Tech Report of PALLAS: PArtial body simuLation with LAtend neural kinematic Solver

Qi Jin[1,2]

[1]Rensselaer Polytechnic Institute
[2]Project of CSCI 4968 RCOS 2024 fall

1 abstract

In the past there were many research imply that deeep learning can easily and accurately solve inverse kinematic problem with inertia motion units information. In addition, there are also some research such as AvatarPoser and AGRoL give a fundamental method to simulate lower body information based upper body motion. However, because of the restriction of diffusion model and previous transformer model, these research cannot simulate model in a short time or simulate accurate motion information.
Recently, there are plenty of research demonstrate LLAMA with rotatory based position encoding can help deep learning model modeling long sequence data feature. Furthermore, diffusion model and vision auto-regression model also show new potential ability of deep learning in image generation and data simulation. Based on these previous research, In this tech report, I proposed a novel 3D human pose estimation and generation model namely **Pallas** as my personal project of RCOS, which can solve and estimate human upper body motion information and based on the upper body information to generate lower body information.

2 Introduction

3 relate work

4 Method

4.1 Problem Definition

In this task, by deploying four IMUs in waist, two wrists and neck, we can track the motion of upper body and simulate motion of lower body. IMU

can obtain information of rotation which represent in rotation matrix and denote as $R(t) \in \mathbb{R}^9$, t represent the number frame we input to the model and each frame in $\mathbb{R}^9$. Additionally, IMU also can obtain linear acceleration information which denote as $A(t) \in \mathbb{R}^3$. For the output, I use invertible inverse kinematic solver which can predict both position, which represent as xyz axis and denote as $P(t) \in \mathbb{R}^3$ and rotation, which represent as r6d form and denote as $\theta(t) \in \mathbb{R}^6$. The total input can be written as $Input(t) = (IMU_1(t), IMU_2(t), IMU_3(t), IMU_4(t))$ and $IMU(t) = (A(t), R(t))$. The total output is $Output(t) = (P_1(t), \theta_1(1), P_2(T), \theta_2(t)...P_n(t), \theta_n(t))$, n represent the number of key-points.

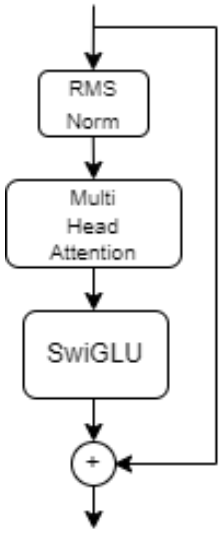
4.2 LLAMA based Encoder

In the designing of Encoder, I mostly follow the structure of LLAMA model. I keep the pre-normalization step with trainable Root Mean Square normalization which can be written as

$$\bar{x}_i = \frac{x_i}{RMS(X)} \cdot g_i \tag{1}$$

$$RMS(X) = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} \tag{2}$$

g_i is a trainable parameter to control the weight of the Root Mean Square normalization in this process.
I use SwiGLU as activation and use residual design to prevent degeneration if deep learning model is deep.



4.2.1 multihead attention with Rotatory Position Embedding

In the current research, Roformer display a significant result of Rotatory Position embedding to model long series data, and rotatory based position embedding have been widely used in both nature language processing, to generate long context, and computer vision, to encode and generate long video sections. In order to enhance the long time series modeling ability, I separate do 2D rotatory position embedding to temporal feature, which can be formulized as

$$q = Linear(x, m) \tag{3}$$

$$k = Linear(x, n) \tag{4}$$

x represent the input to the multihead attention and m,n represent the absolute position embedding of x in different linear layer. Following the equation from Roformer, after rotatory position encoding, q or k can be written as:

$$q_r = rotatory(q) = qe^{im\theta} \tag{5}$$

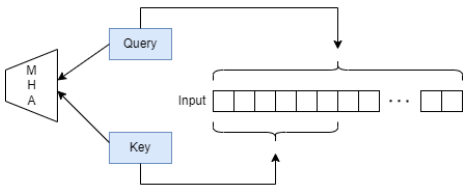
θ represent the degree of rotation, and total multihead attention formula is,

$$MHA = softmax(\frac{q_r^t \cdot k_r}{bs^{0.5}}) \cdot V \tag{6}$$

bs represent batch size of input.

4.2.2 Cross Attention of short and long time series

In the research of PIP, they represent the influence of error accumulation for estimating long term human pose with IMU data without initialize or calibrate neural network frequently. To solve this problem, I purpose the long and short time series cross attention capture the information from different range.



In this attention, Key and Value are used to capture short term time series, because Key and Value play dominate role in extract feature information and I want model focus more in more relevant information and Query is used to capture long term information to provide more long term motion information.

4.3 Invertible Inverse Kinematic solver (IIK)

Estimate and simulate human motion is a complex for single supervise signal. Hence, some past research improve the result of neural network by interpolating extra supervise signal ,such as position information, into media layer of neural network. Additionally, despite the size of input and out tensor and value of Jacobian matrix restrict the generation ability of flow based models, such as NICE, RealNVP and Glow, forward kinematic problem is a single solution and optimize free problem. By introducing the Invertible Neural Network (INN), model can be optimized with bidirectional validation.

4.3.1 Bidirectional kinematic problem

Forward kinematic problems are set of problems which calculating position by rotation information. By setting waist point as root point, every leaf points though kinematic chain can be denoted as:

$$P_n = R_n(TP_n - TP_{n-1}) + P_{n-1} \tag{7}$$

P_n means the position of n index node and $n-1$ means parent node of n. The position of template pose denotes as TP_n and R is calculate the relative position of n and $n-1$. And node 0 means root node.
Inverse kinematic problem can be denoted as:

$$R_{all} = S(P_{all}) \tag{8}$$

$$P_{all} = S^t(P_{all}) \tag{9}$$

S represents the invertible inverse kinematic solver and S^t means inverse mode of the invertible inverse kinematic solver
Based on the formula(7), formula(8) and formula(9), during the forward training, I wish result of solver is approximate to ground truth and during the inverse training, result of inverse solver is approximate to result of formula (7).

4.3.2 Loss

During the forward training of IIK, the loss makes up with forward loss and inverse loss.

$$L_{IIK} = MSE(P, P_{gt}) \tag{10}$$

$$L^t_{IIK} = MSE(R, R_{gt}) \cdot \alpha + (1-\alpha) \cdot MSE(R, \hat{R}) \tag{11}$$

4.4 Generator

4.4.1 Temporal Auto Regression Model