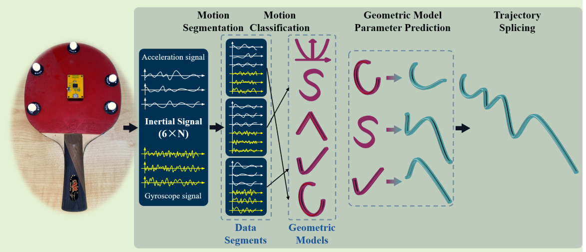
This article has been accepted for publication in IEEE Sensors Journal. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2023.3257867

Arbitrary Spatial Trajectory Reconstruction based on A Single Inertial Sensor

Yifeng Wang, *Student Member, IEEE,* and Yi Zhao

***Abstract*—Compared with vision, infrared rays, and ultrasonic positioning technologies, the signal acquisition of portable inertial sensors is not affected by the external environment, such as light and occlusion. Therefore, motion tracking based on inertial signals is a promising complement. However, accurate trajectory reconstruction based on inertial sensors is a great challenge due to the intrinsic and measurement errors, especially the drift error that exacerbates the accumulative error in trajectory calculation. To address this challenge, we propose a new trajectory reconstruction method, Geometric Dynamic Segmental Reconstruction (GDSR), where we treat the movement trajectory as a combination of basic trajectories. To this end, we design a temporal and spatial interaction segmentation approach to decompose the trajectory into basic segments by combining the dynamic feature of IMU signals with the spatial morphological feature of motion. Accordingly, we design a geometrical model library with undetermined parameters to match these segments. For precise parameter prediction, we propose an extra-supervised learning method that integrates different prediction tasks into one framework, which can not only expand training samples but also enable different subtasks to compete with each other, thus improving the parameter prediction accuracy of each subtask, thereby accurately approximating the trajectory segments. To quantify the trajectory reconstruction accuracy, we propose the Fréchet Spline Sliding Error (FSSE) and Length Error Ratio (LER) to evaluate curve similarity. The range of FSSE is [0, 2], where 0 means that the two curves have the same shape. The range of LER is [1, $+\infty$], where 1 means that the two curves have the same length. We test different IMUs in two experimental scenarios and one public available data set. In all the three tests, the FSSE of the GDSR method is less than 0.09, and the LER is less than 1.31, which is significantly better than all the comparison methods.**



***Index Terms*—Motion tracking, inertial sensor, motion segmentation, motion state recognition, supervised learning.**

I. INTRODUCTION

INERTIAL sensors that can measure acceleration and angular velocity data are widely used in navigation [1], [2], orientation [3]–[5], industrial automation [6], motion state research [7], human motion [8]–[10] and gait analysis [11]–[14]. Especially, they have also been applied to sport performance assessment [15], [16], telerehabilitation and monitoring [17], [18], and joint kinematics [19], [20] with successes. The inertial sensors are usually integrated into the portable devices as an inertial measurement unit (IMU) for diverse application scenes. For instances, Zedda *et al.* [21] and Digo *et al.* [22] provide the upper limb joint kinematics estimation in real-time for both active telerehabilitation purposes and industrial human monitoring, respectively.

However, although the IMU shows the advantages of small volume, low cost, and easy embedding in portable products, the application of IMU in these scenarios is mainly limited to orientation estimation, and there are few attempts at realizing accurate trajectory reconstruction based on IMUs, especially single IMU. Albeit this, there are still some significant work, e.g., in the direction of foot-mounted IMUs [23], pedestrian dead-reckoning, and wheeled vehicles [24]. For the previous gait analysis the idea is to split the trajectory computation stride-by-stride to reset the estimation every 1-2 seconds, thus mitigating the errors accumulation [25]. However, it is challenging to reconstruct the irregular arbitrary motion trajectory that lasts for a longer period of time.

Due to the intrinsic attribution that the inertial navigation algorithms are extremely sensitive to errors [26], people employs the optical sensors for motion capturing and trajectory reconstruction [27]. However, the optical sensor heavily relies on the camera facilities as well as the light environment when recording the motion data [28], so the application range of the optical methods is constrained. Moreover, the optical occlusion problem is a major defect of this measurement scheme [29] while the inertial sensors do not have such problems. IMU-based motion trajectory reconstruction has tremendous application potential, providing a technical basis for navigation, robot path planning, motion control, and other fields. Hence, it is challenging but quite attractive to realize the trajectory reconstruction by using inertial data [30].

This work was supported by the Innovative Research Project of Shenzhen, China under Grant No. KQJSCX20180328165509766 and the Natural Science Foundation of Guangdong, China under Grant No. 2020A1515010812 and 2021A1515011594. (Corresponding author: Yi Zhao.) The authors are with the School of Science, Harbin Institute of Technology, Shenzhen, China (e-mail: wangyifeng@stu.hit.edu.cn; zhao.yi@hit.edu.cn)

The conventional process to estimate the trajectory is based on IMU orientation estimation by means of a sensor fusion algorithm. When appropriately tuning the sensor fusion algorithm [31], the motion trajectory can be given after obtaining the initial conditions [32], [33]. There are some methods which realize human body motion tracking by utilizing multiple inertial sensors. Miezal *et al.* [34] develop a sensor fusion method for multiple inertial signals, which can handle model calibration errors for better body tracking. Huang *et al.* [35] propose a deep neural network capable of reconstructing human motion trajectory in real-time from six IMUs worn on the user's body. Stanzani *et al.* [36] treat a concrete motion as the sequential joint motion. As a result, given the rotation specifications at each joint, the original motion is finally generated by multiplying the corresponding radii in turn up to the last joint [37]. In such a way, the reconstruction of the motion trajectory is roughly separated into a series of reconstruction of basic rotations. Obviously, these approach requires sensors attached on all the necessary joints. This makes it a luxury to restore the motion trajectory. Meanwhile, this forward kinematics approach suffers from orientation drift caused by the integration of the gyroscope offset as documented in [38], [39]. The impact of the orientation inaccuracies on the joint angle estimates are discussed in [22]. Hence, it is necessary to develop a new trajectory reconstruction algorithm based on a single inertial sensor, which is qualified and also convenient for the trajectory reconstruction in general.

There are also few studies that achieve trajectory reconstruction based on a single inertial sensor. Pan *et al.* [40] realize the trajectory restoration of an inertial sensor moving on a horizontal desktop, so the moving process is steady, thereby greatly reducing the noise during data acquisition. In addition, this method requires the sensor to be static for a while after each movement of a small distance so as to eliminate the accumulative error of the inertial sensor. But it also destroys the continuity of the motion such that the reconstruction of general motion trajectory cannot be achieved. Similarly, Wang *et al.* [41] achieve segmental reconstruction of 2D trajectories by integrating an accelerometer and two gyroscopes in a pen. However, since the pen rarely changes its height during writing, it is unfeasible for this method to deal with the arbitrary orbit change, and its application scope is relatively limited. Furthermore, this method also needs the regular standstill when collecting the inertial data for a short while.

With the development of artificial intelligence, there is a new way to achieve motion tracking based on a single IMU. Ribeiro *et al.* [42] present six basic human motion reconstruction in industrial activities. Each motion trajectory is simple and repeated, which makes it easy for the machine learning model to learn the motion characteristics and recover the trajectory. Obviously, such a task is far from the arbitrary trajectory reconstruction that we deal with now. To expand to more action types, Lin *et al.* [43] employ a deep learning model to reconstruct 4 types of walking motion (e.g. walking in a circle or S shape) and 8 types of hand motion (e.g. stretching out or swiping the arm), of which the former can be regarded the large-scale basic curve and the latter can be regarded as the previous rotation around a bearing. As a result, their method appears to be not available to arbitrary trajectory. To realize arbitrary walking trajectory reconstruction, Chen *et al.* [44] segment the walking according to the gait characteristics and then use a deep learning model to predict the walking direction and forward distance between segments, thereby realizing indoor walking trajectory reconstruction. This method simulates each walking segment by a line and predicts the angle and length of each line to reconstruct the whole walking trajectory. With a geometric model of the line, acceptable results are obtained in their work, which indicates the feasibility of the segmentation-reconstruction scheme. However, the current motion segmentation method relies heavily on the regularity of motion and the static moment during walking [45], which is not applicable to complicated trajectories. In summary, there are few attempts at studying arbitrary trajectory reconstruction.

We, therefore, propose a new trajectory reconstruction method, which can achieve arbitrary trajectory reconstruction with a single inertial sensor. We design a temporal and spatial interaction segmentation approach to decompose the trajectory into consecutive segments, which combines the dynamic feature of IMU signals with the spatial morphological feature of motion. Meanwhile, a geometric model library is established, aiming to match these basic segmented trajectories. The matching process can be regarded as the classification of these segments, where the 1D-CNN model implements this task. After determining the geometric model of each segmented trajectory, we design an extra-supervised learning method to solve the problem of high-dimension parameter estimation based on the small sample (i.e., limited segments), which predicts the given geometric model parameters so as to reconstruct the original trajectory segments exactly. Finally, the trajectory reconstruction is available by the interpolation of the reconstructed trajectory segments in sequence. It is worth emphasizing that the optical sensor is only used to provide labels during training deep learning models in the segmentation, classification, and prediction tasks. In practical applications, once the model training is completed, trajectory reconstruction can be achieved utilizing only IMU.

II. Method

A. Geometric Dynamic Segmental Reconstruction (GDSR) framework

The schematic diagram of the proposed method is presented in Fig. 1. Overall the trajectory reconstruction is composed of the four main subtasks: 1. data segmentation for the purpose of dividing the trajectory data into multiple segments according to motion states, 2. motion state recognition for the purpose of matching these segments with the appropriate model from the geometric model library, 3. model parameter prediction for the purpose of estimating the model parameters to make an accurate approximation of the segmented trajectory and 4. trajectory reconstruction for the purpose of concatenating the reconstructed segments and smoothing the joint points. Note that in order to make necessary labels on the trajectory data set, optical sensor data is adopted as a reference. The main worth of our work is a decomposition reconstruction scheme based on the adaptive geometric models, in which we propose a spatiotemporal interactive segmentation method to achieve a precise approximation of motion segments. This scheme can realize an accurate trajectory reconstruction of a moving object by using a single IMU. Furthermore, its configuration ensures the proposed method shows its effectiveness for various experimental data and is robust against the sensor calibration strategy, embedded navigation algorithm, and IMU type.

The first subtask is realized by the segmentation module. Here, We obtain the rough trajectory from the 6-axis inertial data and 3-axis magnetic data according to the inertial navigation algorithm, which is implemented as follows. Firstly, the inertial sensor is calibrated to avoid drift while calculating the trajectory. Then, we calculate the initial roll and pitch of the IMU through the 3-axis acceleration, and calculate the initial yaw of the IMU through the 3-axis magnetic data. The orientation quaternion differential equation is obtained from the angular velocity output by the gyroscope. The quaternion rotation matrix between the IMU coordinate and the East-North-Up (ENU) coordinate is then calculated from the orientation quaternion. Through the quaternion rotation matrix, the 3-axis acceleration under the IMU coordinate is converted to the ENU coordinate, and the gravity acceleration is removed to obtain the linear acceleration of the IMU under the ENU coordinate. The trajectory of IMU is generated by double integration of the linear acceleration data, which is usually imprecise or even distorted. However, the distorted trajectory is helpful for segmentation since we notice that when the motion patterns switch or the direction of motion changes, the trajectory undergo a great degree of deformation. We, therefore, set a multi-resolution window sliding on the trajectory to find the deformation position as the potential segmentation points. On the other hand, we train a deep learning model to search the potential segmentation points from the 6-axis inertial data. Labels required for training is manually marked according to optical recording data. Finally, by fusing the two segmentation point detection results, the motion process is divided into multiple segments.

Implementation of the second subtask is based on the motion state recognition module, which decides a geometric model to match the segmented trajectory. Complicated motion trajectories can be approximately regarded as specific combinations of basic trajectories such as straight lines, arcs, polynomial curves, wavy lines (S-shape curves), etc. We, therefore, build a corresponding geometric model library that consists of these basic geometric curves. The matching process treats the previous segments as the motion state classification, so we adopt the 1D-CNN model that has been proved to have a good performance on the human activity recognition (HAR) problem [46], to identify each segment with the appropriate geometric model.

The third subtask is based on the geometric parameter prediction module. Each geometric model contains necessary parameters, so the parameters of the matched geometric model are then optimized to fit the segmented trajectory accurately. However, deep learning models with a number of parameters are difficult to be fully trained under the small sample, i.e., the limited trajectory segments [47], [48]. We, therefore, design an extra-supervised learning model, which decomposes the high-dimension parameter prediction task into multiple subtasks. By separately solving multiple subtasks and integrating their outputs, the trajectory segments can be accurately restored. From this, the given geometric model with the determined parameters gives the reconstructed trajectory segments that constitute the final trajectory. The last subtask gives the whole trajectory, which is implemented by the trajectory splicing module. Given the trajectory segments, it smooths joints between the sequential segments by interpolation and then concatenates them to generate the final trajectory reconstruction.

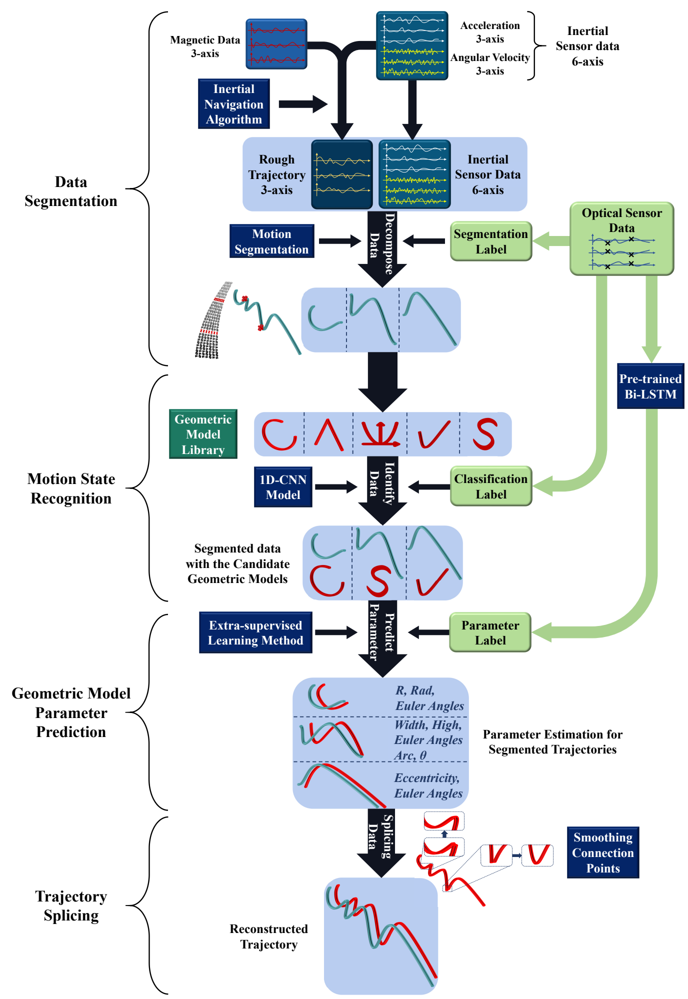


Fig. 1: The schematics of the GDSR method. The trajectory reconstruction implementation is composed of the four main modules: data segmentation, geometric model matching, model parameter prediction and trajectory splicing.

B. Motion segmentation by Time-Spatial Information Interaction

The motion segmentation is fundamental and critical to the process of spatial trajectory reconstruction. The implementation of the subsequent tasks relies on it. We notice that the 1D-CNN tends to take such extreme points (e.g., peaks or troughs) of the original IMU signal as alternative segment points. Ideally, the motion segmentation point should be the extreme point. But there is an obvious delay in IMU measurement. For example, when the sensor recovers from fast motion to a static state, it always takes a period for an IMU signal to return to zero values. When the motion state changes quickly or dramatically, an IMU, especially a low-cost IMU, always responds slower than motion changes. We call this phenomenon a response delay error. Such a factor affects the motion segmentation. As presented in columns 2-4 of Fig. 2, the segmentation points predicted by the 1D-CNN model have a gap with the segments given by the optical data. Hence, it is quite difficult to achieve frame-accurate motion segmentation only with consideration of the original inertial signal. We, therefore, propose a feature fusion algorithm for

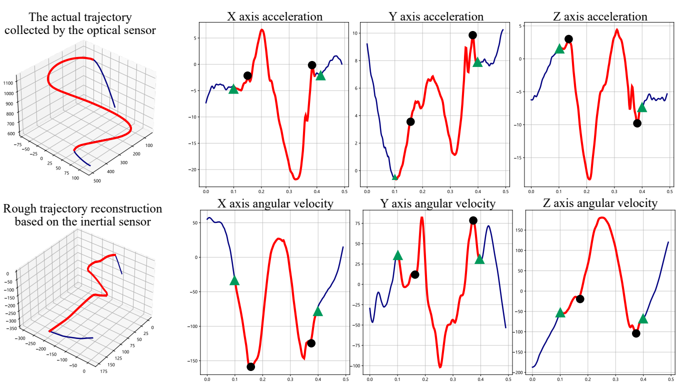


Fig. 2: The actual trajectory recorded by the optical sensor (the left top panel) and the trajectory reconstructed by the inertial data according to the classical trajectory solution algorithm based on the inertial navigation theory (the left bottom panel). The trajectory segment identified by the optical data is represented in the red curve. The waveform of the 6-axis inertial sensor data is displayed in the right columns. The segmentation points of the 1D-CNN model based on the inertial data waveform are marked by black dots. The final segmentation points identified by the collective features of the inertial waveform and reconstructed trajectory are marked by the green triangles.

segmentation points, which is illustrated by Fig. 3. The method consists of two parallel pipelines, including the 1D-CNN for temporal waveform features and the multi-scale spline sliding over the previous rough trajectory for spatial morphological features.

For motion segmentation based on the temporal feature of the IMU signal, the inertial data in a sliding window is fed to the 1D-CNN model. Here, we set the window size to 500 and the step size to 100. When the window slides through the whole signal, the points are identified as potential segmentation points by the 1D-CNN, and some of them are determined as segmentation points many times. If the number of times determined as the segmentation point is greater than the threshold, the corresponding point is determined as the segmentation point.

For motion segmentation based on the spatial feature of the calculated trajectory, we employ a multi-resolution spline function sliding on the trajectory to detect the inflection point. As shown in Fig. 4, an arc spline function with two resolution scales fits the sliding trajectory. An arc spline can be described by the coordinates of the starting point, midpoint and endpoint. Any three points that are not collinear can determine an arc. Here we select the minor arc, rather than the major arc. The Fréchet distance is used to measure the fitting level of the arc spline and its sliding parts, so we obtain a Fréchet-based curve for the whole trajectory. The arc spline function can fit smooth trajectory parts well. However, when it slides the possible motion transformation positions, due to the trajectory deformation, the Fréchet distance increases abruptly, which indicates the segmentation points. Adopting multi-resolution splines makes it qualified for different trajectory curves. Here we select two resolution lengths, i.e., 100 and 300, under which we get two Fréchet-based curves. We perform a weighted summation of the two Fréchet curves and find the potential segmentation points based on the extreme points.

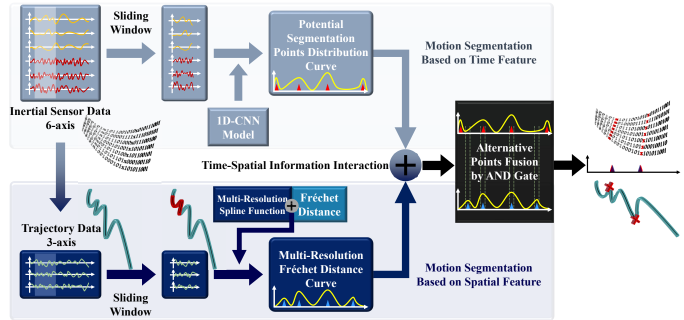


Fig. 3: The schematics of the temporal and spatial interaction segmentation algorithm. An 1D-CNN model is trained to detect the potential segmentation point based on the original 6-axis inertial signal in a sliding window. Meanwhile, another possible segmentation points are obtained by detecting the inflection point of the trajectory fragment in the sliding window. The final segmentation point is determined by fusing the two segmentation results.

Eventually, an AND operation is implemented on the two kinds of segmentation points. That is, if a segmentation point identified by the 1D-CNN model is close to (or coincides with) the corresponding segmentation point given by the spline function, we then take their average position as the final segmentation point. Otherwise, if the segmentation points given by two pipelines are far away, we do not adopt these two points as the segmentation results. Finally, the segmentation points are consistent with the points we manually mark. In fact, even if a small number of segmentation points are lost, the trajectory reconstruction can still be achieved due to the robustness of the geometric model library. For example, if the segmentation points between two arcs are not detected, the S-shape model will be matched with this data segment, so the trajectory reconstruction can still be completed.

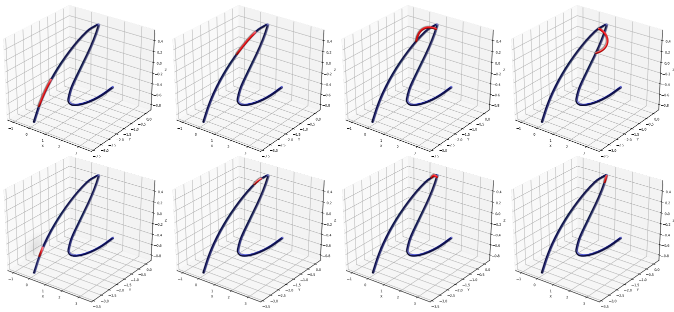


Fig. 4: The process of a multi-resolution spline function sliding over a trajectory (the low resolution in the top panels and the high resolution in the bottom panels).

C. Spatial trajectory simulation by geometric model library

The segmentation algorithm decomposes a complex motion trajectory into relatively simple trajectory segments so that we can use some geometric equations to simulate them separately. Therefore, we set up a geometric model library to cover basic motion curves in general. The candidate models equipped with the necessary parameters can approximate the original segments accurately. When determining the available geometric models, we follow the three rules. First, each geometric model should be simple without too many parameters so that its parameters can be easily and accurately predicted. Second, the different geometric models should have significant morphological differences so that the trajectory segments can be easily matched with an appropriate geometric model. Finally, each geometric model can produce rich morphological variation by virtue of its parameter variations. To sum up, we establish five geometric models: polyline model, arc (C-shape) model, polynomial model, hyperbolic model and wave (S-shape) model, which constitute the geometric model library.

1) The polyline model: The polyline geometric model is designed to simulate the relatively straight trajectory. If the straight line equation is directly set as the geometric model, the subsequent machine learning model tends to predict many curves as straight lines. Therefore, the polyline model has more applicability than the straight line model when simulating real trajectories. The polyline trajectory can be regarded as a combination of two lines in space, and we can use the equations of two spatial lines to describe it as follows:

$$\left\{ \begin{array}{l} \frac{x}{m_1} = \frac{y}{n_1} = \frac{z}{p_1} \\ \frac{x}{m_2} = \frac{y}{n_2} = \frac{z}{p_2} \end{array} \right. \tag{1}$$

It contains six parameters: m_1, n_1, p_1, m_2, n_2, and p_2. (m_1, n_1, p_1) and (m_2, n_2, p_2) denote the direction vectors of the two lines in a three-dimension space, respectively. With changes of the parameters, the polyline geometric model can generate any morphological change, as shown in Fig. 18. The reason for designing the polyline model instead of the straight line model is that a straight line can be simulated when the two lines constructing a polyline are nearly collinear. Therefore, compared with the straight line, the polyline model is more robust to trajectory reconstruction.

2) The arc model: The arc geometric model is described by an elliptic equation, which is designed to simulate the arc trajectory with the closed form and stable curvature. The parametric equations of elliptic models are determined by the axis vector $\vec{a} = (a_x, a_y, a_z)$, the axis vector $\vec{b} = (b_x, b_y, b_z)$ and the center point C, i.e. (c_x, c_y, c_z) in space. Note that in practice we set the center point to the coordinate original, i.e. (0, 0, 0). For any point $M(x(t), y(t), z(t))$ on an ellipse, the parametric equation can be expressed by:

$$\left\{ \begin{array}{c} x(t) \\ y(t) \\ z(t) \end{array} \right\} = \left\{ \begin{array}{c} c_x \\ c_y \\ c_z \end{array} \right\} + \left\{ \begin{array}{c} \cos(t) \cdot a_x \\ \cos(t) \cdot a_y \\ \cos(t) \cdot a_z \end{array} \right\} + \left\{ \begin{array}{c} \sin(t) \cdot b_x \\ \sin(t) \cdot b_y \\ \sin(t) \cdot b_z \end{array} \right\}, \tag{2}$$

where the range of parameter t is $[0, 2\pi]$. A series of geometric curves generated by this model are presented in Fig. 19.

3) The polynomial and hyperbolic model: The polynomial and hyperbolic geometric models are designed to simulate some arcs with open form and large curvature change, which cannot be fitted well by the arc model. Therefore, we prepare the polynomial model: $y = ax^3 + bx^2 + cx$, and the hyperbolic model: $y = ax + \frac{b}{x}$ under this scenario, where a, b and c are the coefficients. Since the hyperbolic function $y = ax + \frac{b}{x}$ has two asymptotes, its two ends are similar to straight lines, which makes it suitable for simulating curves with two ends close to a straight line. The polynomial function is suitable for simulating parabolic or truncated parabolic curves in space. Note that for an object moving in space, it involves the spatial attitude, including roll angle, pitch angle and heading angle, which determines the spatial trajectory of the object. For this reason, Fig. 20 shows the morphological variation of the two functions with different spatial attitude rotations.

4) The S-shape model: In addition to the previous types of polylines and arcs, there is another kind of basic motion curve, which is prevalent but more complex in trajectory segmentation and reconstruction, S-shape curves (i.e., wavy curves). Although there are continuous direction changes in an S-shape curve, the change process is gradual and smooth such that it will not be divided into local segments by the segmentation module, and is more appropriate to be identified as a whole. Therefore, we specially set up the S-shape geometric model. Considering the variety of wavy curves, we design two different S-shape models which have their own application scenarios. The closed S-shape model is composed of three segments, of which the two ends are the arc curves equipped with parameters of radius and radian, and the middle is a sigmoid curve for a smooth connection with the two arcs. The opened S-shape model is composed of two parabolas equipped with the coefficients and value ranges. The concise S-shape model is composed of a cubic function equipped with few coefficients. The morphological changes of the three geometric models are shown in Fig. 21.

D. Extra-supervised learning for geometric parameter prediction

For the geometric parameter prediction of three S-shape model, we propose an extra-supervised learning method, as shown in Fig. 5. This method integrates different S-shape trajectory prediction tasks into one framework. The parameter prediction of each S-shape geometric model can be regarded as a subtask of the extra-supervised learning method. Meanwhile, the method dynamically selects the appropriate S-shape geometric model to reconstruct the given segment, which can not only expand training samples but also enable different subtasks to compete with each other, thus improving the parameter prediction accuracy of each subtask.

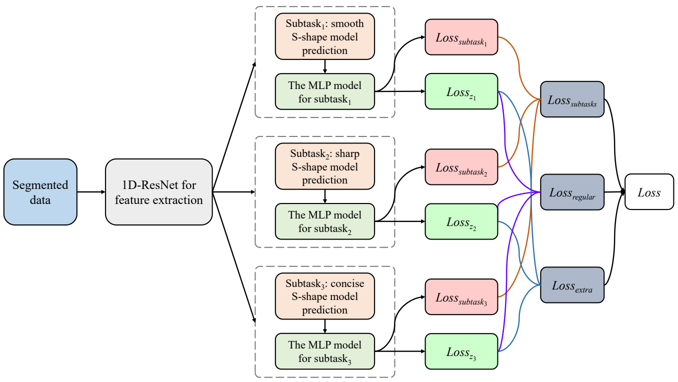


Fig. 5: The structure of the extra-supervised learning method, where the three S-shape models are implemented by three subtasks so as to determine the appropriate model by minimization of the loss objective function.

The data first goes through a deep learning module (here, we use the 1D-ResNet model) for feature extraction. Then the extracted features are fed to three subtasks, respectively, each of which contains one type of S-shape model. The subtask is implemented by an MLP model to realize parameter prediction as well as trajectory fitting. The error of parameter prediction in a subtask is denoted as:

$$Loss_{subtask} = \frac{1}{K}\sum\left(y_k - \hat{y}_k\right)^2, \tag{3}$$

where y_k and $\hat{y}_k$ represent the real value and predicted value of the k-th geometric parameter respectively, and K is the geometric parameters amount under this subtask.

The task objective impels each S-shape geometric model to adapt to the certain type of wavy curves but less suits to the others as the geometric models are suitable to simulate the trajectories with specific characteristics. Therefore, the extra-supervised learning method sets an extra supervision item $\hat{z}_i$ to measure the matching level of the i-th subtask (i.e., geometric models) to the target. The label z_i of the extra supervision item $\hat{z}_i$ is set to the reciprocal of the square logarithm of the prediction loss $Loss_{subtask_i}$ of the i-th subtask, namely:

$$z_i = \frac{1}{2}\log\left(\frac{1}{{Loss_{subtask_i}}^2}\right) = -\frac{1}{2}\log\left({Loss_{subtask_i}}^2\right). \tag{4}$$

Therefore, the extra supervision loss $Loss_{extra}$ is as follows:

$$Loss_{extra} = \frac{1}{2}\sum_{i=1}^{n}\left(z_i - \hat{z}_i\right)^2. \tag{5}$$

In addition, a regularization item $Loss_{regular}$ is given by Equation (7) to further enhance the extra supervision, thereby making the suitable geometric model prominent.

$$Loss_{regular} = \frac{\sum_{i=1}^{n}|\hat{z}_i|}{\max\{\hat{z}_1, \hat{z}_2, \ldots, \hat{z}_n\}}. \tag{6}$$

That is, as the regularization loss $Loss_{regular}$ decreases, the ratio of extra supervision items between the matched and mismatched subtasks will gradually increase and then the prediction errors of those unmatched subtasks make less effect on the training propagation process. Therefore, by the combination of the three loss factors, each subtask emphasizes the intra-task competence and the extra-supervision model as a whole maintains the inter-task pertinence.

The final loss function expression of the extra-supervised learning method is as follows:

$$Loss = \alpha\sum_{i=1}^{n}Loss_{subtask_i} + \beta Loss_{extra} + \gamma Loss_{regular}, \tag{7}$$

where α, β, and γ are the weights of different loss items. By minimization of the loss function, $Loss_{subtask_i}$ makes each subtask fit the target trajectory as much as possible, each subtask learns to find the suitable target by referring to the extra supervision item $\hat{z}_i$, and $Loss_{regular}$ ensures that the subtasks are punished moderately when fitting the unmatched target trajectory such that the subtasks compete with each other to achieve the most accurate reconstruction of the target trajectory.

E. Fréchet Spline Sliding Error for Trajectory Morphological Evaluation

Some studies about the applications of IMU for large-scale motion employ GPS to provide a trajectory reference and global coordinate system. In these cases, the spatial information is important, which is one of the key evaluation indicators. However, the problem which we are concerned with is the human motion tracking in a limited space with its applications to gesture recognition, hand instruction recognition, identity recognition, etc. In these applications, we focus on capturing the shape of the trajectory, so the spatial direction of the trajectory is not important compared with its morphological features. Therefore, we propose a Fréchet spline sliding error to measure the morphological errors between the original and reconstructed trajectories, which are allowed to be within two different coordinate systems. It is not affected by the difference of spatial direction and relative position of two trajectories.

Fréchet distance can measure the morphological differences of different spatial curves. Usually the smaller the Fréchet distance is, the greater the similarity is. However, different curves originating from different coordinate systems may result in large Fréchet distance due to their position or posture deviation although two curves are similar to each other, as shown in Fig. 6(a). Since the real trajectory and the IMU reconstructed trajectory are obtained from their own coordinate systems, Fréchet distance cannot be directly used to quantify their similarity. To address this issue, we adopt a line sliding over the curve to measure the morphological feature of the given curve, as shown in Fig. 6(b). Specifically, a trajectory

This article has been accepted for publication in IEEE Sensors Journal. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2023.3257867

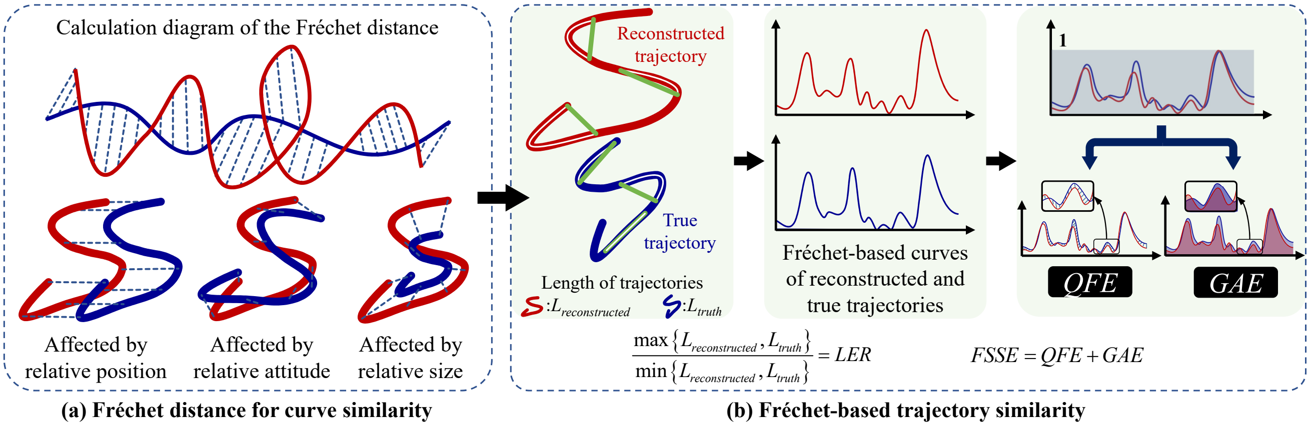


Fig. 6: The process diagram of the Fréchet-based trajectory error. (a) shows that the direct calculation of Fréchet distance is affected by the relative distance, attitude, and size of the trajectories. (b) shows the calculation process of the proposed trajectory error index. The green line represents the sliding spline function.

is divided into a fixed number of segments, and the start point and end point of each segment $s^{(i)}$ are connected to obtain a straight line $l^{(i)}$, where i is the sliding times. The Fréchet distance $f\left(s^{(i)}, l^{(i)}\right)$ between the line $l^{(i)}$ and the trajectory segment $s^{(i)}$ is calculated and then normalized to eliminate the influence of line length $L^{(i)}$ by $d^{(i)} = \frac{f\left(s^{(i)}, l^{(i)}\right)}{L^{(i)}}$. Therefore, when sliding on the whole trajectory (assuming N times of the total sliding), a Fréchet-based curve D can be obtained by a series of Fréchet distance $d^{(i)}$. We record the Fréchet-based curves of the reconstructed and true trajectories as $D_{rec} = [d_{rec}^{(1)}, d_{rec}^{(2)}, \ldots, d_{rec}^{(N)}]$ and $D_{truth} = [d_{truth}^{(1)}, d_{truth}^{(2)}, \ldots, d_{truth}^{(N)}]$, respectively. In order to reduce the influence of the sliding spline scale on the Fréchet-based curve, we adopt the Min-Max method to normalize the two curves. The normalization results of D_{rec} and D_{truth} are represented by D_{rec}^* and D_{truth}^*, respectively.

We then propose the Quadratic Fréchet Error (QFE) and Global Accumulated Error (GAE) to represent the morphological differences between the reconstructed and true trajectories. QFE is the Fréchet distance of the D_{rec}^* and D_{truth}^*, which mainly reflects the maximum local difference of both curves and is bounded by $[0, 1]$. GAE is the mean absolute error of the D_{rec}^* and D_{truth}^*, which mainly reflects the overall accumulated difference between the two trajectories and is also bounded by $[0, 1]$.

$$QFE = f\left(D_{rec}^*, D_{truth}^*\right), \tag{8}$$

$$GAE = \frac{1}{N}\sum_{i=1}^{N} |D_{rec}^*(i) - D_{truth}^*(i)|. \tag{9}$$

Finally, we give the Fréchet-based similarity error by combining the QFE and GAE, which is bounded by [0, 2]. The smaller the value, the more similar the shape of the two space curves.

$$FSSE = QFE + GAE. \tag{10}$$

In addition, to reflect the length difference between two trajectories, we define the Length Error Ratio (LER). The smaller LER is, the more similar the reconstructed trajectory is in length to the true trajectory. When the two curve length is consistent, LER=1.

$$LER = \frac{\max\{L_{reconstructed}, L_{truth}\}}{\min\{L_{reconstructed}, L_{truth}\}}, \tag{11}$$

where $L_{reconstructed}$ and L_{truth} represent geodesic distance (i.e. the curve length) of the reconstructed and true trajectories, respectively.

III. Experiments

A. Experimental settings

We invite 10 volunteers (6 young males and 4 young females) to collect their motion data. The motion is recorded by an inertial sensor module (Yesense YIS300) and an 8-camera optical equipment (Nokov Mars2H), respectively.

Our method is implemented with Python based on PyTorch on a computer with Intel(R) Xeon(R) W-2133 CPU, 64 GB RAM. The 1D-CNN model in the segmentation task contains 6 convolutional layers and 2 fully connected layers, which is trained for 100 epochs. The 1D-CNN model in the classification task contains 3 convolutional layers and 1 fully-connected layer, which is trained for 20 epochs. In the geometric model parameter prediction task, the extra-supervised learning model employs a 1D-ResNet with 18 convolutional layers as a feature extractor and three parallel single-layer MLPs as prediction head, which is trained for 300 epochs.

B. Collection of the trajectory data set

Each volunteer makes a continuous motion at a speed of 0.5-1.5m/s in no more than 3 minutes. The motion process is simultaneously collected by the inertial sensor and the 8-camera optical motion capture system. We use Python3.6 to visualize the motion trajectory collected by the optical system. We then manually label the segmentation points and the geometric model category matched by the divided trajectory fragments by observation of the real optical trajectories. We randomly select the data of nine volunteers as the training set, which includes 1210 effective segmentation points and

1200 trajectories fragments (157 polylines, 198 arcs, 381 polynomial curves, 203 hyperbola curves, and 261 S-shape curves). The data of the last volunteer is used as test data, which includes 239 segmentation points and 240 trajectories fragments (45 polylines, 36 arcs, 59 polynomial curves, 47 hyperbolic curves, and 53 S-shape curves).

The inertial sensor can measure the specific force, which is the vector difference between the actual body acceleration and the gravity vector and three-dimension angular velocity data. The optical facility includes 8 high-speed cameras, which can synchronously record the moving trajectory from their own view. When the object attached with the reflective markers moves within the view field of the cameras. The accuracy of the stereophotogrammetric system can be even higher than centimeters, up to submillimeter, as described in [49]. It should be noted that the layout of reflective balls is asymmetric, which ensures that the distance between any two balls is different. So the optical camera system can still record the racket movement by capturing at least any two balls. The collection of a concrete object movement is shown in Fig. 7. In order to keep time alignment, we uniformly adjust the sampling frequency of the optical and inertial sensors to 200 Hz. Therefore, the data collection of the two sensors can be synchronized.

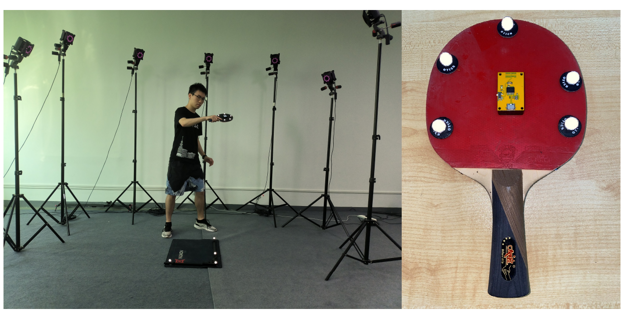


Fig. 7: The scene of motion collection within the optical camera facility (the left panel), and the racket attached with an inertial sensor and five optical markers (the right panel).

The optical trajectory data provides a convenient way of making the segmentation and classification labels. If there is an approximate acute angle in the trajectory (like the angle in the Fig. 4), the position is marked as the segmentation point. Further, the geometric type of the trajectory segment between two segmentation points can be determined by observation. Since the inertial data is synchronized with the optical data, we record the start time A_n, the end time B_n and its geometric type C_n of the n-th trajectory segment for the original 6-axis inertial data. Then we get sequence Q which contains a series of the triple segmentation labels: start point, end point and classification type:

$$Q = (A_1, B_1, C_1), (A_2, B_2, C_2) \cdots (A_n, B_n, C_n). \quad (12)$$

At the stage of the previous segmentation, we identify the classification label of each segment, which indicates that the corresponding geometric model is used to match the segmented curve. Given a geometric model matched by an inertial data segment, the parameters of the geometric model are still unknown, so we employ a deep learning model to predict the parameters of the matched geometric model. The training of the deep learning model requires a training set with the known parameter labels of the geometric model. Therefore, in order to obtain the geometric parameter-prediction labels for the inertial sensor data segment, the synchronized optical sensor data is used to predetermine the geometric model parameters as the label reference. The whole process is shown in Fig. 8.

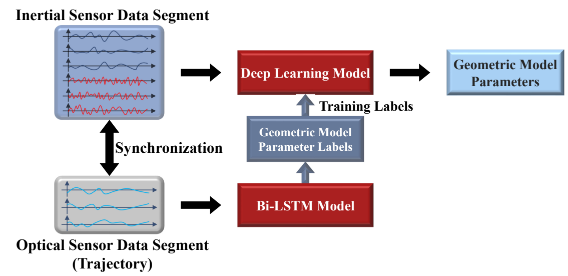


Fig. 8: The geometric model parameters for the given inertial segments are predicted by a deep learning model, which is trained with the guidance of parameter labels predetermined by the Bi-LSTM model under the case of optical data segment.

Finally, the Trajectory Reconstruction Dataset (TR-Dataset), of which each sample data is segmented by labeling the start point, end point, segment geometric type and each sample data is also equipped with a group of geometric parameters, is established for training the models used in trajectory reconstruction of inertial data. As presented in Fig. 1, the motion segmentation method, 1D-CNN model and extra-supervised learning method respectively employ the segmentation labels, the classification labels, and the parameter labels.

IV. Results

A. Motion segmentation

We first examine the accuracy of motion segmentation. the segmentation points are classified as 25 categories according to the types of trajectory segments before and after them. We input the inertial data to the motion segmentation algorithm in Section 3.1 and summarize the segmentation results by two confusion matrices, as shown in Fig. 9, where the rows and columns of the matrices respectively correspond to the types of trajectory before and after the segmentation point. Fig. 9(a) represents the detection accuracy of the algorithm for all the segmentation positions. For instance, 0.909 in the 2nd row means that in the test data, when the current segment is an arc and the next segment is a polyline, the identification accuracy of such segmentation points is 90.9%.

We then compute the average time error in milliseconds between the real and predicted segmentation positions, and the confusion matrix about the deviation in Fig. 9(b) is obtained. It is found that the algorithm has a high detection accuracy for the segmentation points in between the polylines and arcs, and the time error of such segmentation is also small. Since these two types of curves are significantly different, it is easy to be identified by the algorithm when the trajectory changes. The identification accuracy of those segmentation points belonging to the four categories of polynomial and S-shape curves is relatively low. Essentially there exists a high

This article has been accepted for publication in IEEE Sensors Journal. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2023.3257867

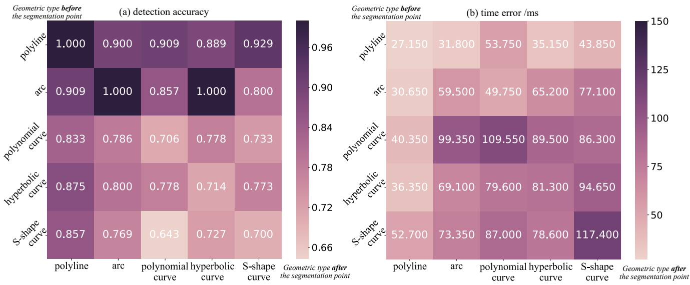


Fig. 9: Confusion matrix obtained from the result of motion segmentation. The left panel shows the detection accuracy of the motion segmentation model for different categories of segmentation points; The right panel shows the time error of the segmentation model for different categories of segmentation points in the unit of millisecond.

similarity between these curves identified as the polynomial and S-shape types. It is challenging for the segmentation method to detect these types of segmentation points. Due to the manual labeling of segmentation points, obviously the selection of segmentation points and location labeling is not unique, so different segmentation appears to be tolerant, which is feasible for the following segment approximation. According to the reconstruction results of the segments, the parameter prediction method exhibits good adaptability to the variation of trajectory segmentation. In addition, it is observed that the maximum deviation between our algorithm and manual labeling is no more than 0.12 seconds, which is acceptable for the subsequent trajectory reconstruction in this case study.

B. Classification and segmental reconstruction

After obtaining the segmentation points, each segment is then matched with a geometric model by the 1D-CNN model. Among 240 segments in the test set, 235 samples are matched with the correct geometric model, and the classification accuracy reaches 97.9%. Finally, the parameters of the geometric models are estimated by the trained extra-supervised learning model. The trajectory reconstruction results based on the four types of geometric models are presented in Fig. 10, where we test each geometric model with three different inertial data. It can be seen that the reconstruction effect of the geometric model on the trajectory is related to the complexity (number of parameters) of the geometric model. The polyline model, which contains six parameters, obtains accurate trajectory reconstruction. In contrast, the arc, polynomial, and hyperbolic models are a little complicated with 8, 7, and 7 parameters, respectively. The corresponding data segment is slightly insufficient compared with the polyline geometric model but their reconstruction performance is still good.

We select three S-shape curves with significant differences from the trajectory fragments to show the reconstruction performance of the extra-supervised learning method, as demonstrated in Fig. 11. It is found that the fitting performance of these geometric models is closely related to the characteristics of the segments, which is highlighted by the matching level

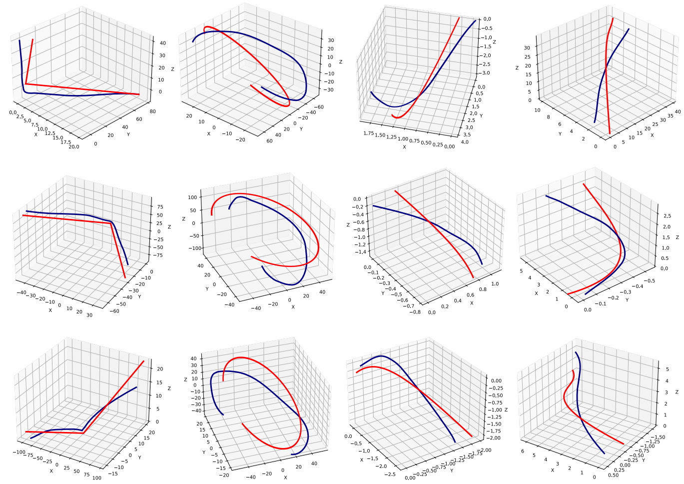


Fig. 10: The trajectory reconstruction by the four types of geometric models. The first, second, third and fourth columns correspond to the results of the polyline model, arc model, polynomial model and hyperbolic model. In each panel the blue one is the real trajectory segments and the red one is the reconstructed curve.

(i.e., the extra supervised item) $\hat{z}_i$ as an assessment of geometric model fitting on the target. The largest $\hat{z}_i$ values indicate the appropriate geometric model for the input segment as well as the accurate trajectory reconstruction. Specifically, albeit the concise S-shape geometric model shows low matching in the first two reconstruction tasks, it can achieve a precise reconstruction of the certain curve, like Trajectory3, which also suggests the rationality of this concise S-shape geometric model and the effectiveness of the extra-supervised learning method. For the parameter prediction of other geometric models, fewer subtasks can be deployed in the proposed framework. For instance, the polynomial geometric model is divided into two subtasks: the cubic and parabolic models. The polyline, arc, and hyperbolic geometric models are relatively simple, and the ideal parameter prediction results can be obtained without task decomposition under the extra-supervised learning framework. We note that the extra-supervised learning method supports the further decomposition of tasks. When the first-level subtasks are still complex and the deep learning model is difficult to achieve the desired accuracy, we can decompose them into the second-level subtasks or even the third-level subtasks with their own extra supervision items. This process artificially introduces the prior information about the trajectory types, thereby making the model adaptive to the diverse trajectory features.

C. Validation on various experimental scenarios and sensors

The aforementioned data collection process constructs the Trajectory Reconstruction Dataset (TR Dataset), which contains 1200 samples for training and 240 samples for testing. This test data collection process is the same as that of the training set. We first test the trajectory reconstruction effect on the TR Dataset. Five spatial trajectories with diverse morpho-

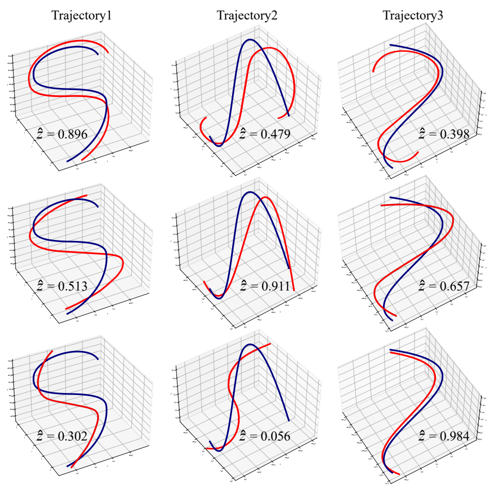


Fig. 11: Reconstruction on three wavy trajectories. Each column shows the reconstruction results of three geometric models (red curve) for one real trajectory sample (blue curve). $\hat{z}$ represents the matching level between the candidate geometric model and the segment.

logical differences are selected to present the performance of the method, as shown in Fig. 12. We also project the trajectory in the coordinate planes of XY, XZ and YZ for a detailed comparison. It demonstrates that the reconstructed trajectory has a high similarity with the real trajectory (recorded by the optical sensor) from different perspectives.

To compare the performance of our method with other trajectory reconstruction methods, we use the FSSE to quantify their reconstruction accuracy for the trajectories in Fig. 12. The baseline method is based on inertial navigation algorithm as introduced. On the basis of the baseline method, the zero-velocity compensation (ZVC) method uses the characteristic of the static state at the motion ending to compensate for the velocity calculation process, so the cumulative error in the trajectory calculation can be alleviated partly [50]. The wavelet transform method can reduce the noise of the IMU signal for better trajectory [51]. On this basis, Li *et al.* [52] perform empirical mode decomposition (EMD) on the signal and then perform wavelet threshold de-noising on the decomposed signal. The FSSE and LER between the trajectory obtained by each method and the real trajectory is given in Table I. Obviously, our method has a higher FSSE for each trajectory, which means that the trajectory reconstructed by our GDSR method is closer to the real trajectory in morphology.

Furthermore, to test the computational efficiency of our method, we test the operation time of the four modules when reconstructing previous the five trajectories, as listed in Table II. It can be found that for most trajectories, the total reconstruction time is about 1 second. For the complicated trajectory like Track4, the extra-supervised learning model needs to make parameter predictions such that it takes about 1.5 seconds for reconstruction. In fact, although the 1D-CNN and BiLSTM models consume some time in the training stage, they are quite efficient in testing.

In addition, we also design another inertial data acquisition

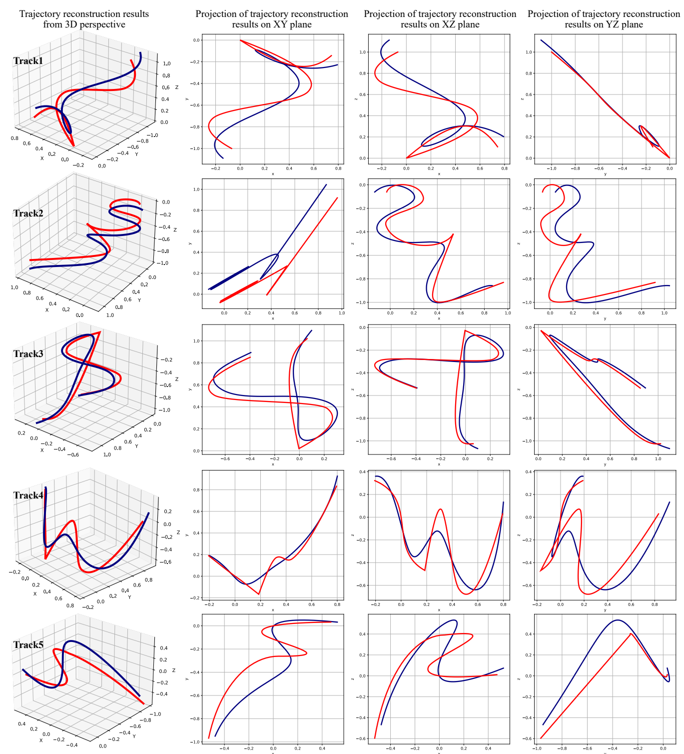


Fig. 12: The reconstruction results of five arbitrary tracks in the first column and their projection in the second, third and fourth columns. In each panel the blue and red curves are the real trajectory and reconstructed curves, respectively.

TABLE I: Comparison of various trajectory reconstruction methods in terms of FSSE and LER for five real trajectories.

Index	Methods	Track1	Track2	Track3	Track4	Track5
LER	Inertial navigation	3.132	2.375	2.943	2.806	2.654
	ZVC	1.574	1.599	1.698	1.449	1.523
	Wavelet	2.056	2.365	2.132	1.914	1.896
	EMD+Wavelet	1.456	1.502	1.557	1.437	1.535
	Ours (GDSR)	**1.221**	**1.053**	**1.018**	**1.200**	**1.075**
FSSE	Inertial navigation	1.178	1.370	0.978	0.691	0.531
	ZVC	0.078	0.082	0.372	0.085	0.133
	Wavelet	0.850	0.153	0.318	0.106	0.157
	EMD+Wavelet	0.180	0.085	0.152	0.171	0.094
	Ours (GDSR)	**0.035**	**0.049**	**0.060**	**0.052**	**0.074**

experiment and test the trajectory reconstruction effect of our GDSR method. As presented in Fig. 13, an IMU is attached at the end of the robot arm. We drag the arm to make an arbitrary movement for 60 seconds, of which slow, medium, and fast movements last for 20 seconds, respectively. The mechanical arm can record its motion and output the reference trajectory. The reconstruction results for slow, medium, and fast motion are reported in Fig. 14 and their reconstruction error is given in Table III.

D. Validation on a public dataset

In order to verify the performance of our GDSR method in different experimental environments, we employ a pub-

TABLE II: The operation time in seconds of the four modules in our method when reconstructing five trajectories.

Modules	Track1	Track2	Track3	Track4	Track5
Motion segmentation	0.179	0.214	0.209	0.244	0.216
Motion recognition	0.012	0.012	0.012	0.013	0.011
Parameter prediction	0.847	0.859	0.821	1.316	0.431
Trajectory splicing	<0.001	<0.001	<0.001	<0.001	<0.001
Total	1.038	1.085	1.042	1.573	0.658

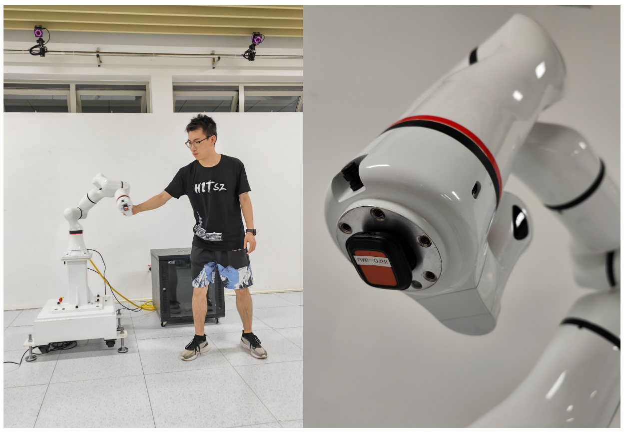


Fig. 13: The experiment scene of motion collection by a robot arm (the left panel), and the arm end with an IMU sensor (the right panel). The IMU is Xsens DOT V2 and the mechanical arm is ROKAE xMate ER3 Pro.

lic dataset, mimu_optical_sassari_dataset, which provides the magneto-inertial signals and corresponding trajectory reference acquired by a stereophotogrammetry system [38]. It is a comprehensive dataset for motion capture based on inertial and optical sensors. The dataset has three experimental scenarios: fast, medium, and slow. For each scenario, we present three trajectory reconstruction results, as shown in Fig. 15. In addition to visible results, we also give quantification analysis in terms of FSSE to measure the morphological error and LER to measure the length error between the nine reconstructed trajectories and corresponding real trajectories, as shown in Table IV.

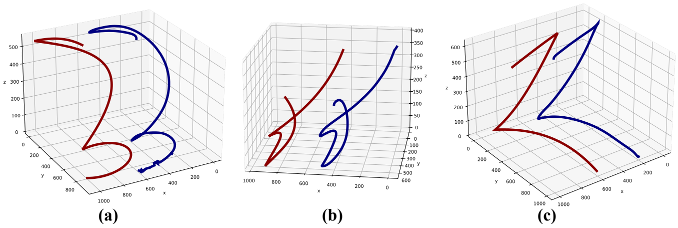


Fig. 14: Trajectory reconstruction results for the inertial data collected from the robot slow speed motion (a), medium speed motion (b) and fast speed motion (c). In each panel the blue and red curves are the real trajectory and reconstructed curves, respectively.

TABLE III: Numerical results of the morphological error and length error between the reconstructed and real trajectories based on the robot motion data.

Quantitative index	Fast	Medium	Slow
Length error (LER)	1.096	1.052	1.012
Morphological error (FSSE)	0.022	0.015	0.047

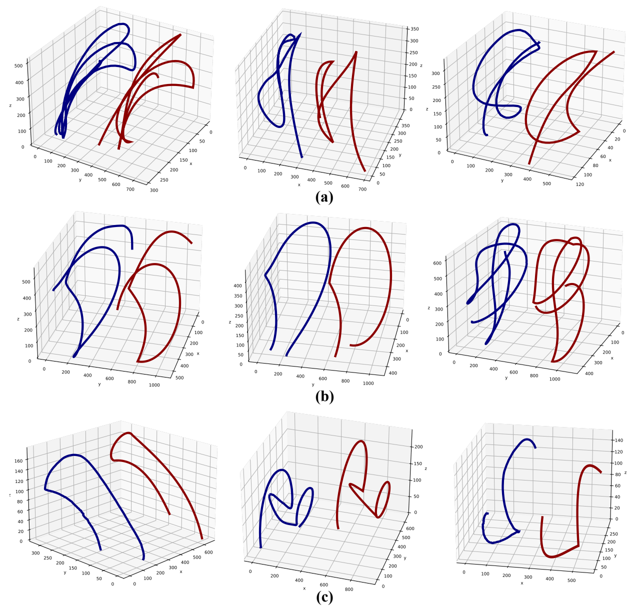


Fig. 15: Trajectory reconstruction results for the public data under three experimental scenarios: fast (a), medium (b), and slow (c). In each panel the blue and red curves are the real trajectory and reconstructed curves, respectively.

E. Generalizability and sensitivity analysis

1) Analysis of training strategy: Nagarajan *et al.* [53] prove that the generalization bound can increase with the dataset size. Therefore, to increase training samples, we adopt a training strategy that does not rely on the validation set [54]. In our experiment, the training loss shows the features that can be used to determine the optimal training epoch. While the training loss decreases to a low level, it then stabilizes for some epochs. Then after more training epochs, the training loss is inclined to fluctuate. Charles *et al.* [55] prove that the fluctuation is a manifestation of overfitting and explain its rationality from the perspective of statistical mechanics. Therefore, we set the time of switching from the stable state to fluctuating state in training loss as the optimal training epoch. The operation is shown in Fig. 16. First, we run with a number of epochs to get the training loss from underfitting to overfitting. Due to the low resolution of the curve, we can not accurately obtain the turning point between the stable state and the fluctuating state, so the certain interval is enlarged for a better identification. Finally, the selected epoch (511) is marked by the red dot in Fig. 16. We also use the early stopping strategy to train the model, and the resulting epoch (487) is marked by the blue dot in the figure. It can be found that the training epoch obtained by our method is very close

TABLE IV: Numerical results of the morphological error and length error between the reconstructed and real trajectories based on the public data samples.

Quantitative index	Fast1	Fast2	Fast3	Medium1	Medium2	Medium3	Slow1	Slow2	Slow3
Length error (LER)	1.028	1.143	1.053	1.059	1.017	1.048	1.029	1.039	1.252
Morphological error (FSSE)	0.039	0.034	0.026	0.037	0.026	0.059	0.060	0.037	0.061

to that of the early stopping method.

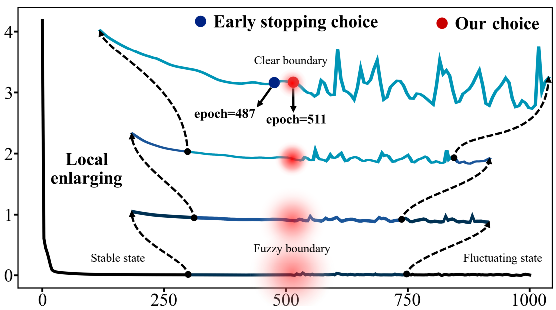


Fig. 16: Training epoch selection based on the training loss. The black curve represents the loss function, and the three blue curves above are the certain range enlarging for better identification. The red dot represents the epoch selected by our strategy. The blue dot represents the epoch selected by the early stopping strategy.

To verify the effect of different training epochs on the trajectory reconstruction accuracy, we select six models with different training epochs (1, 250, 487, 511, 750, 1000) and test their trajectory reconstruction performance in terms of the FSSE and LER indexes, as shown in Fig. 17. Since the early stopping method extracts part of the samples from the training set as the validation set, the actual training set is smaller than our method. Therefore, it is found that our choice epoch is even slightly better than the early stopping choice epoch.

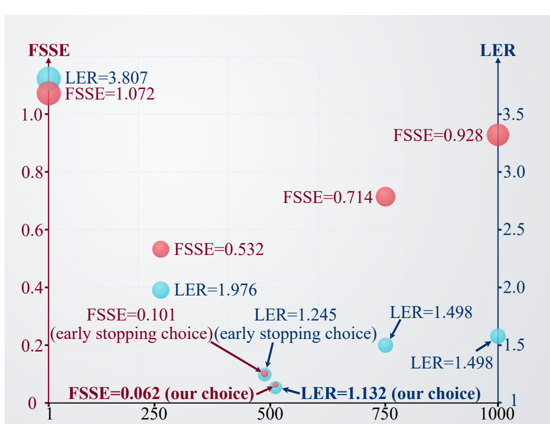


Fig. 17: The trajectory reconstruction effect of models trained with six training epochs. The red dots represents FSSE of trajectory reconstruction. The blue dots represents LER of trajectory reconstruction.

2) Analysis of embedded navigation algorithm and calibration strategy: To verify the reconstruction effect of the GDSR method on the uncalibrated inertial sensors, we collect thirty motion data with three kinds of IMUs (Yesense YIS300, WITE BWT901CL, Xsens DOT V2) under calibrated and uncalibrated conditions, respectively. The average trajectory reconstruction performance is evaluated by LER and FSSE, as shown in Table V. It can be found that the LER and FSSE under both conditions are small, thereby indicating the generalization of our method for different IMUs. In addition, the difference of numerical results between the two conditions is slight, which demonstrates the robustness of our method against IMU calibration condition.

TABLE V: Numerical trajectory reconstruction results of three IMUs under calibrated and uncalibrated conditions based on the GDSR method.

Index	Condition	Yesense YIS300	Xsens DOT V2	WITE BWT901CL
LER	Uncalibrated	1.291	1.229	1.289
	Calibrated	1.285	1.221	1.277
FSSE	Uncalibrated	0.067	0.059	0.061
	Calibrated	0.062	0.057	0.060

In addition, we test the impact of different embedded navigation algorithms on the GDSR method, and the numerical results are given in Table VI. Compared with the most basic inertial navigation (IN) algorithm, other navigation algorithms have been proven to make improvements in motion estimation, so when embedding them in the GDSR method, the reconstruction scale error, LER, can be slightly reduced. However, since these navigation algorithms cannot obtain accurate motion curves, the shape reconstruction error, FSSE, is almost unchanged for all the four algorithms embedded into the GDSR method.

TABLE VI: Trajectory reconstruction results of the GDSR method embedded with four navigation algorithms.

Index	GDSR(IN)	GDSR(ZVC)	GDSR(Wavelet)	GDSR(EMD+Wavelet)
LER	1.098	1.091	1.101	1.084
FSSE	0.051	0.053	0.051	0.050

V. Discussion

Our trajectory reconstruction method performs well on multiple datasets. It has significant value for applications that rely on motion trajectory morphology. Meanwhile, to measure the morphological error of the reconstructed curves, we propose a Fréchet-based similarity error, which can avoid the influence of spatial position and direction difference between two curves.

This indicator verifies the feasibility and effectiveness of the proposed method for diverse datasets.

The visualization and numerical comparison of trajectory reconstruction show that the proposed method can reconstruct various real trajectories accurately. Specifically, the FSSE reveals that the reconstruction accuracy for the complicated motion trajectory is relatively higher as they usually contain more direction or spatial changes for potential segmentation. So the proposed method can find the appropriate segments and give better segment reconstruction. We also note that the FSSE results of fast and medium motion are lower compared with some slow motion according to Table IV. The direction often changes significantly or frequently under fast motion, so it is convenient to identify appropriate segmentation points, thereby leading to better reconstruction performance. Meanwhile, since human motion, especially gesture movement, can be regarded as complicated motion, our method is competent for the trajectory reconstruction of human motion.

We examine the trajectory reconstruction time of our method, including the four embedded modules. According to Table II, the total reconstruction time of Track1, Track2, Track3, and Track4 is about 1 second. For Track5, the reconstruction can be completed in less than 1 second. Note that all the times include the data loading time. It indicates that the proposed method can approximately achieve real-time trajectory reconstruction. In the future, we intend to optimize the geometric model parameter estimation and further improve the trajectory reconstruction efficiency for a shorter time.

VI. Conclusion

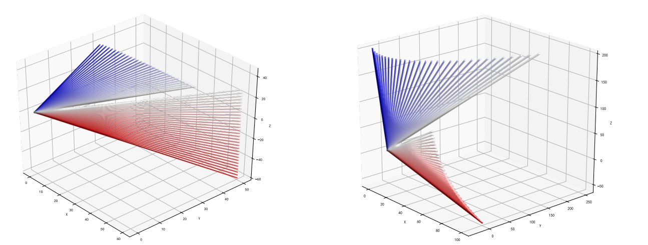
This paper proposes an inspiring method for an accurate trajectory reconstruction. On this basis, it has wide applications in exercise health, human-computer interaction, semantic recognition, and other fields. Our method can be divided into four parts: motion segmentation, motion state recognition, geometric parameter prediction, and segmented trajectory splicing. Among them, motion segmentation and geometric parameter prediction are two key tasks.

For the task of IMU motion segmentation, we propose a temporal and spatial fusion segmentation method that combines the dynamic feature of acceleration and angular velocity in IMU signals with the spatial morphological feature of motion. The dynamic feature extraction of the IMU signal is achieved by the data-driven model (1D-CNN). Meanwhile, the spatial morphological feature extraction is achieved by the multi-scale spline function and the Fréchet distance. By the fusion of the two features, the motion segmentation algorithm has strong adaptability to various trajectory scenarios and wide applicability to practical cases.

For the task of geometric parameter prediction, the 1D-CNN can achieve accurate prediction for the general geometric models such as polyline, arc, and hyperbola. Taking three kinds of S-shape geometric models into account, we propose an extra-supervised learning method, which integrates multiple S-shape model parameter prediction tasks into a deep learning framework, thereby improving the utilization efficiency of training samples and the parameter prediction accuracy. The experimental results demonstrate that the matched geometric model can accurately reconstruct the given trajectory fragments.

To quantify the similarity of the reconstructed and original trajectories, we design the FSSE index with the range of [0, 2] and the LER index with the range of $[1, +\infty]$. In the extensive experiments, the FSSE of our method is less than 0.074, and the LER is less than 1.221, which achieves the SOTA performance. In addition, we test different IMUs in two experimental scenarios and one public dataset. The results demonstrate that the GDSR method presents superiority for different data collection processes and usage scenarios of the inertial sensor. We further perform the generalizability and sensitivity analysis on the GDSR method. Firstly, we provide an analysis of our training epoch selection strategy. Benefiting from the non-dependence on the validation data, our epoch selection strategy obtains stronger generalization than the early stopping strategy since it allows us more samples for training. We then test the GDSR method using three types of IMUs in both calibrated and uncalibrated conditions. The results demonstrate that our method is independent of the IMU types and calibration conditions. Finally, we test the influence of different embedding navigation algorithms on the GDSR method. The various embedding navigation algorithms do not affect the trajectory reconstruction accuracy of our GDSR method.

Appendix
Visualization of the Geometric Models



(a) (b)

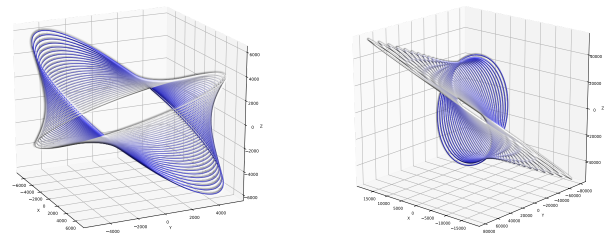
Fig. 18: Visualization of the polyline model in space. As an illustration, (a) shows the polyline model scattered in two fixed planes; (b) shows the polyline model rotating around two central axis. The color lines represent the morphological variation with the different parameter configuration.

References

[1] C.-S. Jao and A. M. Shkel, "A reconstruction filter for saturated accelerometer signals due to insufficient fsr in foot-mounted inertial navigation system," *IEEE Sensors Journal*, vol. 22, no. 1, pp. 695–706, 2021.

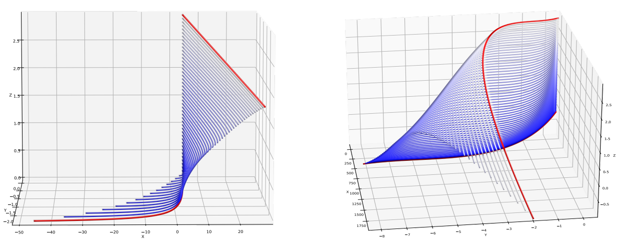
[2] Y. Lin, L. Miao, Z. Zhou, and C. Xu, "A high-accuracy method for calibration of nonorthogonal angles in dual-axis rotational inertial navigation system," *IEEE Sensors Journal*, vol. 21, no. 15, pp. 16 519–16 528, 2021.

[3] J. Sui, L. Wang, W. Wang, and T. Song, "Improvement on the pitch and roll output of rotation inertial navigation system," *IEEE Sensors Journal*, vol. 17, no. 11, pp. 3251–3256, 2017.



(a) (b)

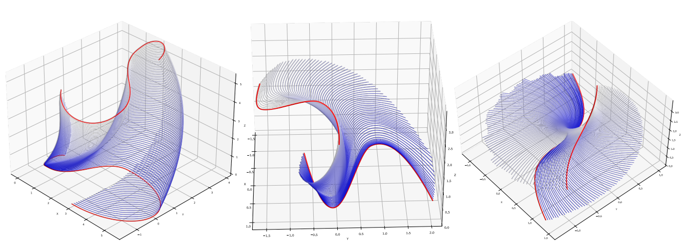
Fig. 19: The arc morphological variation in space. (a) shows the rotation of the model; (b) shows the model curve with one axis telescopic changes.



(a) (b)

Fig. 20: The polynomial and hyperbolic models in space. (a) shows a variety of hyperbolic curves; (b) shows a variety of parabolic curves.

[4] Z.-Q. Zhang, X.-L. Meng, and J.-K. Wu, “Quaternion-based kalman filter with vector selection for accurate orientation tracking,” *IEEE Transactions on Instrumentation and Measurement*, vol. 61, no. 10, pp. 2817–2824, 2012.
[5] H.-W. Chang, J. Georgy, and N. El-Sheimy, “Improved cycling navigation using inertial sensors measurements from portable devices with arbitrary orientation,” *IEEE Transactions on Instrumentation and Measurement*, vol. 64, no. 7, pp. 2012–2019, 2015.
[6] W. Zhu, L. Wang, L. Chen, N. Xu, and Y. Su, “Robotic visual-inertial calibration via deep deterministic policy gradient learning,” *IEEE Sensors Journal*, vol. 22, no. 14, pp. 14 448–14 457, 2022.
[7] C. Chen, R. Jafari, and N. Kehtarnavaz, “A real-time human action recognition system using depth and inertial sensor fusion,” *IEEE Sensors Journal*, vol. 16, no. 3, pp. 773–781, 2016.
[8] A. I. Cuesta-Vargas, A. Galán-Mercant, and J. M. Williams, “The use of inertial sensors system for human motion analysis,” *Physical Therapy Reviews*, vol. 15, no. 6, pp. 462–473, 2010.
[9] A. Filippeschi, N. Schmitz, M. Miezal, G. Bleser, E. Ruffaldi, and



(a) (b) (c)

Fig. 21: Visualization of the three S-shape geometric models, including (a) the smooth S-shape geometric model with arc ends, (b) the sharp S-shape geometric model with parabolic ends, and (c) the concise S-shape geometric model with a cubic curve in between.

D. Stricker, “Survey of motion tracking methods based on inertial sensors: A focus on upper limb human motion,” *Sensors*, vol. 17, no. 6, p. 1257, 2017.
[10] I. H. Lopez-Nava and A. Munoz-Melendez, “Wearable inertial sensors for human motion analysis: A review,” *IEEE Sensors Journal*, vol. 16, no. 22, pp. 7821–7834, 2016.
[11] R. Caldas, M. Mundt, W. Potthast, F. B. de Lima Neto, and B. Markert, “A systematic review of gait analysis methods based on inertial sensors and adaptive algorithms,” *Gait & posture*, vol. 57, pp. 204–210, 2017.
[12] F. Petraglia, L. Scarcella, G. Pedrazzi, L. Brancato, R. Puers, and C. Costantino, “Inertial sensors versus standard systems in gait analysis: a systematic review and meta-analysis,” *European journal of physical and rehabilitation medicine*, vol. 55, no. 2, pp. 265–280, 2019.
[13] N. F. Ribeiro and C. P. Santos, “Inertial measurement units: A brief state of the art on gait analysis,” in *2017 IEEE 5th Portuguese Meeting on Bioengineering (ENBENG)*. IEEE, 2017, pp. 1–4.
[14] C. Mazzà, L. Alcock, K. Aminian, C. Becker, S. Bertuletti, T. Bonci, P. Brown, M. Brozgol, E. Buckley, A.-E. Carsin *et al.*, “Technical validation of real-world monitoring of gait: a multicentric observational study,” *BMJ open*, vol. 11, no. 12, p. e050785, 2021.
[15] V. Camomilla, E. Bergamini, S. Fantozzi, and G. Vannozzi, “Trends supporting the in-field use of wearable inertial sensors for sport performance evaluation: A systematic review,” *Sensors*, vol. 18, no. 3, p. 873, 2018.
[16] P. Picerno, V. Camomilla, and L. Capranica, “Countermovement jump performance assessment using a wearable 3d inertial measurement unit,” *Journal of sports sciences*, vol. 29, no. 2, pp. 139–146, 2011.
[17] D. Giansanti, S. Morelli, G. Maccioni, and G. Costantini, “Toward the design of a wearable system for fall-risk detection in telerehabilitation,” *Telemedicine and e-Health*, vol. 15, no. 3, pp. 296–299, 2009.
[18] H. M. Hondori, M. Khademi, and C. V. Lopes, “Monitoring intake gestures using sensor fusion (microsoft kinect and inertial sensors) for smart home tele-rehab setting,” in *2012 1st Annual IEEE Healthcare Innovation Conference*, 2012.
[19] I. Weygers, M. Kok, M. Konings, H. Hallez, H. De Vroey, and K. Claeys, “Inertial sensor-based lower limb joint kinematics: A methodological systematic review,” *Sensors*, vol. 20, no. 3, p. 673, 2020.
[20] P. Picerno, A. Cereatti, and A. Cappozzo, “Joint kinematics estimate using wearable inertial and magnetic sensing modules,” *Gait & posture*, vol. 28, no. 4, pp. 588–595, 2008.
[21] A. Zedda, E. Gusai, M. Caruso, S. Bertuletti, G. Baldazzi, S. Spanu, D. Riboni, A. Pibiri, M. Monticone, A. Cereatti *et al.*, “Domomea: A home-based telerehabilitation system for stroke patients,” in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 2020, pp. 5773–5776.
[22] E. Digo, L. Gastaldi, M. Antonelli, S. Pastorelli, A. Cereatti, and M. Caruso, “Real-time estimation of upper limbs kinematics with imus during typical industrial gestures,” *Procedia Computer Science*, vol. 200, pp. 1041–1047, 2022.
[23] J.-O. Nilsson, I. Skog, P. Händel, and K. Hari, “Foot-mounted ins for everybody-an open-source embedded implementation,” in *Proceedings of the 2012 IEEE/ION Position, Location and Navigation Symposium*. Ieee, 2012, pp. 140–145.
[24] M. Brossard, A. Barrau, and S. Bonnabel, “Ai-imu dead-reckoning,” *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 4, pp. 585–595, 2020.
[25] F. Salis, S. Bertuletti, K. Scott, M. Caruso, T. Bonci, E. Buckley, U. Della Croce, C. Mazzà, and A. Cereatti, “A wearable multi-sensor system for real world gait analysis,” in *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 2021, pp. 7020–7023.
[26] Y. Jia, S. Li, Y. Qin, and R. Cheng, “Error analysis and compensation of mems rotation modulation inertial navigation system,” *IEEE Sensors Journal*, vol. 18, no. 5, pp. 2023–2030, 2018.
[27] C. N. K. Nam, H. J. Kang, and Y. S. Suh, “Golf swing motion tracking using inertial sensors and a stereo camera,” *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 4, pp. 943–952, 2014.
[28] D. H. Won, E. Lee, M. Heo, S.-W. Lee, J. Lee, J. Kim, S. Sung, and Y. J. Lee, “Selective integration of gnss, vision sensor, and ins using weighted dop under gnss-challenged environments,” *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 9, pp. 2288–2298, 2014.
[29] L. Peng, S. Zheng, P. Li, Y. Wang, and Q. Zhong, “A comprehensive detection system for track geometry using fused vision and inertia,” *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–15, 2021.

[30] S. Zihajehzadeh, P. K. Yoon, B.-S. Kang, and E. J. Park, "Uwb-aided inertial motion capture for lower body 3-d dynamic activity and trajectory tracking," *IEEE Transactions on Instrumentation and Measurement*, vol. 64, no. 12, pp. 3577–3587, 2015.

[31] M. Caruso, A. M. Sabatini, D. Laidig, T. Seel, M. Knaflitz, U. Della Croce, and A. Cereatti, "Analysis of the accuracy of ten algorithms for orientation estimation using inertial and magnetic sensing under optimal conditions: One size does not fit all," *Sensors*, vol. 21, no. 7, p. 2543, 2021.

[32] P. H. Veltink, P. Slycke, J. Hemssems, R. Buschman, G. Bultstra, and H. Hermens, "Three dimensional inertial sensing of foot movements for automatic tuning of a two-channel implantable drop-foot stimulator," *Medical engineering & physics*, vol. 25, no. 1, pp. 21–28, 2003.

[33] A. Cereatti, D. Trojaniello, and U. Della Croce, "Accurately measuring human movement using magneto-inertial sensors: techniques and challenges," in *2015 IEEE International Symposium on Inertial Sensors and Systems (ISISS) Proceedings*. IEEE, 2015, pp. 1–4.

[34] M. Miezal, B. Taetz, and G. Bleser, "On inertial body tracking in the presence of model calibration errors," *Sensors*, vol. 16, no. 7, 2016. [Online]. Available: https://www.mdpi.com/1424-8220/16/7/1132

[35] Y. Huang, M. Kaufmann, E. Aksan, M. J. Black, O. Hilliges, and G. Pons-Moll, "Deep inertial poser: Learning to reconstruct human pose from sparse inertial measurements in real time," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1–15, 2018.

[36] R. Stanzani, P. Dondero, A. Mantero, and M. Testa, "Measurement accuracy of an upper limb tracking system based on two hillcrest labs bno080 imu sensors: An environmental assessment," *IEEE Sensors Journal*, vol. 20, no. 17, pp. 10 267–10 274, 2020.

[37] D. Dinu, M. Fayolas, M. Jacquet, E. Leguy, J. Slavinski, and N. Houel, "Accuracy of postural human-motion tracking using miniature inertial sensors," *Procedia engineering*, vol. 147, pp. 655–658, 2016.

[38] M. Caruso, A. M. Sabatini, M. Knaflitz, M. Gazzoni, U. Della Croce, and A. Cereatti, "Orientation estimation through magneto-inertial sensor fusion: A heuristic approach for suboptimal parameters tuning," *IEEE Sensors Journal*, vol. 21, no. 3, pp. 3408–3419, 2020.

[39] M. Caruso, A. M. Sabatini, M. Knaflitz, U. Della Croce, and A. Cereatti, "Extension of the rigid-constraint method for the heuristic suboptimal parameter tuning to ten sensor fusion algorithms using inertial and magnetic sensing," *Sensors*, vol. 21, no. 18, p. 6307, 2021.

[40] T.-Y. Pan, C.-H. Kuo, H.-T. Liu, and M.-C. Hu, "Handwriting trajectory reconstruction using low-cost imu," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 3, no. 3, pp. 261–270, 2018.

[41] J.-S. Wang, Y.-L. Hsu, and J.-N. Liu, "An inertial-measurement-unit-based pen with a trajectory reconstruction algorithm and its applications," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 10, pp. 3508–3521, 2009.

[42] P. M. S. Ribeiro, A. C. Matos, P. H. Santos, and J. S. Cardoso, "Machine learning improvements to human motion tracking with imus," *Sensors*, vol. 20, no. 21, p. 6383, 2020.

[43] B.-S. Lin, I.-J. Lee, S.-P. Wang, J.-L. Chen, and B.-S. Lin, "Residual neural network and long short-term memory–based algorithm for estimating the motion trajectory of inertial measurement units," *IEEE Sensors Journal*, vol. 22, no. 7, pp. 6910–6919, 2022.

[44] C. Chen, C. X. Lu, J. Wahlström, A. Markham, and N. Trigoni, "Deep neural network based inertial odometry using low-cost inertial measurement units," *IEEE Transactions on Mobile Computing*, vol. 20, no. 4, pp. 1351–1364, 2019.

[45] Z. Chen, X. Pan, C. Chen, and M. Wu, "Contrastive learning of zero-velocity detection for pedestrian inertial navigation," *IEEE Sensors Journal*, vol. 22, no. 6, pp. 4962–4969, 2022.

[46] M. Zeng, L. T. Nguyen, B. Yu, O. J. Mengshoel, J. Zhu, P. Wu, and J. Zhang, "Convolutional neural networks for human activity recognition using mobile sensors," in *6th International Conference on Mobile Computing, Applications and Services*. IEEE, 2014, pp. 197–205.

[47] M. Duan, K. Li, C. Yang, and K. Li, "A hybrid deep learning cnn–elm for age and gender classification," *Neurocomputing*, vol. 275, pp. 448–461, 2018.

[48] S. Zhou and B. Tan, "Electrocardiogram soft computing using hybrid deep learning cnn-elm," *Applied Soft Computing*, vol. 86, p. 105778, 2020.

[49] L. Chiari, U. Della Croce, A. Leardini, and A. Cappozzo, "Human movement analysis using stereophotogrammetry: Part 2: Instrumental errors," *Gait & posture*, vol. 21, no. 2, pp. 197–211, 2005.

[50] S. Y. Cho, J. H. Lee, and C. G. Park, "A zero-velocity detection algorithm robust to various gait types for pedestrian inertial navigation," *IEEE Sensors Journal*, vol. 22, no. 6, pp. 4916–4931, 2022.

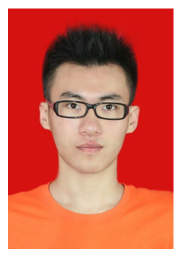
[51] Q. Fan, H. Zhang, P. Pan, X. Zhuang, J. Jia, P. Zhang, Z. Zhao, G. Zhu, and Y. Tang, "Improved pedestrian dead reckoning based on a robust adaptive kalman filter for indoor inertial location system," *Sensors*, vol. 19, no. 2, p. 294, 2019.

[52] D. Li, X. Jia, and J. Zhao, "A novel hybrid fusion algorithm for low-cost gps/ins integrated navigation system during gps outages," *Ieee Access*, vol. 8, pp. 53 984–53 996, 2020.

[53] V. Nagarajan and J. Z. Kolter, "Uniform convergence may be unable to explain generalization in deep learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[54] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 270–279.

[55] C. H. Martin and M. W. Mahoney, "Rethinking generalization requires revisiting old ideas: statistical mechanics approaches and complex learning behavior," *arXiv preprint arXiv:1710.09553*, 2017.



Yifeng Wang received the M.S. degree in Harbin Institute of Technology, Shenzhen, China, where he is currently working toward the Ph.D. degree.

His research interests include machine learning model design and analysis, motion tracking, human activity recognition, and time series analysis.

Yi Zhao received the Master degree from Zhejiang University, Hangzhou, China, in 2003 and the Ph.D. degree from Hong Kong Polytechnic University, Hong Kong, China, in 2007. Since graduating, he has been with Harbin Institute of Technology, Shenzhen, China, in 2007 and is currently a Professor. His research interests include nonlinear dynamics, nonlinear time series analysis, and complex system modeling. His recent works have been on the application of mathematical methods to a diverse range of application problems.