传感器的噪声和偏置水平较高，导致长期惯性导航中的系统误差无限增长 [1]。

神经网络能够采用连续激活函数，这类函数能够天然地理解时间，并能够对复杂的非线性系统行为进行建模；这些行为通常过于复杂，难以通过经典数学方法处理 [2]。循环神经网络（RNN）长期以来一直是序列到序列建模的首选。现有的大多数深度惯性导航解决方案都采用循环神经网络，有些还辅以卷积神经网络（CNN）（见第 2 节）。然而，这些架构存在一些众所周知的局限性，例如无法捕捉长期依赖关系，以及无法并行化的序列计算 [3]。

这些缺陷促使人们开发新的架构。其中最引人注目的是基于自注意力的 Transformer模型，该模型首次提出于 [4], ，自问世以来已在自然语言处理（NLP）领域得到广泛应用。它们在自然语言处理领域取得的成功，推动了其在多个领域的应用。近期，Transformer已应用于计算机视觉（CV） [5], 、自然语言处理 [6], 、时间序列预测 [7], 、图像识别 / 生成 [8], 、文本摘要 [9], 、语音识别 [10] 以及音乐生成 [11]。与循环神经网络不同，这些应用展现了网络对输入序列元素之间长期依赖关系的建模能力，并且能够并行处理。正如 [12], 所指出的，这些能力有助于解决循环神经网络中常见的内存瓶颈。此外，对长短期记忆网络（LSTM，一种循环神经网络变体）与基于自注意力的 Transformer 的效果进行比较后发现，在具有长程依赖关系的数据集上，基于自注意力的技术取得了显著的性能提升 [13]。

与卷积神经网络不同，Transformer 无需特定的设计规范，并且擅长处理集合函数。此外，其简洁的架构借助同质处理单元，能够处理多种模态；事实证明，这种架构对于扩展至大型网络和数据集具有出色的可扩展性。在神经网络中引入自注意力机制，使输入能够彼此交互，并根据其与最终预测的相关性进行评估。尽管这一方法已得到广泛研究，但在惯性里程计领域，利用自注意力和从低成本、含噪惯性传感器获取的原始序列读数进行研究的工作仍然有限。在其他序列到序列学习问题中取得的重大进展表明，应用基于自注意力的技术有望消除对精确动力学模型的需求，并提供比以往基于循环神经网络或卷积神经网络的方法更稳健的解决方案。

2 相关工作

近期研究表明，通过融合惯性传感器，实现一种精确的惯性里程计算法可作为构建更稳健、可靠的自运动估计系统的基础。以下介绍文献中以往提出的深度惯性里程计方法：

[14] 的作者展示了利用智能手机的六维惯性传感器（由三维加速度和三维角速度组成）以及其应用程序编程接口（API）提供的姿态估计来定位行人的可行性。他们的方法名为 RIDI，通过利用人体自然运动中的模式，学习如何预测速度，并使用线性最小二乘法校正线性加速度。近年来，深度学习（DL）的进步进一步推动了数据驱动惯性导航的发展。在 [15], 中，研究人员提出了一种名为 PDRNet 的深度学习方法，用于行人航位推算（PDR）。PDRNet 由一个用于智能手机位置识别的分类网络和一个用于确定距离与航向变化的回归网络组成。这些方法严重依赖大量经过精心调节的参数和用户的步行习惯，因此本质上容易出现快速漂移，且缺乏泛化能力。IONet [16] 采用双向长短期记忆网络（Bi-LSTM）和运动学模型，对速度大小和方向变化率进行回归。RONIN [17] 将为计算机视觉（CV）构建的 ResNet [18], 和 LSTM 用作骨干网络，再次对速度向量进行回归。 [19] 的作者采用预积分和 LSTM，作为补充 IMU 运动模型、实现深度惯性里程计的一种方案。这些统一的深度神经网络在高动态条件下提供了更稳健的解决方案，但仍依赖直接积分。此外，这些方法依赖 IMU 姿态估计，在维度上均有所限制，并且依赖需要预先了解系统动力学的动力学模型。

在 [20] 中，作者提出了 TLIO，同样采用 ResNet 回归三维位移估计和不确定性，使他们能够将相对状态测量紧密融合到随机克隆扩展卡尔曼滤波器（EKF）中，以求解位姿、速度和传感器偏置。由于依赖 EKF，研究表明，该方法在运动高度动态或异常时容易发生系统故障，这与先前的研究结果一致 [21]。类似地，在 [22], 中，作者提出了一种混合方法，采用 LSTM 和 EKF，并通过模块化设计构建，其中包括方向和

位置子系统，称为 IDOL。与依赖系统 API 的先前方法相比，配备专用方向模块并纳入三维磁力计读数的做法被证明更具优势。 [23]的作者为基于智能手机的深度惯性里程计提出了一种新颖的损失函数形式。作者采用 ResNet 风格的神经网络，利用来自 IMU 的 2 秒惯性信号来估计平均速度和运动方向。该研究指出，尽管纳入磁力计读数显然有诸多益处，网络却无法收敛。
最新方法见 [24]，并在 [25], 中进行了进一步阐述。作者提出了一种基于注意力、由旋转等变性监督的惯性里程计方法，该方法基于六维惯性测量单元测量值。他们采用 ResNet 证明，加入基于旋转等变性的自监督辅助任务，并对模型进行联合训练，可以提升模型性能；通过测试时训练策略，还能进一步提升性能。在后续工作中，作者提出了一种用于惯性里程计的混合神经网络模型，该模型结合了 CNN 模块、注意力机制和双向长短期记忆网络。CNN 模块用于从归一化的六维惯性测量单元测量值中提取空间特征，而包括空间注意力机制和通道注意力机制在内的注意力机制则用于优化这些特征。随后，双向长短期记忆网络用于捕捉时间特征。

数据驱动的惯性里程计解决方案的有效性已有充分记录，但这些方法在网络设计方面存在一个共同问题。精心设计的网络能够提升各种应用中的性能 [26] ，而高频采集的惯性测量单元数据，使用循环神经网络和卷积神经网络等传统机器学习方法处理起来颇具挑战。使用循环神经网络处理惯性测量单元数据的一个缺点是存在“冲刷效应”，即网络记忆过去输入的能力会随时间减弱 [27], ，因此难以准确处理较长的数据序列。另一方面，卷积神经网络需要采用深层架构才能覆盖足够大的感受野，从而有效处理惯性测量单元数据，但这可能导致训练和部署期间产生显著的计算开销[28]。

3 预备知识

在本研究中，我们提出了一种基于自注意力编码器-解码器网络模块的深度惯性里程计新方法。为应对高频时间序列数据带来的挑战，我们提出完全依赖自注意力机制来计算输入和输出的表示，而不使用序列对齐循环神经网络或卷积。鉴于自注意力机制有望成为从输入中提取信息并生成输出的主要手段，相较于依赖循环神经网络或卷积神经网络，它有望带来更高的效率和灵活性。这是因为自注意力机制能够捕捉数据中的长程依赖，同时还能在训练阶段实现并行化。此外，自注意力机制具有一定的可解释性，因为它能让模型识别每个时间步最重要的输入特征。

我们将提出的方法命名为：递归惯性里程计 Transformer（RIOT）和姿态递归惯性里程计 Transformer（ARIOT）。据我们所知，我们的方法是唯一利用自注意力和所有可用惯性测量单元信息（来自三维加速度计、三维陀螺仪和三维磁力计），提供端到端三维惯性里程计解决方案的网络。

ARIOT 模型是一种分层变换器，与 RIOT 不同，它引入了一个额外的内部姿态估计网络，用于根据传感器测量值回归估计惯性测量单元的姿态。该子系统受益于 [29] 中基于自注意力的框架设计。其输出用于里程计网络，以进一步回归加速度计读数和先前位置，从而得到更新后的惯性测量单元定位。RIOT 和 ARIOT 的有效性通过来自不同用户、活动和智能手机惯性测量单元设备的未见序列进行全面验证。

4 问题表述

4.1 传感器模型

首先，我们考虑如何对九维惯性测量单元的测量值进行建模。可以认为，这类系统具有噪声水平高、加性偏置随时间变化的特点。典型惯性测量单元可提供的测量值来自三轴角速率陀螺仪、三轴加速度计和三轴磁力计。惯性测量单元的参考坐标系称为机体坐标系（B），它相对于某个固定的惯性坐标系（I）旋转，例如地心惯性（ECI）坐标系或北-东-地（NED）坐标系。不过，为简洁起见，本文假定这些参考坐标系已知，不将其纳入符号表示。
陀螺仪测量 B 相对于 I的角速度，但测量值会受到缓慢变化的偏置和噪声的影响。因此，我们可以将陀螺仪测量值定义为$\mathbf{I}_{\omega,\ t}$，如下所示：

$$\mathbf{I}_{\omega,t} = \boldsymbol{\omega}_t + \boldsymbol{\delta}_{\omega,t} + \boldsymbol{e}_{\omega,t} \tag{1}$$

其中， $\boldsymbol{\omega}_t$ 表示每个时刻 t的真实角速度， $\boldsymbol{\delta}_{\omega,\ t}$ 表示时变偏置，$\boldsymbol{e}_{\omega,\ t}$ 表示噪声，通常假设其服从高斯分布，$\boldsymbol{e}_{\omega,\ t} \sim \mathcal{N}(0,\ \Sigma_\omega)$。
加速度计测量 B 相对于 I的线性加速度。考虑加入噪声和偏置后，加速度计测量值$\mathbf{I}_{a,\ t}$可表示为

$$\mathbf{I}_{a,t} = \mathbf{f}_t + \boldsymbol{\delta}_{a,t} + \boldsymbol{e}_{a,t} \tag{2}$$

其中，$\mathbf{f}_t$ 表示每个时刻 t的比力，$\boldsymbol{\delta}_{a,\ t}$ 和 $\boldsymbol{e}_{a,\ t}$ 分别表示偏置和噪声，其中$\boldsymbol{e}_{a,\ t} \sim \mathcal{N}(0,\ \Sigma_\omega)$。

磁力计提供有关传感器周围局部磁场方向和强度的信息。局部磁场由地球磁场以及因存在磁性材料而产生的其他磁场组成。由于磁力计测量值主要用于姿态确定，我们假设局部磁场向量的模（记为$\mathbf{m}^l$）等于 1，即 $\left\| \mathbf{m}^l \right\| = 1$。假设磁力计只测量局部磁场方向，其测量值$\mathbf{I}_{\mathrm{m},\ t}$可建模为

$$\mathbf{I}_{m,t} = \mathbf{R}_t^{bn}\mathbf{m}^l + \boldsymbol{e}_{m,t}, \tag{3}$$

其中$\mathbf{R}_t^{bn}$ 表示　　从导航坐标系到机体坐标系的旋转矩阵，且 $\boldsymbol{e}_{\mathrm{m},t} \sim \mathcal{N}(0, \Sigma_{\mathrm{m}})$ 是高斯噪声　　e.
通过纳入磁力计测量值，我们使系统能够确定其初始姿态。这基于这样一个原理：给定两个不同参考坐标系中的一组两个或更多线性无关向量，即可计算这两个坐标系之间的旋转。这里的基本假设是，加速度计只测量重力向量，而磁力计只测量局部磁场。因此，我们有四个线性无关向量：测量值$\mathbf{I}_{a,t}$ 和$\mathbf{I}_{m,t}$、局部重力向量 $\boldsymbol{g}^n$ 以及局部磁场向量 $\mathbf{m}^l$。尽管这被视为一项重大优势，但其缺点是需要掌握局部磁场信息，才能将式 3 转换到局部坐标。

4.2 姿态与位置估计

传统的姿态估计方法依赖陀螺仪积分作为推导姿态的基准。然而，已有充分文献指出，陀螺仪测量值缺乏实现绝对姿态确定所需的信息。因此，采用数值陀螺仪积分会导致累积误差不断增长且没有上限。因此，通常会将加速度计测得的比力与磁力计测得的磁场测量值结合使用，以校正估计结果，因为它们能够提供有关绝对角位置的信息。这些方法通常涉及复杂的数学模型和计算，并且需要先验状态信息以及特定的传感器参数 [30]。

与姿态估计类似，传统的位置估计方法也容易因缺乏绝对位置变化信息而产生无界误差。为克服这一局限，我们提出在梯度下降优化中使用自注意力和原始数据，以分析并保留与加速度计误差和偏置相关的信息，尤其是在较长的序列中。注意力机制会提取相关特征，从而学习加速度、姿态和位置之间的关系。对于 ARIOT，这种方法还有一个注意事项：需要识别并补偿初始姿态估计网络中的姿态估计误差，如 [29] 所示。

我们将加速度计和陀螺仪测量值作为动力学模型的输入，以估计位置。状态向量包括位置和姿态的四元数参数化表示（详见第5.2节）。我们结合惯性测量和先验位置来估计姿态和位置。

时间区间 Δt 内的位置动力学由下式给出

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \Delta t \boldsymbol{v}_t + \frac{\Delta t^2}{2}\left(\mathbf{R}_t^{nb}\left(\mathbf{I}_{a,t} - \boldsymbol{\delta}_{a,t}\right) + \boldsymbol{g}^n + \boldsymbol{e}_{a,t}\right) \tag{4}$$

其中 $\boldsymbol{v}_t$ 表示　　速度和时间t ，而$\mathbf{R}_t^{nb}$表示从机体坐标系到导航坐标系的旋转矩阵。为方便起见，我们将　e
噪声项的符号　　姿态的动力学方程为

$$\mathbf{q}_{t+1} = \mathbf{q}_t \odot \exp_q\left(\frac{\Delta t}{2}\left(\boldsymbol{I}_{\omega,t} - \boldsymbol{\delta}_{\omega,t} - \boldsymbol{e}_{\omega,t}\right)\right) \odot \exp_q\left(\frac{\Delta t}{2} f\left(\boldsymbol{I}_{a,t}, \boldsymbol{I}_{m,t}\right)\right) \tag{5}$$

其中， $f(\cdot)$ 是加速度计和磁力计测量值的函数，用于计算四元数的校正项；符号 $\odot$表示四元数乘法，定义如下

$$\begin{pmatrix} j_1 \\ j_2 \\ j_3 \\ j_4 \end{pmatrix} \odot \begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ k_4 \end{pmatrix} = \begin{pmatrix} j_1k_1 - j_2k_2 - j_3k_3 - j_4k_4 \\ j_1k_2 + j_2k_1 + j_3k_4 - j_4k_3 \\ j_1k_3 - j_2k_4 + j_3k_1 + j_4k_2 \\ j_1k_4 + j_2k_3 - j_3k_2 + j_4k_1 \end{pmatrix}. \tag{6}$$

与神经网络相比，传统状态估计算法的一项显著优势是能够保留先前的状态估计，以便更新后续状态。这使算法能够利用过去的信息来校正或优化当前估计，从而提高准确性和可靠性。相比之下，神经网络通常经过训练，基于输入数据进行预测，而不会明确保留过去的估计。

我们承认，循环神经网络在这里多少是个例外，因为它们可以设计为具有能够保留过去状态信息的内部状态记忆。然而，这实际上并没有赋予网络所需的递归属性。相反，它更像是一种伪递归：隐藏状态存储来自先前步骤的部分信息，并利用这些信息影响未来步骤。理解循环神经网络递归行为的本质及其局限性时，这一区分至关重要。此外，采用这种机制的网络还存在前述的记忆瓶颈和梯度消失问题。

5 提出的解决方案

5.1 网络组件

图 1 和图 2 展示了各模型的概览。下面我们将介绍两个网络中共有的组件，具体的模块化适配将在后文介绍。在模型设计方面，我们遵循 [4], 中提出的原始自然语言处理 Transformer，其由编码器-解码器块和多头注意力（MHA）组成。这种有意保持简洁的系统设计具有优势：实现高效，并且能够直接提供开箱即用的解决方案。标准 Transformer 的输入是由词元嵌入构成的一维序列。为了处理惯性测量单元数据，我们将序列嵌入扩展为 N 维，以对应特征输入；每个特征输入还带有一组额外的位置嵌入，用于表示时间信息。该网络会为每个输入时间步生成一个表示序列，随后将其用作下游任务中的特征向量。

5.1.1 位置编码

在 Transformer 模型中，如 [4], 所述，相对序列位置并未被明确编码。为了纳入相对序列位置信息，我们在第一层之前对输入添加正弦位置编码函数。编码值使用正弦和余弦三角函数计算。这些函数的参数是序列位置（pos）与缩放因子 $10000^{2i/d_{\text{model}}}$ 的乘积，其中 i 是索引变量，$0 \leq i \leq \frac{d_{\text{model}}}{2} - 1$，用于计算位置编码向量的不同维度。对于 i 的每个值，都会计算位置编码向量的两个维度，一个使用正弦函数（$PE_{(pos,2i)}$），另一个使用余弦函数（$PE_{(pos,2i+1)}$）。同时使用正弦函数和余弦函数的思想，是为了捕捉序列位置信息的幅值和相位，即序列中元素之间的相对顺序和距离。这样，模型便能够在处理的不同阶段关注输入序列的不同部分。此外，这种方法使模型能够泛化到不同的序列长度，并根据元素的相对位置而非绝对位置关注元素。当模型需要处理不同长度的序列或采样不匹配的情况时，这一点会很有用 [31]。两个网络中的位置编码定义为

$$\begin{aligned} PE_{(pos,2i)} &= \sin\left(pos/10000^{2i/d_{\text{model}}}\right) \\ PE_{(pos,2i+1)} &= \cos\left(pos/10000^{2i/d_{\text{model}}}\right) \end{aligned} \tag{7}$$

其中，pos 表示位置，i 表示维度，d_{model} 表示模型维度；在本文中，姿态网络和位置网络的 d_{model} 分别为 64 和 224（见第 7 节）。

5.1.2 自注意力

这些网络中的自注意力子层采用 $h = 2$ 个注意力头。自注意力机制首先通过一组权重参数化的线性变换，将 IMU测量值投影到高维空间中$\mathbf{W}^Q$，$\mathbf{W}^K$，$\mathbf{W}^V \in \mathbb{R}^{d_I \times d_{\text{model}}}$。该投影由一组权重参数化，这些权重在训练过程中学习得到。每一层和每个注意力头都有各自独有的参数矩阵。随后，变换后的输入数据会传入一个函数（通常称为“注意力函数”），该函数为每个输入元素生成一组注意力权重，用以表示各输入元素在回归姿态或位置估计时的重要性。

每个注意力头都对一个由 n 个元素组成的输入序列 $\mathbf{I} = (I_1, \ldots, I_n)$ 进行处理，其中 $I_i \in \mathbb{R}^{d_I}$。计算得到一个长度相同的新序列 $\mathbf{z} = (z_1, \ldots, z_n)$，其中 $z_i \in \mathbb{R}^{d_{\text{model}}}$，且每个输出元素都计算为

每个输入经线性变换后的加权和

$$z_i = \sum_{j=1}^{T} \alpha_{i,j} \left(x_j \mathbf{W}^V \right) \tag{8}$$

其中，每个权重系数 $\alpha_{i,j}$ 均通过 softmax 函数计算，该函数会对每个元素的兼容性分数进行归一化，从而得到输入序列上的概率分布

$$\alpha_{i,j} = \frac{\exp e_{i,j}}{\sum_{k=1}^{n} \exp e_{i,k}} \tag{9}$$

e_{ij} 通过一个比较两个输入元素的兼容性函数计算得出，

$$e_{ij} = \frac{\left(x_i \mathbf{W}^Q \right) \left(x_j \mathbf{W}^K \right)^T}{\sqrt{d_z}} \tag{10}$$

采用缩放点积作为兼容性函数，以实现高效计算。对输入进行线性变换可提供足够的表达能力。自注意力层通过多头注意力（MHA）实现。

在利用惯性测量单元（IMU）信息估计姿态四元数或位置时，自注意力机制会根据不同传感器测量值与位置估计的相关程度，赋予它们不同的权重。例如，在估计旋转运动时，陀螺仪测量值可能会被赋予比加速度计测量值更高的权重；而在估计线性加速度时，加速度计测量值可能会被赋予更高的权重。

5.1.3 编码器

输入向量与位置编码向量逐元素相加后，输入两个相同的编码器层。每个编码层由两个子层组成：一个 MHA 子层和一个全连接前馈（FF）子层。对于 ARIOT 姿态模块，我们尝试使用多个卷积层来提取数据的空间结构特征；然而，自注意力机制已足以捕获相关信息，未观察到额外收益。

我们的编码器遵循 [4], 提出的查询–键–值模型，其中所使用的缩放点积注意力表示为

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{D_k}} \right) \mathbf{V} \tag{11}$$

其中，输入 $\mathbf{I}$ 用于获取查询 $\mathbf{Q} = \mathbf{I}(k)\mathbf{W}^Q \in \mathbb{R}^{N \times D_k}$、键 $\mathbf{K} = \mathbf{I}(k)\mathbf{W}^K \in \mathbb{R}^{M \times D_k}$ 和值 $\mathbf{V} = \mathbf{I}(k)\mathbf{W}^V \in \mathbb{R}^{M \times D_v}$；每个 $\mathbf{W}$ 都是训练过程中更新的相应权重矩阵，N, M 表示查询和键（或值）的长度，D_k, D_v 表示键（或查询）和值的维度。MHA 由 H 组不同的学习得到的投影组成，而不是使用单一的注意力函数，具体如下

$$\text{MultiHeadAttn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_H)\mathbf{W}^O$$

其中，注意力头$_i = \text{Attention}\left(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}_i^V \right)$。这些投影是参数矩阵$W_i^Q \in \mathbb{R}^{D_{\text{model}} \times D_k}$，$W_i^K \in \mathbb{R}^{D_{\text{model}} \times D_k}, W_i^V \in \mathbb{R}^{D_{\text{model}} \times D_v}$和 $W^O \in \mathbb{R}^{hD_v \times D_{\text{model}}}$。在本文中，我们采用 $h = 2$ 个并行注意力层，即注意力头。对于每个注意力头，我们使用$D_k = D_v = D_{\text{model}}/h$。

除了注意力子层外，每个编码器/解码器层还包含一个全连接前馈网络，由线性变换和激活函数组成。我们在前馈网络中使用 aLeakyReLU [32] 激活函数，如下所示

$$\text{LeakyRe}LU(x) = \begin{cases} x, & \text{if } x \geq 0 \\ 1 \times 10^{-3} \cdot x, & \text{otherwise} \end{cases} \tag{12}$$

逐点前馈网络是一个全连接模块

$$\text{FFN}\left(\mathbf{H}' \right) = \text{LeakyRe}LU\left(\mathbf{H}'\mathbf{W}^1 + \mathbf{b}^1 \right) \mathbf{W}^2 + \mathbf{b}^2 \tag{13}$$

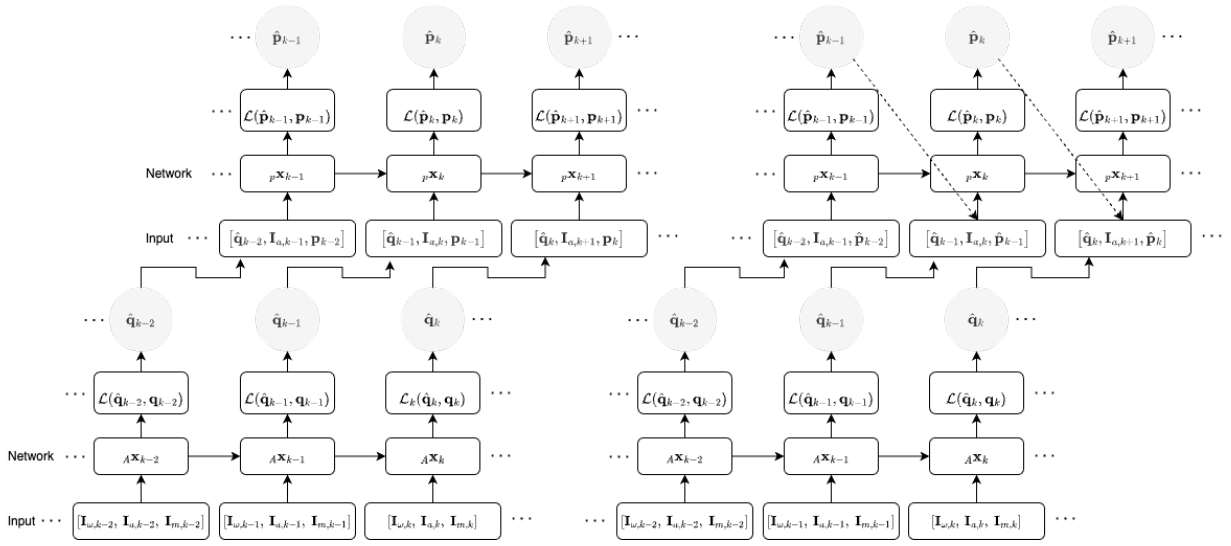


图1：ARIOT模型在第一轮（左）和第二轮（右）训练期间的情况。 ${}_A\mathbf{x}$ 表示第5.2节和${}_p\mathbf{x}$ 5.3节中所述的姿态神经网络。网络输入见第4.1节，具体为公式1、2和3。推理过程与第二轮的图示类似，但不进行损失最小化，详见第6.3节。

其中，$\mathbf{H}'$是前一层的输出，$\mathbf{W}^1 \in \mathbb{R}^{D_m \times D_f}$、$\mathbf{W}^2 \in \mathbb{R}^{D_f \times D_m}$、$\mathbf{b}^1 \in \mathbb{R}^{D_f}$和$\mathbf{b}^2 \in \mathbb{R}^{D_m}$是可训练参数，而 D_f 表示内层维度。每个子层的各个模块周围都插入了一个层归一化模块。也就是说，

$$\mathbf{H}' = \text{LayerNorm}\left(\text{SelfAttn}(\mathbf{X}) + \mathbf{X}\right) \tag{14}$$

其中，SelfAttn(·)表示自注意力模块，LayerNorm(·)表示层归一化操作。随后将所得向量输入解码器。

5.1.4 解码器

解码器由2个相同的层组成。解码器包含编码器中的子层，此外还增加了第三个子层，用于对编码器的输出向量执行多头注意力。多头注意力机制使模型能够并行关注输入序列的多个部分，从而捕捉输入更细致、更丰富的表示。这通过将注意力机制划分为多个“注意力头”来实现。每个注意力头都使用不同的线性投影执行注意力。此外，解码器中的自注意力机制会屏蔽后续位置，防止其影响当前位置，确保 k 处的预测仅依赖于 $k-1$处及之前已知的输出。在我们的姿态网络中，输出层通过双曲正切函数将最终层映射为估计的四元数。对于位置，线性化后不使用任何输出函数。

5.2 姿态递归惯性里程计 Transformer

姿态递归惯性里程计 Transformer 是一种分层框架，由两个基于自注意力的编码器-解码器网络组成。初始网络以先前的工作 [29] 为基础，用于根据 9D 惯性测量（见公式 1、公式 2 和公式 3）回归姿态估计。借此，可在单一框架中分别估计姿态和位置，为惯性里程计提供稳健的解决方案。在这两个模块中使用自注意力机制，能够对数据中的长期依赖关系进行建模，从而有效处理长序列中出现的高动态运动。

接下来，我们使用四元数对姿态进行参数化。四元数是一种表示$\mathbb{R}^4$中姿态的方式，相较于使用 $\mathbb{R}^3$进行表示具有多项优势。四元数不存在不连续性和奇异点，并且计算效率更高、数值稳定性更好。要作为有效的姿态表示，四元数必须是单位四元数。单位四元数与旋转矩阵之间存在一一对应关系，并且它们对群$\mathcal{SO}(3)$构成二重覆盖，这意味着$\mathbf{q}$ 和 $-\mathbf{q}$ 表示相同的姿态。然而，通过要求$\mathbf{q}_0 \geq 0$，我们可以确保四元数与旋转矩阵之间存在唯一对应关系 [33]。

我们提出在梯度下降优化中使用自注意力机制和原始九维惯性测量单元数据，以分析并保留长序列中与陀螺仪误差和偏置相关的信息。这样可以通过放弃预积分来降低复杂性。此外，由于不强制网络采用预定义的动力学模型，该方案不受约束。这些特性以及磁力计测量值的加入，还使网络成为一种开箱即用的解决方案，适用于已知局部磁场的情况。

5.2.1 损失函数

我们提出一种新的四元数损失函数，并称之为四元数损失。为了定义四元数损失，我们首先介绍一些四元数背景知识和记号。四元数是一个 4 元组 $(x,\ y,\ z,\ w)$，其中 $x,\ y,\ z,\ w$ 为实数。四元数可以表示为

$$\mathbf{q} = w + xi + yj + zk \tag{15}$$

其中 $i,\ j,\ k$ 为虚数单位，满足 $i^2 = j^2 = k^2 = ijk = -1$。

通过将 w 设为旋转角的余弦，并将 $x,\ y,\ z$ 设为旋转角的正弦与旋转轴 [34]的乘积，可以用四元数表示三维空间中的旋转。给定一对四元数 $(\mathbf{q}_1,\mathbf{q}_2)$，我们可以使用内积来衡量它们之间的相似度：

$$\langle \mathbf{q}_1, \mathbf{q}_2 \rangle = x_1x_2 + y_1y_2 + z_1z_2 + w_1w_2 \tag{16}$$

该乘积与四元数之间的夹角存在如下关系，

$$\cos(\theta) = \frac{\langle \mathbf{q}_1, \mathbf{q}_2 \rangle}{|\mathbf{q}_1| \cdot |\mathbf{q}_2|} \tag{17}$$

其中，θ 表示四元数之间的夹角，$|\cdot|$ 表示 L2范数。

随后，我们将四元数损失函数定义为

$$\mathcal{L}(\mathbf{q}_1, \mathbf{q}_2) = \cos^{-1}(\text{clamp}(\langle \mathbf{q}_1, \mathbf{q}_2 \rangle, -1 + \epsilon, 1 - \epsilon)) \tag{18}$$

其中

$$\text{clamp}(x, a, b) = \begin{cases} a & \text{if } x < a \\ x & \text{if } a \le x \le b \\ b & \text{if } x > b \end{cases}$$

并且，ϵ 是一个较小的正数常量，用于在内积超出范围 $[-1, 1]$ 时避免数值不稳定性。

批次中的平均夹角则计算为

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \theta_i \tag{19}$$

其中 N 为批次大小。

5.3 递归惯性里程计 Transformer

递归惯性里程计 Transformer 是一种基于自注意力的编码器-解码器网络。在梯度下降优化中，放弃姿态模块，直接对原始九维 IMU 数据（来自公式 1、公式 2 和公式 3）应用自注意力，以进行三维位移回归；如图 2 所示。

在训练的第一个周期中，网络的输入是惯性测量与真实位置先验的拼接；随后，真实位置先验被估计位置先验所替代。输入经过嵌入层，以生成嵌入表示。然后，编码器应用自注意力机制，为每个时间步计算嵌入表示的加权和，并据此计算上下文向量。接着，上下文向量经过解码器，以估计每个时间步的三维位置。输入序列、嵌入函数和自注意力机制的方程见第 5.1 节。随后，使用式 20 中的均方误差（MSE）损失函数，通过 ADAM 优化算法对模型进行训练，使其最小化该损失。 [35]

$$\mathcal{L}(\hat{\mathbf{p}}, \mathbf{p}) = \frac{1}{NT} \sum_{n=1}^{N} \sum_{t=1}^{T} \left\| \hat{\mathbf{p}}_{n,t} - \mathbf{p}_{n,t} \right\|^2 \tag{20}$$

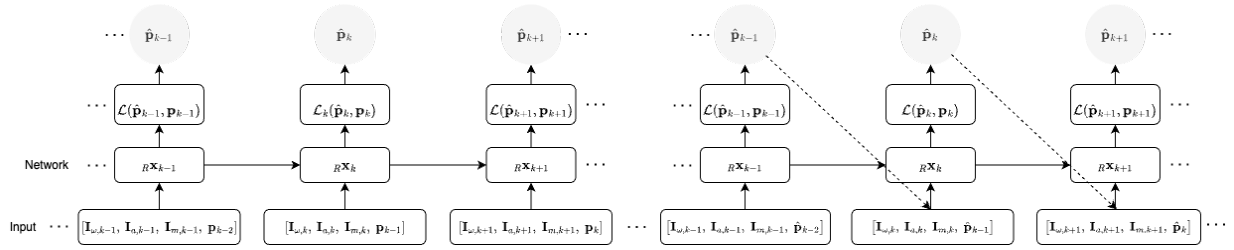


图 2：RIOT 模型在第一个周期（左）和第二个周期（右）的训练过程。 ${}_R\mathbf{x}$ 表示第5.3节所述的姿态神经网络。网络输入见第4.1节，具体为公式 1、2 和 3。推理过程与第二个周期的示意相似，但不进行损失最小化，详情见第6.3节。

其中， $\|\cdot\|^2$ 表示平方欧几里得范数， N 表示批次大小， T 表示序列长度，$\hat{}\mathbf{p}$ 和$\mathbf{p}$ 分别表示估计位置和真实位置。

6 评估

尽管文献中提出了许多旨在解决惯性导航问题的方法，但这些方法使用各自的数据集，并采用各种预处理和对齐技术（例如 Umeyama算法 [36]）来评估算法。在这些条件下，很难直接比较不同算法。此外，据我们所知，其他方法都没有利用所有可用的惯性测量单元信息。然而，纳入磁力计测量值也有其缺点：我们的解决方案依赖局部磁场，因为磁力计读数被用于消除惯性测量单元姿态的歧义。这使得我们的网络校准具有区域特异性；如果不了解局部磁场，就无法泛化到其他数据集。为此，我们基于自行实现的循环神经网络进行比较。

6.1 门控循环单元

近期关于循环神经网络的研究表明，在多种场景下，门控循环单元（GRU）的表现优于常用的长短期记忆网络 [37, 38, 39]。此外，GRU 的参数更少，计算效率更高，并且已被证明对噪声和缺失数据具有更强的鲁棒性 [40]。

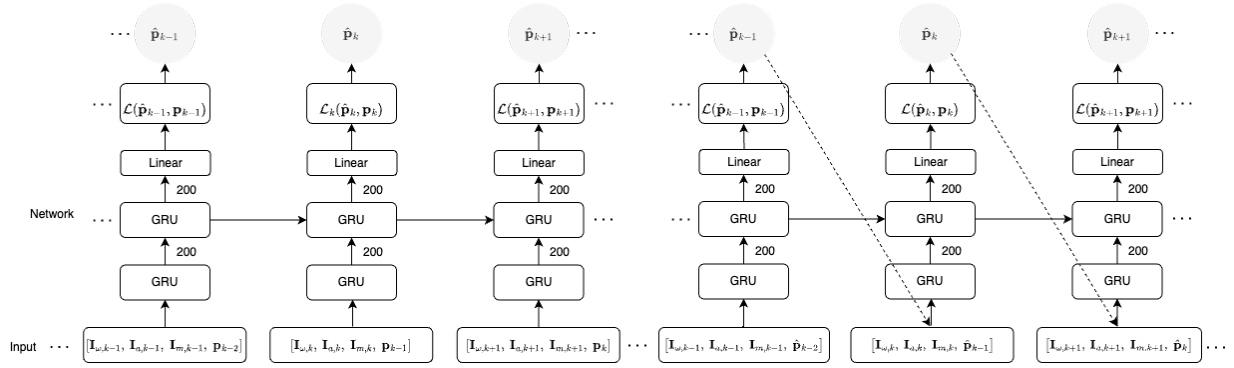


图 3：GRU 模型在第一个周期（左）和第二个周期（右）中的训练过程。我们的模型由 2 个 GRU单元组成，如第 6.1 节所述，每层有 200 个神经元。网络的输入见第4.1节，具体为公式 1、2 和 3。推理采用与第二个周期相似的示意方式进行，但不对损失进行最小化；详情见第 6.3 节。

我们添加了自己实现的两层 GRU，以作比较。GRU 已被证明在惯性姿态估计中有效 [41], ，然而据我们所知，GRU 尚未在惯性里程计领域经过测试。该网络以时间步 t 的隐藏状态 h_t 表示如下

$$r_t = \sigma(\mathbf{W}_{ir}x_t + \mathbf{b}_{ir} + \mathbf{W}_{hr}h_{t-1} + \mathbf{b}_{hr}) \tag{21}$$

$$z_t = \sigma(\mathbf{W}_{iz}x_t + \mathbf{b}_{iz} + \mathbf{W}_{hz}h_{t-1} + \mathbf{b}_{hz}) \tag{22}$$

$$\tilde{h_t} = \text{LeakyRe}LU(\mathbf{W}_{ix}x_t + \mathbf{b}_{ix} + r_t * (\mathbf{W}_{hx}h_{t-1} + \mathbf{b}_{hx})) \tag{23}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h_t} \tag{24}$$

其中， x_t 是时间步 t 的输入，$\mathbf{W}_{i*}$ 和 $\mathbf{b}_{i*}$ 分别是输入到隐藏层的权重和偏置，$\mathbf{W}_{h*}$ 和 b_{h*} 分别是隐藏层到隐藏层的权重和偏置， σ 是逻辑 sigmoid 函数。式 21 和式 22 分别计算重置门 r_t 和更新门 z_t。这些门控制传递到下一时间˜步的信息量。式 23 计算候选隐藏状态 h_t，式 24 则通过结合˜先前的隐藏状态 h_{t-1} 和候选隐藏状态 h_t来更新隐藏状态 h_t 。

我们以与 RIOT 相同的方式实现该网络，如图 3 所示，其中两层门控循环单元（GRU）堆栈将采样时刻 t 的九维惯性测量单元输入与时刻 $t-1$ 的三维位置向量拼接后，转换为一个N_n维特征向量 h_t，其中 $N_n = 200$表示每层的神经元数量。

6.2 训练与数据集

我们使用 Chen 等人发布的公开智能手机数据训练和测试该方法 [42]。该数据集包含 158 个序列，总距离超过 42 千米，并涵盖多种安装方式、活动和用户，以尽可能反映现实生活中的广泛使用场景。数据由五名不同用户使用四种不同类型的市售消费级智能手机采集。惯性测量单元数据以 100 Hz 的频率采集并同步，这一频率在各种应用和研究中普遍得到认可 [43, 44, 45]。我们使用高精度光学动作捕捉系统（Vicon）采集完整位姿真实值，位置精度为 $0.01m$ ，姿态精度为 0.1 度 [46]。按照 [47]，我们将数据集随机划分为训练集、验证集和测试集。对于每个变量，我们各留出一个序列，以便与其他技术进行完整且未见过数据的比较。为避免过拟合并提高计算效率，我们采用滑动窗口，每次截取 100 个测量值、步长为 50，并将其输入编码器；同时使用随机搜索调整超参数。由此得到 63,614 个训练样本、18,175 个验证样本和 9089 个测试样本。所有改进的实现均使用 PyTorch 完成。姿态网络在 300 个训练周期后收敛。位置网络在使用真实输入时经过 120 个周期收敛，使用递归输入时经过 30 个周期收敛。所有实现均采用 0.001 的学习率、ADAM 优化器和 0.2 的丢弃率。训练在 $4\times$ 台 Nvidia V100 GPU 上并行进行。

6.3 推理

各模型的推理过程与第二轮训练过程十分相似，如图1、图2和图3所示，分别对应 ARIOT、RIOT 和门控循环单元。每个序列的初始窗口均以零值进行前填充，随后接入给定的初始位置。之后，通过处理后续数据窗口，迭代更新位置窗口。该推理方法旨在体现循环架构和递归数学模型，同时利用自注意力的优势。图4展示了这一过程。

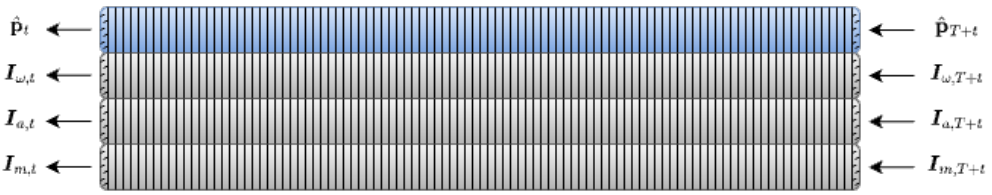


图4：RIOT 的滑动窗口递归推理过程示意图。底部的三个输入见第4.1节，具体为公式1、2和3；顶部为网络位置估计，$\hat{}\mathbf{p}$ 其中 T 为序列长度， t为时间步。

6.4 评估指标

为了定量评估各方法在每个长度为 K 的未观测序列上的表现，采用了以下三项指标：

表 1：二维位置误差指标对比。针对数据集中的每个变量，均将一个完整序列排除在训练数据之外。这样做是为了在完整序列上进行未见数据对比，从而评估不同用户、活动和设备的表现，并考察各方法的泛化能力。请注意，每个网络都能够生成三维位置估计；然而，由于数据大多采集于水平面上，z 轴上的偏差远小于 x-y 平面上的偏差，因此加入垂直维度会使误差指标产生偏差。每个序列、每项指标中表现最佳的模型均以粗体标出。在大多数情况下，RIOT 的表现最佳；但在动态运动幅度很大的情况下，ARIOT 的跟踪效果更好。

	用户 2		用户 3		用户 4		用户 5		口袋		跑步	
模型	绝对轨迹误差（m）	相对轨迹误差（m）	绝对轨迹误差（m）	相对轨迹误差（米）	绝对轨迹误差（米）	相对轨迹误差（米）	绝对轨迹误差（米）	相对轨迹误差（米）	绝对轨迹误差（米）	相对轨迹误差（米）	绝对轨迹误差（米）	相对轨迹误差（米）
GRU	0.0796	0.0110	0.0692	0.0100	0.0757	0.0121	0.0856	0.0114	0.1013	0.125	0.1589	0.0171
ARIOT	0.0994	0.0093	0.0934	0.0088	0.0960	0.0094	0.1027	0.0100	0.1059	0.0088	0.1279	0.0144
RIOT	**0.0681**	**0.0090**	**0.0655**	**0.0085**	**0.0654**	**0.0091**	**0.0721**	**0.0096**	**0.0676**	**0.0085**	**0.0990**	**0.0140**
	慢走		手推车		手提包		手持设备		iPhone 5		iPhone 6	
模型	绝对轨迹误差（米）	相对轨迹误差 (m)	绝对轨迹误差 (m)	相对轨迹误差 (m)	绝对轨迹误差 (m)	相对轨迹误差 (m)	绝对轨迹误差 (m)	相对轨迹误差（m）	绝对轨迹误差（m）	相对轨迹误差（m）	绝对轨迹误差（m）	相对轨迹误差（m）
GRU	0.2634	0.0077	0.0881	0.0116	0.2021	0.0112	7.352	0.0357	0.1172	0.0138	0.1133	0.0110
ARIOT	0.1082	0.0060	0.1033	0.0099	0.1096	0.0091	**0.3196**	0.0129	0.1046	0.0090	0.1036	0.0089
RIOT	**0.0660**	**0.0058**	**0.0690**	**0.0096**	**0.0694**	**0.0089**	0.4438	**0.0109**	**0.0690**	**0.0086**	**0.0667**	**0.0085**

- **绝对轨迹误差 (ATE)**（米）

$$\text{ATE} = \sqrt{\frac{1}{K}\sum_{k=1}^{K}\left\|(\hat{\mathbf{p}}_k - \mathbf{p}_k)\right\|^2} \tag{25}$$

绝对轨迹误差通常用于评估引导或导航系统的性能，表示估计位置的全局精度。

- **相对轨迹误差 (RTE)**（米）

$$\sqrt{\frac{1}{K}\sum_{k=1}^{K}\left\|(\hat{\mathbf{p}}_{k+\Delta t} + \mathbf{p}_{k+\Delta t}) - (\hat{\mathbf{p}}_k + \mathbf{p}_k)\right\|^2} \tag{26}$$

相对轨迹误差用于衡量给定时刻估计位置与真实位置之间的差异，并以两个位置之间的距离为基准。它常用于量化预定义时长内的位置一致性 Δt；在本研究中为 $\Delta t = 1s$ 。

- **累积分布函数 (CDF)**

$$\int_0^e f(x)dx, \quad \boldsymbol{e} = \left[\frac{1}{K}\sum_{k=1}^{K}\sqrt{\left\|(\hat{\mathbf{p}}_k - \mathbf{p}_k)\right\|^2}\right] \tag{27}$$

累积分布函数是分布函数 $f(x)$，用于描述变量的分布。在此语境中，它用于表示估计位置的误差小于或等于某一特定值的概率。 $f(x)$ 是定位误差 $\boldsymbol{e}$ 的概率密度函数。

绝对轨迹误差和相对轨迹误差常用于深度惯性里程计论文中 [17, 25] ，而累积分布函数是室内定位研究中的常用指标 [48]。

6.5 评估

本文提出了三种方法，用于在来自不同用户、设备和活动的未见序列上进行评估。表1列出了绝对轨迹误差（ATE）和相对轨迹误差（RTE）的评估结果，并以粗体标出每个序列和指标表现最佳的方法。此外，我们还对模型的输出进行了定性分析，发现预测轨迹与真实轨迹高度吻合。附录A展示了这一结果，并提供了各方法在数据首尾一分钟内的位置估计可视化结果。总体而言，RIOT表现最佳，绝对轨迹误差和相对轨迹误差均为最低，除惯性测量单元由手持设备持握的情况外。当惯性测量单元由手持设备持握，且伴随明显且持续的动态运动时，ARIOT中的姿态估计模块能够发挥作用，帮助区分受线性加速度和

重力共同影响的加速度计测量值，从而得到更准确的轨迹估计。然而，当惯性测量单元在运动过程中基本稳定或呈周期性运动时，姿态估计模块带来的额外复杂性不仅多余，实际上还会影响性能。我们推测，模型过度依赖姿态表示，而这只有在高度动态的场景中才有益。

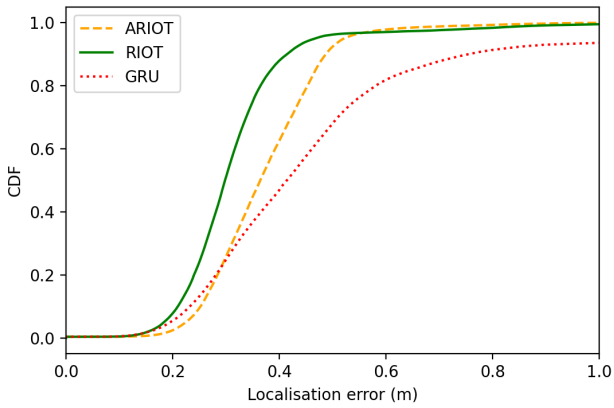


图5：所有测试序列中各方法定位误差的累积分布函数。该图绘制了低于特定百分比阈值的定位误差值所占的百分比，从而可以评估各方法在不同定位误差水平（米）下的表现。与 ARIOT 和 GRU 相比，RIOT 的表现最佳，低误差值更为集中。

分析模型的性能时，考虑数据特征和具体场景非常重要。我们推测，RIOT 表现更优的原因在于其架构更简单，似乎更适用于 IMU 运动较不剧烈的场景。另一方面，ARIOT 的姿态估计模块增加了复杂度，因此能够更好地处理动态运动。

显然，在所有序列中，GRU 的表现都明显逊于 RIOT 和 ARIOT。这可能是因为 RNN 模型的设计从根本上较为低效，导致其无法有效处理长序列中复杂的运动。不过，值得注意的是，GRU 模型的表现仍然相当不错，这凸显了我们在模型开发过程中所采用的学习过程的有效性。

图 5 进一步印证了这一分析，该图展示了整个测试序列集上定位误差的平均累积分布函数（CDF）。RIOT 的表现几乎始终如一，这一点可由低误差范围内 CDF 的陡峭梯度看出；相比之下，ARIOT 和 GRU 的误差则分布在更宽的数值范围内。

我们的模型采用多头自注意力机制，该机制通过多个并行的注意力机制实现。自注意力通过允许模型动态关注输入序列的不同部分，能够捕捉复杂的关系和依赖性。每个自注意力机制都会计算一个大小为 $T \times T$ 的注意力矩阵 $\boldsymbol{A}$，其中 T 为序列长度，并采用公式 11 所述的 Softmax运算。注意力分数决定了输入时间特征对更高层输出时间特征的影响。
图 6a、6c、7a 和 7c 中的矩阵可视化展示了模型如何对多个输入进行加权和组合以作出预测。注意力矩阵的值表示每个编码器的第一自注意力层中的两个注意力头，并将其表示为输入节点与输出节点之间的邻接矩阵。如图 6b、6d、7b 和 7d 所示，该矩阵也可以表示为二分图。边权重表示注意力的强度，边的不透明度表示其大小。输入时间序列显示在注意力图上方，作为参考；注意力分数则以垂直线表示，其对应于 $\boldsymbol{A}$ 中的值。

通过对注意力矩阵进行可视化，我们可以直接观察到不同自注意力头和编码器之间的差异。第一个编码器中的注意力头似乎高度集中于序列的后半部分，而第二个编码器中的注意力头则集中于序列开头，但整体注意力更强。从图6和图7中可以观察到，模型在进行预测时会同时考虑数据中的短期和长期时间依赖关系，而不是像传统方法那样只关注前一时间步。这在很大程度上解释了其能够获得准确且稳定的位置估计，尤其是在运动复杂或存在噪声的情况下。

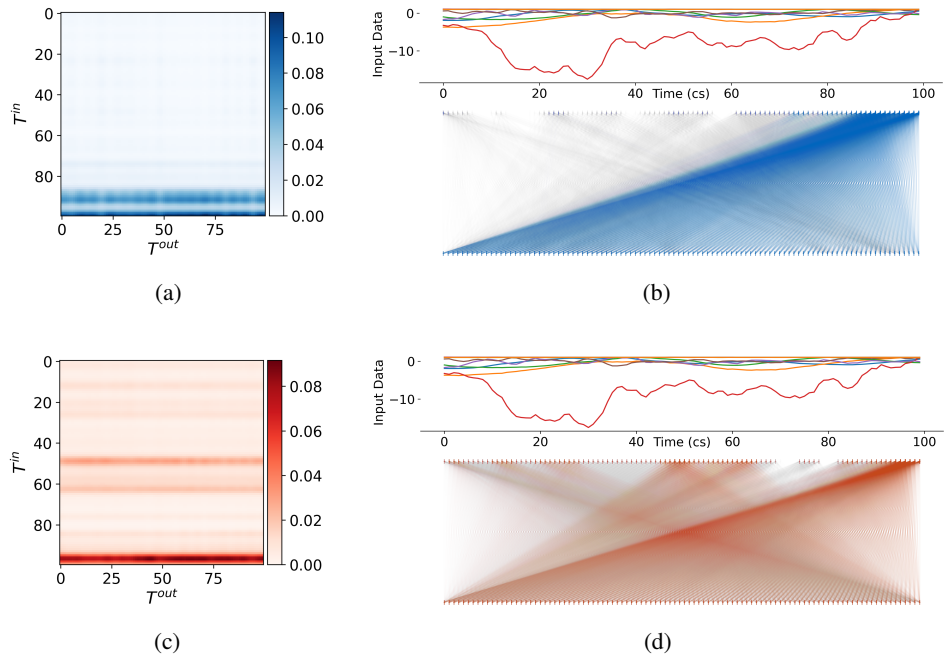


图 6：RIOT 中第一个编码器对任意输入数据序列所产生的自注意力分数可视化。**(a)** 和**(b)** 分别以矩阵和二分图的形式展示第一个编码器的第一个注意力头所产生的注意力分数。**(c)** 和**(d)** 分别以矩阵和二分图的形式展示第二个编码器的第二个注意力头所产生的注意力分数。**左侧：**热力矩阵显示序列中每个输入元素对应的注意力分数。颜色越深，分配给该元素的注意力权重越高，表明它对最终输出的影响越大。**右侧：**图中将每个输入元素表示为一侧的节点，而模型分配的注意力分数则表示为另一侧的节点。连接这些节点的边表示注意力权重，即模型对各输入元素的关注程度。边的粗细表示注意力权重的大小；边越粗，注意力分数越高。

总之，我们评估了三种新型递归式深度惯性里程计框架的性能。结果表明，基于自注意力的网络性能优于基于门控循环单元的循环神经网络，其中 RIOT 的总体表现最佳，其按序列长度加权的平均绝对轨迹误差为 0.0865m，相对轨迹误差为 0.0091m。ARIOT 和 GRU 的平均相对轨迹误差和绝对轨迹误差分别为 0.1134m、0.0095m，以及 0.4594m、0.0130m。我们的结果还表明，更简单的架构通常能够取得更好的结果；然而，在惯性测量单元经历高度动态运动的特定场景中，加入姿态模块可以显著提升性能，这凸显了在多样化数据集上评估解决方案的重要性。

7 RIOT 消融实验

模型维度。 我们使用不同大小的维度向量训练模型，范围从 56 到 896。增加模型维度后，性能略有提升，但提升幅度较小且可测量，直到维度达到 224。这一发现符合深度学习中的普遍规律：模型复杂度达到一定程度后，继续增加复杂度会导致过拟合，从而降低模型在新数据上的表现。

编码器-解码器块。 增加编码器-解码器块的数量并未降低模型的困惑度。我们训练了 3 个不同的模型，分别包含 2、4 和 6 个块。

注意力头。 我们训练了模型，每个编码器-解码器块分别包含 2、3 和 4 个注意力头。网络在测试集上的表现略有提升，但几乎无法测量。然而，在未见序列上进行测试时，包含 3 个或 4 个注意力头的模型表现明显更差。我们认为，将注意力头的数量增加到 2 个以上会导致网络过拟合。

窗口大小。 我们使用不同的窗口大小训练模型，范围从 50 到 500（0.5 秒到 5 秒）。随着窗口大小逐步增加到 100 以上，我们发现测试集上的结果有所改善，但在未见

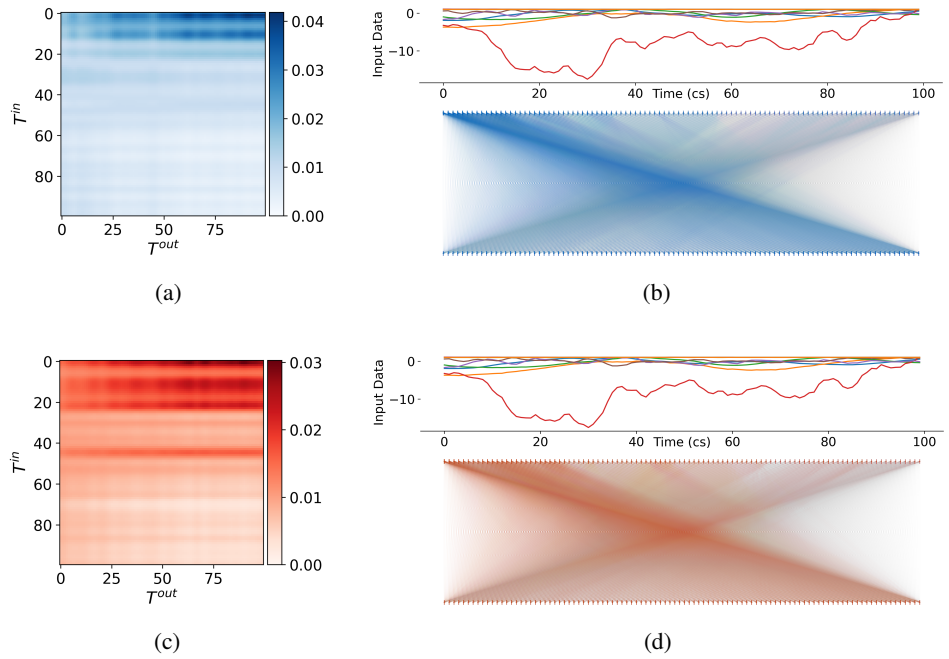


图 7：RIOT 中第二个编码器对任意输入数据序列的自注意力分数可视化。**(a)** 和**(b)** 分别以矩阵和二分图的形式展示了第二个编码器第一个注意力头的注意力分数。**(c)** 和**(d)** 分别以矩阵和二分图的形式展示了第二个编码器第二个注意力头的注意力分数。**左侧：** 热力矩阵显示了序列中每个输入元素所获得的注意力分数。颜色越深，表示该元素获得的注意力权重越高，说明它对最终输出的影响越大。**右侧：** 图中将每个输入元素表示为图一侧的节点，而模型分配的注意力分数则表示为另一侧的节点。连接这些节点的边表示注意力权重，即模型关注每个输入元素的程度。边的粗细表示注意力权重的大小，边越粗，表示注意力分数越高。

序列上的结果却更差。增大输入数据的窗口大小会使模型复杂度呈指数级增长，因为 RIOT 有 12 个输入特征。增加的复杂度会导致网络过拟合。

8 结论

本文提出了用于姿态不变惯性里程计的新型自注意力递归神经网络模型 RIOT 和 ARIOT。所提出的方法在训练过程中融入真实位置先验，并使用惯性测量和真实位移数据进行训练，从而能够递归运行，并学习运动特征以及系统误差中的偏置和漂移。评估结果表明，在位置误差指标方面，RIOT 优于 ARIOT 和门控循环单元（GRU）：其按序列长度加权的平均绝对轨迹误差（ATE）为 0.0865 米，按序列长度加权的平均相对轨迹误差（RTE）为 0.0091 米。这些结果显著优于文献中现有的深度学习惯性里程计方法，彰显了所提出方法及学习方法的有效性。未来的工作将探讨这些方法的可扩展性，并使其不受局部磁场影响。

资金

本研究得到了澳大利亚国家计算基础设施（NCI Australia）提供的资源支持。该基础设施是由澳大利亚政府资助的NCRIS支持能力（资助编号：LE160100051）。

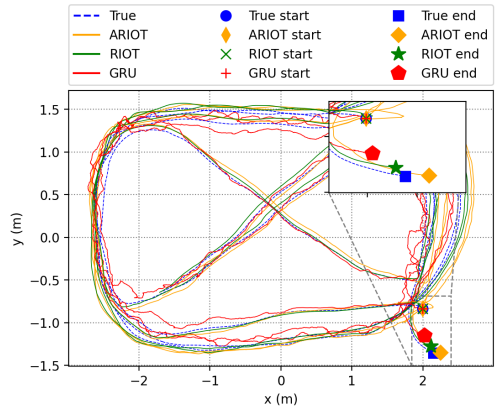
A P每个网络的前一分钟和最后一分钟的位置估计可视化结果

参考文献

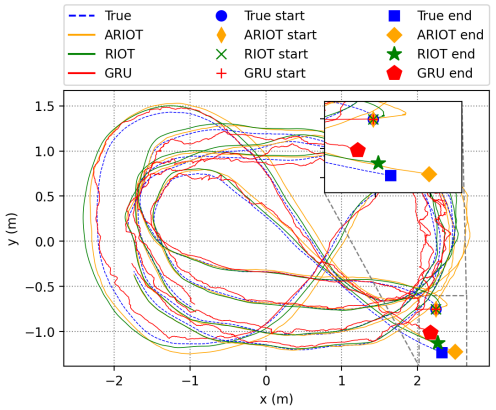
[1] Naser El-Sheimy、Haiying Hou和Xiaoji Niu。使用阿兰方差分析和建模惯性传感器。IEEE仪器与测量汇刊，57(1):140–149，2007。[2] Oludare Isaac Abiodun、Aman Jantan、Abiodun Esther Omolara、Kemi Victoria Dada、Nachaat AbdElatif Mohamed和Humaira Arshad。人工神经网络应用的最新进展：综述。Heliyon，4 (11):e00938，2018。[3] Razvan Pascanu、Tomas Mikolov和Yoshua Bengio。循环神经网络训练的难点。载于《国际机器学习会议》，第1310–1318页。PMLR，2013。[4] Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Łukasz Kaiser和Illia Polosukhin。注意力就是你所需要的一切。载于《神经信息处理系统进展》，第5998–6008页，2017。[5] Salman Khan、Muzammal Naseer、Munawar Hayat、Syed Waqas Zamir、Fahad Shahbaz Khan和Mubarak Shah。视觉中的Transformer：综述。ACM计算调查（CSUR），54(10s):1–41，2022。[6] Jacob Devlin、Ming-Wei Chang、Kenton Lee和Kristina Toutanova。BERT：用于语言理解的深度双向Transformer预训练。arXiv预印本 arXiv:1810.04805，2018。[7] R Mohammdi Farsani和E Pazouki。用于时间序列预测的Transformer自注意力模型。《电气与计算机工程创新期刊》（JECEI），9(1):1–10，2021。[8] Niki Parmar、Ashish Vaswani、Jakob Uszkoreit、Lukasz Kaiser、Noam Shazeer、Alexander Ku和Dustin Tran。图像Transformer。载于《国际机器学习会议》，第4055–4064页。PMLR，2018。[9] Peter J Liu、Mohammad Saleh、Etienne Pot、Ben Goodrich、Ryan Sepassi、Lukasz Kaiser和Noam Shazeer。通过总结长序列生成维基百科。arXiv预印本 arXiv:1801.10198，2018。[10] Daniel Povey、Hossein Hadian、Pegah Ghahremani、Ke Li和Sanjeev Khudanpur。用于自动语音识别的时间受限自注意力层。载于2018年IEEE国际声学、语音与信号处理会议（ICASSP），第5874–5878页。IEEE，2018。[11] Cheng-Zhi Anna Huang、Ashish Vaswani、Jakob Uszkoreit、Noam Shazeer、Ian Simon、Curtis Hawthorne、Andrew M Dai、Matthew D Hoffman、Monica Dinculescu和Douglas Eck。音乐Transformer。arXiv预印本 arXiv:1809.04281，2018。[12] Danny Merkx和Stefan L Frank。比较Transformer和循环神经网络对人类句子加工数据的预测。arXiv预印本 arXiv:2005.09471，2020。[13] Alec Radford、Karthik Narasimhan、Tim Salimans、Ilya Sutskever等。通过生成式预训练提升语言理解能力。2018。[14] Hang Yan、Qi Shan和Yasutaka Furukawa。RIDI：鲁棒惯性测量单元二重积分。载于《欧洲计算机视觉会议》（ECCV）论文集，第621–636页，2018。[15] Omri Asraf、Firas Shama和Itzik Klein。PDRNet：深度学习行人航位推算框架。《IEEE传感器期刊》，22(6):4932–4939，2021。[16] Changhao Chen、Xiaoxuan Lu、Andrew Markham和Niki Trigoni。IONet：学习消除惯性里程计的漂移难题。载于《AAAI人工智能会议》论文集，第32卷，2018。[17] Hang Yan、Sachini Herath和Yasutaka Furukawa。RONIN：真实环境中的鲁棒神经惯性导航——基准测试、评估与新方法。arXiv预印本 arXiv:1905.12853，2019。[18] Kaiming He、Xiangyu Zhang、Shaoqing Ren和Jian Sun。用于图像识别的深度残差学习。载于《IEEE计算机视觉与模式识别会议》论文集，第770–778页，2016。[19] Rooholla Khorrambakht、Chris Xiaoxuan Lu、Hamed Damirchi、Zhenghua Chen和Zhengguo Li。通过精确的惯性测量单元预积分实现深度惯性里程计。arXiv预印本 arXiv:2101.07061，2021。[20] Wenxin Liu、David Caruso、Eddy Ilg、Jing Dong、Anastasios I Mourikis、Kostas Daniilidis、Vijay Kumar和Jakob Engel。TLIO：紧耦合学习式惯性里程计。《IEEE机器人与自动化快报》，5(4):5653–5660，2020。[21] James Brotchie、Wenchao Li、Allison Kealy和Bill Moran。使用最大熵分布评估智能手机应用中的旋转跟踪。《IEEE Access》，9:168806–168815，2021。

[22] Scott Sun、Dennis Melamed 和 Kris Kitani。Idol：惯性深度姿态估计与定位。载于《AAAI人工智能会议论文集》，第35卷，第6128–6137页，2021年。[23] Yingying Wang、Hu Cheng、Chaoqun Wang 和 Max Q-H Meng。用于行人定位的姿态不变惯性里程计。《IEEE仪器与测量汇刊》，70：1–12，2021年。[24] Xiya Cao、Caifa Zhou、Dandan Zeng 和 Yongliang Wang。Rio：用于鲁棒惯性里程计的旋转等变监督学习。载于《IEEE/CVF计算机视觉与模式识别会议论文集》，第6614–6623页，2022年。[25] Yingying Wang、Hu Cheng 和 Max Q-H Meng。A2dio：基于六维IMU、用于行人定位的注意力驱动深度惯性里程计。载于2022年国际机器人与自动化会议（ICRA），第819–825页。IEEE，2022年。[26] Oludare Isaac Abiodun、Aman Jantan、Abiodun Esther Omolara、Kemi Victoria Dada、Nachaat AbdElatif Mohamed 和 Humaira Arshad。人工神经网络应用的最新进展：综述。《Heliyon》，4（11）：e00938，2018年。[27] Nima Mohajerin 和 Steven L Waslander。利用循环神经网络对动态系统进行多步预测。《IEEE神经网络与学习系统汇刊》，30（11）：3370–3383，2019年。[28] Laith Alzubaidi、Jinglan Zhang、Amjad J Humaidi、Ayad Al-Dujaili、Ye Duan、Omran Al-Shamma、José Santa-maría、Mohammed A Fadhel、Muthana Al-Amidie 和 Laith Farhan。深度学习综述：概念、CNN架构、挑战、应用及未来方向。《大数据期刊》，8（1）：1–74，2021年。[29] James Brotchie、Wei Shao、Wenchao Li 和 Allison Kealy。利用自注意力机制进行智能手机姿态估计。《Sensors》，22（22）：9011，2022年。[30] John L Crassidis、F Landis Markley 和 Yang Cheng。非线性姿态估计方法综述。《制导、控制与动力学杂志》，30（1）：12–28，2007年。[31] Giuseppe Coviello、Gianfranco Avitabile、Antonello Florio 和 Claudio Talarico。无线同步平台的IMU采样率不匹配研究。载于2020年IEEE第63届国际中西部电路与系统研讨会（MWSCAS），第229–232页。IEEE，2020年。[32] Andrew L Maas、Awni Y Hannun、Andrew Y Ng 等。整流器非线性改善神经网络声学模型。载于ICML会议论文集，第30卷，第3页。Citeseer，2013年。[33] Manon Kok、Jeroen D Hol 和 Thomas B Schön。使用惯性传感器进行位置与姿态估计。arXiv预印本 arXiv:1704.06053，2017年。[34] Du Q Huynh。三维旋转的度量：比较与分析。《数学成像与视觉杂志》，35（2）：155–164，2009年。[35] Diederik P Kingma 和 Jimmy Ba。Adam：一种随机优化方法。arXiv预印本 arXiv:1412.6980，2014年。[36] Shinji Umeyama。两个点模式之间变换参数的最小二乘估计。《IEEE模式分析与机器智能汇刊》，13（04）：376–380，1991年。[37] Shudong Yang、Xueying Yu 和 Ying Zhou。LSTM与GRU神经网络性能比较研究：以Yelp评论数据集为例。载于2020年国际电子通信与人工智能研讨会（IWECAI），第98–101页。IEEE，2020年。[38] Nicole Gruber 和 Alfred Jockisch。在文本动机分类中，GRU单元是否更具特异性，而LSTM单元是否更敏感？《Frontiers in Artificial Intelligence》，3：40，2020年。[39] Roberto Cahuantzi、Xinye Chen 和 Stefan Güttel。用于学习符号序列的LSTM与GRU网络比较。arXiv预印本 arXiv:2107.02248，2021年。[40] Zhengping Che、Sanjay Purushotham、Kyunghyun Cho、David Sontag 和 Yan Liu。用于处理缺失值多变量时间序列的循环神经网络。《Scientific Reports》，8（1）：1–12，2018年。[41] Daniel Weber、Clemens Gühmann 和 Thomas Seel。Riann——一种性能优于姿态估计滤波器的鲁棒神经网络。《人工智能》，2（3）：444–463，2021年。[42] Changhao Chen、Peijun Zhao、Chris Xiaoxuan Lu、Wei Wang、Andrew Markham 和 Niki Trigoni。Oxiod：深度惯性里程计数据集。arXiv预印本 arXiv:1809.07491，2018年。[43] Robbe Vleugels、Ben Van Herbruggen、Jaron Fontaine 和 Eli De Poorter。用于冰球分析的超宽带室内定位与基于IMU的活动识别。《Sensors》，21（14）：4650，2021年。

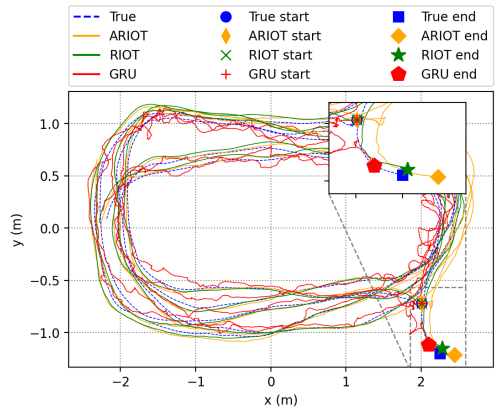
[44] Vicent Girbés-Juan、Leopoldo Armesto、Daniel Hernández-Ferrándiz、Juan F Dols 和 Antonio Sala。面向重型车辆的全球定位系统、IMU与基于CAN的里程计异步传感器融合。*IEEE*车辆技术汇刊，70（9）：8617–8626，2021年。[45] Sharmita Dey 和 Arndt Schilling。基于单个IMU传感器、用于鲁棒在线足部角度轨迹预测的函数逼近模型：对主动假肢足控制的启示。*IEEE*工业信息学汇刊，2022年。[46] Vicon。Vicon动作捕捉系统。2017年。[47] Ian Goodfellow、Yoshua Bengio 和 Aaron Courville。深度学习。麻省理工学院出版社，2016年。[48] Weixing Xue、Weining Qiu、Xianghong Hua 和 Kegen Yu。改进用于室内定位的Wi-Fi接收信号强度测量。*IEEE*传感器期刊，17（7）：2224–2230，2017年。doi:10.1109/JSEN.2017.2660522。



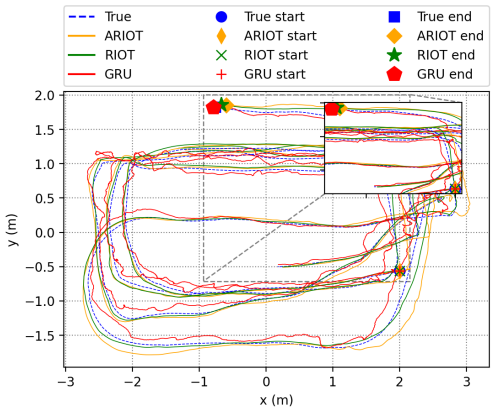
(a) 用户 2：总时间：263.5s，真实距离：220.9m，ARIOT 距离：234.1m，RIOT 距离：228.1m，GRU 距离：260.7m



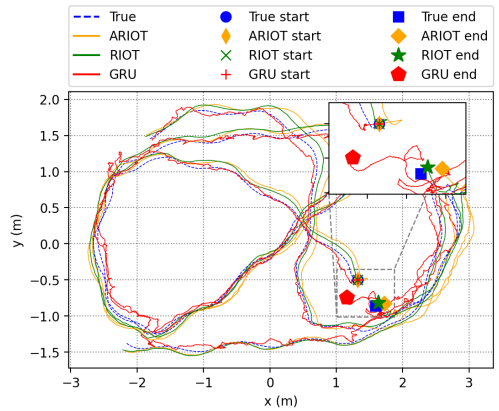
(b) 用户 3：总时间：258.0s，真实距离：203.3m，ARIOT 距离：215.5m，RIOT 距离：209.7m，GRU 距离：229.3m



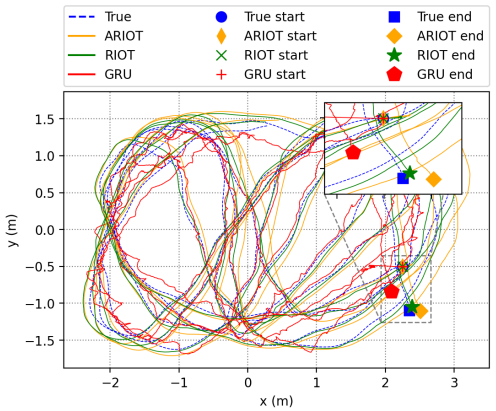
(c) 用户 4：总时间：434.7s，真实距离：364.3m，ARIOT 距离：387.6m，RIOT 距离：377.5m，GRU 距离：448.6m



(d) 用户 5：总时间：152.0s，真实距离：130.3m，ARIOT 距离：138.6m，RIOT 距离：134.0m，GRU 距离：153.7m

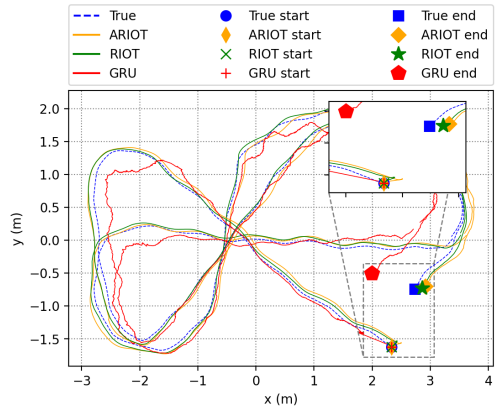


(e) 口袋：总时间：622.9s，真实距离：493.7m，ARIOT 距离：526.3m，RIOT 距离：512.5m，GRU 距离：650.7m

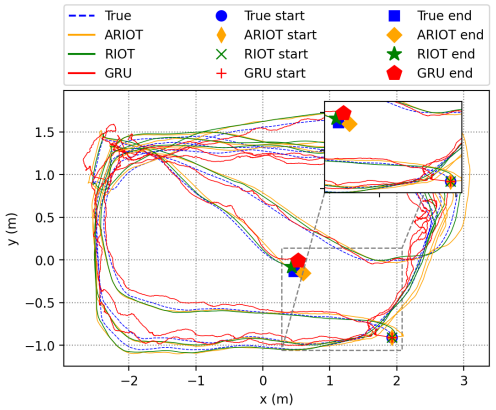


(f) 跑步：总时间：302.9s，真实距离：393.1m，ARIOT 距离：422.2m，RIOT 距离：410.5m，GRU 距离：447.6m

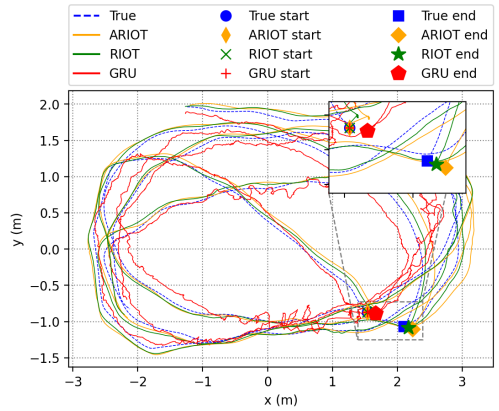
图 8：各网络在序列开始和结束的第一分钟内的位置估计，以及真实路径。图中还给出了序列运行时间以及各方法的总路径长度。



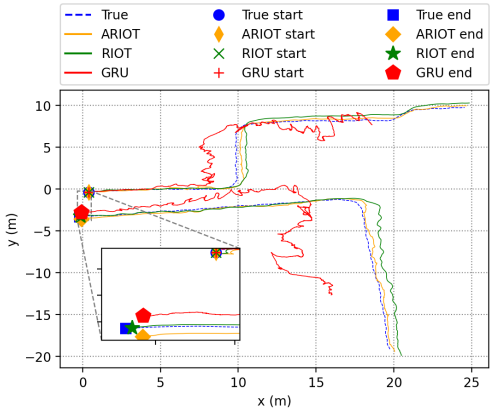
(a) 慢走：总时间：303.7s，真实距离：166.6m，ARIOT 距离：174.7m，RIOT 距离：171.1m，GRU 距离：185.2m



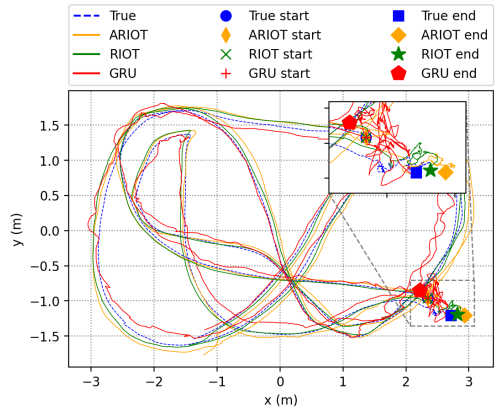
(b) 手推车：总时间：370.0s，真实距离：328.4m，ARIOT 距离：346.5m，RIOT 距离：337.4m，GRU 距离：376.8m



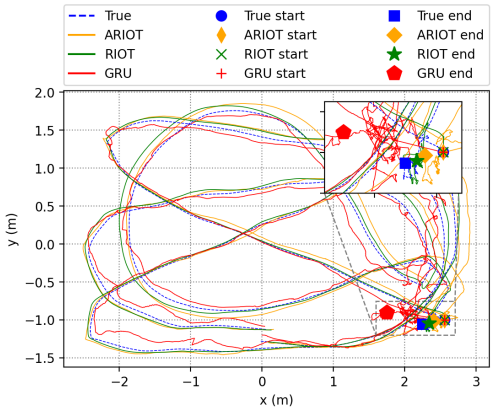
(c) 手提包：总时间：365.8s，真实距离：305.9m，ARIOT 距离：321.9m，RIOT 距离：317.1m，GRU 距离：341.1m



(d) 手持设备：总时间：156.0s，真实距离：152.8m，ARIOT 距离：169.7m，RIOT 距离：159.8m，GRU 距离：358.6m



(e) iPhone 5：总时间：183.8s，真实距离：142.4m，ARIOT 距离：149.8m，RIOT 距离：145.3m，GRU 距离：158.3m



(f) iPhone 6：总时间：173.4s，真实距离：133.4m，ARIOT 距离：141.6m，RIOT 距离：137.4m，GRU 距离：150.3m

图9：各网络在前一分钟和最后一分钟的位置信息估计，以及真实路径。图中还给出了序列运行时间和各方法的总路径长度。