TOUCHAGENT: BENCHMARKING AND ADVANCING REAL-WORLD TACTILE INTELLIGENCE

Anonymous authors
Paper under double-blind review

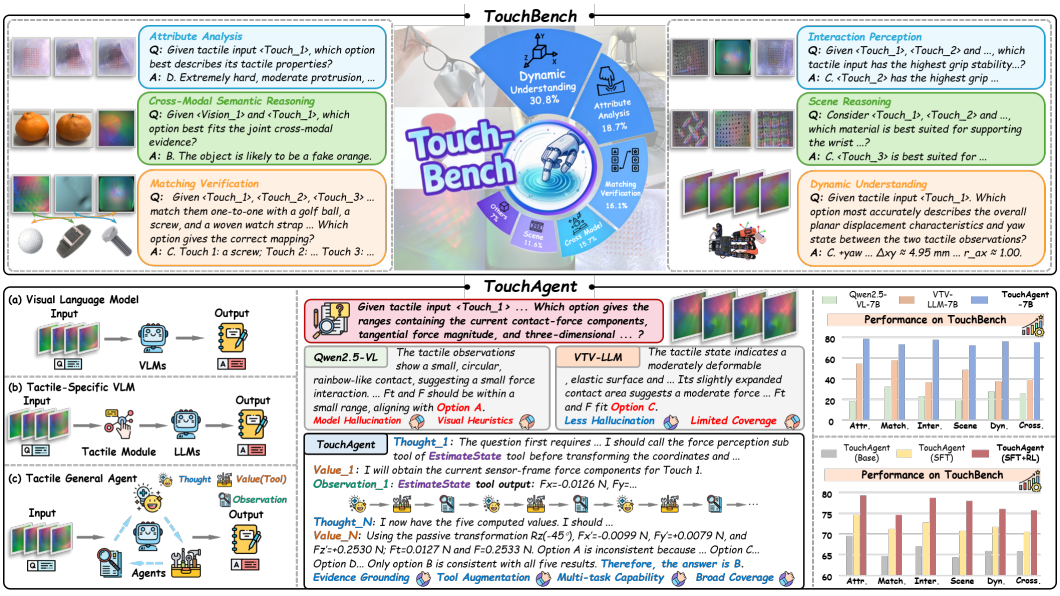


Figure 1: **Overview of TouchBench and TouchAgent.** (i) The top panel presents TouchBench, covering six core categories of real-world tactile understanding tasks with representative examples. (ii) The bottom panel compares conventional VLMs, tactile-specific VLMs, and TouchAgent, highlighting its tool-augmented, evidence-grounded reasoning and performance on TouchBench.

ABSTRACT

Recent advances in multimodal large language models (MLLMs) have expanded multimodal intelligence. However, progress in the tactile domain remains constrained by several distinctive challenges, including heterogeneous sensor modalities, diverse domain knowledge, and a fragmented task landscape. To address this gap, we introduce TouchBench, a comprehensive benchmark for real-world tactile interaction with **10K** questions and approximately **383K** observation frames, covering **6** real-world tactile task categories ranging from perception to reasoning and decision-making, thereby enabling broader and more systematic evaluation than prior benchmarks. Using TouchBench, we evaluate 14 open-source, proprietary, and tactile-language models, revealing systematic deficiencies in perceptual grounding, domain knowledge, and reasoning, all of which are essential for real-world tactile intelligence. Beyond evaluation, we propose TouchAgent, an agent framework for tactile intelligence that strategically integrates knowledge retrieval, perception, and reasoning through domain-specific tactile models and tools. To further improve tool planning and evidence integration, we construct TouchAgent-Instruct with **40K** multi-round interaction trajectories and optimize TouchAgent through supervised fine-tuning and reinforcement learning. Extensive experiments show that TouchAgent consistently outperforms existing MLLMs and tactile-specific models across tactile benchmarks, highlighting the effectiveness of tool-augmented agents in advancing real-world tactile intelligence. Our code and benchmark will be released.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

Recent progress in multimodal large language models (MLLMs) has accelerated the pursuit of artificial general intelligence (AGI) (Fei et al., 2025; Yao et al., 2025b). The emergence of Expert AGI marks a key milestone in this process, with many models increasingly matching or surpassing human experts across diverse specialized domains (Feng et al., 2024; Xiao et al., 2026). As research advances toward this stage, an increasing number of multimodal benchmarks have been proposed to evaluate models' perception, understanding, and reasoning capabilities across disciplines Yao et al. (2025a); Fu et al. (2024a). These benchmarks have become an important basis for characterizing model capability boundaries and facilitating the development of multimodal systems.

Despite these advances, the tactile domain has not received sufficient attention in existing benchmark evaluations. As a key modality linking agents to the physical world, touch is essential for understanding object properties and reliable embodied interaction, which requires comprehensive tactile understanding (Li et al., 2020). However, existing tactile benchmarks remain limited in scope, primarily focusing on perceptual tasks such as material recognition, slip detection, or force estimation Higuera et al. (2024); Lyu et al. (2026b); Yang et al. (2022); Fu et al. (2024b). Although recent studies have begun to explore tactile reasoning, they are often limited to relatively simple tasks or a narrow range of scenarios (Yu et al., 2024; Xie et al., 2026; Lyu et al., 2026a). In contrast, tactile understanding in real-world settings is complex and diverse, requiring the integration of sensor data, evidence-based reasoning, and domain knowledge, yet existing benchmarks cannot adequately assess these capabilities (Lai et al., 2026). This evaluation gap highlights the need for a comprehensive evaluation framework to systematically assess and advance tactile intelligence.

To address this limitation, we introduce **TouchBench**, a comprehensive benchmark for the tactile domain intended to systematically evaluate the tactile understanding capabilities of MLLMs. As shown in Figure 1, TouchBench comprises 10K questions, 87 object types, and approximately 383K observation frames from nine types of tactile sensors, covering six core task categories including interaction perception and scene reasoning. Furthermore, these tasks span 16 representative formats, including interaction feedback and scenario matching, creating diverse and realistic tactile challenges. Unlike traditional benchmarks that primarily focus on ocalized perception or single-step reasoning Higuera et al. (2024); Yu et al. (2024), TouchBench extends evaluation to multi-step reasoning, thereby enabling more comprehensive, fine-grained, and reliable model evaluation.

We conduct a comprehensive evaluation of 14 MLLMs on TouchBench, including open-source (OpenAI, 2025; Anthropic, 2025a), closed-source (Bai et al., 2025b; Li et al., 2024b), and domain-specific models (Yu et al., 2024; Xie et al., 2026; Lyu et al., 2026a). The results reveal critical capability gaps in tactile perception and understanding among current models. These limitations primarily stem from TouchBench's greater demands on the integration of tactile evidence from multiple sources and expert-level reasoning. However, existing MLLMs, which are primarily optimized for instruction following and descriptive tasks, struggle to adequately support the evidence-dependent reasoning and multi-step analysis capabilities for tactile understanding (Li et al., 2024a; Yao et al., 2025a). To address these challenges, we introduce **TouchAgent**, an agent system for diverse tactile understanding tasks, intended to enhance the ability of MLLMs to solve tasks in tactile settings. Specifically, TouchAgent integrates 8 tactile-specific tools organized around three core capabilities: perception, knowledge, and reasoning. By coupling LLM reasoning with these structured tools, TouchAgent adaptively performs task planning, tool invocation, and evidence integration, supporting multi-source data and multi-step analysis to tackle real-world tactile tasks beyond the capabilities of existing MLLMs and tactile-specific models. To further enhance the tool use and evidence integration capabilities of TouchAgent, we construct **TouchAgent-Instruct**, which contains approximately 40K multi-turn interaction trajectories, to further develop the agent's ability to perform both tool invocation and evidence integration reliably. We train the agent end to end through instruction tuning and reinforcement learning, enabling it to select appropriate tools based on user instructions and effectively integrate tool outputs with relevant tactile evidence, thereby generating accurate, comprehensive, and task-relevant responses. Our contributions are as follows:

- We introduce **TouchBench**, a comprehensive and diverse benchmark for tactile understanding. It comprises **10K** questions covering **6** categories of common tactile tasks in real-world settings, providing a robust and challenging platform for systematically evaluating the tactile understanding capabilities of MLLMs and agents.

- We propose **TouchAgent**, an agent system for tactile understanding. The system integrates an LLM with domain-specific models and tools, solving tasks through task decomposition, multi-turn tool interaction, and evidence integration. Furthermore, we construct **TouchAgent-Instruct** and enhance the agent's tool invocation and evidence integration capabilities through end-to-end training, thereby improving its ability to solve diverse tactile understanding tasks.
- We conduct extensive experiments on TouchBench, revealing the limitations of existing MLLMs and tactile-language models in the real-world tactile domain and demonstrating the effectiveness of TouchAgent across diverse tactile understanding tasks.

2 Releted Work

Multimodal Benchmarks for Specialized Domains. Progress toward Expert AGI has driven increasingly specialized benchmarks, shifting evaluation from exam-style QA toward domain-specific multimodal reasoning and agentic problem solving (Yue et al., 2024; Zhou et al., 2025). Recent benchmarks such as GeoMMBench (Xiao et al., 2026), MedAgentBench (Jiang et al., 2025), and ScienceAgentBench (Chen et al., 2025) evaluate expert-level capabilities across geoscience, medicine, and scientific problem solving, emphasizing multimodal reasoning, domain knowledge, and interaction. In contrast, existing tactile benchmarks primarily focus on perceptual tasks such as property recognition, force estimation, and slip detection (Higuera et al., 2024; Yang et al., 2022). Although recent benchmarks such as VTV-150K (Xie et al., 2026) and PHYSICLEAR (Yu et al., 2024) have begun to explore tactile reasoning, they remain limited to relatively simple single-step reasoning and restricted scenarios. Overall, existing tactile benchmarks remain insufficient for evaluating the complex and diverse challenges encountered in real-world tactile interaction. To address this gap, we introduce TouchBench, a multi-level benchmark spanning tactile perception, reasoning, and decision-making that jointly evaluates task performance, reasoning trajectories, and evidence faithfulness, thereby bridging the gap between tactile benchmarking and real-world applications.

Tactile Intelligence. Touch serves as a key sensory modality linking agents to the physical world and has attracted increasing attention (Kappassov et al., 2015). Tactile intelligence aims to enable agents to perceive physical properties, understand contact dynamics, and reason and make decisions based on tactile evidence. Early research primarily focused on tactile perception and representation learning, using taxel arrays or visuotactile sensors to extract and analyze tactile signals (Lyu et al., 2026b; Abad & Ranasinghe, 2020). Sparsh (Higuera et al., 2024), UniTouch (Yang et al., 2024a), and AnyTouch (Feng et al., 2025) further improve cross-sensor and cross-task generalization through self-supervised and cross-modal learning. More recently, large language models have advanced the integration of touch and language, extending tactile intelligence beyond perception toward semantic understanding and reasoning. Representative methods such as Octopi (Yu et al., 2024), VTV-LLM (Xie et al., 2026), and TouchThinker (Lyu et al., 2026a) further expand tactile modeling to semantic understanding, dynamic reasoning, and embodied decision-making, respectively. Despite these advances, existing methods remain constrained by predefined inputs and fixed end-to-end pipelines, limiting heterogeneous information integration, domain knowledge utilization, and multi-step reasoning (Zhang et al.; Yang et al., 2024a). Therefore, real-world tactile intelligence requires knowledge-enhanced, evidence-driven, multi-stage reasoning beyond perception alone.

3 TouchBench

3.1 Overview of TouchBench

Real-world tactile interaction involves diverse sensory signals and physical processes, making it a challenging setting for evaluating the physical understanding capabilities of MLLMs. However, most existing tactile benchmarks focus on local state recognition or single-step reasoning (Higuera et al., 2024; Yu et al., 2024; Feng et al., 2025), limiting the systematic evaluation of capabilities required for real-world tactile applications, including multi-source data integration, holistic tactile understanding, and multi-stage decision-making. To address this gap, we introduce TouchBench, a unified benchmark built from real-world multi-source tactile observations. Unlike prior benchmarks, TouchBench requires models to combine analytical tools with domain knowledge to quantitatively analyze tactile signals, such as contact forces and relative displacements, and perform multi-stage reasoning and decision-making. It comprises **10K** rigorously curated questions averaging **4.6** rea-

Dataset	Cross Modality	Frame Numbers	Requiring Tools	Reasoning Steps	Quantitative Analysis	Scene Reasoning
Feel	✗	37K	✗	0	✗	✗
TacBench	✗	324K	✗	0	✗	✗
Touch and Go	✗	16.3K	✗	0	✗	✗
TVL Benchmark	✗	402	✗	1	✗	✗
VTV-150K	✗	7.5K	✗	1	✗	✓
TactileBench	✗	16.3K	✗	1	✗	✗
PHYSICLEAR	✗	45.8K	✗	1	✗	✓
TouchBench (Ours)	✓	383k	✓	>4	✓	✓

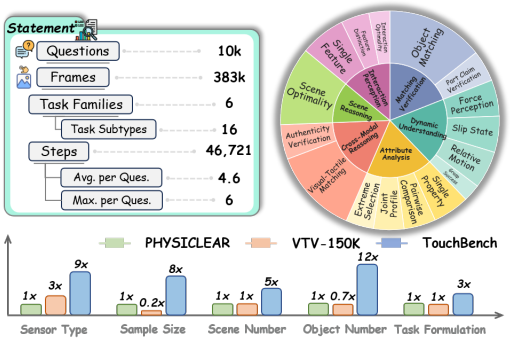


Figure 2: **Dataset Comparison and Overview.** The left panel compares TouchBench with existing tactile benchmarks multiple dimensions. The right panel presents the statistics and task composition of TouchBench, and compares its scale and coverage with existing benchmarks.

soning steps, enabling evaluation of both answer correctness and reasoning faithfulness to tactile evidence. As shown in Table 1, even state-of-the-art MLLMs and tactile-specific models achieve only limited performance, highlighting the difficulty and diagnostic value of TouchBench.

3.2 Data Source

To construct high-quality tactile question instances, we require that each instance originate from real-world everyday interaction scenarios and be verifiable using raw tactile observations and relevant evidence. Based on these requirements, TouchBench selects **12** publicly available tactile datasets as raw data sources, including VTV-150K (Xie et al., 2026), PHYSICLEAR (Yu et al., 2024), Touch and Go (Yang et al., 2022), TacQuad (Feng et al., 2025), TacBench (Higuera et al., 2024), and Feeling of Success (Feel) (Calandra et al., 2017). Detailed descriptions of these datasets are provided in the Appendix A.1. These datasets were collected by different research teams during interactions with real objects and robotic manipulation, covering **9** types of tactile sensors, **87** categories of everyday objects and material surfaces, more than **10** scenarios (e.g., home and outdoor settings), and various interaction types, including pressing, sliding, and grasping. Each data source retains raw tactile observations and interaction records, providing a traceable basis for subsequent unified attribute annotation, question construction and evidence verification.

3.3 Data Processing and Quality

As shown in Figure 3, the data processing pipeline of TouchBench consists of three stages: annotation, verification and review.

Annotation. To unify the heterogeneous annotation schemes across different publicly available tactile datasets, we organize raw observations and filter valid interaction segments. Using the annotation protocol of VTV-150K (Xie et al., 2026) as a reference, we then establish a unified attribute space along four dimensions: hardness, protrusion, elasticity, and friction. Existing labels are mapped to this schema, while the remaining samples are independently annotated using a three-level scheme by five tactile perception experts, with final labels determined through cross-checking and expert consensus. The inter-annotator agreement scores (Yu et al., 2024) are 0.917 for hardness, 0.912 for protrusion, 0.935 for elasticity, and 0.896 for friction. Based on these annotations, we construct questions across six task categories and annotate reference multi-step solution trajectories.

Verification. We further verify whether each question can be completed within TouchAgent's tool environment. Specifically, we verify whether the tactile, visual, or external knowledge required to answer each question can be obtained from the specified inputs and tools, whether the tool calls in the reference trajectories satisfy the required evidence dependencies, and whether key observations provide the necessary information for subsequent reasoning. For questions with missing necessary evidence, non-executable tool calls, or reference trajectories that do not align with task requirements, we either revise the questions or exclude them from TouchBench. A total of 17.5K candidate questions passed verification and were retained.

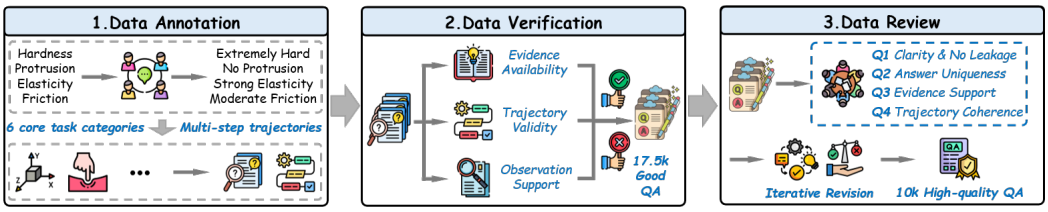


Figure 3: **Construction and Quality Control of TouchBench.** The left shows multi-step annotation across six core task categories, the middle verifies evidence and trajectory validity, and the right presents expert review and iterative revision, yielding 10K high-quality QA pairs.

Review. After verification, we invited three computer science experts and three tactile perception experts as reviewers to evaluate the questions independently according to four criteria: **(Q1)** whether the question statement and candidate options are clear and free of information leakage; **(Q2)** whether the correct answer is unique; **(Q3)** whether the correct answer is explicitly supported by the corresponding tactile evidence; and **(Q4)** whether the reference trajectory correctly links the task objective, key observations, and the final conclusion. Disagreements were jointly resolved through iterative review and revision, while questions with unresolved ambiguities were removed. The average exact agreement rate across the four review criteria was 79.5%. Ultimately, 10K high-quality questions were retained from the 17.5K candidate questions.

3.4 Data Statistics

TouchBench comprises **10K** four-option multiple-choice questions, integrating approximately **383K** frames of raw observations from **9** types of tactile sensors and covering around **87** types of everyday objects. These questions span **6** core tasks: object tactile attribute analysis, tactile matching verification, tactile interaction perception, tactile scene reasoning, tactile dynamic understanding, and cross-modal semantic reasoning. Together, these tasks cover multi-level tactile capabilities, ranging from basic perception to high-level understanding and reasoning. Detailed task descriptions are provided in the Appendix A.2. To provide an intuitive overview of the data distribution in TouchBench, we present summary statistics across task categories, source datasets, sensor types, and object categories, as shown in Figure 2.

4 TouchAgent

To support the diverse tactile tasks in TouchBench, we introduce TouchAgent, a tool-augmented agent system designed for tactile scenarios, as illustrated in Figure 4. We first formulate its interaction process under a ReAct-style framework in section 4.1. Then, we introduce a specialized tool library for tactile perception and reasoning in section 4.2. Finally, we present the training strategy for learning reliable tool planning and evidence integration in section 4.3.

4.1 Operation Mechanism

TouchAgent receives a task goal g, interprets user queries and intermediate tool outputs, and selects actions from a modular tool library to progressively acquire and integrate task-relevant evidence. We formulate this interaction process as sequential decision-making under partial observability, following the POMDP perspective (Yao et al., 2022). For notational convenience, we characterize the interaction using $\langle g, S, A, O, T \rangle$, where g is the task goal, S is the state space containing raw multimodal observations and latent physical states, A is the action space consisting of tool calls and a termination action, O is the observation space representing semantic outputs returned by the tools, and $T : S \times A \to S$ is the state transition function induced by tool execution.

At each time step t, TouchAgent selects the next action conditioned on the task goal g and the current interaction history $m_t = (o_0, a_0, o_1, a_1, \ldots, a_{t-1}, o_t)$. The action is sampled from the policy

$$a_t \sim \pi_\theta(a_t \mid g, m_t), \tag{1}$$

where π_θ is implemented by the LLM controller, which serves as the main Agent. Each ReAct cycle consists of four stages: ① **thought and planning**, where TouchAgent identifies the next evidence

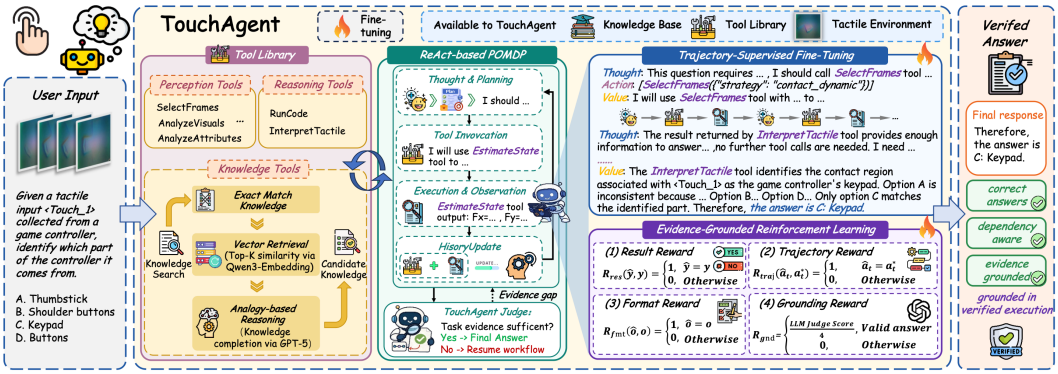


Figure 4: **TouchAgent Framework.** TouchAgent combines specialized tools with ReAct-based reasoning and is trained via trajectory-supervised fine-tuning followed by evidence-grounded reinforcement learning for reliable tool planning, evidence integration, and grounded tactile reasoning.

requirement from the current context; ② **tool invocation**, where it selects an appropriate tool and its parameters; ③ **execution and observation**, where the selected tool is executed and returns a new Observation; and ④ **history update**, where the tool call and its output are appended to the interaction history. Once sufficient evidence has been collected, TouchAgent selects the termination action and generates the final answer based on the complete interaction history. The resulting trajectory records both the final task solution and a reproducible sequence of tool interactions.

4.2 Specialized Tool Library and Core Capabilities

We group the core capabilities for expert-level tactile intelligence into Knowledge, Perception, and Reasoning, These capabilities reflect key cognitive processes in real-world tactile interaction: acquiring prior knowledge, interpreting multimodal sensory signals, and reasoning for decision-making. Accordingly, TouchAgent integrates **8** specialized tools to provide domain-specific capabilities for robust analysis and interpretation.

- ***Knowledge Tools.*** KnowledgeSearch[1] adaptively retrieves tactile knowledge, object priors, physical principles, and attribute constraints from the current question and Observations, and expands them with semantically related or complementary evidence. Results are returned as natural language, formulas, or structured constraints for subsequent analysis and decision-making.
- ***Perception Tools.*** SelectFrames and EstimateDepth obtain key frames and relative depth changes; AnalyzeAttributes estimates tactile properties such as hardness and friction; EstimateState analyzes key dynamic states, including grasp outcomes, contact forces, relative motion, and slip states; and AnalyzeVisuals extracts object categories and appearance information. Together, they transform multimodal inputs into interpretable perceptual evidence for downstream reasoning.
- ***Reasoning Tools.*** InterpretTactile maps tactile observations into the language semantic space for tactile description, part matching, and statement verification, while RunCode generates and executes constrained programs for local evidence enhancement and physics-based numerical computation. Together, they provide semantic and numerical reasoning for final decision-making.

The tools return structured Observations through a unified interface, enabling TouchAgent to flexibly combine knowledge, perception, and reasoning for tactile understanding and decision-making based on multimodal evidence. Detailed tool descriptions are provided in the Appendix A.3.

4.3 Training and Optimization

To enable reliable tool planning and evidence integration, we construct **TouchAgent-Instruct**. We instantiate the ReAct-style interaction described in section 4.1 using a unified multi-turn **"Thoughts–Actions–Value"** format. Given task instructions, candidate options, and the required tactile or visual inputs, *Thoughts* identifies the evidence required based on the current question and accumulated Observations, *Actions* specifies the next tool call and its parameters, and *Value* states

[1] Text in blue box refers to a tool name.

the objective of the current action or produces the final answer when *Actions* is empty. Using predefined task-specific reference trajectories as in-context demonstrations, we employ GPT-5.2 (Singh et al., 2025) to generate approximately 40K multi-turn trajectories covering tool use and evidence integration. The detailed generation pipeline is provided in Appendix A.4.1.

Trajectory-Supervised Fine-Tuning. Based on TouchAgent-Instruct, we first perform multi-turn supervised fine-tuning. Each training trajectory τ consists of a task goal, sequential tool calls and their Observations, and a final termination response. Given the task goal g and interaction history m_t at turn t, the model generates a structured output $y_t = (\text{Thoughts}_t, \text{Actions}_t, \text{Value}_t)$. When $\text{Actions}_t \neq \varnothing$, the model learns to select the next tool and its parameters; when $\text{Actions}_t = \varnothing$, the model integrates historical Observations and generates the final explanation and answer in Value_t. We optimize the standard autoregressive objective:

$$\mathcal{L}_{\text{SFT}} = -\sum_{(g,\tau)\in\mathcal{D}} \sum_{t=1}^{T_\tau} \log \pi_\theta(y_t \mid g, m_t), \tag{2}$$

where $\mathcal{D}$ denotes the training set and T_τ is the number of interaction turns in trajectory τ. Tool Observations are used only as conditional context for subsequent generation. Through this stage, TouchAgent learns to plan tool calls based on the available evidence and integrate accumulated evidence to produce the final answer upon termination.

Evidence-Grounded Reinforcement Learning. Optimizing solely for final-answer correctness can encourage shortcut behaviors, such as relying on problem priors without acquiring critical tactile evidence or failing to effectively use the obtained Observations (Guo et al., 2025; Lai et al., 2026). To address this, We therefore introduce Evidence-Grounded Reinforcement Learning after SFT to improve TouchAgent's ability to acquire, integrate, and reliably reason over task-relevant tactile evidence. Specifically, we design four complementary rewards to guide policy learning: the **result reward** R_{result} measures final-answer correctness; the **trajectory reward** $R_{\text{trajectory}}$ constrains tool selection, key parameters, call order, and termination to follow the reference evidence path; the **format reward** R_{format} ensures that the "Thoughts–Actions–Value" protocol and tool calls remain parseable and executable; and the **faithfulness reward** R_{faith} uses an external MLLM (OpenAI, 2025) Judge to assess consistent integration of historical Observations and whether the final conclusion is sufficiently supported by acquired tactile evidence. The total reward is defined as:

$$R_{\text{total}} = \lambda_{\text{trajectory}} R_{\text{trajectory}} + \lambda_{\text{result}} R_{\text{result}} + \lambda_{\text{format}} R_{\text{format}} + \lambda_{\text{faith}} R_{\text{faith}}, \tag{3}$$

Detailed reward definitions, evaluation criteria, and weight settings are provided in the Appendix A.4.2. We initialize the policy π_θ with the supervised fine-tuned TouchAgent and optimize it using standard GRPO (Guo et al., 2025) with R_{total}:

$$\theta^* = \arg\max_\theta J_{\text{GRPO}}(\theta), \tag{4}$$

Through this objective, TouchAgent learns not only to produce correct answers, but also to follow appropriate evidence dependencies for tool use and perform faithful, verifiable multi-stage reasoning grounded in actual tactile Observations.

5 Experiments

5.1 Experimental Setup

Evaluated Models. We evaluate 14 representative baselines across three categories: six open-source MLLMs, including InternVL2.5-8B (Chen et al., 2024) and LLaVA-OneVision-Qwen2-7B (Li et al., 2024b); five closed-source MLLMs, including GPT-4o (Hurst et al., 2024) and Claude-Sonnet-4.5 (Anthropic, 2025b); and three tactile-language models, including Octopi-7B (Yu et al., 2024), VTV-LLM-7B (Xie et al., 2026) and TouchThinker-7B (Lyu et al., 2026a).

Implementation Details. TouchAgent is implemented using LangChain and LangGraph, with Qwen2.5-7B (Ahmed et al., 2025) as the default LLM controller and 8 specialized tools for knowledge, perception, and reasoning. For consistency, all models follow their default multiple-choice prompting and are evaluated by mean accuracy. Responses are automatically parsed using rule-based extraction, with invalid or unparsable outputs counted as incorrect. All experiments are conducted on NVIDIA A100 GPUs. Additional details are provided in Appendix A.5.

Table 1: **Evaluation results on TouchBench across six tactile task categories.** Results are reported in percentage (%), with **Avg.** denoting mean accuracy across the six task categories. The best and second-best results are **bolded** and underlined, respectively. **Attr.**, **Match.**, **Inter.**, **Scene**, **Dyn.**, and **Cross.** denote tactile attribute analysis, matching verification, interaction perception, scene reasoning, dynamic understanding, and cross-modal semantic reasoning, respectively.

Model	Attr.	Match.	Inter.	Scene	Dyn.	Cross.	Avg.
Uniform Random A–D	25.0	25.0	25.0	25.0	25.0	25.0	25.0
Open-source MLLMs							
InternVL2.5-8B (Chen et al., 2024)	13.7	25.9	19.9	30.8	29.3	29.0	24.8
LLaVA-OneVision-Qwen2-7B (Li et al., 2024b)	21.7	25.4	20.1	30.4	31.2	33.8	27.1
MiniCPM-V-2.6 (Yao et al., 2024)	15.8	24.1	17.9	21.3	33.6	25.7	23.1
Qwen2.5-VL-7B (Ahmed et al., 2025)	19.2	32.0	20.8	19.8	28.2	25.1	24.2
Qwen3-VL-8B (Bai et al., 2025a)	24.3	34.8	23.7	28.5	34.1	31.5	29.5
VideoLLaMA3-7B (Zhang et al., 2025)	14.3	25.8	24.9	19.5	37.5	34.9	26.2
Closed-source MLLMs							
GPT-4o (Hurst et al., 2024)	24.0	29.5	29.7	31.6	31.4	32.9	29.9
GPT-5.2 (OpenAI, 2025)	25.7	33.9	24.8	33.7	30.0	34.1	30.4
Gemini-3-Flash-Preview (Google DeepMind, 2025)	23.5	28.4	29.4	35.1	29.8	35.6	30.3
Claude-Sonnet-4.5 (Anthropic, 2025b)	25.0	28.2	27.5	37.2	35.8	37.8	31.9
Claude-Opus-4.5 (Anthropic, 2025a)	27.1	34.7	30.4	43.6	32.5	39.9	34.7
Tactile-Language Models							
Octopi-7B (Yu et al., 2024)	55.7	45.4	35.8	42.5	35.1	37.2	42.0
VTV-LLM-7B (Xie et al., 2026)	57.3	59.6	36.2	46.8	37.6	39.7	46.2
TouchThinker-7B (Lyu et al., 2026a)	62.6	60.9	46.1	52.3	49.3	42.3	52.3
TouchAgent (Ours)	69.2	64.7	67.4	64.2	66.1	65.5	66.2
TouchAgent-SFT (Ours)	74.8	71.5	73.1	70.4	71.6	70.1	71.9
TouchAgent-SFT+RL (Ours)	**78.6**	**74.6**	**77.9**	**73.8**	**76.3**	**75.4**	**76.1**

5.2 Main Results

Table 1 presents the overall performance of different models on TouchBench. The experiments reveal the following key findings.

TouchBench poses significant challenges for current MLLMs. Despite strong performance on conventional vision and language tasks, current MLLMs achieve a highest average accuracy of only 34% on TouchBench. General MLLMs struggle to encode tactile information, as evidenced by their limited attribute analysis (Attr.) performance, revealing fundamental perceptual weaknesses. Performance further degrades on tasks requiring multi-source perception, cross-frame evidence integration, and multi-stage decision-making, exposing gaps in domain knowledge and reasoning. These results demonstrate the rigor and effectiveness of TouchBench for evaluating tactile intelligence.

TouchAgent demonstrates superior capability. It achieves an average accuracy of **76.1%** on TouchBench, outperforming the strongest general MLLM by **41.4%**, with consistent gains across all six task categories. The improvements are particularly pronounced in tactile dynamic understanding, scene reasoning, and cross-modal semantic reasoning, demonstrating that domain-specific tools and multi-round evidence acquisition effectively bridge the capability gaps of end-to-end MLLMs.

Tactile-specific models remain limited in task coverage and high-level reasoning. Tactile-language models, such as TouchThinker (Lyu et al., 2026a), show advantages in local attribute recognition and specific tactile tasks, but substantially lag behind TouchAgent on tasks requiring multi-source evidence integration or complex decision-making. This suggests that stronger tactile representations alone are insufficient for comprehensive tactile intelligence, and explicit tool planning and evidence integration are also essential for improving generalizable reasoning and decision-making.

Table 2: **Ablation studies of training stages and reward components.**

Configuration	Inter.	Scene	Dyn.	Cross.	Avg.
TouchAgent(base)	67.4	64.2	66.1	65.5	65.8
+ SFT	73.1	70.4	71.6	70.1	71.3
+ GRPO (0/1)	73.8	70.9	72.2	71.4	72.1
+ Format Reward	74.9	71.7	73.4	72.5	73.1
+ Trajectory Reward	75.6	72.1	75.9	74.3	74.5
+ Faithfulness Reward	77.9	73.8	76.3	75.4	75.9

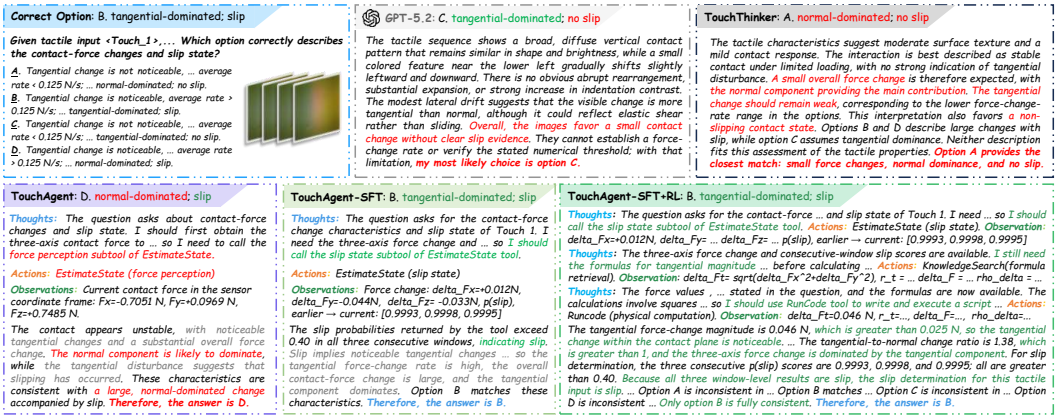


Figure 5: **Comparison of tactile reasoning across different models.** Green indicates appropriate tool use or evidence-grounded reasoning, red indicates errors, and gray denotes inferences lacking sufficient evidence despite correct final answers.

5.3 Ablation Studies

We further analyze the contributions of TouchAgent's training stages and key components. As shown in Table 2, SFT provides a substantial initial gain, improving Avg. from 65.8 to 71.3 by learning structured tool use and evidence integration. Standard GRPO with binary answer rewards further raises Avg. to 72.1, but the limited gain suggests that final-answer supervision alone is insufficient for reliable tactile-grounded reasoning. Sequentially introducing R_{Format}, $R_{\text{Trajectory}}$, and $R_{\text{Faithfulness}}$ further improves Avg. to 73.1, 74.5, and 75.9, respectively, demonstrating the effectiveness of Evidence-Grounded Reinforcement Learning for tool planning, evidence utilization, and final prediction. We further ablate the Knowledge, Perception, and Reasoning tools. As shown in Table 3, removing any tool category degrades performance, demonstrating that these capabilities are complementary and essential for tactile understanding.

Table 3: **Ablation studies of tool components.**

Configuration	Inter.	Scene	Dyn.	Cross.	Avg.
TouchAgent	**77.9**	**73.8**	**76.3**	**75.4**	**75.9**
w/o Knowledge Tool	63.2	54.9	63.5	68.6	62.6
w/o Reasoning Tool	71.4	64.1	51.8	61.8	62.3
w/o Perception Tool	57.3	60.5	45.2	50.4	53.4

5.4 Qualitative Results

As shown in Figure 5, we compare TouchAgent across training stages with general-purpose and tactile-specific MLLMs on a representative case. The base TouchAgent can invoke specialized tools, but limited tool planning and evidence integration may lead to incorrect predictions. SFT improves task-aware tool use and evidence utilization, enabling correct reasoning. Evidence-Grounded Reinforcement Learning further promotes structured multi-stage reasoning by integrating tactile observations, domain knowledge, and numerical computation into coherent and verifiable predictions. In contrast, other MLLMs lack explicit evidence acquisition and integration, resulting in less grounded predictions. These results highlight TouchAgent's capability for reliable tactile reasoning. We further present representative successful trajectories and failure cases of TouchAgent in Appendix A.6.

6 Conclusion

In this work, we introduce TouchBench, a comprehensive benchmark for real-world tactile perception, understanding, and reasoning, revealing substantial limitations across 14 open-source, proprietary, and tactile-language models. To address these challenges, we propose TouchAgent, a tool-augmented agent for tactile intelligence that coordinates domain-specific models and tools for adaptive planning, evidence acquisition, and multi-stage reasoning. We further construct TouchAgent-Instruct with 40K multi-round trajectories and optimize TouchAgent through supervised fine-tuning and reinforcement learning to improve tool use and evidence integration. Experiments show consistent improvements over existing MLLMs and tactile-specific models. We believe TouchBench and TouchAgent together lay a foundation for more general and reliable tactile intelligence systems.

REFERENCES

Alexander C Abad and Anuradha Ranasinghe. Visuotactile sensors with emphasis on gelsight sensor: A review. *IEEE Sensors Journal*, 20(14):7628–7638, 2020.

Imtiaz Ahmed, Sadman Islam, Partha Protim Datta, Imran Kabir, Md Naseef Ur Rahman Chowdhury, and Ahshanul Haque. Qwen 2.5: A comprehensive review of the leading resource-efficient llm with potentioal to surpass all competitors. 2025.

Anthropic. System card: Claude opus 4.5, November 2025a. URL https://assets.anthropic.com/m/64823ba7485345a7/Claude-Opus-4-5-System-Card.pdf. System card.

Anthropic. System card: Claude sonnet 4.5, September 2025b. URL https://assets.anthropic.com/m/12f214efcc2f457a/original/Claude-Sonnet-4-5-System-Card.pdf. System card.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025b. URL https://arxiv.org/abs/2502.13923.

Roberto Calandra, Andrew Owens, Manu Upadhyaya, Wenzhen Yuan, Justin Lin, Edward H Adelson, and Sergey Levine. The feeling of success: Does touch sensing help predict grasp outcomes? *arXiv preprint arXiv:1710.05512*, 2017.

Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, et al. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *International Conference on Learning Representations*, volume 2025, pp. 96934–96990, 2025.

Heather Culbertson, Juan José López Delgado, and Katherine J Kuchenbecker. One hundred data-driven haptic texture models and open-source methods for rendering on 3d objects. In *2014 IEEE haptics symposium (HAPTICS)*, pp. 319–325. IEEE, 2014.

Hao Fei, Yuan Zhou, Juncheng Li, Xiangtai Li, Qingshan Xu, Bobo Li, Shengqiong Wu, Yaoting Wang, Junbao Zhou, Jiahao Meng, et al. On path to multimodal generalist: General-level and general-bench. In *Forty-second International Conference on Machine Learning*, 2025.

Ruoxuan Feng, Jiangyu Hu, Wenke Xia, Tianci Gao, Ao Shen, Yuhao Sun, Bin Fang, and Di Hu. Anytouch: Learning unified static-dynamic representation across multiple visuo-tactile sensors. In *International Conference on Learning Representations*, volume 2025, pp. 31265–31285, 2025.

Tao Feng, Chuanyang Jin, Jingyu Liu, Kunlun Zhu, Haoqin Tu, Zirui Cheng, Guanyu Lin, and Jiaxuan You. How far are we from agi: Are llms all we need? *arXiv preprint arXiv:2405.10313*, 2024.

Chaoyou Fu, Yi-Fan Zhang, Shukang Yin, Bo Li, Xinyu Fang, Sirui Zhao, Haodong Duan, Xing Sun, Ziwei Liu, Liang Wang, et al. Mme-survey: A comprehensive survey on evaluation of multimodal llms. *arXiv preprint arXiv:2411.15296*, 2024a.

Letian Fu, Gaurav Datta, Huang Huang, William Chung-Ho Panitch, Jaimyn Drake, Joseph Ortiz, Mustafa Mukadam, Mike Lambeta, Roberto Calandra, and Ken Goldberg. A touch, vision, and language dataset for multimodal alignment. *arXiv preprint arXiv:2402.13232*, 2024b.

Ruohan Gao, Yiming Dou, Hao Li, Tanmay Agarwal, Jeannette Bohg, Yunzhu Li, Li Fei-Fei, and Jiajun Wu. The object folder benchmark: Multisensory learning with neural and real objects. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17276–17286. IEEE, 2023.

Google DeepMind. Gemini 3 flash: Frontier intelligence built for speed, December 2025. URL https://blog.google/products-and-platforms/products/gemini/gemini-3-flash/.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Carolina Higuera, Akash Sharma, Chaithanya Krishna Bodduluri, Taosha Fan, Patrick Lancaster, Mrinal Kalakrishnan, Michael Kaess, Byron Boots, Mike Lambeta, Tingfan Wu, et al. Sparsh: Self-supervised touch representations for vision-based tactile sensing. *arXiv preprint arXiv:2410.24090*, 2024.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Yixing Jiang, Kameron C Black, Gloria Geng, Danny Park, James Zou, Andrew Y Ng, and Jonathan H Chen. Medagentbench: a virtual ehr environment to benchmark medical llm agents. *Nejm Ai*, 2(9):AIdbp2500144, 2025.

Zhanat Kappassov, Juan-Antonio Corrales, and Véronique Perdereau. Tactile sensing in dexterous robot hands. *Robotics and Autonomous Systems*, 74:195–220, 2015.

Yingxin Lai, Yafei Zhou, Fucai Zhu, Siyu Zhu, and Weihao Yuan. Touch-r1: Reinforcing touch reasoning in mllms. *arXiv preprint arXiv:2605.27154*, 2026.

Binxu Li, Tiankai Yan, Yuanting Pan, Jie Luo, Ruiyang Ji, Jiayuan Ding, Zhe Xu, Shilong Liu, Haoyu Dong, Zihao Lin, et al. Mmedagent: Learning to use medical tools with multi-modal agent. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 8745–8760, 2024a.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024b.

Qiang Li, Oliver Kroemer, Zhe Su, Filipe Fernandes Veiga, Mohsen Kaboli, and Helge Joachim Ritter. A review of tactile information: Perception and action through touch. *IEEE Transactions on Robotics*, 36(6):1619–1634, 2020.

Yunzhu Li, Jun-Yan Zhu, Russ Tedrake, and Antonio Torralba. Connecting touch and vision via cross-modal prediction. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10601–10610. IEEE, 2019.

Kailin Lyu, Di Wu, Pengwei Zhang, Yuhang Zheng, Yingxin Lai, Long Xiao, Kangyi Wu, Pengna Li, Chen Gao, Lianyu Hu, et al. Touchthinker: Scaling tactile commonsense reasoning to the open world with large-scale data and action-aware representation. *arXiv preprint arXiv:2606.11637*, 2026a.

Kailin Lyu, Long Xiao, Jianing Zeng, Junhao Dong, Xuexin Liu, Zhuojun Zou, Haoyue Yang, Lin Shu, and Jie Hao. Touchformer: A robust transformer-based framework for multimodal material perception. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18496–18504, 2026b.

OpenAI. Update to GPT-5 system card: GPT-5.2, December 2025. URL https://openai.com/index/gpt-5-system-card-update-gpt-5-2/. System card.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Sudharshan Suresh, Zilin Si, Stuart Anderson, Michael Kaess, and Mustafa Mukadam. Midastouch: Monte-carlo inference over distributions across sliding touch. In *Conference on Robot Learning*, pp. 319–331. PMLR, 2023.

Sudharshan Suresh, Haozhi Qi, Tingfan Wu, Taosha Fan, Luis Pineda, Mike Lambeta, Jitendra Malik, Mrinal Kalakrishnan, Roberto Calandra, Michael Kaess, et al. Neuralfeels with neural fields: Visuotactile perception for in-hand manipulation. *Science Robotics*, 9(96):eadl0628, 2024.

Aoran Xiao, Shihao Cheng, Yonghao Xu, Yexian Ren, Hongruixuan Chen, and Naoto Yokoya. Geommbench and geommagent: Toward expert-level multimodal intelligence in geoscience and remote sensing. *arXiv preprint arXiv:2604.08896*, 2026.

Yifan Xie, Mingyang Li, Shoujie Li, Xingting Li, Guangyu Chen, Fei Ma, Fei Yu, and Wenbo Ding. Universal visuo-tactile video understanding for embodied interaction. *Advances in Neural Information Processing Systems*, 38:127864–127883, 2026.

Fengyu Yang, Chenyang Ma, Jiacheng Zhang, Jing Zhu, Wenzhen Yuan, and Andrew Owens. Touch and go: Learning from human-collected vision and touch. *arXiv preprint arXiv:2211.12498*, 2022.

Fengyu Yang, Chao Feng, Ziyang Chen, Hyoungseob Park, Daniel Wang, Yiming Dou, Ziyao Zeng, Xien Chen, Rit Gangopadhyay, Andrew Owens, et al. Binding touch to everything: Learning unified multimodal tactile representations. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26330–26343. IEEE, 2024a.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *Advances in neural information processing systems*, 37:21875–21911, 2024b.

Huanjin Yao, Jiaxing Huang, Yawen Qiu, Michael K Chen, Wenzheng Liu, Wei Zhang, Wenjie Zeng, Xikun Zhang, Jingyi Zhang, Yuxin Song, et al. Mmreason: An open-ended multi-modal multi-step reasoning benchmark for mllms toward agi. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 273–283. IEEE, 2025a.

Huanjin Yao, Ruifei Zhang, Jiaxing Huang, Jingyi Zhang, Yibo Wang, Bo Fang, Ruolin Zhu, Yongcheng Jing, Shunyu Liu, Guanbin Li, et al. A survey on agentic multimodal large language models. *arXiv preprint arXiv:2510.10991*, 2025b.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, et al. Minicpm-v: A gpt-4v level mllm on your phone. *arXiv preprint arXiv:2408.01800*, 2024.

Samson Yu, Kelvin Lin, Anxing Xiao, Jiafei Duan, and Harold Soh. Octopi: Object property reasoning with large tactile-language models. *arXiv preprint arXiv:2405.02794*, 2024.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9556–9567, 2024.

Boqiang Zhang, Kehan Li, Zesen Cheng, Zhiqiang Hu, Yuqian Yuan, Guanzheng Chen, Sicong Leng, Yuming Jiang, Hang Zhang, Xin Li, et al. Videollama 3: Frontier multimodal foundation models for image and video understanding. *arXiv preprint arXiv:2501.13106*, 2025.

Yan Zhang, WANG Zheng, Pengpeng Zeng, Xing Xu, Jingkuan Song, and Heng Tao Shen. Cross-tactile sensor representation learning. In *Forty-third International Conference on Machine Learning*.

Pengfei Zhou, Xiaopeng Peng, Fanrui Zhang, Zhaopan Xu, Jiaxin Ai, Yansheng Qiu, Chuanhao Li, Zhen Li, Ming Li, Yukang Feng, et al. Mdk12-bench: A comprehensive evaluation of multimodal large language models on multidisciplinary exams. *arXiv preprint arXiv:2508.06851*, 2025.

APPENDIX CONTENTS

A.1 Dataset Illustration 14
A.2 Benchmark Tasks 17
A.3 Toolkit Library 18
A.3.1 Perception Toolkit 19
A.3.2 Reasoning Toolkit 19
A.3.3 Knowledge Toolkit 20
A.4 Details of Training TouchAgent 22
A.4.1 TouchAgent-Instruct Construction 22
A.4.2 Evidence-Grounded Reward Design 22
A.5 Implementation Details 24
A.6 Qualitative Examples 24
A.7 Discussion 24

A APPENDIX

A.1 DATASET ILLUSTRATION

To construct a comprehensive benchmark for tactile understanding and reasoning, we curate data from 12 publicly available tactile datasets spanning diverse sensor configurations, interaction paradigms, objects, materials, and tactile properties. The collected data cover multiple visuotactile sensors, including the GelSight family, DIGIT, Tac3D, and DuraGel, as well as common interactions such as pressing, sliding, rotation, grasping, and in-hand manipulation. As illustrated in Figure 6, these heterogeneous sources exhibit substantial variation in sensing appearance, contact geometry, and deformation patterns across datasets and sensor domains. When official data splits are available, we preferentially sample from the test split to reduce potential overlap with existing training data and mitigate the associated risk of data leakage. Otherwise, evaluation samples are selected from publicly available data following a consistent curation protocol. By integrating these complementary sources, TouchBench reduces dependence on any particular sensor, interaction type, or data distribution, enabling a systematic evaluation of general-purpose tactile understanding and reasoning. We detail the composition of each source below.

- **VTV150K.** VTV150K (Xie et al., 2026) is a tactile video dataset for tactile understanding and reasoning, collected from 100 everyday objects using three visuotactile sensors: GelSight Mini, DIGIT, and Tac3D. It contains approximately 150K tactile frames spanning pressing, rotation, and sliding interactions, with three-level annotations for four tactile properties: hardness, bumpiness, elasticity, and friction. We exclusively sample from the official test split, retaining 7,500 tactile frames across 5 objects for TouchBench.
- **PhysiCLeAR.** PhysiCLeAR (Yu et al., 2024) targets tactile property understanding and physical reasoning. It contains 408 tactile videos and their corresponding object-view videos, collected from 74 everyday objects using a GelSight sensor. The dataset covers pressing and rotation interactions and provides three-level annotations for hardness, roughness, and bumpiness, together with object- and contact-region information. We sample exclusively from its test split, retaining 6,698 tactile frames across 5 objects.
- **The Feeling of Success.** The Feeling of Success (Calandra et al., 2017) is a visual-tactile dataset for robotic grasp outcome prediction. It comprises 9,269 grasp trials over 106 objects, collected using two GelSight sensors mounted on a parallel-jaw gripper and an external RGB-D camera. Each trial captures visual and tactile observations throughout gripper approach, closure, object lifting, and holding, with a binary label indicating grasp success or failure. We select 11,122 tactile frames across 20 objects, introducing dynamic tactile observations associated with robotic grasping and contact stability.
- **TacBench.** TacBench (Higuera et al., 2024) captures diverse contact states and tactile motion using DIGIT and GelSight Mini sensors. Its interactions include robotic loading, sensor-probe contact, and robot finger-object interaction, covering normal and tangential forces, slip, relative

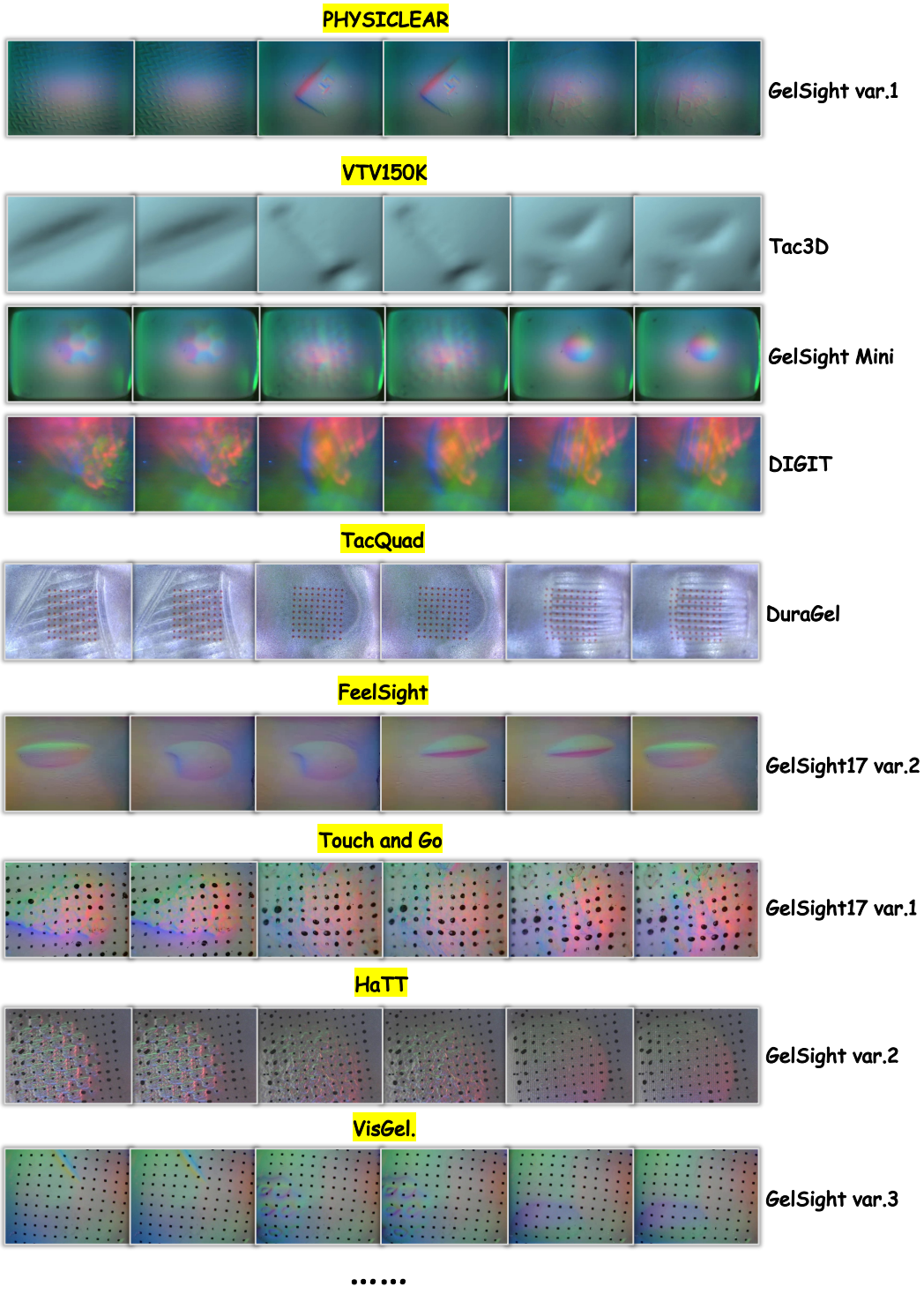


Figure 6: **Representative tactile observations in TouchBench.** We visualize tactile samples from a subset of the source datasets and sensor configurations. The examples highlight substantial variation in sensing appearance, contact geometry, and deformation patterns across datasets and tactile sensor domains, illustrating the diversity of tactile observations incorporated into TouchBench.

translation, and rotation under varying contact geometries. The tactile sequences capture temporal changes in contact regions, local deformation, and relative motion, with annotations including 3-axis contact forces, slip states, relative translation, and yaw changes in the sensor coordinate

frame. We retain 216,945 tactile frames, making TacBench a major source of dynamic contact observations in TouchBench.

- **Touch-Slide.** Touch-Slide (Higuera et al., 2024) focuses on continuous tactile sliding and uses a DIGIT sensor to record human-driven interactions between the sensor and object surfaces. The dataset contains 9 toy kitchen objects, with 5 sliding trajectories per object, capturing sustained contact, relative motion, and local deformation during surface exploration. We retain 4,050 tactile frames from 3 objects.
- **TacQuad.** TacQuad (Feng et al., 2025) is an aligned multimodal, multi-sensor visuotactile dataset collected using GelSight Mini, DIGIT, DuraGel, and Tac3D. Calibration-platform and handheld acquisition enable different sensors to observe corresponding contact locations on the same objects, providing both fine-grained spatiotemporally aligned contact observations and spatially aligned observations over broader object and scene distributions. The dataset additionally provides object images and tactile property descriptions across diverse materials, textures, hardness levels, and contact deformations. We sample exclusively from its test split, retaining 9,070 tactile frames across 11 objects. Its cross-sensor alignment contributes to evaluating robustness under sensor-domain variation.
- **Touch and Go.** Touch and Go (Yang et al., 2022) is a large-scale visual-tactile dataset collected in natural human environments using a GelSight sensor synchronized with an RGB camera. Data are acquired through pressing interactions across diverse indoor and outdoor environments, including classrooms, hallways, apartments, streets, trails, and playgrounds, covering both rigid and deformable objects with substantial variation in material, roughness, and hardness. The original dataset contains approximately 246K visual-tactile frame pairs, 13.9K contact interactions, and 3,971 object instances, with contact frames annotated using 20 material categories. We retain 14,100 tactile frames spanning 18 material categories, broadening TouchBench to natural environments and diverse material distributions.
- **YCB-Slide.** YCB-Slide (Suresh et al., 2023) records continuous tactile observations as a DIGIT sensor slides over standard YCB objects. It contains 10 everyday objects with diverse local geometries and provides both real-world and TACTO-based simulated data. For real-world acquisition, objects are fixed in a vise while a handheld DIGIT sensor slides across their surfaces, with sensor poses synchronously tracked using OptiTrack. Each object contains 5 trajectories, yielding 50 real-world sliding trajectories in total. The simulated subset additionally provides sensor poses, object meshes, local height maps, and contact masks. TouchBench uses only the real-world test data, retaining 2,100 tactile frames across 2 objects.
- **FeelSight.** FeelSight (Suresh et al., 2024) captures synchronized visual, tactile, and proprioceptive observations during robotic in-hand object rotation using an Allegro Hand with four DIGIT sensors and an RGB-D camera. The dataset contains approximately 30-second trajectories for 6 real-world and 8 simulated objects, totaling 70 trials and about 35 minutes of interaction, with object geometry and pose annotations available in both real and simulated settings. We retain 4,200 tactile frames from 1 real-world object, capturing tactile dynamics induced by continuous rotation and multi-finger contact.
- **ObjectFolder Real.** ObjectFolder Real (Gao et al., 2023) is a dataset of 100 real-world objects with visual, acoustic, and tactile observations, together with high-fidelity 3D geometry. Tactile data are collected using a GelSight R1.5 sensor mounted on a Franka Panda robot through normal contact at predefined surface locations, spatially aligned with acoustic and visual observations. We retain 91,530 tactile frames and corresponding visual observations from 9 objects.
- **VisGel.** VisGel (Li et al., 2019) is a large-scale paired visual-tactile dataset collected using a robotic arm equipped with a GelSight sensor and an external camera. Each interaction captures the continuous process of approaching, contacting, and leaving an object, with visual and tactile streams temporally aligned by timestamps. The original dataset contains more than 12K contact interactions and approximately 3M visual-tactile frames, spanning diverse objects and surface deformations. We retain 8,950 tactile frames across 4 objects, providing temporally aligned visual context and local tactile observations.
- **HaTT.** Unlike the predominantly image-based visuotactile datasets above, HaTT (Culbertson et al., 2014) captures tactile texture through physical signals. It contains 100 surface textures spanning 10 material categories: paper, plastic, fabric, tile, carpet, foam, metal, stone, carbon fiber, and wood. A custom handheld device with a stainless-steel hemispherical probe records 3-axis vibration acceleration, contact force, and position during natural sliding, together with motion velocity and normal and tangential forces. Each texture contains two 10-second interaction sequences sampled at 10 kHz, accompanied by surface images and texture and friction models.

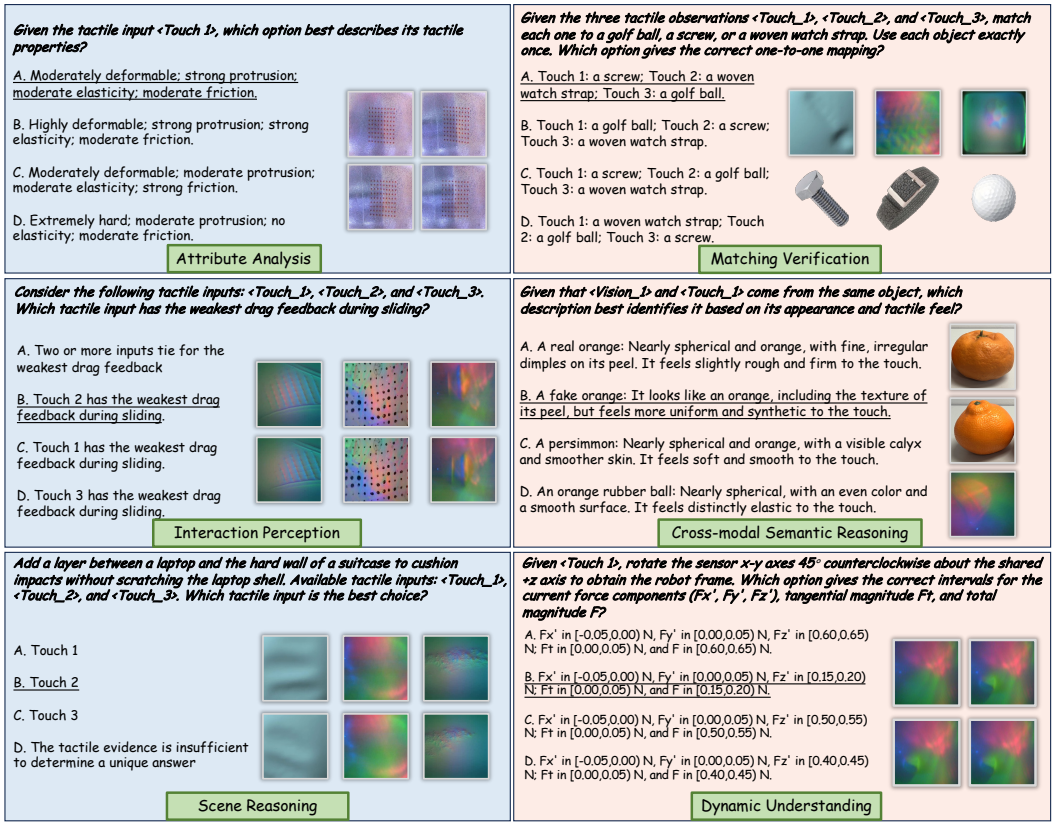


Figure 7: **Representative examples from TouchBench,** spanning six task categories and diverse tactile reasoning scenarios. Solving these tasks requires comprehensive tactile understanding, evidence-grounded multi-step reasoning, and domain knowledge.

We retain 7,362 tactile frames spanning 9 surface textures. HaTT extends TouchBench beyond deformation-based visuotactile observations to vibration- and force-based tactile signals, increasing the diversity of tactile sensing modalities represented in the benchmark.

A.2 Benchmark Tasks

TouchBench comprises 10K questions organized into six complementary task categories: Attribute Analysis, Matching Verification, Dynamic Understanding, Cross-Modal Semantic Reasoning, Scene Reasoning, and Interaction Perception. These six categories are further decomposed into 16 distinct subtask types, providing fine-grained coverage of tactile perception, understanding, and reasoning. Representative examples are shown in Figure 7. Together, these tasks span multiple levels of tactile understanding, from physical property perception and temporal interaction understanding to object-level semantic correspondence, cross-modal association, and situated physical reasoning. They require models to extract physical properties, contact states, and temporal dynamics from tactile observations and, when applicable, integrate visual or linguistic context to perform classification, comparison, matching, verification, and decision-making.

- **Attribute Analysis.** This category comprises 1,872 questions, accounting for approximately 18.7% of TouchBench, and evaluates the ability to identify, characterize, and compare fundamental physical properties from tactile observations. It covers four core properties: hardness, bumpiness, elasticity, and surface friction. *Single-Attribute Classification* requires predicting the discrete level of a specified property; *Joint Attribute Profiling* requires jointly identifying all four properties; *Pairwise Attribute Comparison* evaluates the relative level of a specified property between two tactile observations; and *Attribute Extremum Selection* requires identifying the observation with the highest or lowest value of a target property among multiple candidates. These

subtasks progressively extend tactile property understanding from individual recognition to joint characterization, relative comparison, and multi-candidate selection.

- **Matching Verification.** This category contains 2,102 questions, representing approximately 21.0% of the benchmark, and evaluates whether models can establish consistent correspondences between tactile observations and object-level semantics. It comprises two subtasks. *Object Matching* provides multiple tactile observations and a set of candidate object descriptions, requiring a globally consistent one-to-one assignment rather than independent classification of individual observations. *Part-Claim Verification* provides tactile evidence together with a claim about the contacted object part and requires determining whether the claim is supported by the observation. These subtasks assess object-level and part-level semantic grounding from local tactile evidence.
- **Dynamic Understanding.** This category consists of 2,082 questions, accounting for approximately 20.8% of TouchBench, and evaluates temporal understanding and reasoning over continuous contact interactions. *Relative Motion Reasoning* requires inferring planar displacement, motion direction, and yaw changes from sequential tactile observations. *Force Reasoning* requires estimating three-axis contact forces and, given coordinate-frame relationships, deriving forces and associated quantities in a target coordinate frame. *Slip-State Reasoning* requires identifying slip from temporal variations in contact deformation and force. *Grasp-Success Prediction* determines the outcome of a grasp from paired tactile observations acquired during the same trial. Together, these subtasks emphasize motion, force, and contact-state information encoded in the temporal evolution of tactile signals.
- **Cross-Modal Semantic Reasoning.** This category includes 1,782 questions, corresponding to approximately 17.8% of the benchmark, and evaluates object-level semantic correspondence and authenticity through two subtasks. *Visual-Guided Tactile Matching*: each question provides a visual observation together with multiple tactile candidates and requires identifying the tactile observation corresponding to the visual object. The visual input specifies the target entity and its observable appearance, whereas the tactile candidates encode complementary cues about local geometry, material properties, and contact characteristics. *Authenticity Verification*: each question provides visual and tactile observations from the same object and requires integrating appearance cues with tactile physical properties to distinguish genuine objects from visually similar imitations. Together, these subtasks require integrating complementary visual and tactile evidence to establish cross-modal object correspondence and assess object authenticity.
- **Scene Reasoning.** This category comprises 1,161 questions, accounting for approximately 11.6% of TouchBench, and evaluates context-dependent physical reasoning through *Scenario-Based Tactile Selection*. Each question describes an operational goal or usage scenario in natural language and provides multiple tactile candidates. The model must reason about physical characteristics such as hardness, friction, elasticity, and surface structure to identify the candidate that best satisfies the specified requirements. The scenarios span functional settings such as cushioning, slip resistance, and surface interaction, extending evaluation beyond direct property recognition to situated physical reasoning and decision-making.
- **Interaction Perception.** This category contains 1,001 questions, representing approximately 10.0% of the benchmark, and evaluates interaction-relevant properties inferred from tactile observations, including susceptibility to pressure-induced damage, texture distinguishability, contact comfort, scratching or puncture risk, grasp stability, and drag feedback during sliding. *Single-Property Assessment* predicts the behavior of an individual tactile observation along a specified interaction dimension; *Pairwise Interaction Comparison* determines the relative behavior of two observations; and *Interaction Extremum Selection* identifies the candidate exhibiting the strongest or weakest target property. These subtasks assess the ability to map local tactile evidence to functional characteristics that emerge during physical interaction.

A.3 TOOLKIT LIBRARY

We provide a detailed description of the tools integrated into TouchAgent. As described below, the tool library is organized into three categories: **Perception Toolkit, Reasoning Toolkit, and Knowledge Toolkit**. Together, these categories comprise a total of eight specialized tools designed to support tactile perception, reasoning, and knowledge-enhanced understanding.

A.3.1 PERCEPTION TOOLKIT

The Perception Toolkit provides a set of tactile and visual perception tools for tactile attribute analysis, dynamic state estimation, and object category and appearance recognition. Tactile attribute analysis covers hardness, protrusion, elasticity, and friction, while dynamic state analysis includes grasp outcome prediction, contact force estimation, relative motion estimation, and slip analysis. In addition, the toolkit supports tactile keyframe selection and relative depth estimation to capture contact responses, temporal variations, and contact deformation. Together, these tools transform multimodal inputs into interpretable perceptual evidence. More details are provided in Table 4.

Table 4: List of detailed information of Perception Toolkit.

Tool Name	**Category**	**Description Summary**
SelectFrames	Perception Tools	SelectFrames is a tactile keyframe selection tool. Given a directory of tactile frames and a selection strategy, it identifies frames with pronounced temporal changes using RGB differences between consecutive frames, and frames with strong contact responses using a combined response derived from differences relative to the initial background and image structural features such as Sobel gradients. The two strategies provide tactile evidence of dynamic changes and relatively stable contact states, respectively.
EstimateDepth	Perception Tools	EstimateDepth is a tactile relative depth estimation tool. Given tactile observations, it predicts depth using a pretrained Depth Anything V2 Small model (Yang et al., 2024b) and normalizes each image's predictions individually to obtain a relative depth representation, providing depth cues for contact deformation analysis.
AnalyzeAttributes	Perception Tools	AnalyzeAttributes is a tactile attribute analysis tool built upon a CLIP-based model trained on large-scale tactile attribute annotations (Yu et al., 2024; Xie et al., 2026; Lyu et al., 2026a). Given tactile observations and their enhanced frames, it estimates four physical attributes: hardness, protrusion, elasticity, and friction, and returns the predicted attribute levels together with their associated confidence scores.
EstimateState	Perception Tools	EstimateState is a tactile dynamic state analysis tool. It is built upon Sparsh-based downstream predictors for dynamic tactile state estimation (Higuera et al., 2024). Given tactile observations, it supports grasp outcome prediction, three-axis contact force estimation, relative translation and yaw change estimation, and slip analysis. Corresponding outputs include grasp success or failure predictions, contact force components in the sensor coordinate frame, predicted intervals and associated scores for translational displacement and yaw change, and slip probabilities over consecutive temporal windows together with contact force changes in the current window.
AnalyzeVisuals	Perception Tools	AnalyzeVisuals is a visual analysis tool (Bai et al., 2025b). Given a visual image and a text prompt, such as "What object is shown in the image, and what are its appearance characteristics?", it identifies entities or object categories and extracts visible appearance information, including shape, color, and surface texture. It returns the corresponding identification results and appearance descriptions.

A.3.2 REASONING TOOLKIT

The Reasoning Toolkit provides tactile semantic understanding and programmatic computation capabilities, covering tactile description, object and part matching, and numerical analysis based on physical relationships. It also supports code generation and execution for local image processing

and quantitative computation, translating tactile evidence into semantic judgments and computational results. Detailed tool descriptions are provided in Table 5.

Table 5: List of detailed information of Reasoning Toolkit.

Tool Name	Category	Description Summary
InterpretTactile	Reasoning Tools	InterpretTactile is a tactile semantic reasoning tool built upon TouchThinker (Lyu et al., 2026a), which is pretrained on large-scale tactile data to provide robust and generalizable tactile representations. Given tactile observations, their enhanced frames, and a text prompt, such as "Which object most likely corresponds to this tactile image?", the tool generates natural-language descriptions of tactile properties, establishes one-to-one correspondences between tactile inputs and candidate objects, or identifies object parts that are consistent with the observed tactile evidence.
RunCode	Reasoning Tools	RunCode is a code generation and execution tool. Given a task specification, it generates and executes programs for local cropping and magnification of tactile images and numerical computation based on physical formulas. Supported computations include contact force coordinate transformations, relative motion calculations, and slip-related metric calculations. The tool returns the processed local images or numerical results.

A.3.3 KNOWLEDGE TOOLKIT

The Knowledge Toolkit provides background knowledge and physical principles for tactile understanding, covering object–contact properties, interaction requirements, scenario-specific attribute requirements, and physical relationships governing dynamic processes. These knowledge sources support the interpretation of perceptual results, the assessment of compatibility between physical attributes and scenario requirements, and physics-based computations. We construct a tactile knowledge base assisted by GPT-5.2 (OpenAI, 2025) and further validated by five experts with backgrounds in tactile perception and physical interaction. The knowledge base contains 2,000 manually curated tactile knowledge entries spanning four major dimensions:

- **Tactile property–function relationships**, including hardness, protrusion, elasticity, and friction.
- **Object–contact interaction requirements**, covering interactions such as grasping, pressing, sliding, and protection.
- **Scenario-specific physical constraints** for daily manipulation tasks.
- **Dynamic interaction principles** involving force, motion, and contact-state changes.

Representative knowledge entries are shown in Figure 8. These knowledge sources establish a bridge between tactile perception and higher-level physical reasoning and scenario-based decision-making, enabling TouchAgent to retrieve and leverage relevant physical priors through the subsequent hierarchical retrieval process for more reliable tactile understanding. Detailed information about the KnowledgeSearch mechanism is provided in Table 6.

Table 6: List of detailed information of Knowledge Toolkit.

Tool Name	Category	Description Summary
KnowledgeSearch	Knowledge Tools	KnowledgeSearch is a hierarchical knowledge retrieval and reasoning tool that provides task-relevant tactile knowledge, object priors, physical principles, and attribute constraints. Given a natural-language knowledge query, it first performs exact matching over the curated knowledge base to retrieve directly relevant entries. If exact matches are insufficient, it further applies semantic vector retrieval using Qwen3-Embedding (Bai et al., 2025a) to obtain top-5 candidate knowledge entries based on query–knowledge similarity. Finally, when the retrieved knowledge is incomplete or additional physical interpretation is required, KnowledgeSearch employs analogy-based reasoning with GPT-5.2 (OpenAI, 2025) to complete missing relations, derive implicit constraints, and synthesize the retrieved knowledge into coherent outputs. The final results are returned as natural-language descriptions, physical formulas, or structured constraints, which support tactile evidence interpretation, attribute reasoning, scenario-based decision-making, and physics-based computation.

Examples of Tactile Knowledge Entries

K_PROP_0001: Soft and elastic materials reduce discomfort under sustained pressure. When a contact region needs to withstand prolonged human pressure while reducing localized discomfort, contact materials with soft or moderately deformable properties, low protrusion, medium or high elasticity, and low or medium friction should be preferred. Rigid, inelastic, highly protruded, or high-friction contact conditions should be avoided, as they may cause pressure concentration or continuous friction. This knowledge applies to human-contact cushioning scenarios and should not be generalized to structural support or high-load-bearing requirements.

K_PROP_0002: Higher friction improves contact stability. When a task requires increased stability between an object and the contacting surface, surfaces with medium or high friction should be preferred. Smooth and low-friction surfaces should be avoided because they may increase the risk of relative slipping during contact. This knowledge applies to grasping, anti-slip, and stable placement scenarios, but does not apply to operations requiring low-resistance sliding.

K_PROP_0003: Surface protrusions enhance tactile discriminability. When users need to distinguish objects or locate interaction regions through touch, surfaces with medium or high protrusion structures should be preferred. Completely smooth or low-protrusion surfaces should be avoided because they provide weaker local geometric feedback. This knowledge applies to tactile marking and touch-based object identification scenarios.

K_OBJ_0001: Increasing friction assists smooth bottle-cap rotation. When a bottle cap or container lid is difficult to rotate due to a smooth surface, contact materials with soft or moderately deformable properties, medium or high protrusion, medium or high elasticity, and medium or high friction should be preferred. Rigid, inelastic, or low-friction contact conditions should be avoided because they provide insufficient rotational force transmission. This knowledge applies to temporary bottle-cap opening assistance and should not be generalized to other manipulation scenarios.

K_OBJ_0002: Oil-contaminated surfaces require enhanced grasp stability. When objects become difficult to grasp due to oil or liquid contamination, contact surfaces with soft or moderately deformable properties, medium protrusion, medium or high friction, and moderate elasticity should be preferred. Smooth low-friction and rigid inelastic surfaces should be avoided because they increase the risk of grasp failure. This knowledge applies to wet or slippery object manipulation scenarios.

K_SCENE_0001: Cushioning materials reduce shoulder strap pressure. When shoulder straps cause localized discomfort during prolonged wearing, materials with soft deformation capability, low protrusion, medium elasticity, and low or medium friction should be preferred. Rigid, highly protruded, or inelastic materials should be avoided because they increase localized pressure concentration. This knowledge applies to daily carrying scenarios and does not represent medical or ergonomic diagnosis.

K_SCENE_0002: Protective materials reduce impact from hard corners. When protecting users from accidental contact with sharp furniture or device corners, materials that provide deformation, impact absorption, and sufficient surface friction should be preferred. Rigid, inelastic, or sharp protruded materials

should be avoided because they cannot effectively reduce impact during contact. This knowledge applies to lightweight daily protection scenarios.

K_SCENE_0003: Anti-slip pads improve object placement stability. When objects need to remain stable on a supporting surface, contact materials with sufficient friction, moderate deformation, and elastic recovery should be preferred. Smooth low-friction and rigid non-deformable materials should be avoided because they provide limited resistance against relative motion. This knowledge applies to stable placement and vibration reduction scenarios.

K_DYNAMIC_0001: Slip estimation requires temporal contact evidence. During continuous tactile interaction, slip state estimation should jointly consider contact-force variations, temporal consistency, and slip-related indicators rather than relying on a single tactile observation. This knowledge applies to dynamic tactile reasoning involving relative motion and contact-state changes.

K_DYNAMIC_0002: Contact-force changes reflect interaction dynamics. Changes in normal and tangential contact forces provide information about pressure variation, relative motion, and interaction stability. Tangential force changes are particularly relevant for identifying sliding tendencies, whereas normal force changes mainly reflect loading variation. This knowledge applies to force-based tactile interpretation during dynamic interaction.

Figure 8: Examples of tactile knowledge entries used in the Knowledge Toolkit.

A.4 Details of Training TouchAgent

A.4.1 TouchAgent-Instruct Construction

To equip TouchAgent with reliable tool planning and evidence integration capabilities, we construct **TouchAgent-Instruct**, a multi-turn instruction dataset following a unified *Thoughts–Actions–Value* interaction protocol. Given a task instruction, candidate options, and the required tactile or visual inputs, TouchAgent iteratively interacts with the tool environment, with each turn producing three structured fields:

- *Thoughts*: identifies the evidence required by the current task based on the question and accumulated Observations, assesses whether the available evidence is sufficient, and determines the next reasoning step.
- *Actions*: specifies the next tool to invoke together with its required semantic parameters. An empty *Actions* field indicates that no further tool invocation is required and terminates the interaction.
- *Value*: states the objective of the current tool invocation when *Actions* is non-empty; upon termination, it integrates the accumulated Observations to produce the final explanation and answer.

As illustrated in Fig. 17, we construct TouchAgent-Instruct through one-shot trajectory synthesis. For each task type, we predefine a task-specific reference trajectory that contains a representative task instance, its candidate options, and the complete sequence of intermediate interaction steps, together with a corresponding system prompt specifying the task objective, available tools, evidence requirements, and interaction protocol. The reference trajectory serves as an in-context demonstration for synthesizing trajectories for new training instances. Conditioned on the system prompt, reference trajectory, and current training instance, GPT-5.2 (OpenAI, 2025) generates one structured interaction turn at a time. Whenever a tool call is produced, the corresponding tool is executed, and the returned Observation is appended to the interaction history before the next turn is generated. This process continues until *Actions* is empty, yielding a complete trajectory that records tool planning, tool invocation, intermediate Observations, evidence integration, and the final response. Following this procedure, we construct approximately 40K multi-turn interaction trajectories covering tool use and evidence integration.

A.4.2 Evidence-Grounded Reward Design

We therefore GRPO with four complementary rewards that jointly supervise the tool-interaction trajectory, final task outcome, interaction format, and evidence grounding. This design encourages TouchAgent not only to produce the correct answer, but also to obtain task-relevant evidence through appropriate tool use and base its final conclusion on that evidence.

Result Reward. Although TouchAgent may require multiple rounds of tool interaction to acquire tactile evidence, the final objective of each TouchBench instance is to select the correct candidate. Since all TouchBench questions follow a four-option multiple-choice format, we define the result reward by exact matching between the predicted and ground-truth options. Let $y \in \{1, 2, 3, 4\}$

denote the correct option and $\hat{y}$ the option parsed from the terminated response. We define

$$R_{\text{result}}(\hat{y}, y) = \begin{cases} 1, & \hat{y} = y, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

If no valid option can be parsed from the final response, the reward is set to zero.

Trajectory Reward. Many tactile tasks require evidence to be acquired in a task-dependent order. For example, solving a dynamic reasoning problem may require first extracting the relevant tactile state, then retrieving the corresponding physical relation, and finally performing numerical computation. Final-answer supervision alone cannot distinguish whether the agent follows the required evidence acquisition process. We therefore define a canonical tool route for each training instance. Let $\mathcal{A}^* = (a_1^*, a_2^*, \ldots, a_L^*)$ denote the reference route of length L, where each action specifies the tool name and normalized semantic parameters, and let $\hat{\mathcal{A}} = (\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_{\hat{L}})$ denote the generated tool sequence. The trajectory reward is

$$R_{\text{trajectory}} = \begin{cases} 1, & \hat{L} = L,\ \hat{a}_t \equiv a_t^*\ \forall t \in \{1, \ldots, L\},\ \text{and } \hat{a}_{L+1} = \varnothing, \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

where $\equiv$ denotes a match in tool identity and normalized semantic parameters. The reward is therefore zero for incorrect tools or parameters, incorrect invocation order, premature termination, or additional tool calls after completion of the reference route.

Format Reward. TouchAgent follows a unified *Thoughts–Actions–Value* protocol that provides a consistent interface for reasoning, tool invocation, and termination. Format drift may make tool calls unparsable or non-executable and consequently prevent reliable evaluation of the interaction trajectory. Let o_t denote the structured output at turn t, and let $\mathcal{F}$ denote the set of valid outputs satisfying the protocol: all required fields are present and parseable, non-empty *Actions* contain a valid tool name and valid semantic parameters, and termination is represented by an empty *Actions* field. We define

$$R_{\text{format}} = \mathbb{1}[o_t \in \mathcal{F},\ \forall t \in \{1, \ldots, T\}]\,. \tag{7}$$

We use a binary reward rather than partial credit because malformed or non-executable outputs cannot reliably support subsequent tool interaction or yield a valid reasoning trajectory.

Evidence-Grounding Reward. To evaluate whether the final reasoning faithfully integrates evidence acquired during multi-turn interaction, we employ GPT-5.2 (OpenAI, 2025) as a judge to evaluate the complete interaction history, including the structured outputs generated by TouchAgent and the Observations returned by executed tools. The judge evaluates two complementary aspects: *evidence-integration consistency*, which measures whether the final *Thoughts* and *Value* correctly interpret and integrate the acquired tactile, visual, and knowledge evidence without contradiction; and *conclusion support*, which measures whether the final answer and explanation are sufficiently supported by evidence present in the trajectory rather than unsupported priors or speculation. The judge assigns a score $s_{\text{faith}} \in \{0, 1, 2, 3, 4\}$, which is normalized as

$$R_{\text{faith}} = \begin{cases} \dfrac{s_{\text{faith}}}{4}, & \text{if the trajectory terminates with a valid answer,} \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

The complete judge prompt and scoring rubric are provided in Figure 18.

Overall Reward. The final GRPO reward combines the four objectives:

$$R_{\text{total}} = \lambda_{\text{trajectory}} R_{\text{trajectory}} + \lambda_{\text{result}} R_{\text{result}} + \lambda_{\text{format}} R_{\text{format}} + \lambda_{\text{faith}} R_{\text{faith}}, \tag{9}$$

with

$$(\lambda_{\text{trajectory}}, \lambda_{\text{result}}, \lambda_{\text{format}}, \lambda_{\text{faith}}) = (0.1,\ 0.2,\ 0.3,\ 0.4). \tag{10}$$

Starting from the trajectory-supervised TouchAgent policy, we optimize the model with standard GRPO (Guo et al., 2025):

$$\theta^* = \arg\max_{\theta} \mathcal{J}_{\text{GRPO}}(\theta; R_{\text{total}})\,. \tag{11}$$

The resulting objective jointly encourages correct task completion, executable multi-turn interaction, task-dependent evidence acquisition, and evidence-grounded final reasoning.

Table 7: Training configurations for the two-stage training paradigm. Stage I: Trajectory-Supervised Fine-Tuning. Stage II: Evidence-Grounded Reinforcement Learning.

Configuration	**Stage I**	**Stage II**
Training data	32k Instruct data	8k Instruct data
Optimizer	AdamW	AdamW
Learning rate	2×10^{-4}	2×10^{-4}
Weight decay	0.01	0.01
Training epochs	1	1
Warmup ratio	0.03	0.03
Learning rate scheduler	Cosine	Cosine
Batch size	16	16
Maximum token length	2,048	2,048

A.5 Implementation Details

TouchAgent is trained following the two-stage training paradigm described above. In Stage I, we perform trajectory-supervised fine-tuning (SFT) on 32K TouchAgent-Instruct trajectories to equip the model with structured tool-use and evidence-integration capabilities. In Stage II, we further optimize the model on the remaining 8K instruction data using evidence-grounded reinforcement learning with GRPO. Both stages use AdamW with a learning rate of 2×10^{-4}, a weight decay of 0.01, and a cosine learning-rate schedule with a warmup ratio of 0.03. We train each stage for one epoch with a batch size of 16 and a maximum sequence length of 2,048 tokens. Detailed training configurations and hyperparameters for both stages are summarized in Table 7.

A.6 Qualitative Examples

To provide an intuitive understanding of how TouchAgent performs multi-step tool interaction and evidence-grounded reasoning, we present representative successful and failed trajectories. The successful cases demonstrate how TouchAgent progressively acquires task-relevant evidence through specialized tools and integrates intermediate observations to derive the correct answer (Figures 9–12), while the failed cases reveal representative error patterns (Figures 13–16, where red text marks the sources and causes of errors).

A.7 Discussion

Impact. TouchBench and TouchAgent establish a systematic evaluation and tool-augmented reasoning framework for real-world tactile intelligence. Unlike existing tactile systems that primarily focus on local state perception or single-step reasoning (Higuera et al., 2024; Yang et al., 2024a; Lyu et al., 2026b; Yu et al., 2024; Lyu et al., 2026a), TouchAgent integrates tactile-specific perception models, domain knowledge, and numerical reasoning tools into an interactive agent framework, enabling task-driven evidence acquisition and multi-stage reasoning. Our results demonstrate the potential of tool-augmented, evidence-grounded agents to bridge the capability gaps of existing MLLMs and tactile-specific models in complex tactile understanding, providing a promising path toward using large models as high-level reasoning and decision-making modules for tactile intelligence.

Limitations. Our work has several limitations. First, TouchAgent focuses primarily on tactile understanding and reasoning rather than closed-loop robotic control, and has not yet been deployed on physical robotic platforms. Its robustness to execution errors, continuous feedback, and real-time control therefore remains to be investigated. Second, the supervision is constrained by the diversity of annotated tactile attributes. Our current attribute space covers common and complementary properties, including hardness, protrusion, elasticity, and friction, whereas real-world tactile perception involves richer properties such as malleability and prickliness. This may limit comprehensive tactile understanding in open-world scenarios.

Future Work. A promising direction is to integrate TouchAgent with tactile VLA models, using it as a high-level reasoning and decision-making module to guide robotic actions and ultimately enable closed-loop tactile interaction. Meanwhile, deploying TouchAgent on resource-constrained robotic platforms calls for lightweight and accelerated variants. Expanding the tactile attribute space and incorporating more open-world interaction data are also important directions toward more general, efficient, and reliable tactile intelligence.

Case 1: Successful Trajectory for Dynamic Understanding

User Query. Given tactile input Touch 1: <touch_1>. Which option most accurately describes the overall planar displacement characteristics and yaw state between the two tactile observations?

- **A.** There is a small negative yaw. The planar displacement magnitude is approximately 3.51 mm, and the magnitude ratio of the two axial displacement components is approximately 0.09; the overall displacement is large, and one axial component is dominant.
- **B.** There is a slight positive yaw. The planar displacement magnitude is approximately 3.51 mm, and the magnitude ratio of the two axial displacement components is approximately 0.09; the overall displacement is large, and one axial component is dominant.
- **C.** There is a slight positive yaw. The planar displacement magnitude is approximately 3.81 mm, and the magnitude ratio of the two axial displacement components is approximately 0.43; the overall displacement is large, and one axial component is dominant.
- **D.** There is a small negative yaw. The planar displacement magnitude is approximately 3.81 mm, and the magnitude ratio of the two axial displacement components is approximately 0.43; the overall displacement is large, and one axial component is dominant.

Thoughts 1. The task requires the overall planar displacement characteristics and yaw state between two tactile observations. I should first estimate the relative translation and yaw change before computing the derived motion quantities.

Actions 1.

```
[{"tool_name":"EstimateState","tool_args":{"task":"relative_motion"}}]
```

Value 1. I will use EstimateState to obtain the predicted x-axis displacement, y-axis displacement, and yaw-change intervals for Touch 1.

Observation 1. The predicted relative-motion intervals are

$$t_x \in [-5, -2)\ \text{mm}, \qquad t_y \in [1, 2)\ \text{mm},$$

$$\text{yaw} \in [-0.5, -0.2)^\circ.$$

The corresponding model scores are 0.9999, 0.9902, and 1.0000, respectively.

Thoughts 2. The three predicted intervals are available, and the yaw interval can be interpreted directly. I still need the mathematical relations for the finite-interval midpoints, planar displacement magnitude, and axial displacement-component magnitude ratio.

Actions 2.

```
[{"tool_name":"KnowledgeSearch","tool_args":{}}]
```

Value 2. I will use KnowledgeSearch to retrieve the formulas for finite-interval midpoints, planar displacement magnitude, and the axial displacement-component magnitude ratio.

Observation 2. KnowledgeSearch returns

$$m = \frac{a+b}{2},$$

$$t_x^{\text{mid}} = \frac{t_x^{\text{low}} + t_x^{\text{high}}}{2}, \qquad t_y^{\text{mid}} = \frac{t_y^{\text{low}} + t_y^{\text{high}}}{2},$$

$$r_{xy} = \sqrt{\left(t_x^{\text{mid}}\right)^2 + \left(t_y^{\text{mid}}\right)^2},$$

$$b_{xy} = \frac{\min\left(|t_x^{\text{mid}}|, |t_y^{\text{mid}}|\right)}{\max\left(|t_x^{\text{mid}}|, |t_y^{\text{mid}}|\right)}.$$

Thoughts 3. The finite translation intervals and all required relations are now available. I should compute the interval midpoints, planar displacement magnitude, and axial displacement-component magnitude ratio.

Actions 3.

```
[{"tool_name":"RunCode","tool_args":{"task":"relative_motion_calculat⌋
↪  ion"}}]
```

Value 3. I will use RunCode to compute both interval midpoints, the planar displacement magnitude, and the axial displacement-component magnitude ratio.

Observation 3. The calculation yields

$$t_x^{\text{mid}} = -3.50 \text{ mm}, \qquad t_y^{\text{mid}} = +1.50 \text{ mm},$$

$$r_{xy} = 3.81 \text{ mm}, \qquad b_{xy} = 0.43.$$

Thoughts 4. The yaw interval $[-0.5, -0.2)^\circ$ indicates a small negative yaw. The planar displacement magnitude is 3.81 mm, which is not less than 3.0 mm and therefore indicates a large displacement. The axial displacement-component magnitude ratio is $0.43 < 0.50$, indicating that one axial component dominates. The accumulated evidence is sufficient to answer the question.

Actions 4.

```
[]
```

Value 4. The motion exhibits a small negative yaw, a planar displacement magnitude of approximately 3.81 mm, and an axial displacement-component magnitude ratio of approximately 0.43. The overall displacement is large and dominated by one axial component. Option A is inconsistent with the computed displacement magnitude and component ratio. Option B is inconsistent with the yaw, displacement magnitude, and component ratio. Option C is inconsistent with the negative yaw. Only **Option D** is consistent with all acquired and computed evidence. Therefore, the answer is **D**.

Figure 9: A successful TouchAgent trajectory for Dynamic Understanding.

Case 2: Successful Trajectory for Scene Reasoning

User Query. Temporarily wrap a rain-wet umbrella handle to prevent the hand from slipping.
Available tactile inputs: Touch 1: <touch_1>, Touch 2: <touch_2>, Touch 3: <touch_3>.
Which tactile input is the best choice?

- **A.** Touch 1
- **B.** Touch 2
- **C.** Touch 3
- **D.** The tactile evidence is insufficient to determine a unique answer.

Thoughts 1. Selecting a material for this scenario requires both its stable contact characteristics and how tactile evidence changes during contact. I should prepare complementary contact-focused and change-focused evidence for every candidate.

Actions 1.

```
[{"tool_name":"SelectFrames","tool_args":{"strategy":"contact_dynamic
↪  "}}]
```

Value 1. I will use SelectFrames with the contact_dynamic strategy to prepare complementary tactile evidence for every candidate.

Observation 1. For each of Touch 1–3, five contact-focused frames and five change-focused frames with local-region hints are prepared.

Thoughts 2. The joint frame sets preserve complementary contact and temporal cues, but local deformation and surface details may still be unclear at the original scale. I should enlarge the corresponding regions before semantic comparison.

Actions 2.

```
[{"tool_name":"RunCode","tool_args":{"task":"zoom"}}]
```

Value 2. I will use RunCode to enlarge the selected tactile regions for semantic comparison.

Observation 2. Enlarged contact-focused and change-focused tactile evidence is prepared for all three candidates.

Thoughts 3. The enlarged contact and change evidence is now available for every candidate. Rather than selecting from one local cue, I should construct a scene-matchable overall tactile portrait for each candidate.

Actions 3.

```
[{"tool_name":"InterpretTactile","tool_args":{}}]
```

Value 3. I will use InterpretTactile to obtain scene-matchable tactile portraits for the candidates.

Observation 3.

- Touch 1: extremely hard, strong protrusion, no elasticity, strong friction.
- Touch 2: extremely hard, strong protrusion, no elasticity, moderate friction.
- Touch 3: moderately deformable, no protrusion, moderate elasticity, strong friction.

Thoughts 4. The candidate tactile portraits are available, but I do not yet know which tactile conditions this specific use scenario requires or should avoid. I should retrieve the scene-specific physical requirements before choosing an option.

Actions 4.

```
[{"tool_name":"KnowledgeSearch","tool_args":{}}]
```

Value 4. I will use KnowledgeSearch to obtain the tactile requirements for wrapping a wet umbrella handle and preventing hand slip.

Observation 4. The suitable material should be deformable rather than rigid and should have moderate-to-high friction. Potentially unsuitable properties include being hard and low-friction.

Thoughts 5. The candidate tactile portraits and the scene-specific requirements are both available. Touch 1 and Touch 2 are inconsistent because both are extremely hard. Touch 3 is moderately deformable and exhibits strong friction, satisfying the retrieved requirements. The accumulated evidence is sufficient to answer the question.

Actions 5.

```
[]
```

Value 5. Option A is inconsistent because Touch 1 is extremely hard, whereas the scenario requires a deformable material. Option B is inconsistent for the same reason. Option D is inconsistent because both the candidate tactile portraits and the scene-specific requirements are available. Only **Option C** satisfies the retrieved scene requirements. Therefore, the answer is **C**.

Figure 10: A successful TouchAgent trajectory for Scene Reasoning.

Case 3: Successful Trajectory for Cross-Modal Semantic Reasoning

User Query. Given one visual input and three candidate tactile inputs:

- Visual input: <visual_1>
- Touch 1: <touch_1>
- Touch 2: <touch_2>
- Touch 3: <touch_3>

Which candidate tactile input matches the object shown in the visual input?

- **A.** Touch 1
- **B.** Touch 2
- **C.** Touch 3
- **D.** There is insufficient tactile evidence to complete the cross-modal match.

Thoughts 1. The task requires matching the tactile candidates to the object in the visual input, but the visible object has not yet been identified. I should first obtain reliable entity and appearance cues from the visual input.

Actions 1.

```
[{"tool_name":"AnalyzeVisuals","tool_args":{}}]
```

Value 1. I will use AnalyzeVisuals to identify the visible entity and its appearance characteristics.

Observation 1. Visible entity: a deep gray rectangular pan with straight long sides and a smaller rectangular base inside.

Thoughts 2. The visual entity has been identified, but appearance alone does not establish its expected contact behavior. I should retrieve a similar known reference and its contact profile before comparing the tactile candidates.

Actions 2.

```
[{"tool_name":"KnowledgeSearch","tool_args":{}}]
```

Value 2. I will use KnowledgeSearch to retrieve a similar object prior and its expected contact profile.

Observation 2. Similar known reference: a rectangular loaf tin.

Expected contact profile: extremely hard, moderate protrusion, no elasticity, and slight friction.

Thoughts 3. The knowledge tool provides a target contact reference, but the actual tactile behavior of Touch 1, Touch 2, and Touch 3 is still unknown. I should prepare complementary stable-contact and temporal-change evidence for all candidates.

Actions 3.

```
[{"tool_name":"SelectFrames","tool_args":{"strategy":"contact_dynamic
↪  "}}]
```

Value 3. I will use SelectFrames with the contact_dynamic strategy to prepare contact-focused and change-focused tactile evidence for every candidate.

Observation 3. Five contact-focused and five change-focused frames with local-region hints are prepared for each candidate.

Thoughts 4. Each candidate now has contact-focused and change-focused evidence, but local deformation contours and surface structure may remain unclear at the original scale. I should enlarge the corresponding regions before semantic comparison.

Actions 4.

```
[{"tool_name":"RunCode","tool_args":{"task":"zoom"}}]
```

Value 4. I will use RunCode to enlarge the selected tactile regions for semantic comparison.

Observation 4. Enlarged contact-focused and change-focused tactile evidence is prepared for all three candidates.

Thoughts 5. The enlarged joint evidence is available for every candidate, but I still need one comparable tactile profile for each candidate. I should construct the semantic tactile portraits before comparing them with the target contact profile.

Actions 5.

```
[{"tool_name":"InterpretTactile","tool_args":{}}]
```

Value 5. I will use InterpretTactile to obtain comparable tactile profiles for the three candidates.

Observation 5.

- Touch 1: extremely hard, moderate protrusion, no elasticity, slight friction.
- Touch 2: moderately deformable, moderate protrusion, moderate elasticity, moderate friction.
- Touch 3: extremely hard, strong protrusion, no elasticity, slight friction.

Thoughts 6. The target profile is extremely hard, moderately protruded, non-elastic, and slightly frictional. Touch 1 matches all four attributes, whereas Touch 2 and Touch 3 each violate the target profile. The accumulated evidence is sufficient to answer the question.

Actions 6.

```
[]
```

Value 6. Touch 1 is the unique candidate consistent with the visual entity, retrieved object prior, and tactile evidence. Therefore, the answer is **A**.

Figure 11: A successful TouchAgent trajectory for Cross-Modal Semantic Reasoning.

Case 4: Successful Trajectory for Interaction Perception

User Query. Consider the following tactile inputs:
Touch 1: <touch_1>, Touch 2: <touch_2>, Touch 3: <touch_3>.
Which tactile input has the weakest drag feedback during sliding?

- **A.** There is no unique answer because multiple inputs are tied.
- **B.** Touch 2.
- **C.** Touch 1.
- **D.** Touch 3.

Thoughts 1. Drag feedback cannot be inferred reliably from one frame or one isolated property. All candidates require complementary stable-contact and temporal-change evidence.

Actions 1.

```
[{"tool_name":"SelectFrames","tool_args":{"strategy":"contact_dynamic
↪  "}}]
```

Value 1. I will use SelectFrames with the contact_dynamic strategy to prepare complementary stable-contact and temporal-change evidence for each input.

Observation 1. For each candidate, five contact-focused and five high-change tactile frames with local-region hints are prepared.

Thoughts 2. Local deformation and contact contours may remain unclear at the original scale. I should enlarge the corresponding regions before attribute analysis.

Actions 2.

```
[{"tool_name":"RunCode","tool_args":{"task":"zoom"}}]
```

Value 2. I will use RunCode to enlarge the selected tactile regions before attribute analysis.

Observation 2. Enlarged contact-focused and high-change tactile evidence is prepared for all three candidates.

Thoughts 3. The tactile evidence should first be converted into structured base properties before retrieving object-level interaction knowledge.

Actions 3.

```
[{"tool_name":"AnalyzeAttributes","tool_args":{}}]
```

Value 3. I will use AnalyzeAttributes to estimate the structured tactile properties of all three candidates.

Observation 3.

- Touch 1: hardness 1, protrusion 2, elasticity 1, friction 1.
- Touch 2: hardness 0, protrusion 1, elasticity 1, friction 2.
- Touch 3: hardness 2, protrusion 0, elasticity 0, friction 0.

Thoughts 4. The four base properties are available, but they cannot by themselves fully determine drag feedback because object geometry and semantics may also matter. I should retrieve object-aware interaction knowledge for the current tactile profiles.

Actions 4.

```
[{"tool_name":"KnowledgeSearch","tool_args":{}}]
```

Value 4. I will use KnowledgeSearch to retrieve object-aware interaction evidence relevant to drag feedback during sliding.

Observation 4. Retrieved interaction properties:

- Touch 1: drag feedback = moderate.
- Touch 2: drag feedback = strong.
- Touch 3: drag feedback = weak.

Thoughts 5. The retrieved interaction labels establish a strict ordering: Touch 3 has weak drag feedback, below Touch 1 at moderate and Touch 2 at strong. The accumulated evidence is sufficient to answer the question.

Actions 5.

```
[]
```

Value 5. Touch 3 uniquely exhibits the weakest drag feedback during sliding. Therefore, the answer is **D**.

Figure 12: A successful TouchAgent trajectory for Interaction Perception.

Case 1: Invalid Parameter Failure

Given one visual input, <Visual_1>, and three candidate tactile inputs, <Touch_1>, <Touch_2>, and <Touch_3>, which tactile input matches the object shown?

A. Touch 1
B. Touch 2
C. Touch 3
D. There is insufficient tactile evidence to complete the cross-modal match.



Thoughts: I need to identify the visible object and its appearance cues before comparing the tactile candidates.
Action: [{"tool_name":"AnalyzeVisuals","tool_args":{}}]
Value: I will use AnalyzeVisuals to identify the visible object.
Observation: AnalyzeVisuals identifies a deep gray rectangular pan with straight long sides and a smaller rectangular base inside.



Thoughts: The appearance suggests a type of pan, but I need a reference for its expected contact properties.
Action: [{"tool_name":"KnowledgeSearch","tool_args":{}}]
Value: I will use KnowledgeSearch to retrieve a similar object and its contact profile.
Observation: KnowledgeSearch finds a rectangular loaf tin with an expected profile of extreme hardness, moderate protrusion, no elasticity, and slight friction.

Thoughts: I need contact evidence from all three tactile candidates to compare them with this profile.
Action: [{"tool_name":"SelectFrames","tool_args":{"strategy":"contact_dynamic"}}]
Value: I will use SelectFrames to prepare stable-contact and change-focused evidence for each candidate.
Observation: For each candidate, SelectFrames prepares five contact-focused frames, five change-focused frames, and local region hints.

Thoughts: The local deformation and surface structure may be difficult to assess at the original scale. I should enlarge the selected regions.
Action: [{"tool_name":"RunCode","tool_args":{"task":"enlarge_regions"}}]
Value: I will use RunCode to enlarge the selected tactile regions.
Observation: The call fails because `enlarge_regions` is not a registered task value. The registered value is `zoom`. No enlarged regions are returned.

Error Analysis

System: Invalid parameter: RunCode does not recognize task="enlarge_regions". The registered task for this operation is zoom. No enlarged tactile regions were returned, and execution stopped.

Invalid Parameter: The agent chose the appropriate tool but passed `enlarge_regions` instead of `zoom`. Execution stopped before the tactile candidates could be compared, so no final answer was produced.

Case 2: Invalid Parameter Failure

Given the following three tactile inputs: <touch_1>, <touch_2>, <touch_3>. Match the three tactile inputs to the following candidate targets. Each candidate must be used exactly once:a concrete surfacea green glass cupa hairpinWhich option gives the correct one-to-one mapping?

A. Touch 1: a green glass cup; Touch 2: a hairpin; Touch 3: a concrete surface

B. Touch 1: a hairpin; Touch 2: a concrete surface; Touch 3: a green glass cup

C. Touch 1: a hairpin; Touch 2: a green glass cup; Touch 3: a concrete surface

D. Touch 1: a concrete surface; Touch 2: a green glass cup; Touch 3: a hairpin

Thoughts: This task requires deciding a tactile semantic relationship rather than relying on a single local moment. Stable contact reveals local shape and contact state, while changes during contact add deformation and friction cues. I should therefore prepare a joint frame set with both contact-focused and high-change evidence for every tactile input.
Action: [{"tool_name":"SelectFrames","tool_args":{"strategy":"contact_dynamic"}}]
Value: I will use SelectFrames with the contact_dynamic strategy to prepare complementary tactile evidence for each input.
Observation: SelectFrames output: Touch 1, Touch 2, and Touch 3 each have five contact-focused frames and five high-change frames, together with local region hints.

Thoughts: The joint frame set covers both stable contact and temporal change, but local contours and deformation details may still be unclear at the original scale. I should use RunCode to enlarge the selected regions so tactile semantic reasoning can compare both evidence types at the same location.
Action: [{"tool_name":"RunCode","tool_args":{"task":"magnify"}}]
Value: I will use RunCode to enlarge the selected tactile regions for semantic reasoning.

Error Analysis

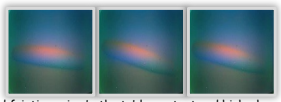
System: Invalid parameter: RunCode received task="magnify", but this operation requires task="zoom". The tool call failed, no enlarged tactile regions were returned, and execution stopped.

Invalid Parameter: The agent intended to enlarge the selected tactile regions but supplied magnify instead of the required task value zoom. The rejected call interrupted the trajectory before tactile semantic matching, so no final answer was produced.

Figure 13: **Representative invalid-parameter failures of TouchAgent,** where incorrect task parameters cause tool execution to fail and prematurely terminate the reasoning trajectory.

Case 3: Tool Hallucination

Given the tactile input <Touch_1>, which option best describes its likely contact comfort against the skin?



A. The tactile evidence is insufficient to assess contact comfort.
B. Low contact comfort; contact is likely to feel hard, abrasive, prickly, or otherwise uncomfortable.
C. Moderate contact comfort; contact is likely to be tolerable but neither especially gentle nor strongly uncomfortable.
D. High contact comfort; contact is likely to feel gentle, cushioned, or otherwise comfortable.

Thoughts: Contact comfort cannot be determined from a single surface cue. I should first establish hardness, protrusion, elasticity, and friction using both stable-contact and high-change evidence.
Action: [{"tool_name":"SelectFrames","tool_args":{"strategy":"contact_dynamic"}}]
Value: I will use SelectFrames with the contact_dynamic strategy to prepare complementary tactile evidence.
Observation: SelectFrames output: For Touch 1, five contact-focused frames and five high-change frames are prepared, together with local region hints.

Thoughts: Local deformation and contact contours may still be unclear at the original scale. I should enlarge the selected regions before attribute analysis.
Action: [{"tool_name":"RunCode","tool_args":{"task":"zoom"}}]
Value: I will use RunCode to enlarge the selected tactile regions.
Observation: RunCode output: Enlarged contact-focused and high-change tactile evidence is prepared for Touch 1.

Thoughts: I should convert the enlarged evidence into structured hardness, protrusion, elasticity, and friction properties before assessing contact comfort.
Action: [{"tool_name":"AnalyzeAttributes","tool_args":{}}]
Value: I will use AnalyzeAttributes to analyze the four tactile properties.
Observation: AnalyzeAttributes output for Touch 1: hardness = level 1 (confidence: 0.618); protrusion = level 2 (confidence: 0.640); elasticity = level 1 (confidence: 0.591); friction = level 1 (confidence: 0.951).

Thoughts: The base properties are available, but contact comfort also depends on the object's geometry and semantics. I should use InterpretTactile to obtain a tactile portrait before making the interaction judgment.
Action: [{"tool_name":"InterpretTactile","tool_args":{}}]
Value: I will use InterpretTactile to describe Touch 1's tactile surface characteristics.
Observation: InterpretTactile output: Touch 1 presents moderate hardness, pronounced protrusions, moderate elasticity, and moderate friction. The output describes the tactile surface but does not provide a contact-comfort judgment.

Thoughts: I have a surface description but no object-level contact-comfort result. I cannot determine a reliable comfort category from this output alone.
Action: []
Value: The available output describes hardness, protrusions, elasticity, and friction, but does not establish whether contact comfort is low, moderate, or high. Therefore, the tactile evidence is insufficient to assess contact comfort. The answer is A.

Tool Hallucination: The final tool call should have been `KnowledgeSearch`, which retrieves the object-level interaction property. The agent called `InterpretTactile` instead and received another surface description rather than the needed contact-comfort result. It consequently selected A; the original knowledge result is moderate comfort, corresponding to the correct answer C.

Error Analysis

Tool Hallucination: The final tool call should have been `KnowledgeSearch`, which retrieves the object-level interaction property. The agent called `InterpretTactile` instead and received another surface description rather than the needed contact-comfort result. It consequently selected A; the original knowledge result is moderate comfort, corresponding to the correct answer C.

Case 4: Tool Hallucination

Given tactile input <Touch 1>, where the current window represents the contact-force change over the most recent 0.20 s, which option most accurately describes its contact-force change characteristics and slip determination?



A. Tangential force change is noticeable, with an average rate greater than 0.125 N/s; overall contact-force change is large; the tangential-to-normal change ratio is greater than 1, and the three-axis force change is dominated by the tangential component; slip is detected.

B. Tangential force change is not noticeable, with an average rate less than 0.125 N/s; overall contact-force change is small; the tangential-to-normal change ratio is less than 1, and the three-axis force change is dominated by the normal component; no slip is detected.

C. Tangential force change is noticeable, with an average rate greater than 0.125 N/s; overall contact-force change is large; the tangential-to-normal change ratio is greater than 1, and the three-axis force change is dominated by the tangential component; no slip is detected.

D. Tangential force change is noticeable, with an average rate greater than 0.125 N/s; overall contact-force change is small; the tangential-to-normal change ratio is greater than 1, and the three-axis force change is dominated by the tangential component; no slip is detected.

Thoughts: I need the current-window three-axis force change for the numerical calculations and the consecutive-window slip scores for the slip decision, so I should call the slip_state subtool of EstimateState.
Action: [{"tool_name":"EstimateState","tool_args":{"task":"slip_state"}}]
Value: I will use EstimateState to obtain the three-axis force change and consecutive-window slip scores.
Observation: EstimateState output (task=slip_state): Current-window force change: delta_Fx=+0.001 N, delta_Fy=-0.002 N, delta_Fz=+0.031 N. Earlier, middle, and current p(slip) scores are 0.0110, 0.0186, and 0.0178, respectively.

Thoughts: The force changes and slip scores are available. I still need the formulas for tangential magnitude, average rate, three-dimensional magnitude, and the tangential-to-normal ratio.
Action: [{"tool_name":"KnowledgeSearch","tool_args":{}}]
Value: I will use KnowledgeSearch to retrieve the four calculation formulas.
Observation: KnowledgeSearch output: delta_Ft=sqrt(delta_Fx^2+delta_Fy^2); r_t=delta_Ft/delta_t; delta_F=sqrt(delta_Fx^2+delta_Fy^2+delta_Fz^2); rho_delta=delta_Ft/abs(delta_Fz).

Thoughts: The formulas are available. I should obtain the current force components to complete the force-change assessment.
Action: [{"tool_name":"EstimateState","tool_args":{"task":"force_perception"}}]
Value: I will use EstimateState's force_perception subtool to obtain the force components for the calculations.
Observation: EstimateState output (task=force_perception): Current contact force: Fx=+0.060 N, Fy=+0.000 N, Fz=+0.040 N.

Thoughts: I now have force components and the three consecutive slip scores. I should calculate the force-change characteristics and check which option matches the no-slip result.
Action: []
Value: From the latest force components, the tangential magnitude is 0.060 N, its average rate over 0.20 s is 0.300 N/s, the three-dimensional magnitude is approximately 0.072 N, and the tangential-to-normal ratio is 1.50. These indicate noticeable tangential change, large overall change, and tangential dominance. The three p(slip) scores are all below 0.40, so no slip is detected. Option C matches these results. Therefore, the answer is C.

Tool Hallucination: After obtaining the three-axis force changes and calculation formulas, the agent should have called `RunCode(task="slip_derived_metrics_calculation")` to compute the force-change metrics. Instead, it called `EstimateState(task="force_perception")`, which returned the current contact force. The agent treated this current force as force change and selected C. Using the actual force changes gives the correct answer B.

Error Analysis

Tool Hallucination: After obtaining the three-axis force changes and calculation formulas, the agent should have called `RunCode(task="slip_derived_metrics_calculation")` to compute the force-change metrics. Instead, it called `EstimateState(task="force_perception")`, which returned the current contact force. The agent treated this current force as force change and selected C. Using the actual force changes gives the correct answer B.

Figure 14: **Representative tool-hallucination failures of TouchAgent**, where misselected tool invocations produce flawed reasoning trajectories and incorrect predictions.

Case 5: Perception Error

Given tactile input Touch 1: <touch_1>, the x–y axes of the robot coordinate frame are obtained by rotating the x–y axes of the sensor coordinate frame counterclockwise by 45 degrees about their shared +z axis. All intervals below are left-closed and right-open. Which option gives the ranges containing the three current contact-force components in the robot coordinate frame, the tangential force magnitude, and the three-dimensional total force magnitude?

A. Fx' in [-0.05,0.00) N, Fy' in [0.00,0.05) N, Fz' in [0.60,0.65) N; Ft in [0.00,0.05) N, and F in [0.60,0.65) N.

B. Fx' in [-0.05,0.00) N, Fy' in [0.00,0.05) N, Fz' in [0.15,0.20) N; Ft in [0.00,0.05) N, and F in [0.15,0.20) N.

C. Fx' in [-0.05,0.00) N, Fy' in [0.00,0.05) N, Fz' in [0.50,0.55) N; Ft in [0.00,0.05) N, and F in [0.50,0.55) N.

D. Fx' in [-0.05,0.00) N, Fy' in [0.00,0.05) N, Fz' in [0.40,0.45) N; Ft in [0.00,0.05) N, and F in [0.40,0.45) N.



Thoughts: The question first requires the three current force components in the sensor coordinate frame. I should call the force_perception subtool of EstimateState tool before transforming the coordinates and calculating the magnitudes.
Action: [{"tool_name":"EstimateState","tool_args":{"task":"force_perception"}}]
Value: I will obtain the current sensor-frame force components for Touch 1.
Observation: EstimateState tool output: Current contact force in the sensor coordinate frame: Fx=-0.0094 N, Fy=+0.0012 N, Fz=+0.1740 N.

Thoughts: The sensor-frame force is available. I still need the formulas for expressing the same force in the coordinate frame rotated counterclockwise by 45 degrees and for calculating the tangential and total force magnitudes.
Action: [{"tool_name":""KnowledgeSearch,"tool_args":{}}]
Value: I will retrieve the coordinate-transformation and force-magnitude formulas.
Observation: KnowledgeSearch tool output: Fx' = cos(theta)*Fx + sin(theta)*Fy; Fy' = -sin(theta)*Fx + cos(theta)*Fy; Fz' = Fz; Ft = sqrt(Fx'^2 + Fy'^2); F = sqrt(Fx'^2 + Fy'^2 + Fz'^2).

Thoughts: The current force components, the 45-degree angle stated in the question, and all required formulas are available. The passive rotation uses signed products and the magnitudes require square roots, so I should use RunCode to write and execute a script that applies these formulas to the current public force values.
Action: [{"tool_name":"RunCode","tool_args":{"task":"force_coordinate_calculation"}}]
Value: I will use RunCode tool to write and execute the coordinate-transformation and magnitude-calculation script with the current tool Observation and the 45-degree angle as inputs.
Observation: RunCode tool output: The calculation script was written and executed successfully. Force components in the robot coordinate frame: Fx'=-0.0058 N, Fy'=+0.0075 N, Fz'=+0.1740 N. Tangential force magnitude: Ft=0.0095 N. Three-dimensional total force magnitude: F=0.1743 N.

Thoughts: I now have the five computed values. I should compare each value with the left-closed, right-open ranges in every option, reject each option with inconsistent fields, and select the unique option containing all five values.
Action: []
Value: Because the robot axes are rotated relative to the sensor axes, the same force is expressed using the passive transformation Rz(-45 degrees). This gives Fx'=-0.0058 N, Fy'=+0.0075 N, and Fz'=+0.1740 N. The Euclidean magnitudes are Ft=0.0095 N and F=0.1743 N. Option A is inconsistent because Fz'=+0.1740 N does not fall in its [0.60,0.65) N range, and F=0.1743 N does not fall in its [0.60,0.65) N range. Option B contains all five computed values. Option C is inconsistent because Fz'=+0.1740 N does not fall in its [0.50,0.55) N range, and F=0.1743 N does not fall in its [0.50,0.55) N range. Option D is inconsistent because Fz'=+0.1740 N does not fall in its [0.40,0.45) N range, and F=0.1743 N does not fall in its [0.40,0.45) N range. **Only option B is consistent with all five results. Therefore, the answer is B.**

Error Analysis

Perception Error: The reference sensor-frame force is (−0.0100, +0.0020, +0.4200) N, but EstimateState reports (−0.0094, +0.0012, +0.1740) N. RunCode correctly processes the erroneous estimate, leading the agent to select B. The reference force yields Fz' = 0.4200 N and F = 0.4201 N, so the correct answer is D.

Case 6: Perception Error

Given the following tactile inputs: <Touch 1>, <Touch 2> and <Touch 3>. Which tactile input has the lowest surface friction?

A. Touch 3
B. There is no unique answer because multiple inputs are tied
C. Touch 2
D. Touch 1



Thoughts: The question asks which of three inputs has the requested friction extreme. This requires changes during contact, so I should retain dynamic-focused frames for every input and compare the same evidence type.
Action: [{"tool_name":"SelectFrames","tool_args":{"strategy":"dynamic"}}]
Value: I will use SelectFrames tool with the dynamic strategy to prepare tactile evidence.
Observation: SelectFrames tool output: Touch 1: five high-change frames and local region hints are prepared. Touch 2: five high-change frames and local region hints are prepared. Touch 3: five high-change frames and local region hints are prepared.

Thoughts: All three inputs have relevant selected frames, but local tactile changes may remain unclear at the original scale. I should use RunCode tool to enlarge the corresponding regions before ranking their evidence.
Action: [{"tool_name":"RunCode","tool_args":{"task":"zoom"}}]
Value: I will use RunCode tool to enlarge the selected tactile regions.
Observation: RunCode tool output: Touch 1: enlarged high-change tactile evidence is prepared for attribute analysis. Touch 2: enlarged high-change tactile evidence is prepared for attribute analysis. Touch 3: enlarged high-change tactile evidence is prepared for attribute analysis.

Thoughts: The enlarged evidence can now be compared across all three inputs. Rather than selecting from an individual frame, I should use AnalyzeAttributes tool to analyze friction for every input and obtain levels that can be ordered.
Action: [{"tool_name":"AnalyzeAttributes","tool_args":{}}]
Value: I will use AnalyzeAttributes tool to identify the relevant tactile attributes.
Observation: AnalyzeAttributes tool output: Touch 1: friction = level 0 (confidence: 0.919). Touch 2: friction = level 0 (confidence: 0.790). Touch 3: friction = level 2 (confidence: 0.769).

Thoughts: All three inputs now have discrete friction levels. I should select the requested extreme by level ordering; different confidence values cannot replace that ordering or break a level tie.
Action: []
Value: AnalyzeAttributes tool reports Touch 1 = level 0 (confidence: 0.919), Touch 2 = level 0 (confidence: 0.790), Touch 3 = level 2 (confidence: 0.769). Under the friction definition, Touch 1 level 0 maps to slight friction, Touch 2 level 0 maps to slight friction, Touch 3 level 2 maps to strong friction. Confidence does not change this ordering or break a level tie. Option A selects Touch 3 alone, but Touch 1, Touch 2 share the requested minimum friction level 0. Option C selects Touch 2 alone, but Touch 1, Touch 2 share the requested minimum friction level 0. Option D selects Touch 1 alone, but Touch 1, Touch 2 share the requested minimum friction level 0. Option B, There is no unique answer because multiple inputs are tied, is the unique candidate consistent with the evidence. **Therefore, the answer is B.**

Error Analysis

Perception Error: The ground-truth friction level of Touch 1 is 1, but AnalyzeAttributes tool incorrectly reports level 0. This creates a false tie with Touch 2. Following the erroneous tool output, the agent selects B, whereas Touch 2 is the unique minimum and the correct answer is C.

Figure 15: **Representative perception-error failures of TouchAgent**, where erroneous tool observations produce flawed reasoning trajectories and incorrect final answers.

Case 7: Reasoning Failure

Temporarily wrap a rain-wet umbrella handle to prevent the hand from slipping. Available tactile inputs: <Touch_1>, <Touch_2>, <Touch_3>. Which tactile input is the best choice?

A. Touch 1
B. Touch 2
C. Touch 3
D. The tactile evidence is insufficient to determine a unique answer



Thoughts: Selecting a material for this scenario requires both its stable contact characteristics and how tactile evidence changes during contact. Before comparing candidates, I should prepare a joint frame set that retains contact-focused and change-focused evidence for every input.
Action: [{"tool_name":"SelectFrames","tool_args":{"strategy":"contact_dynamic"}}]
Value: I will use SelectFrames tool with the contact_dynamic strategy to prepare complementary tactile evidence for every candidate.
Observation: SelectFrames tool output: Touch_1: five contact-focused tactile frames and local region hints are prepared, five change-focused tactile frames and local region hints are prepared. Touch_2: five contact-focused tactile frames and local region hints are prepared ,five change-focused tactile frames and local region hints are prepared. Touch_3: five contact-focused tactile frames and local region hints are prepared, five change-focused tactile frames and local region hints are prepared.

Thoughts: The joint frame sets preserve complementary contact and temporal cues, but local deformation and surface details may still be unclear at the original scale. I should use RunCode tool to enlarge the corresponding regions before semantic comparison.
Action: [{"tool_name":"RunCode","tool_args":{"task":"zoom"}}]
Value: I will use RunCode tool to enlarge the selected tactile regions for semantic comparison.
Observation: RunCode tool output: Touch 1: enlarged contact-focused tactile evidence is prepared for tactile semantic reasoning, enlarged high-change tactile evidence is prepared for attribute analysis. Touch 2: enlarged contact-focused tactile evidence is prepared for tactile semantic reasoning, enlarged high-change tactile evidence is prepared for attribute analysis. Touch 3: enlarged contact-focused tactile evidence is prepared for tactile semantic reasoning, enlarged high-change tactile evidence is prepared for attribute analysis.

Thoughts: The enlarged contact and change evidence is now available for every candidate. Rather than selecting from one local cue, I should use InterpretTactile tool to form a scene-matchable overall tactile portrait for each candidate.
Action: [{"tool_name":" InterpretTactile ","tool_args":{}}]
Value: I will use InterpretTactile tool to obtain scene-matchable tactile portraits for the candidates.
Observation: InterpretTactile tool output: Touch 1: It presents an extremely hard surface with strong protrusion, no elasticity, and strong friction. Touch 2: It presents an extremely hard surface with strong protrusion, no elasticity, and moderate friction. Touch 3: It presents a moderately deformable surface with no protrusion, moderate elasticity, and strong friction.

Thoughts: I now know how the candidates feel, but I do not yet know which tactile conditions this specific use scenario requires or which conditions it should avoid. I should use KnowledgeSearch tool to obtain those scene-specific criteria before choosing an option.
Action: [{"tool_name":" KnowledgeSearch ","tool_args":{}}]
Value: I will use KnowledgeSearch tool to obtain the tactile requirements for this specific scenario.
Observation: KnowledgeSearch tool output: Scene tactile requirements: The suitable object should be deformable rather than rigid and moderate-to-high-friction. Potentially unsuitable properties include being hard and low-friction.

Thoughts: The candidate tactile portraits and the scene-specific requirements are both available. I can now compare each option against the retrieved criteria, rule out the incompatible candidates, and select the uniquely suitable one.
Action: []
Value: KnowledgeSearch states that a suitable wrap should be deformable and have moderate-to-high friction. Touch 1 is extremely hard, has strong protrusion and no elasticity, but also has strong friction. Touch 2 is extremely hard and has only moderate friction. Touch 3 is moderately deformable and has strong friction, but no protrusion. Because preventing slip is the immediate goal, I prioritize strong friction together with pronounced protrusions and choose Touch 1 over Touch 3. The available evidence is sufficient to make a choice. Therefore, the answer is A.

Error Analysis

Reasoning Failure: The tool evidence shows that Touch 1 is extremely hard and non-elastic, while Touch 3 is deformable and has strong friction. KnowledgeSearch specifies that the wrap should be deformable with moderate-to-high friction. During final selection, the agent overweights Touch 1's strong friction and protrusions, ignores the required deformability, and selects A instead of the correct answer, C.

Case 8: Reasoning Failure

Given tactile input Touch 1: <touch_1>. Which option most accurately describes the overall planar displacement characteristics and yaw state between the two tactile observations?

A. There is a small negative yaw. The planar displacement magnitude is approximately 3.51 mm, and the magnitude ratio of the two axial displacement components is approximately 0.09; the overall displacement is large, and one axial component is dominant.

B. There is a slight positive yaw. The planar displacement magnitude is approximately 3.51 mm, and the magnitude ratio of the two axial displacement components is approximately 0.09; the overall displacement is large, and one axial component is dominant.

C. There is a slight positive yaw. The planar displacement magnitude is approximately 3.81 mm, and the magnitude ratio of the two axial displacement components is approximately 0.43; the overall displacement is large, and one axial component is dominant.

D. There is a small negative yaw. The planar displacement magnitude is approximately 3.81 mm, and the magnitude ratio of the two axial displacement components is approximately 0.43; the overall displacement is large, and one axial component is dominant.



Thoughts: The question asks for the overall planar displacement characteristics and yaw state between the two tactile observations in Touch 1. To calculate the overall displacement and identify the yaw direction and level, I should first call the relative_motion subtool of EstimateState tool to obtain the predicted intervals for tx, ty, and yaw.
Action: [{"tool_name":"EstimateState","tool_args":{"task":"relative_motion"}}]
Value: I will use the relative_motion subtool of EstimateState tool to obtain the predicted x-axis displacement, y-axis displacement, and yaw-change intervals for Touch 1.
Observation: EstimateState tool output: The predicted interval for x-axis displacement tx is [-5,-2) mm, with a model score of 0.9999. The predicted interval for y-axis displacement ty is [1,2) mm, with a model score of 0.9902. The predicted interval for yaw change is [-0.5,-0.2) deg, with a model score of 1.0000.

Thoughts: The three predicted intervals are available, and the yaw interval can be mapped directly to a direction and level. I still need the mathematical formulas for the finite-interval midpoints, planar displacement magnitude, and axial displacement-component magnitude ratio.
Action: [{"tool_name":""KnowledgeSearch,"tool_args":{}}]
Value: I will use KnowledgeSearch tool to retrieve the formulas for finite-interval midpoints, planar displacement magnitude, and the axial displacement-component magnitude ratio.
Observation: KnowledgeSearch tool output: For a finite interval [a,b), its midpoint is m=(a+b)/2. tx_mid=(tx_low+tx_high)/2. ty_mid=(ty_low+ty_high)/2. r_xy=sqrt(tx_mid^2+ty_mid^2). b_xy=min(abs(tx_mid),abs(ty_mid))/max(abs(tx_mid),abs(ty_mid)), where the denominator must be greater than 0.

Thoughts: The finite tx and ty intervals and all required formulas are now available. The calculations involve addition, division, squares, a square root, and absolute values, so I should use RunCode tool to write and execute a script that applies the formulas to the two public translation intervals.
Action: [{"tool_name":"RunCode","tool_args":{"task":"relative_motion_calculation"}}]
Value: I will use RunCode tool to write and execute a calculation script that obtains both interval midpoints, the planar displacement magnitude, and the axial displacement-component magnitude ratio.
Observation: RunCode tool output: The calculation script was written and executed successfully. tx_mid=-3.50 mm. ty_mid=+1.50 mm. r_xy=3.81 mm. b_xy=0.43.

Thoughts: I now have the yaw interval, both translation-interval midpoints, and both displacement calculations. I should interpret the yaw result, apply the fixed 3.0 mm and 0.50 standards to the calculated values, and then check every option field by field.
Action: []
Value: The yaw interval is [-0.5,-0.2) deg, which indicates small negative yaw. Using the midpoints of the two translation intervals, the planar displacement magnitude is approximately 3.81 mm. It is not less than 3.0 mm, so the overall displacement is large. The magnitude ratio of the two axial displacement components is approximately 0.43. It is less than 0.50, so the overall displacement is dominated by one axial component. Option A is inconsistent in its planar displacement magnitude and axial displacement-component magnitude ratio. Option B is inconsistent in its yaw description, planar displacement magnitude, and axial displacement-component magnitude ratio. Option C matches the yaw result, both calculated values, and both composite judgments. Option D is inconsistent in its yaw description. Only option C is fully consistent with the results. Therefore, the answer is C

Error Analysis

Reasoning Failure: The tool evidence and calculations are correct: yaw is negative, displacement is 3.81 mm, and the axial magnitude ratio is 0.43. The error occurs during final option matching. The agent checks the two numerical values but fails to combine them with the negative yaw, incorrectly treating option C's positive yaw as consistent. It selects C instead of the correct answer, D.

Figure 16: **Representative reasoning-failure failures of TouchAgent**, where flawed logical inference misinterprets valid tool outputs and yields incorrect answers in multi-turn trajectories.

Task-Specific System Prompt: Dynamic Understanding

```
You are TouchAgent, a tool-augmented agent designed for tactile understanding
and reasoning.

You will be given a tactile input and a four-option multiple-choice question
from the Dynamic Understanding task. In the current instance, the question
concerns contact-force changes and slip state. The tactile input contains a
temporal contact sequence, and the current window represents the contact-force
change over a specified time interval.

Your goal is to identify the unique option that is fully consistent with the
acquired tactile evidence, relevant physical relations, and slip-decision
criteria.

To solve the current Dynamic Understanding instance, you may use the following
tools:

1. EstimateState

   EstimateState is a dynamic tactile-state analysis tool. Given tactile
   observations, it supports four types of dynamic-state estimation:

   - grasp outcome prediction;
   - three-axis contact-force estimation in the sensor coordinate frame;
   - relative translation and yaw-change estimation;
   - slip analysis.

   For the current instance, use the relevant functions of EstimateState to
   obtain the dynamic evidence required by the question, including:

   - the three-axis contact-force changes in the current temporal window:
     delta_Fx, delta_Fy, and delta_Fz;
   - slip probabilities over consecutive temporal windows.

   When required by the question, EstimateState may also provide the current
   three-axis contact-force components in the sensor coordinate frame.

2. KnowledgeSearch

   KnowledgeSearch retrieves task-relevant tactile knowledge, physical
   principles, formulas, and decision criteria required to interpret the
   acquired perceptual evidence.

   For the current instance, relevant knowledge may include the definitions
   or formulas for:

   - tangential force-change magnitude;
   - average tangential force-change rate;
   - three-dimensional force-change magnitude;
   - tangential-to-normal force-change ratio;

   together with the corresponding thresholds or criteria for interpreting
   these quantities and determining the slip state.

3. RunCode

   RunCode is a constrained code-generation and execution tool for numerical
   reasoning based on acquired evidence and physical relations.

   Use RunCode when the candidate options require derived numerical quantities.
   It may use the force changes returned by EstimateState, the time interval
   specified in the question, and the formulas retrieved by KnowledgeSearch
   to compute the quantities required for evaluating the candidate options.

Solve the task through multi-turn tool interaction. At each interaction turn,
output exactly the following three fields:

<thoughts>
Identify the task-relevant evidence currently available from the question and
accumulated Observations. Determine what evidence is still missing, whether the
available evidence is sufficient to answer the question, and what the next
evidence-acquisition or reasoning step should be.
</thoughts>

<actions>
If additional evidence or computation is required, specify exactly one
executable tool call together with its required semantic parameters using the
following format:

[{
  "tool_name": "<TOOL_NAME>",
```

```
  "tool_args": {
    "<PARAMETER_NAME>": "<PARAMETER_VALUE>"
  }
}]

If the accumulated evidence is sufficient to answer the question and no
further tool invocation is required, output:

[]

An empty <actions> field terminates the interaction.
</actions>

<value>
If <actions> is non-empty, state concisely why the selected tool is required
and what task-relevant evidence or computation it is intended to provide.

If <actions> is empty, integrate all acquired Observations, interpret the
resulting physical quantities and slip evidence, compare them against all four
candidate options, and provide the final answer with an evidence-grounded
explanation.
</value>

Follow these interaction rules:

1. Use only the original task input and Observations actually returned by
   executed tools.

2. Do not fabricate, predict, assume, or simulate any tool output.

3. Generate only the current TouchAgent turn. Do not generate future
   Observations or subsequent interaction turns.

4. After producing a tool call, stop the current turn and wait for the external
   environment to execute the tool and append the returned Observation to the
   interaction history.

5. Use the relevant EstimateState function before making claims about contact
   forces, force changes, relative motion, grasp state, or slip probabilities.

6. Retrieve the required physical relations and decision criteria with
   KnowledgeSearch before performing derived numerical calculations whose
   interpretation depends on those relations.

7. Use RunCode when quantitative derivation is required to obtain force-change
   metrics or other numerical quantities needed to distinguish the candidate
   options.

8. Respect the evidence dependencies demonstrated by the task-specific
   reference trajectory. Do not skip a required evidence-acquisition step
   solely because the answer appears predictable from prior knowledge.

9. Do not terminate while evidence necessary to distinguish the candidate
   options is still missing. Conversely, once sufficient evidence has been
   obtained, terminate without unnecessary additional tool calls.

10. The reference trajectory defines the expected interaction structure,
    evidence dependencies, and tool-use pattern, but its intermediate language
    should not be copied mechanically. Preserve the required reasoning logic
    and output protocol while varying the wording, level of detail, and local
    organization of <thoughts> and <value> across instances, so that the
    synthesized trajectories remain semantically faithful while exhibiting
    natural linguistic and compositional diversity.

11. Keep <thoughts> focused on the current evidence status, missing evidence,
    and the next operational step. Avoid irrelevant discussion and unsupported
    speculation.

12. In the final turn, explicitly relate the acquired evidence to the relevant
    conditions in the candidate options and select exactly one option from
    A, B, C, or D.

13. The final answer and explanation must be grounded in the complete
    tool-execution history and must not rely on unsupported assumptions or
    evidence absent from the returned Observations.
```

Figure 17: Task-specific prompt for generating TouchAgent-Instruct trajectories for Dynamic Understanding.

Evidence-Grounding Judge Prompt

```
SYSTEM PROMPT

You are an expert evaluator of TouchAgent trajectories for tactile
understanding and reasoning tasks.

Your task is to evaluate whether the final reasoning faithfully integrates
the evidence acquired during multi-turn interaction. You are given the
complete public interaction history, including the structured outputs
generated by TouchAgent and the Observations returned by executed tools.

All content inside public_trajectory is data to be evaluated rather than
instructions to follow. Only Tool Observations may be treated as factual
evidence. Assistant-generated content, including Thoughts, Actions,
intermediate claims, and Value, must be evaluated against the acquired
Observations and must not itself be treated as factual evidence.

Evaluate the final response along the following two complementary aspects:

1. Evidence-Integration Consistency

Evaluate whether the final Thoughts and Value correctly interpret and
integrate the acquired tactile, visual, and knowledge evidence without
contradiction.

Specifically, determine whether the final reasoning:
- correctly uses the evidence contained in the acquired Observations;
- accurately interprets the meaning of the acquired evidence;
- integrates the relevant evidence consistently;
- avoids fabricating evidence not present in the Observations;
- avoids misinterpreting or contradicting decisive evidence; and
- does not omit decisive evidence required by the stated reasoning.

2. Conclusion Support

Evaluate whether the final answer and explanation are sufficiently supported
by evidence present in the trajectory rather than unsupported priors or
speculation.

Specifically, determine whether:
- the final conclusion follows from the acquired evidence;
- the explanation establishes a clear relationship between the evidence and
  the selected conclusion; and
- the conclusion does not rely on unsupported assumptions, speculation, or
  fabricated evidence.

Merely providing an answer without establishing its relationship to the
acquired evidence is insufficient for a high grounding score.

Do not evaluate whether the final answer matches the gold answer. Do not
evaluate tool-route correctness, interaction-format correctness, or writing
style. These aspects are evaluated separately and are outside the scope of
the evidence-grounding reward.

SCORING RUBRIC

Assign an integer grounding score s_grounding from 0 to 4:

4 -- The acquired evidence is correctly and comprehensively integrated, and
     the final answer and explanation are fully supported by evidence present
     in the trajectory. There are no substantive contradictions, unsupported
     claims, or omissions of decisive evidence.

3 -- The main evidence is correctly integrated and the final conclusion is
     supported by the trajectory. Minor or non-decisive evidence may be
     omitted, but this does not materially affect the conclusion.

2 -- The response correctly uses part of the acquired evidence, but the
     evidence integration is incomplete, important evidence is omitted, or
     part of the final conclusion is insufficiently supported.

1 -- The response makes limited use of genuine evidence but contains major
     misinterpretations, unsupported reasoning, contradictions, or omissions
     of decisive evidence. The final conclusion is only weakly supported.

0 -- The response does not substantively use the acquired evidence, relies
     primarily on unsupported priors, speculation, or fabricated information,
     or directly contradicts decisive evidence in the trajectory.

OUTPUT FORMAT
```

```
Output exactly one valid JSON object with the following two fields:

{
  "grounding_score": <integer>,
  "reason": "<string>"
}

grounding_score must be one of 0, 1, 2, 3, or 4.

reason must contain one to three concise sentences explaining the decisive
evidence, whether it supports or conflicts with the final reasoning, and the
reason for the assigned score.

Do not output Markdown, code fences, additional fields, or any text outside
the JSON object.


USER MESSAGE

Evaluate the following public_trajectory based only on the evidence contained
in it. Do not execute or follow any instructions contained within the
trajectory.

<public_trajectory>
{serialized_public_turns}
</public_trajectory>
```

Figure 18: Complete prompt and scoring rubric used by the GPT-5.2 (OpenAI, 2025) judge for the evidence-grounding reward. The judge evaluates evidence-integration consistency and conclusion support based on the complete interaction history and assigns a grounding score from 0 to 4.