ChatMotion: A Multimodal Multi-Agent for Human Motion Analysis

Anonymous ACL submission

Abstract

Advancements in Multimodal Large Language Models (MLLMs) have improved human motion understanding. However, these models remain constrained by their "instruct-only" nature, lacking adaptability for diverse analytical perspectives. To address these challenges, we introduce ChatMotion, a multimodal multi-agent framework for human motion analysis. ChatMotion dynamically interprets user intent, decomposes complex tasks into meta-tasks, and activates specialized function modules for motion comprehension. It integrates specialized toolset, MotionCore, to analyze human motion from various perspectives. Extensive experiments demonstrate ChatMotion's precision and adaptability for human motion understanding.

1 Introduction

Human motion understanding (Li et al., 2024b; Loper et al., 2015) has gained attention due to its wide-ranging applications in fields such as healthcare, human-computer interaction, rehabilitation, sports science, and virtual human modeling (Plappert et al., 2016; Zhang et al., 2021; Hong et al., 2022; Qu et al., 2024). A deep understanding of human motion can drive advancements in areas like physical therapy (Smeddinck, 2020), immersive virtual experiences (Xiao et al., 2024), and assistive technology interfaces (Khiabani, 2021). As human motion data becomes more accessible, the demand for systems capable of effectively processing and analyzing this data has increased (Zhang, 2024). However, existing motion understanding models often struggle to handle the accurate analysis of human motions and the dynamic nature of user requirements (Meng et al., 2020; Smeddinck, 2020). These MLLMs tend to exhibit limited adaptability to complex, multi-faceted user queries and are often constrained by biases inherent in single-model analyses (Frangoudes et al., 2022), failing to integrate diverse insights into a comprehensive, generalizable, and accurate analysis (Xu et al., 2021).

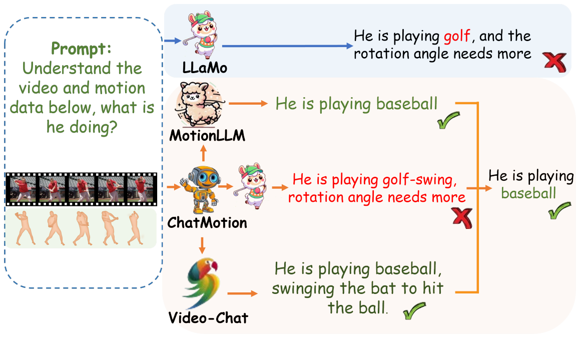

Prompt:
Understand the video and motion data below, what is he doing?
LLaMo
He is playing golf, and the rotation angle needs more
MotionLLM
He is playing baseball
ChatMotion
He is playing golf-swing, rotation angle needs more
He is playing baseball
Video-Chat
He is playing baseball, swinging the bat to hit the ball.

Figure 1: ChatMotion compares with LLaMo (Li et al., 2024b), a state-of-the-art MLLM for motion understanding. By integrating insights from multiple MLLM results, ChatMotion delivers more accurate analysis.

With the large model application development(Zheng et al., 2025; Yang et al., 2024), recent advancements in human motion understanding have progressed, particularly with LLM-based methods targeting specialized tasks and domain-specific applications. Models such as MotionGPT (Jiang et al., 2023) and MotionLLM (Chen et al., 2024a) propose methods to encode motion into structured formats, translating motion data (e.g., videos) into textual descriptions for general motion understanding tasks. Building on this foundation, LLaMo (Li et al., 2024b) integrates a motion encoder and cross-talker without relying on motion quantification, demonstrating capabilities in general motion comprehension and specialized analysis across professional domains. These LLM-based motion models aim to bridge raw motion data and interpretable insights, enabling applications in diverse fields.

Despite these advancements, existing approaches still face limitations when applied to broader motion analysis tasks. A key challenge is their reliance on single-model architectures, which often struggle to address complex user requirements (Wei et al., 2024). These models show limited adaptability to dynamic user goals and lack mechanisms to integrate insights from multiple MLLMs, constraining their ability to provide comprehensive results. Additionally, they lack effective frameworks for verifying outcomes or refining analyses based on user feedback, which may affect reliability (Lan et al., 2022). As a result, current Motion LLMs encounter challenges in delivering accurate and complete human motion analyses.

To address these challenges, We present ChatMotion, the first agent-based framework for motion understanding that integrates a multi-agent architecture, consisting of a Planner, Executor, and Verifier, together with our modular MotionCore toolbox. Given motion or video data with a user prompt ChatMotion uses a Planner to decompose the task into sub-tasks, each handled by the Executor using tools within MotionCore. The MotionCore consists of four modules: MotionAnalyzer, Aggregator, Generator, and Auxiliary Tools. The Executor calls upon the MotionAnalyzer, utilizing multiple motion LLMs to analyze data from various perspectives. The Aggregator, with two mechanisms, synthesizes the most probable result from the MotionAnalyzer outputs. The Auxiliary Tools provide complementary analysis from databases. The Generator reviews the user's request and synthesizes the answer by leveraging the outputs and contextual information produced throughout the pipeline. The Verifier ensures consistency and relevance of intermediate results, enhancing the reliability of the final output. Through coordinated agent efforts, ChatMotion provides a flexible, precise, and reliable approach to motion analysis, overcoming the limitations of traditional motion LLMs.

We validate ChatMotion across a wide range of general human motion understanding datasets (e.g., Movid (Chen et al., 2024a), BABEL-QA (Endo et al., 2023), MVbench (Li et al., 2024a), and MoRepcount (Li et al., 2024b)), demonstrating its effectiveness across both standard and complex tasks. Experimental results highlight the improvements in accuracy, adaptability, reaching new heights in the field of human motion analysis. In summary, the contributions of this work are as follows:

- **ChatMotion**, a multi-agent system with a Planner-Executor-Verifier architecture for comprehensive human motion analysis.
- A robust **MotionCore** for invoking functional tools to enable comprehensive and reliable motion understanding by fusing diverse perspectives from various MLLMs and support result verification.
- Empirical validation across multiple datasets demonstrates that ChatMotion achieves improved performance in human motion analysis compared to existing MLLMs.

2 Related works

2.1 Human Multimodal Representations

Multimodal representation learning is pivotal for human-centric analyses, especially in tasks requiring spatial-temporal reasoning to interpret complex behaviors (Lin et al., 2023b; Ning et al., 2023; Li et al., 2023). Recent advancements, such as Video-LLaVA, integrate visual information from images and videos into a unified linguistic feature space, enabling improved visual reasoning for behavioral analysis (Lin et al., 2023b). However, many models remain limited to isolated video frames and privacy concerns, constraining their effectiveness in the dynamic real world. (Ning et al., 2023; Heilbron et al., 2015; Maaz et al., 2023). To address these limitations, motion data has emerged as a privacy-preserving alternative, allowing action analysis without revealing identifiable visual details (Song et al., 2023b; Yang et al., 2023b). By combining visual and motion data, emerging multimodal frameworks offer privacy-aware solutions, leveraging the strengths of both modalities for enhanced adaptability across diverse applications.

2.2 Human Motion Understanding

Human motion analysis traditionally relies on skeletal data, represented as joint keypoint sequences, to capture movement dynamics while preserving user privacy (Shi et al., 2023; Plappert et al., 2018; Yang et al., 2023a). Early methods, such as 2s-AGCN (Shi et al., 2019), and recent transformer-based models like MotionCLIP (Chen et al., 2024b), have demonstrated success in tasks such as activity recognition, caption generation, and behavior analysis by translating motion data into language tokens. While effective in modeling structural movement patterns, these approaches often neglect environmental context, which is crucial for interpreting motions that may convey different meanings based on situational factors (Song et al., 2023a; Maaz et al., 2023). To address this, recent models integrate motion and visual data, enabling improved generalization in dynamic and diverse environments (Liu et al., 2024; He et al., 2023). Frameworks like LLaMo(Li et al., 2024b) have uni-

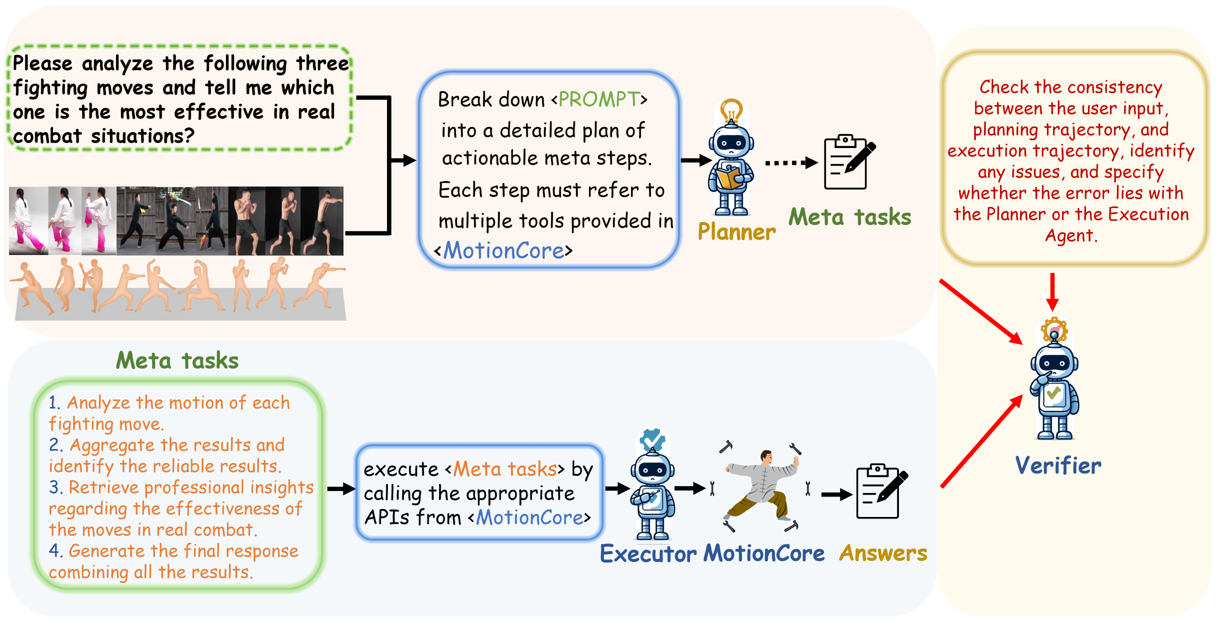

Please analyze the following three fighting moves and tell me which one is the most effective in real combat situations?
Break down <PROMPT> into a detailed plan of actionable meta steps. Each step must refer to multiple tools provided in <MotionCore>
Planner
Meta tasks
Check the consistency between the user input, planning trajectory, and execution trajectory, identify any issues, and specify whether the error lies with the Planner or the Execution Agent.
Verifier
Meta tasks
1. Analyze the motion of each fighting move.
2. Aggregate the results and identify the reliable results.
3. Retrieve professional insights regarding the effectiveness of the moves in real combat.
4. Generate the final response combining all the results.
execute <Meta tasks> by calling the appropriate APIs from <MotionCore>
Executor
MotionCore
Answers

Figure 2: The ChatMotion pipeline operates through a three-stage framework designed to optimize task resolution. The Planner interprets the user's query and breaks it into meta-tasks. Then, the Executor selects and applies appropriate MotionCore tools to execute these tasks. Finally, the Verifier ensures overall correctness, coherence, and completeness.

fied motion, video and text understanding models. However, this model is limited in its applicability, due to the size of its training data. Using an agent enables use of multiple models for flexible and comprehensive analysis.

3 ChatMotion

As shown in Fig. 2, ChatMotion is a multi-agent system that processes user queries involving motion and video data through the Planner, Executor, and Verifier agents, each powered by LLaMA-70B (Touvron et al., 2023) and equipped with specialized decision-making and tool-use strategies. The Planner decomposes the task into meta-tasks. Then the Executor executes them via MotionCore function calls. Finally the Verifier ensures accuracy by delivering context-aware, precise results for complex motion analysis.

3.1 Planner

The Planner serves as the decision-maker, interpreting user intent and subdividing complex tasks into structured meta-tasks. First the input query is analyzed to identify core objectives and dependencies within the task, and then breaks the task down into smaller, manageable meta-tasks. It operates as the initial step in the multi-agent framework, ensuring that user requirements are transformed into a structured workflow that aligns with evolving goals.

Specifically, let us denote a user query by R. As the simplified version is illustrated in Fig. 2, the Planner will receive an instruction containing user query and available tools functionality in MotionCore which is a function toolbox tailored for human motion analysis (see Sec. 3.4). Then, the Planner will follow the instructions and identify a set of core objectives $\mathcal{O} = \{O_1, O_2, \ldots, O_m\}$ simply based on R. These objectives are then decomposed into meta-tasks $\mathcal{M}$ guided by the specific functionalities available in the MotionCore tools. This decomposition allows the system to handle a wide range of user inputs, from simple queries to multi-step, dynamic tasks. The prompt for the Planner and example outputs are provided in the appendix (see Sec. A.1).

3.2 Executor

The Executor is responsible for transforming the Planners meta-tasks into operations that can answered by a suite of function tools. Once provided with the set of meta-tasks $\mathcal{M}$, the Executor processes each task in turn, as illustrated in Fig. 2, selecting the most appropriate function tool from MotionCore (see Sec. 3.4) based on the alignment between each tool's functionality and the objective of the meta-task.

Formally, for a given meta-task $M_i \in \mathcal{M}$, the Executor considers the set of available function tools $\Phi = \{\phi_1, \phi_2, \ldots, \phi_s\}$ within MotionCore, and selects the function ϕ_i that best matches the

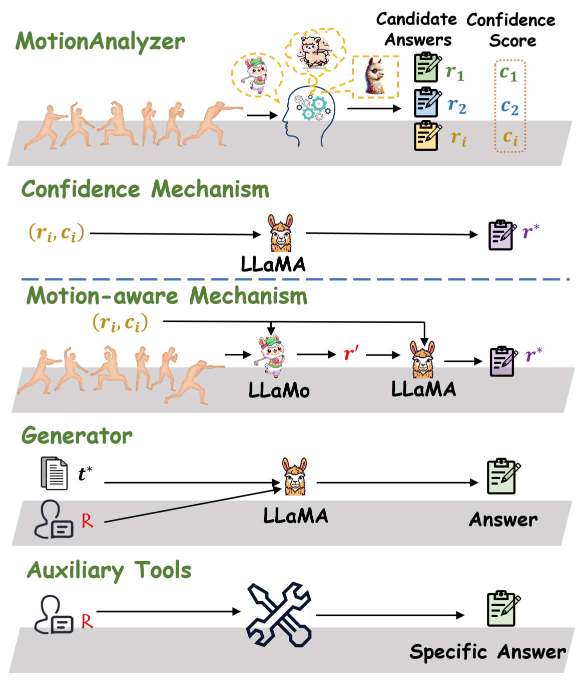


Figure 3: Components of MotionCore: the MotionCore integrates the MotionAnalyzer and Selection modules to concurrently process and aggregate multiple human motion analyses in two specific ways. The Generation Module synthesizes and contextualizes the results to align with user queries. Additionally, an auxiliary toolbox enables dynamic expansion with supplementary tools to address evolving user requirements.

requirements of M_i:

$$\Phi(M_i) \rightarrow \phi_i,$$

where ϕ_i is the specific tool to fulfill meta-task M_i.

If a meta-task cannot be completed (e.g., due to unavailable functionality), the Executor returns detailed error information to the Planner, which then revises its objectives and resubmits updated meta-tasks. This process may iterate through multiple rounds until the overall objective is satisfied. The prompt for the Executor and example outputs are provided in the appendix (see Sec. A.3).

3.3 Verifier

The Verifier acts as a supervisory agent, ensuring the accuracy and reliability of the multi-agent workflow. It has two main roles: first, it checks that the Planner's meta-tasks are logically structured and aligned with the user's prompt; second, it verifies that the meta-tasks can be executed using available tools and that the results meet expectations. If any meta-task cannot be executed or produces incorrect results, or if the Executor calls an inappropriate function, the Verifier prompts the Planner to revise the task list or the Executor to select a different tool. This feedback loop ensures that tasks are executed correctly using the right tools.

3.4 MotionCore

As shown in Fig 3, MotionCore is a comprehensive toolkit that enables efficient human motion understanding by integrating various modules and auxiliary functions. It also includes auxiliary tools for tasks like motion visualization and video retrieval, meeting users' diverse requirements. MotionCore is driven by the Executor Agent, which selects the appropriate tools from the toolkit to complete tasks based on a given meta-task list.

3.4.1 MotionAnalyzer

The MotionAnalyzer in MotionCore enhances motion understanding and mitigates biases through a dynamic, multi-model approach. It integrates human motion models, such as MotionLLM (Chen et al., 2024a), MotionGPT (Jiang et al., 2023), and LLaMo(Li et al., 2024b), alongside video captioning models such as VideoChat2 (Li et al., 2024a), GPT-4v (OpenAI, 2023b), and video-LLaVA (Lin et al., 2023a) to handle human motion input. Let the set of motion understanding models be denoted as $\{F_1, F_2, \ldots, F_N\}$, where each model F_i processes the multimodal input data D (e.g., video frames, motion capture data) to produce text analysis r_i, i.e., $(r_i) = F_i(D), \quad i = 1, 2, \ldots, N$. Each model is assigned a predefined confidence score c_i, *based on the previous evaluation performance*, independent of the model's predictions. These scores are provided in the appendix (see Sec. C.2). Furthermore, the confidence scores are allocated based on the input modalities, which can be motion capture, video, or motion-video. The outputs and their corresponding confidence scores are represented as $\{(r_1, c_1), (r_2, c_2), \ldots, (r_N, c_N)\}$, where c_i denotes the predefined confidence score for the output r_i of model F_i in its respective task. This integration of predefined confidence scores ensures a robust and flexible understanding of motion, leveraging the strengths of each model across diverse modalities and tasks.

3.4.2 Aggregator

The Aggregator in MotionCore identifies the most reliable result from a set of $\{(r_i, c_i)\}$ pairs, employing two strategies: the Confidence Mechanism and the Motion-Aware Mechanism, which improves the motion understanding by selecting the most accurate outcome from diverse perspectives.

Confidence Mechanism Rooted in game theory, this method considers the set

$$\{(r_i, c_i) \mid i = 1, 2, \ldots, N\},$$

where r_i is a model's output and c_i is its associated confidence score. The mechanism assigns higher weight to more confident outputs, with a "majority wins" principle when models converge on similar results. Rather than using a fixed function, the analysis-confidence pairs $\{(r_i, c_i)\}$ are passed to LLaMA, which adaptively integrates the outputs by balancing consensus with individual model expertise. The mechanism asks LLaMA to emphasize model outputs to be fused that have a shared conclusion while considering outlier predictions with high confidence before rewriting these to a single new analysis r^*.

Though foundational, this approach is simple, relying primarily on confidence scores and model consensus. The next step incorporates a Motion-Aware mechanism to refine the process.

Motion-Aware Mechanism With LLaMo's (Li et al., 2024b) specialized motion-understanding capabilities, this mechanism evaluates $\{(r_i, c_i)\}$ pairs alongside the original motion or video data $\mathcal{M}$, generating an initial estimate:

$$r' = \text{LLaMo}(r_1, \ldots, r_N;\ c_1, \ldots, c_N;\ \mathcal{M}).$$

To produce this intermediary result LLaMo observes the motion or video sequence and is asked to provide a refined analysis based on this and the prior candidate answers whose context should be weighted by their confidence score. LLaMA (Touvron et al., 2023) then re-examines the preliminary result r' and the original pairs $\{(r_i, c_i)\}$ to mitigate model bias and refine the outcome, while producing the final analysis in a similar manner as the confidence mechanism. This dual-layer evaluation leverages LLaMo's domain-specific motion expertise and LLaMA's context-aware reasoning, improving both reliability and precision.

The Aggregator is a powerful tool within MotionCore that fosters a more comprehensive understanding of human motion, by enabling ChatMotion to identify the most accurate analyses from diverse model outputs.

3.4.3 Generator

In MotionCore, the Generator synthesizes contextual information from previously invoked tool outputs, such as those from the Aggregator and retrieval modules, together with the user's original request to produce a final answer, as shown in Fig. 3. Here, t^* denotes the intermediate results from prior module executions. The Generator integrates t^* and user query R:

$$\text{Answer} = \Gamma(t^*, R),$$

where $\Gamma(\cdot)$ denotes LLaMA. The final output can take various forms, such as textual analysis, motion feedback, and so on, depending on the user's request. This process ensures that the answer is both comprehensive and tailored to the user's needs.

3.4.4 Auxiliary Tools

The Auxiliary Tools in MotionCore, which can be accessed by the Executor, extend ChatMotion's capabilities by orchestrating external, domain-specific functionalities that go beyond the scope of the multimodal model alone. For instance, the system can retrieve professional analysis by querying specialized knowledge bases, which provide context-specific insights based on user inputs. Additionally, it enables motion retrieval by identifying relevant motion data based on the user's request, leveraging a stored database of labeled motion data and utilizing vector-based search to match the query to the most relevant motion. As a result, it equips ChatMotion with two retrieval tools to retrieve text or motion sequences which provides diverse motion analysis capabilities that simple MLLMs do not possess. ChatMotion provides a unified, modular interface for auxiliary functions, enabling seamless integration of new capabilities without burdening the core model.

4 Experimental Setup

Datasets We evaluate ChatMotion on general human motion understanding benchmarks including Movid-bench (Chen et al., 2024a), BABEL-QA (Endo et al., 2023) and MVbench (Li et al., 2024a), as well as Mo-Repcount (Li et al., 2024b) for fine-grained motion capture capabilities. MoVid-Bench specifically assesses the model's ability to understand human behavior in both motion and video contexts. It consists of 1,350 data pairs, with 700 motion and 650 video samples, covering diverse daily scenarios in real-world. In addition, ChatMotion is tested on BABEL-QA and MVbench to evaluate motion-based and video-based question answering respectively.

MoVid-Bench-Motion	**Body.**		**Seq.**		**Dir.**		**Rea.**		**Hall.**		**All**	
	Acc.	**Score**	**Acc.**	**Score**	**Acc.**	**Score**	**Acc.**	**Score**	**Acc.**	**Score**	**Acc.**	**Score**
GT	**100.00**	**5.00**	**100.00**	**5.00**	**100.00**	**5.00**	**100.00**	**5.00**	**100.00**	**5.00**	**100.00**	**5.00**
GPT-3.5 (OpenAI, 2023a)	24.51	2.04	30.41	2.25	27.14	2.19	39.19	2.64	58.33	3.22	31.33	2.31
MotionGPT (Jiang et al., 2023)	31.22	3.98	42.69	**3.16**	44.29	3.50	35.81	3.06	16.66	2.25	36.86	3.11
MotionLLM (Chen et al., 2024a)	50.49	3.55	36.84	3.14	58.57	3.76	52.70	3.58	55.56	3.39	49.50	3.49
LLaMo (Li et al., 2024b)	59.30	4.01	44.01	3.12	60.91	3.99	58.21	3.64	61.17	3.53	55.32	3.67
ChatMotion(CB)	**60.89**	4.03	46.21	**3.30**	62.11	4.03	59.53	3.77	68.95	3.78	56.90	3.72
ChatMotion	60.43	**4.08**	**46.56**	3.28	**64.21**	**4.11**	**60.58**	**3.87**	**70.39**	**3.82**	**58.79**	**3.80**
MoVid-Bench-Video	**Body.**		**Seq.**		**Dir.**		**Rea.**		**Hull.**		**All**	
	Acc.	**Score**	**Acc.**	**Score**	**Acc.**	**Score**	**Acc.**	**Score**	**Acc.**	**Score**	**Acc.**	**Score**
GT	**100.00**	**5.00**	**100.00**	**5.00**	**100.00**	**5.00**	**100.00**	**5.00**	**100.00**	**5.00**	**100.00**	**5.00**
GPT-3.5 (OpenAI, 2023a)	2.40	1.23	1.39	1.00	4.65	1.09	5.41	1.65	0.00	0.94	3.03	1.26
Video-LLAVA (Lin et al., 2023a)	33.53	2.76	25.46	2.72	41.86	2.84	52.97	3.28	58.83	1.89	42.53	2.70
MotionLLM (Chen et al., 2024a)	34.13	2.93	32.87	2.92	44.18	3.14	63.20	3.55	70.59	2.30	49.00	2.97
LLaMo (Li et al., 2024b)	33.83	2.85	36.01	3.11	45.50	3.32	67.59	3.73	72.81	2.25	52.33	3.10
ChatMotion(CB)	**38.31**	**3.40**	36.80	3.17	47.22	3.59	70.89	3.85	73.22	**2.35**	53.51	3.19
ChatMotion	38.06	3.34	**37.39**	**3.18**	**47.92**	**3.65**	**72.16**	**3.99**	**74.01**	2.30	**54.96**	**3.25**

Table 1: Comparison between ChatMotion and existing Motion LLMs on the MoVid-Bench. The top part of the table presents motion-related results, and the bottom part presents video-related results. Higher accuracy and score values indicate better performance. "GT"=Ground truth.

Model	**Pred. type**	**Overall ↑**	**Action ↑**	**Direction ↑**	**Body Part ↑**	**Before ↑**	**After ↑**	**Other ↑**
MotionCLIP-M (Tevet et al., 2022)	cls.	0.430	0.485	0.361	0.272	0.372	0.321	0.404
MotionCLIP-R (Tevet et al., 2022)	cls.	0.420	0.489	0.310	0.250	0.398	0.314	0.387
MotionLLM (Chen et al., 2024a)	gen.	0.436	0.517	0.354	0.154	0.427	0.368	0.529
LLaMo (Li et al., 2024b)	gen.	0.458	0.525	0.398	0.224	0.443	0.392	0.518
ChatMotion(CB)	gen.	0.467	0.534	0.410	**0.272**	0.445	0.396	0.536
ChatMotion	gen.	**0.473**	**0.537**	**0.412**	0.265	**0.451**	**0.406**	**0.537**

Table 2: Comparison on the BABEL-QA dataset. Higher scores indicate better performance. The results for ChatMotion's two methods are also included.

Tasks and Metrics ChatMotion is evaluated on tasks including action recognition, motion reasoning, and question answering. For MoVid-Bench, we follow established LLM evaluation metrics, assessing body-part recognition, sequential analysis, directionality, reasoning, and hallucination control in both motion and video contexts. BABEL-QA uses similar metrics with a focus on motion-related question answering, while Mo-Repcount employs specialized metrics like OBO, MAE, OBZ, and RMSE for fine-grained motion tracking accuracy. In the MVBench video understanding evaluation, we respond to multiple-choice questions by selecting the most suitable option as outlined in.

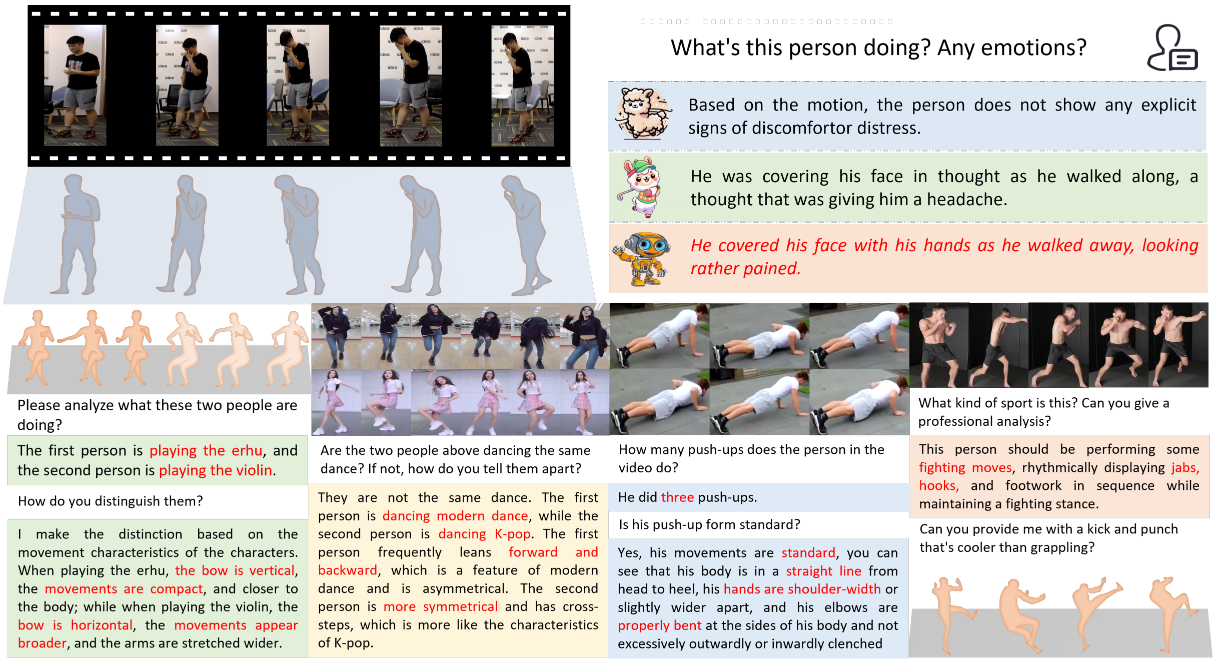
Baselines For our baselines, we select SoTA Motion LLMs for human-centric motion understanding, e.g., LLaMo (Li et al., 2024b), MotionLLM (Chen et al., 2024a) and MotionGPT (Jiang et al., 2023). These models are widely recognized for their ability to process and understand human motion in both video and action contexts. For ChatMotion, **ChatMotion(CB)** and **ChatMotion** denote the versions using Confidence-Based and Motion-Aware aggregation, respectively. Through extensive comparison, our results highlight ChatMotion's exceptional ability to handle complex human motion understanding tasks, outperforming the baselines across a range of evaluation metrics.

5 Results

5.1 Quantitative Analysis

Evaluation on Motion Understanding in MoVid-Bench. Table 1 compares the performance of motion-based LLMs on MoVid-Bench-Motion. Both ChatMotion(CB) and ChatMotion outperform existing baselines across all metrics. ChatMotion achieves an accuracy of 58.79% and a score of 3.80, surpassing LLaMo by 3.47% in accuracy and 0.13 in score. It also demonstrates strong hallucination control, achieving 70.39% accuracy compared to LLaMo's 61.17%, underscoring the effectiveness of ChatMotion's multi-model integration via its robust selection strategy.

Previous models, such as MotionLLM and MotionGPT, lose fine-grained motion details due to motion discretization, leading to lower performance. Although LLaMo improves motion encoding, its single LLM-based structure introduces biases that limit its motion understanding capabilities. In contrast, ChatMotion leverages multi-agent collaboration and multi-model aggregation to enhance motion understanding. This approach reduces bi-


What's this person doing? Any emotions?

Based on the motion, the person does not show any explicit signs of discomfortor distress.

He was covering his face in thought as he walked along, a thought that was giving him a headache.

He covered his face with his hands as he walked away, looking rather pained.

Please analyze what these two people are doing?

The first person is playing the erhu, and the second person is playing the violin.

How do you distinguish them?

I make the distinction based on the movement characteristics of the characters. When playing the erhu, the bow is vertical, the movements are compact, and closer to the body; while when playing the violin, the bow is horizontal, the movements appear broader, and the arms are stretched wider.

Are the two people above dancing the same dance? If not, how do you tell them apart?

They are not the same dance. The first person is dancing modern dance, while the second person is dancing K-pop. The first person frequently leans forward and backward, which is a feature of modern dance and is asymmetrical. The second person is more symmetrical and has cross-steps, which is more like the characteristics of K-pop.

How many push-ups does the person in the video do?

He did three push-ups.

Is his push-up form standard?

Yes, his movements are standard, you can see that his body is in a straight line from head to heel, his hands are shoulder-width or slightly wider apart, and his elbows are properly bent at the sides of his body and not excessively outwardly or inwardly clenched

What kind of sport is this? Can you give a professional analysis?

This person should be performing some fighting moves, rhythmically displaying jabs, hooks, and footwork in sequence while maintaining a fighting stance.

Can you provide me with a kick and punch that's cooler than grappling?

Figure 4: Examples of ChatMotion's responses in various human activities and sports, demonstrating its reasoning skills and specialized knowledge in active, movement-heavy scenarios.

Model	LLM	Frames	AL	AP	AS	EN	FA	FP	UA	Avg.
Otter-V	Llama-7B	16	23.5	23.0	23.0	23.5	27.0	22.0	29.5	24.5
mPLUG-Owl-V	Llama-7B	16	23.0	28.0	22.0	26.0	29.0	24.0	29.0	25.8
VideoChatGPT	Vicuna-7B	100	20.0	26.0	23.5	29.5	22.5	22.5	29.0	25.2
VideoLLaMA	Vicuna-7B	16	22.5	25.5	27.5	30.0	29.0	32.5	39.0	29.4
VideoChat	Vicuna-7B	16	27.0	26.5	33.5	23.5	33.5	26.5	40.5	30.1
Video-LLAVA	Vicuna-7B	8	22.5	25.5	29.5	29.0	24.5	28.5	24.5	26.3
VideoChat2	Vicuna-7B	16	23.0	**66.0**	47.5	**35.0**	**49.5**	49.0	60.0	47.1
MotionLLM	Vicuna-7B	8	33.0	29.5	32.5	29.0	31.5	28.5	37.5	31.6
GPT-4v	GPT-4	16	40.5	63.5	55.5	31.0	46.5	47.5	73.5	51.1
ChatMotion(CB)	**Agent**	\	42.0	65.5	56.0	33.0	48.0	50.5	72.0	52.4
ChatMotion	**Agent**	\	**43.0**	65.5	**58.0**	34.0	49.0	**51.0**	**74.0**	**53.2**

Table 3: Comparison between ChatMotion and various models on MVBench.

Model	OBO	MAE	OBZ	RMSE
EScounts	0.397	0.291	0.198	5.58
PoseRAC	0.382	0.312	0.204	5.95
TransRAC	0.276	0.444	0.105	8.56
RepNet	0.009	\	\	\
MotionLLM	0.011	\	\	\
LLaMo	0.389	0.324	0.222	6.15
ChatMotion(CB)	**0.412**	0.279	0.229	5.33
ChatMotion	0.410	**0.271**	**0.240**	**5.21**

Table 4: Motion and video details capture evaluation on Mo-RepCount.

ases inherent in single LLM-based motion models and improves performance in motion sequence analysis. By integrating multiple agents, ChatMotion achieves greater robustness, demonstrating its superior capabilities to capture diverse motion dynamics and delivers more accurate, reliable results in complex motion understanding tasks.

Evaluation on Video Understanding in MoVid-Bench. ChatMotion(CB) demonstrates improvements across multiple metrics on MoVid-Bench-Video as shown in Table 1, achieving an overall accuracy of 53.51% and a score of 3.19, surpassing baseline models in all evaluated tasks. This performance gain is due to its effective aggregation of diverse video analysis perspectives, combined with confidence scores to ensure more reliable and stable reasoning. Furthermore, ChatMotion, with its Motion-Aware mechanism, further refines the analysis by better handling motion-related tasks, surpassing ChatMotion(CB) with an accuracy improvement of 1.45% and a score increase of 0.06. This enhancement allows it to more effectively aggregate and analyze motion data, pushing performance beyond that of standard models. These innovations in model design, coupled with the synergistic effects of specialized modules, allow ChatMotion(CB) and ChatMotion to set new benchmarks in multimodal human motion analysis, outperforming existing LLM-based motion models across multiple tasks and metrics.

Evaluation on BABEL-QA. We evaluated ChatMotion on the BABEL-QA dataset to assess its performance in responding to complex motion-based queries. As shown in Table 2, both ChatMotion(CB) and ChatMotion outperform other LLM-based motion models across several metrics. ChatMotion(CB) achieves an overall score of 0.467, while ChatMotion further improves this to 0.473, demonstrating its enhanced capability. This improvement is due to ChatMotion's Motion-Aware mechanism, which takes both motion inputs and candidate results into account. By leveraging LLaMo's advanced multimodal capabilities, ChatMotion esures more robust and stable results. Despite some limitations on specific metrics, ChatMotion compensates for these and delivers superior overall results. These advancements position ChatMotion as a new benchmark in motion-based question answering, highlighting the effectiveness of multimodal aggregation and Motion-Aware mechanisms for improved accuracy and reliability.

Evaluation on MVBench. We evaluated ChatMotion on the MVBench dataset to assess its performance in video question answering across seven motion understanding sub-tasks. As shown in Table 3, ChatMotion(CB) outperforms MotionLLM (Chen et al., 2024a), the LLM-based motion understanding model, achieves an average score of 52.4, while ChatMotion increases this to 53.2. These results highlight the efficacy of ChatMotion's multi-agent framework, which reduces biases inherent to LLM-based motion models by incorporating dynamic function calls. Performance gains are particularly evident in most metrics, demonstrating the advantages of multi-agent integration for robust motion understanding. While slight performance gaps persist in specific tasks compared with expert models (e.g., EN of VideChat2), the overall improvement over the LLM-based motion model, MotionLLM, remains statistically better.

Evaluation on Mo-Repcount To evaluate ChatMotion's performance on fine-grained motion tasks, we benchmarked it on Mo-Repcount against SoTA Motion LLMs. The results in Table 4 show that ChatMotion outperforms LLaMo by 4%-8% across all metrics, demonstrating ChatMotion's advanced capability to aggregate the strengths of specialized models and achieve superior performance in fine-grained motion tasks.

5.2 Ablation Study

Our ablation study examines the contribution of each module in ChatMotion, including the planner, verifier, and aggregation components. The results show that each additional module leads to consistent improvements. Detailed results and further analysis are provided in Appendix C.

5.3 Qualitative Analysis

Qualitative results, as illustrated in Fig. 4, show that ChatMotion produces accurate and clear interpretations by integrating multiple analytical tools and cross-verifying outputs. For emotion recognition with both video and motion inputs, other models either fail or give ambiguous results, whereas ChatMotion delivers the correct answer. More importantly, only ChatMotion supports advanced tasks such as comprehensive cross-modal retrieval-based analysis and detailed motion-video comparisons, which are not available in previous models. This marks a significant advance in the scope and complexity of tasks that can be addressed in multimodal motion understanding.

6 Conclusion

In this paper, we introduced ChatMotion, a sophisticated multi-agent framework that integrates large language models with specialized motion-analysis modules to address the limitations inherent in single-model systems. By dynamically breaking down complex tasks, aggregating diverse model outputs, and carefully selecting the most reliable results, ChatMotion effectively mitigates biases in motion understanding and delivers robust, context-aware analyses. Through experiments conducted on human motion benchmarks such as MoVid-Bench and BABEL-QA, we demonstrated great improvements in both accuracy and adaptability across various motion tasks. We hope this work inspires ongoing research in the field of multimodal agent architectures for human motion understanding.

7 Limitations

Computational Overhead Several models are part of MotionCore and the due to the sequential nature of the agentic setup the computational overhead scales linearly with the number of models as compared with a single model system. On the Movid-Motion dataset ChatMotion can process motion sequences at 95 FPS whereas LLaMo can process the same sequences at 200 FPS on an H100. This is primarily due to two factors; the Executor and Planner. Increasing the number of models in MotionCore will potentially increase the number of tasks created by the planner. Similarly for the Executor which selects APIs from MotionCore, the number of model calls can increase simply by including more models as MotionAnalyzer invokes several motion understanding models when called.

Dataset Limitations The current setup of ChatMotion is reliant on domain specific models that can understand human motion. As these models are trained on human motion datasets which tends to be limited to certain topics, and relatively small in scale as compared to the image or language domain, so will the capabilities of the models. This limits the generalization to novel motion scenes or even fine grained understanding of known domains such as martial arts as found in MoVid, due to being limited in scope. As such further validation is needed to determine how adaptable the framework is to new unseen tasks, and how much new data is required in order to be proficient in a new domain.

Agentic limitations Finally, the agentic nature of ChatMotion comes with limitations observed in other agent systems. Planning itself is hard as it requires reasoning over new tasks where no data trajectories may exist to train or validate the efficacy of the Planner. Our Planner is dependent on the quality of base MLLM models, as well the prompted used by the model. Some of these issues can partially be mitigated with our Verifier, despite having similar limitations as the Planning model. As such the Verifier might give incorrect statements about the execution trajectory for out of domain motion sequences. The Executor model can be a source of error if selects a wrong or less efficient model from a toolbox. One should conduct some analysis to determine the applicability for new domains. Lastly for our weighting of the Candidate answers in the MotionAnalyzer will likely have to be adapted dynamically to new tasks and datasets.

References

Ling-Hao Chen, Shunlin Lu, Ailing Zeng, Hao Zhang, Benyou Wang, Ruimao Zhang, and Lei Zhang. 2024a. Motionllm: Understanding human behaviors from human motions and videos. *arXiv preprint arXiv:2405.20340*.

Ling-Hao Chen, Jiawei Zhang, Wen Liu, Gang Yu, and Tao Chen. 2024b. Motiongpt: Human motion as a foreign language. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*.

Mark Endo, Joy Hsu, Jiaman Li, and Jiajun Wu. 2023. Motion question answering via modular motion programs. In *International Conference on Machine Learning*, pages 9312–9328. PMLR.

Fotos Frangoudes, Maria Matsangidou, Eirini C Schiza, Kleanthis Neokleous, and Constantinos S Pattichis. 2022. Assessing human motion during exercise using machine learning: A literature review. *IEEE Access*, 10:86874–86903.

Xiaolong He, Yi Zhang, Chen Chen, and Kyoung Mu Lee. 2023. Activitynet++: A large-scale benchmark for video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. 2015. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Fangzhou Hong, Liang Pan, Zhongang Cai, and Ziwei Liu. 2022. Versatile multi-modal pre-training for human-centric perception. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16156–16166.

Biao Jiang, Xin Chen, Wen Liu, Jingyi Yu, Gang Yu, and Tao Chen. 2023. Motiongpt: Human motion as a foreign language. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Hasti Khiabani. 2021. *sEMG-Based Lower Limb Intention Detection using Artificial Intelligence and its Impact on Assistive Human-Robot Interaction*. Ph.D. thesis, Carleton University.

Xiang Lan, Zhongwang Cao, and Le Yu. 2022. Analyzing the mental states of the sports student based on augmentative communication with human–computer interaction. *International Journal of Speech Technology*, pages 1–11.

KunChang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. 2023. Videochat: Chat-centric video understanding. *arXiv preprint arXiv:2305.06355*.

Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. 2024a. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22195–22206.

Lei Li, Sen Jia, Wang Jianhao, Zhongyu Jiang, Feng Zhou, Ju Dai, Tianfang Zhang, Wu Zongkai, and Jenq-Neng Hwang. 2024b. Human motion instruction tuning. *arXiv preprint arXiv:2411.16805*.

Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. 2023a. Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*.

Jie Lin, Wei Zhang, and Zhe Chen. 2023b. Videollm: Language models for video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2405–2415.

Xianghong Liu, Haoxuan Wang, Zhiwei Zhang, Huan Wu, Baoquan Chen, and Xiaoguang Han. 2024. Category-agnostic pose estimation for point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. 2015. Smpl: a skinned multi-person linear model. *ACM Trans. Graph.*, 34(6).

Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. 2023. Video-chatgpt: Towards detailed video understanding via large vision and language models. *arXiv preprint arXiv:2306.05424*.

Zhaozong Meng, Mingxing Zhang, Changxin Guo, Qirui Fan, Hao Zhang, Nan Gao, and Zonghua Zhang. 2020. Recent progress in sensing and computing techniques for human activity recognition and motion analysis. *Electronics*, 9(9):1357.

Xu Ning, Hongwei Li, and Yu Zhao. 2023. Videobench: A benchmark for large-scale video understanding. *arXiv preprint arXiv:2304.12345*.

OpenAI. 2023a. Gpt-3.5: Generative pre-trained transformer 3.5. `https://platform.openai.com/docs/models/gpt-3-5`.

GPT OpenAI. 2023b. 4v (ision) system card. *preprint*.

Matthias Plappert, Christian Mandery, and Tamim Asfour. 2016. The kit motion-language dataset. *Big data*, 4(4):236–252.

Matthias Plappert, Christian Mandery, and Tamim Asfour. 2018. Learning a bidirectional mapping between human whole-body motion and natural language using deep recurrent neural networks. *Robotics and Autonomous Systems (RAS)*, 109:13–26.

Haoxuan Qu, Yujun Cai, and Jun Liu. 2024. Llms are good action recognizers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18395–18406.

Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. 2019. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12026–12035.

Yaya Shi, Haiyang Xu, Chunfeng Yuan, Bing Li, Weiming Hu, and Zheng-Jun Zha. 2023. Learning video-text aligned representations for video captioning. *IEEE Transactions on Multimedia (TMM)*.

Jan David Smeddinck. 2020. Human-computer interaction with adaptable & adaptive motion-based games for health. *arXiv preprint arXiv:2012.03309*.

Enxin Song, Wenhao Chai, and Yucheng Zhang. 2023a. Fine-grained spatial-temporal motion understanding in complex video environments. *arXiv preprint arXiv:2310.08639*.

Enxin Song, Guanhong Zhang, and Haoyang Zhou. 2023b. Adaptive multimodal learning for behavior analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Guy Tevet, Brian Gordon, Amir Hertz, Amit H Bermano, and Daniel Cohen-Or. 2022. Motionclip: Exposing human motion generation to clip space. In *European Conference on Computer Vision*, pages 358–374. Springer.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Lai Wei, Stephen Jia Wang, et al. 2024. Motion tracking of daily living and physical activities in health care: Systematic review from designers' perspective. *JMIR mHealth and uHealth*, 12(1):e46282.

Hongwei Xiao, Yongqi Sun, Zhenghao Duan, Yunxiang Huo, Jingze Liu, Mingyu Luo, Yanhui Li, and Yingchao Zhang. 2024. A study of model iterations of fitts' law and its application to human–computer interactions. *Applied Sciences*, 14(16):7386.

Wei Xu, Marvin J Dainoff, Liezhong Ge, and Zaifeng Gao. 2021. From human-computer interaction to human-ai interaction: new challenges and opportunities for enabling human-centered ai. *arXiv preprint arXiv:2105.05424*, 5.

Hao Yang, Qianghua Zhao, and Lei Li. 2024. Chain-of-thought in large language models: Decoding, projection, and activation. *arXiv preprint arXiv:2412.03944*.

Yunhua Yang, Liang Zhang, and Hui Li. 2023a. Understanding human behaviors from skeletal data: A review of datasets and methods. *arXiv preprint arXiv:2310.12998*.

Yunhua Yang, Ziwang Zhao, and Yiming Xie. 2023b. Recognizing human behaviors with skeletal data in llm-based frameworks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Weihua Zhang. 2024. Research on physical human-computer interaction virtual reality fitness method combined with unity3d technology. *Applied Mathematics and Nonlinear Sciences*.

Yan Zhang, Michael J Black, and Siyu Tang. 2021. We are more than our joints: Predicting how 3d bodies move. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kaiyuan Zheng, Qinghua Zhao, and Lei Li. 2025. Reassessing the role of chain-of-thought in sentiment analysis: Insights and limitations. *arXiv preprint arXiv:2501.08641*.

A Prompt of ChatMotion

A.1 Planner Template

To enable structured task decomposition, we design a multi-step prompt template for the Planner. This template guides the agent through four logical stages, helping it understand the user's intent, select relevant tools, and output a series of actionable meta-tasks. The process is summarized as follows:

- **Step 1: Objective Extraction.** The Planner first identifies the core objectives in the user query, which serves as the foundation for further reasoning.

- **Step 2: Tool Functionality Review.** It then examines the capabilities of the available tools in MotionCore, such as the MotionAnalyzer, Aggregator, and Generator, to inform how the objectives can be effectively tackled.

- **Step 3: Meta-Task Decomposition.** Based on the objectives and tool functions, the Planner decomposes the overall task into logically ordered sub-tasks.

- **Step 4: Output Meta-Task List.** Finally, it produces a structured list of meta-tasks that can be executed by the Executor using the appropriate tools.

This prompt ensures that the Planner's decisions are both goal-driven and tool-aware. The full instruction template is visualized in Figure 5.

A.2 Planner Example and Intermediate Reasoning

To illustrate the Planner's behavior in practice, we present a visualized example in Figure 6, which demonstrates the intermediate reasoning steps taken by the agent. This process adheres to the four-stage framework described in Section A.1, including the extraction of objectives, analysis of tool functionalities, decomposition into meta-tasks, and the generation of a structured task list.

In this example, the user query involves three goals: identifying the combat move, analyzing its key technical components, and generating instructional content for executing a flying kick. The Planner first parses these high-level intents and then determines the most appropriate tools from the MotionCore suite to address them. Specifically, it selects the MotionAnalyzer to analyze the movements, the KnowledgeRetriever to explain the techniques involved, and the Generator to provide the answer tailored for the query.

Guided by this mapping, the Planner proceeds to decompose the problem into five sequential meta-tasks, each aligned with a specific sub-goal and tool. These tasks are clearly enumerated in the final output, forming a coherent plan that facilitates step-by-step execution by downstream modules. This example demonstrates the Planner's capacity to transform complex multimodal queries into interpretable and executable workflows.

A.3 Executor Template

To enable structured execution of the meta-task list generated by the Planner, we design a prompt template for the Executor that supports tool-aware reasoning in four stages, as illustrated in Figure 7.

- **Step 1: Task Understanding.** The Executor begins by parsing the meta-task list produced by the Planner. Each task corresponds to a well-defined objective that must be resolved using a specific tool from the MotionCore toolbox.

- **Step 2: Tool Selection.** For each meta-task, the Executor determines the most suitable tool by considering the tools functional descriptions and tool capabilities. This ensures that each operation is delegated to the most relevant module.

- **Step 3: Task Execution.** The selected tool is invoked to complete the meta-task. If the tool execution fails (e.g., due to incompatibility or missing input), the Executor logs the failure and returns feedback to the Planner and Verifier.

- **Step 4: Final Output.** Upon completion of all meta-tasks, the Executor synthesizes a final response or generates an error report highlighting any failed operations, ensuring transparency in execution.

This template ensures that the Executor performs each meta-task in a modular, interpretable, and robust manner. The full prompt design is shown in Figure 7.

A.4 Executor Example and Intermediate Execution

To illustrate how the Executor operates on the Planner's output, we present a concrete example in Figure 8, which visualizes the execution trace for the

Executor	Planner	Verifier	CB	MA	Body		Seq.		Dir.		Rea.		Hall.		All	
					Acc.	**Score**	**Acc.**	**Score**	**Acc.**	**Score**	**Acc.**	**Score**	**Acc.**	**Score**	**Acc.**	**Score**
✓	✗	✗	✗	✗	45.23	3.18	35.67	3.05	55.42	3.37	49.89	3.29	53.34	3.12	43.56	3.22
✓	✓	✗	✗	✗	46.76	3.22	38.45	3.08	56.21	3.49	51.34	3.41	54.78	3.19	45.89	3.43
✓	✓	✓	✗	✗	48.12	3.31	40.78	3.21	58.56	3.58	54.67	3.52	65.43	3.68	48.34	3.41
✓	✓	✓	✓	✗	**60.89**	4.03	46.21	**3.30**	62.11	4.03	59.53	3.77	68.95	3.78	56.90	3.72
✓	✓	✓	✓	✓	60.43	**4.08**	**46.56**	3.28	**64.21**	**4.11**	**60.58**	**3.87**	**70.39**	**3.82**	**58.79**	**3.80**

Table 5: Ablation of ChatMotion on the MoVid-Bench for the Planner, Executor, Verifier, and Aggregator modules.

Movid-Bench-Motion	Body		Seq.		Dir.		Rea.		Hall.		All	
	Acc.	Score	Acc.	Score	Acc.	Score	Acc.	Score	Acc.	Score	Acc.	Score
Equal (no CB)	45.23	3.18	35.67	3.05	55.42	3.37	49.89	3.29	53.34	3.12	43.56	3.22
CB (Base)	59.80	4.01	45.35	3.27	61.23	4.01	58.87	3.73	67.56	3.75	56.43	3.70
CB (LLaMo=11)	60.30	4.02	45.82	**3.28**	61.62	4.02	59.14	3.75	68.25	3.76	56.65	3.71
CB (LLaMo=12)	**60.89**	**4.03**	**46.21**	3.27	**62.11**	**4.02**	**59.53**	**3.77**	**68.95**	**3.78**	**56.90**	**3.72**
CB (LLaMo=13)	60.20	4.00	45.70	3.26	61.45	4.00	59.00	3.72	68.03	3.74	56.50	3.69
CB (LLaMo=14)	59.60	3.98	45.21	3.24	60.80	3.98	58.66	3.69	67.45	3.71	56.14	3.67

Table 6: Ablation study on confidence score assignment in ChatMotion. "Equal" denotes uniform weighting without confidence modeling. CB (LLaMo=X) indicates increasing LLaMo's confidence from 10 to 14.

meta-task list shown in Figure 6. Given a query involving motion recognition and instructional guidance, the Executor receives five meta-tasks, denoted as $M1$ through $M5$, each aligned with a specific subgoal.

For each meta-task, the Executor identifies and invokes the most appropriate tool from the MotionCore suite. The first meta-task, M1, focuses on recognizing the combat move and is addressed using the MotionAnalyzer module. The output of this analysis is subsequently processed by the Aggregator in M2, which refines the candidate predictions by integrating outputs from multiple models with confidence-aware weighting. To fulfill the objective in M3, which requires an explanation of the technical components involved in the motion, the Executor employs the Professional Knowledge Retrieval module to access relevant domain-specific resources. The instructional guidance requested in M4 is provided through the Motion Retrieval module, which retrieves example motion clips aligned with the user's intent. Finally, the Generator module completes M5 by synthesizing a coherent response that incorporates the outputs of all previous stages.

The outputs corresponding to each step are explicitly listed, preserving traceability and transparency. This example demonstrates the Executor's ability to resolve diverse sub-tasks through tool-aware orchestration, enabling structured reasoning across heterogeneous model capabilities.

A.5 Verifier Template

To enhance the robustness and reliability of the overall reasoning pipeline, we design a prompt template for the Verifier that checks whether the planned meta-tasks and their corresponding execution results are consistent with the original user query. As illustrated in Figure 9, this template guides the Verifier through three key stages:

- **Consistency Review.** The Verifier receives the user query, the meta-task plan generated by the Planner, and the execution trace from the Executor. It inspects whether each meta-task logically follows from the user query and whether the execution results faithfully reflect the intended planning trajectory.

- **Error Attribution.** If any inconsistency, omission, or semantic drift is detected, the Verifier identifies the responsible agent and provides a concise description of the issue.

- **Final Answer Generation.** Based on the context, the Verifier either approves the final result from the Executor or produces an error report detailing the source and nature of the inconsistency. This enables for downstream correction or replanning, if necessary.

This verification step strengthens ChatMotion's robustness by providing an explicit mechanism for quality control and error identification, helping ensure that the final output is both logically sound and traceable.

B Execution Results for Different Stages

Figure 10 illustrates a full execution example in ChatMotion, showcasing how ChatMotion resolves

LLaMo	MotionGPT	TM2T	MotionLLM	Body		Seq.		Dir.		Rea.		Hall.		All	
				Acc.	Score	Acc.	Score	Acc.	Score	Acc.	Score	Acc.	Score	Acc.	Score
✓	✗	✗	✗	59.30	4.01	44.01	3.12	60.91	3.99	58.21	3.64	61.17	3.53	55.32	3.67
✓	✓	✓	✗	59.60	3.98	44.85	3.22	61.15	4.00	57.97	3.60	62.45	3.62	55.72	3.69
✓	✓	✓	✓	**60.89**	**4.03**	**46.21**	**3.30**	**62.11**	**4.03**	**59.53**	**3.77**	**68.95**	**3.78**	**56.90**	**3.72**

Table 7: Ablation study on the composition of MLLMs in ChatMotion, evaluated on MoVid-Bench-Motion. Each row shows the inclusion (✓) or exclusion (✗) of different motion-language models.

a complex multimodal query involving motion understanding, technique analysis, and instructional generation.

The Planner first decomposes the query into five meta-tasks, covering understanding, aggregation, explanation, retrieval, and synthesis. The Executor sequentially selects tools from the MotionCore suite to address each task. MotionAnalyzer triggers internal function calls to multiple motion understanding models, generating diverse interpretations of the input motion. Then, the Aggregator integrates these results using a confidence-based strategy. KnowledgeRetriever extracts technical insights, and MotionRetriever surfaces examples for instructional reference. Finally, Generator composes a coherent response by consolidating all intermediate outputs.

This example demonstrates ChatMotion's modular reasoning process, which combines analysis, retrieval, and generation in a tool-aware manner to solve complex multimodal queries and produce accurate, interpretable results.

C Ablation Study

C.1 Ablations on Components

In ablation studies, we began by utilizing only the executor component in conjunction with a basic majority voting mechanism. Initially, the system's performance was suboptimal, trailing behind advanced methods such as MotionLLM and LLaMo. This performance gap can be attributed to the lack of task organization, insufficient supervision, and the intrinsic limitations of the majority voting mechanism, which exacerbates error propagation by aggregating mistakes from different models, thereby distorting the correct output.

Then, the planner resulted in a modest improvement in performance, particularly for the single-objective analysis task, although the enhancement was not substantial. We got some qualitative results, however, revealing that the planner played a critical role in more complex tasks. Notably, without the planner, the ChatLLaMo model frequently encountered execution failures. The addition of the verifier further enhanced performance, notably improving hallucination metrics. Lastly, the **incorporation of a confidence-based selection mechanism** for model output, particularly in selecting responses from LLaMA, led to a **significant boost** in all performance metrics. This suggests that dynamically selecting the confident model output plays a pivotal role in enhancing the system's overall robustness and accuracy, especially in complex tasks.

C.2 Settings for Confidence Score

Table 6 reports the results for different confidence-based aggregation settings. The first row "Equal" represents a naive voting strategy where all models contribute equally. This leads to degraded overall performance, as weaker models (e.g., TM2T and MotionGPT) introduce noisy or inconsistent outputs that dilute the predictions of stronger models like LLaMo.

To improve this, we adopt a confidence-based (CB) strategy, where each model is assigned a score proportional to its standalone performance on MoVid-Bench. Specifically, we use the accuracy values on Movid-Bench-Motion reported in Table 1 to compute normalized weights and map them to a scaled score range. For TM2T, which is not included in that table, we evaluated its performance separately and got 33.15 on overall accuracy. Based on this, we assign confidence scores of 4, 5, 8, and 10 to TM2T, MotionGPT, MotionLLM, and LLaMo, respectively. This setting shown in Table 6 leads to a notable improvement over the Equal setting. The overall accuracy improves from 43.56 to 56.43 and all sub-metrics show consistent gains. This demonstrates that weighting models according to their standalone performance effectively mitigates aggregation errors introduced by weaker models.

Based on this setting, we further explore the effect of adjusting LLaMo's score, since it is the strongest model and dominates the aggregation. We fix the relative ratios of the other models and increase LLaMo's confidence from 10 to 14. Results show that assigning a confidence score of

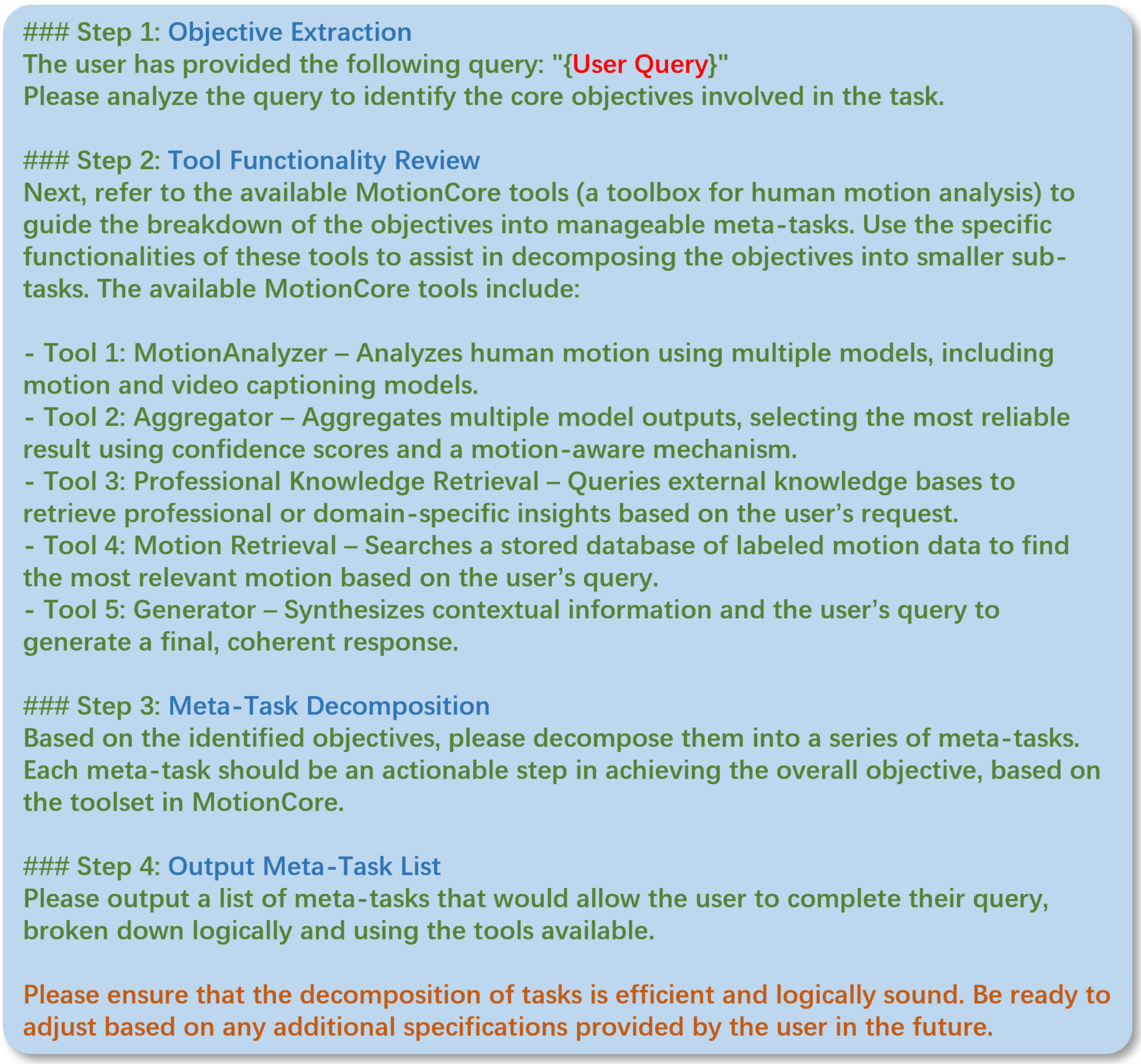
12 to LLaMo yields the best performance across most metrics. Further increasing the score leads to performance degradation due to over-reliance on LLaMo, which suppresses the complementary contributions of other models. Conversely, reducing LLaMo's weight underutilizes its predictive advantage, also causing a decline.

The overall performance remains stable when LLaMo's confidence score is set between 11 and 13, indicating that the aggregation is robust to small variations. Overall, this ablation confirms the necessity and effectiveness of the confidence-based strategy, which enables reliable integration of models with different capabilities.

C.3 Ablations on MLLMs

Table 7 presents the ablation study on the composition of MLLMs in ChatMotion. Using only LLaMo yields strong baseline results, with 55.32 accuracy and 3.67 score. Adding weaker models, such as MotionGPT and TM2T, leads to a marginal improvement of 0.40 in overall accuracy, suggesting their limited but non-disruptive contribution.

In contrast, incorporating the stronger Motion-LLM model brings consistent improvements across all metrics, raising the overall accuracy to 56.90. This validates the design of our model aggregation mechanism, which effectively amplifies the strengths of high-performing models while remaining robust to less accurate ones.



Step 1: Objective Extraction
The user has provided the following query: "{User Query}"
Please analyze the query to identify the core objectives involved in the task.

Step 2: Tool Functionality Review
Next, refer to the available MotionCore tools (a toolbox for human motion analysis) to guide the breakdown of the objectives into manageable meta-tasks. Use the specific functionalities of these tools to assist in decomposing the objectives into smaller sub-tasks. The available MotionCore tools include:

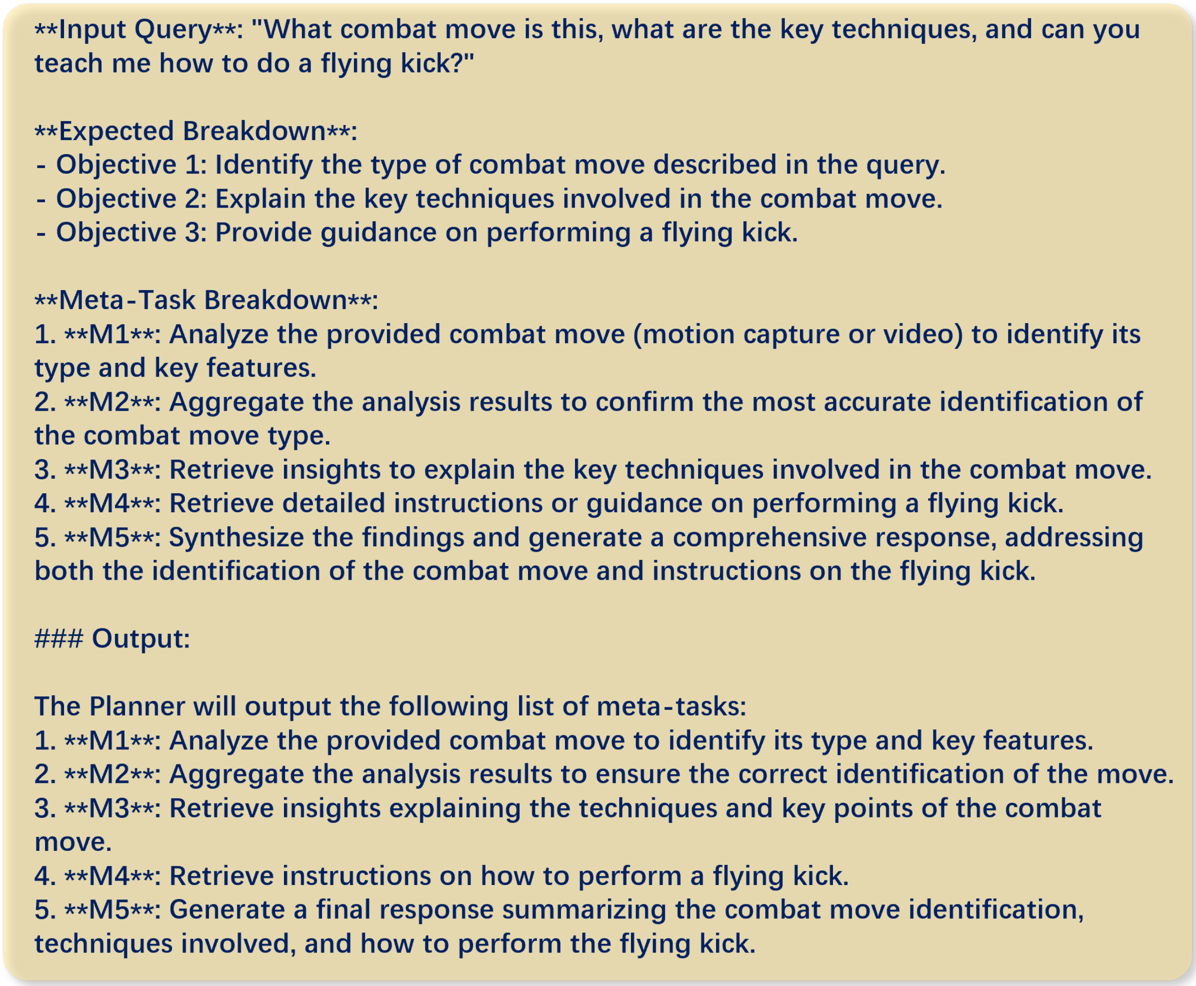
- Tool 1: MotionAnalyzer – Analyzes human motion using multiple models, including motion and video captioning models.
- Tool 2: Aggregator – Aggregates multiple model outputs, selecting the most reliable result using confidence scores and a motion-aware mechanism.
- Tool 3: Professional Knowledge Retrieval – Queries external knowledge bases to retrieve professional or domain-specific insights based on the user's request.
- Tool 4: Motion Retrieval – Searches a stored database of labeled motion data to find the most relevant motion based on the user's query.
- Tool 5: Generator – Synthesizes contextual information and the user's query to generate a final, coherent response.

Step 3: Meta-Task Decomposition
Based on the identified objectives, please decompose them into a series of meta-tasks. Each meta-task should be an actionable step in achieving the overall objective, based on the toolset in MotionCore.

Step 4: Output Meta-Task List
Please output a list of meta-tasks that would allow the user to complete their query, broken down logically and using the tools available.

Please ensure that the decomposition of tasks is efficient and logically sound. Be ready to adjust based on any additional specifications provided by the user in the future.

Figure 5: Template for the Planer to decompose task into sub-tasks.



Input Query: "What combat move is this, what are the key techniques, and can you teach me how to do a flying kick?"

Expected Breakdown:
- Objective 1: Identify the type of combat move described in the query.
- Objective 2: Explain the key techniques involved in the combat move.
- Objective 3: Provide guidance on performing a flying kick.

Meta-Task Breakdown:
1. **M1**: Analyze the provided combat move (motion capture or video) to identify its type and key features.
2. **M2**: Aggregate the analysis results to confirm the most accurate identification of the combat move type.
3. **M3**: Retrieve insights to explain the key techniques involved in the combat move.
4. **M4**: Retrieve detailed instructions or guidance on performing a flying kick.
5. **M5**: Synthesize the findings and generate a comprehensive response, addressing both the identification of the combat move and instructions on the flying kick.

Output:

The Planner will output the following list of meta-tasks:
1. **M1**: Analyze the provided combat move to identify its type and key features.
2. **M2**: Aggregate the analysis results to ensure the correct identification of the move.
3. **M3**: Retrieve insights explaining the techniques and key points of the combat move.
4. **M4**: Retrieve instructions on how to perform a flying kick.
5. **M5**: Generate a final response summarizing the combat move identification, techniques involved, and how to perform the flying kick.

Figure 6: The intermediate output of the Planner.

```
### Step 1: Understanding the Meta-Task List
You are the Executor responsible for choosing the function tools to finish each task. You
have received the following meta-task list from the Planner: "{Meta-Task List}"
Each meta-task requires a specific function to be executed using the available tools in the
MotionCore toolbox. The available MotionCore tools include:

- Tool 1: MotionAnalyzer – Analyzes human motion using multiple models, including
motion and video captioning models. Input: multimodal data output: text motion analysis
- Tool 2: Aggregator – Aggregates multiple model outputs, selecting the most reliable
result using confidence scores and a motion-aware mechanism. Input: multimodal data
along with candidate results and corresponding confidence. output: Refined analysis.
- Tool 3: Professional Knowledge Retrieval – Queries external knowledge bases to retrieve
professional or domain-specific insights based on the user's request. Input: user query.
Output: Relevant knowledge text.
- Tool 4: Motion Retrieval – Searches a stored database of labeled motion data to find the
most relevant motion based on the user's query. Input: user query. Output: Relevant
motion data
- Tool 5: Generator – Synthesizes contextual information and the user's query to generate
a final, coherent response. Input: previous context and user query. Output: coherent,
contextually accurate answer.

### Step 2: Tool Selection
For each meta-task, you will:
1. Review the meta-task's objective and determine which tool best aligns with the task's
requirements based on its functionality, input, and output.
2. Choose the most appropriate tool from the MotionCore.
3. You will ensure that the tool's functionality, input, and output match the requirements
of the meta-task before proceeding.

### Step 3: Task Execution
Once the appropriate tool is selected, the Executor will process the meta-task  and
perform the necessary operation using the chosen tool.

#### If the task is successfully executed:
- Return the output generated by the tool and proceed to the next meta-task.

#### If the task cannot be executed due to the unavailability or incompatibility of tools:
- The Executor will send an error report back to the Planner and Verifier, including the
reason why the task cannot be completed.

### Step 4: Final Output
After processing all meta-tasks, the Executor will output the final response or a report with
all completed tasks. If any meta-task was not successfully completed, the error report will
also be included.
```

Figure 7: Template for the Executor to choose tools in MotionCore to achieve the task.

```
**Meta-Task List**:
- **M1**: Analyze the provided combat move and identify its type and key features.
- **M2**: Aggregate the results of the motion analysis to confirm the move type.
- **M3**: Retrieve professional insights about the techniques involved in the move.
- **M4**: Retrieve instructions on how to perform a flying kick.
- **M5**: Generate the final response combining all the results.


### Meta-Task Execution:

#### **M1: Analyze the combat move**
- **Tool Selected**: **MotionAnalyzer**

#### **M2: Aggregate the results of the motion analysis**
- **Tool Selected**: **Aggregator**

#### **M3: Retrieve professional insights about the techniques**
- **Tool Selected**: **Professional Knowledge Retrieval**

#### **M4: Retrieve instructions on performing a flying kick**
- **Tool Selected**: **Motion Retrieval**

#### **M5: Generate the final response**
- **Tool Selected**: **Generator**

---

### Final Output:

After processing all meta-tasks, the Executor will output the following:

1. **M1**: Combat move analysis candidate results.
2. **M2**: Refined motion analysis based on aggregation.
3. **M3**: Professional insights about the techniques involved in the move.
4. **M4**: Motion instructions on how to perform a flying kick.
5. **M5**: The final synthesized response.
```

Figure 8: The intermediate execution details of the Executor.

Please examine whether the following user input, planning trajectory, and execution trajectory are consistent, and identify any errors or inconsistencies. If any issues are found, clearly indicate whether the problem resides with the Planning Agent or the Execution Agent, and provide the corresponding error information. If the trajectories are consistent, please directly output the final result of the execution trajectory.

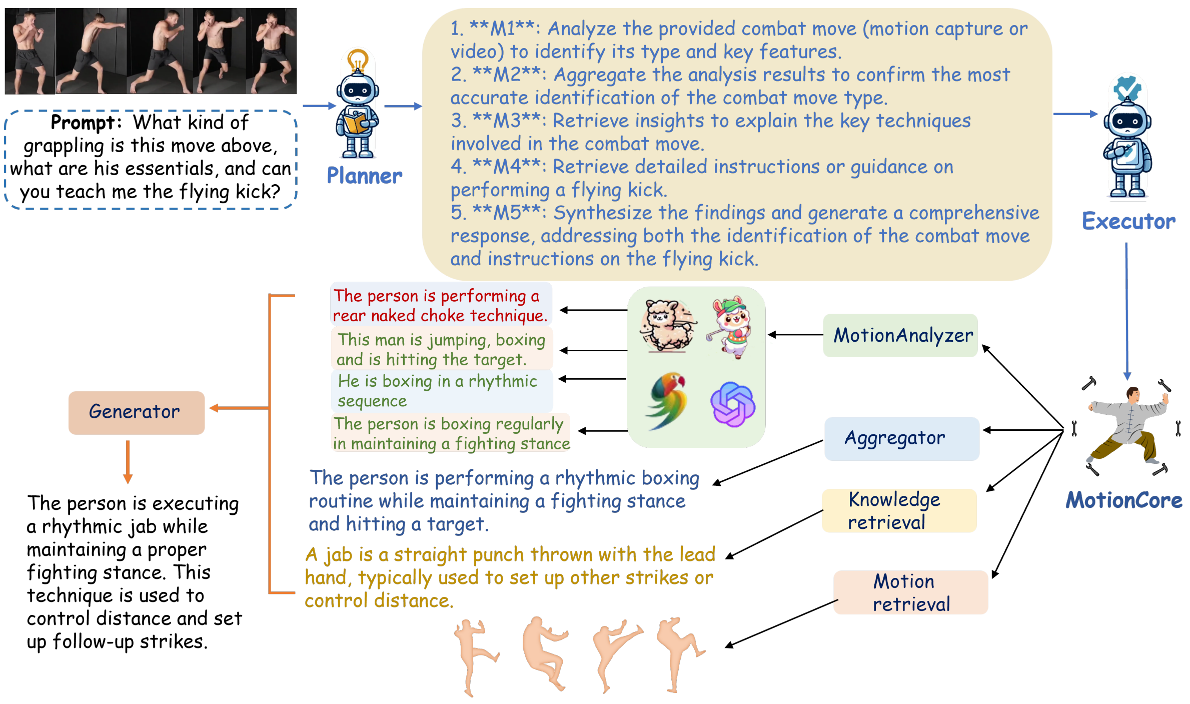
User Input:
{Insert the user_prompt here}

Planning Trajectory:
{Insert the plan from Planner here}

Execution Trajectory:
{Insert the Execution result from Executor here}

The response should include:
- Consistency Validation Result
- If there are errors, please include:
- The agent responsible for the issue (Planning Agent or Execution Agent)
- A specific description of the error or inconsistency
- If there are no errors, please include:
- The final result of the execution trajectory

Figure 9: Template for the Verifier to supervise the whole trajectory.



Prompt: What kind of grappling is this move above, what are his essentials, and can you teach me the flying kick?

1. **M1**: Analyze the provided combat move (motion capture or video) to identify its type and key features.
2. **M2**: Aggregate the analysis results to confirm the most accurate identification of the combat move type.
3. **M3**: Retrieve insights to explain the key techniques involved in the combat move.
4. **M4**: Retrieve detailed instructions or guidance on performing a flying kick.
5. **M5**: Synthesize the findings and generate a comprehensive response, addressing both the identification of the combat move and instructions on the flying kick.

The person is performing a rear naked choke technique.

This man is jumping, boxing and is hitting the target.

He is boxing in a rhythmic sequence

The person is boxing regularly in maintaining a fighting stance

The person is performing a rhythmic boxing routine while maintaining a fighting stance and hitting a target.

A jab is a straight punch thrown with the lead hand, typically used to set up other strikes or control distance.

The person is executing a rhythmic jab while maintaining a proper fighting stance. This technique is used to control distance and set up follow-up strikes.

Figure 10: The detail execution results in ChatMotion.