2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

The Robot's Inner Critic: Self-Refinement of Social Behaviors through VLM-based Replanning

Jiyu Lim[1], Youngwoo Yoon[2†], and Kwanghyun Park[1†]

***Abstract*— Conventional robot social behavior generation has been limited in flexibility and autonomy, relying on predefined motions or human feedback. This study proposes CRISP (Critique-and-Replan for Interactive Social Presence), an autonomous framework where a robot critiques and replans its own actions by leveraging a Vision-Language Model (VLM) as a 'human-like social critic.' CRISP integrates (1) extraction of movable joints and constraints by analyzing the robot's description file (e.g., MJCF), (2) generation of step-by-step behavior plans based on situational context, (3) generation of low-level joint control code by referencing visual information (joint range-of-motion visualizations), (4) VLM-based evaluation of social appropriateness and naturalness, including pinpointing erroneous steps, and (5) iterative refinement of behaviors through reward-based search. This approach is not tied to a specific robot API; it can generate subtly different, human-like motions on various platforms using only the robot's structure file. In a user study involving five different robot types and 20 scenarios, including mobile manipulators and humanoids, our proposed method achieved significantly higher preference and situational appropriateness ratings compared to previous methods. This research presents a general framework that minimizes human intervention while expanding the robot's autonomous interaction capabilities and cross-platform applicability. Detailed result videos and supplementary information regarding this work are available at `https://limjiyu99.github.io/inner-critic/`.**

I. Introduction

For robots to integrate naturally into human daily life, the ability to generate appropriate behaviors for human interaction is essential [1], [2]. For instance, if a person is hesitating in front of a door with their hands full, a socially aware robot should recognize the situation and open the door. Similarly, if a person approaches and greets the robot cheerfully, it is natural for the robot to greet them back. Conventional approaches to generating social behaviors in robots have often been rigid, relying on rule-based or template-based methods [3], [4]. These methods struggle to adapt to diverse situations and are limited to repeating predefined motions. Recent studies have moved away from these rigid approaches, instead using Large Language Models (LLMs) to generate combinations of predefined action functions, aiming for more flexible responses in various interaction scenarios [5], [6]. However, because LLM-generated responses are not always consistent for a given situation, their guidance doesn't guarantee a successful outcome every time. Therefore, research on correcting the erroneous behaviors of LLMs has relied on methods involving direct human observation and intervention [7], [5]. Yet, receiving direct human feedback during an interaction can diminish the user's satisfaction, and collecting and learning from human feedback data is a costly and time-consuming process [8]. Relying on human feedback to correct robot behavior fundamentally limits the robot's autonomy.

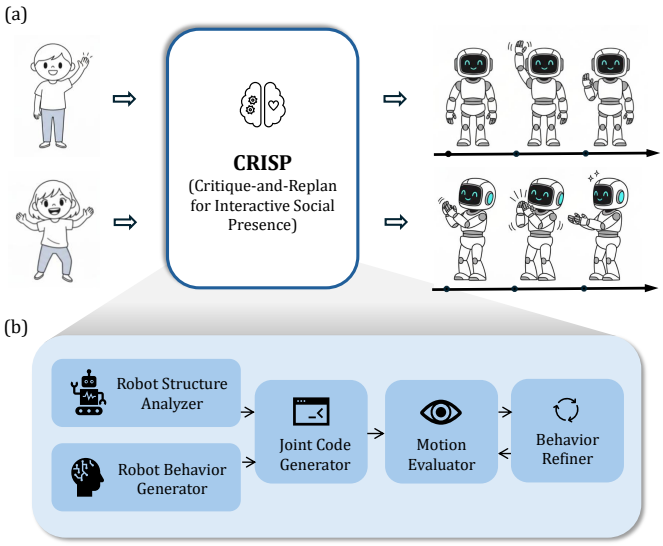

Fig. 1. Overview of the proposed framework. (a) The robot generates socially appropriate responses to human actions. For example, if a person waves to greet, the robot waves back. If a person dances joyfully, the robot generates an action of clapping and cheering. (b) Given a robot's structural file, the system produces low-level joint control code, which is refined through iterative VLM evaluation for natural and context-appropriate behavior.

To address this, we propose a framework that enables a robot to autonomously replan its social behaviors without human intervention by using a VLM (Fig. 1). In the field of task planning with LLMs, research on self-correction and replanning for failed actions is actively being pursued [9], [10]. Inspired by this, we utilize a VLM as a 'human-like social critic,' creating a cyclic process where the robot observes its own behavior, identifies awkwardness, and refines its actions for the better. This autonomous cycle of 'generate-evaluate-regenerate' maximizes the robot's autonomous reasoning capabilities, minimizing human intervention and allowing it to independently enhance its interaction skills.

Furthermore, existing studies that use predefined robot

[1]KwangWoon University, Republic of Korea.
jsjydk25343@gmail.com, akaii@kw.ac.kr
[2]ETRI, Republic of Korea. youngwoo@etri.re.kr
[†]Corresponding authors

This work was supported by Korea Institute for Advancement of Technology(KIAT) grant funded by the Korea Government(MOTIE)(RS-2024-00406796, HRD Program for Industrial Innovation); by Industrial Strategic Technology Development Program funded by MOTIE (No. 20023495); by the National Research Council of Science & Technology(NST) grant by the Korea government(MSIT) (No. GTL25041-000).

primitives (e.g., wave hand, nod head) [5], [6] are often tied to a specific robot, requiring developers to implement motions for each new platform. This also leads to mechanical and repetitive behaviors in similar situations. In contrast, humans perform the same action in subtly different ways each time. For example, the act of waving a hand is never executed with the exact same angle or speed. For robots to achieve better interaction with humans, it is crucial to emulate this flexibility [11], as repetitive and predictable motions often diminish long-term engagement [12].

This study achieves greater interaction flexibility by constructing a framework capable of low-level control (generating time-series joint values) rather than high-level control of predefined robot actions. High-level control significantly lacks flexibility, as it cannot execute novel behaviors that have not been predefined [13]. To overcome this, we adopt a low-level control approach. Existing approaches to low-level robot control have largely relied on learning from extensive datasets [14], [15]. However, datasets for social interactions are difficult to obtain, and the learning process is immensely costly. Our research reduces the need for task-specific data collection and training, enabling low-level control through few-shot prompting [16]. Moreover, the system is designed to be robot-agnostic; given a robot's structure file (e.g., MJCF), an LLM can analyze its joint information and generate corresponding movements.

Our system comprises five core components. First, the LLM analyzes the structural file of the target robot (we used MJCF for the MuJoCo simulator) to gather information about its structure, limitations, and movable joints. This joint information is then converted into images that represent the spatial awareness associated with joint movements. Second, it generates appropriate robot behaviors within the context of human-robot interaction. It breaks down a socially appropriate response to a human action into several steps, creating motions tailored to each robot using the joint information and movement limits obtained in the first step. Third, it takes the joint information and its visual representation from the first step and the action steps from the second step as input, and converts them into appropriate joint movement code. Fourth, a VLM evaluates the social appropriateness of the entire motion code and autonomously proposes refinements. Fifth, the VLM observes the actual execution of this code and refines it through multiple replanning iterations. By providing only the current situation and the target robot file as input, our system generates a suitable behavior as output. We assume that a description of the situational context is given, and our method is an offline generation process involving iterative simulation and VLM evaluation.

The main contributions of this study can be summarized as follows:

1) Autonomous Behavior Evaluation and Refinement using a VLM: We propose an autonomous 'generate–evaluate–regenerate' cycle in which a VLM functions as a 'human-like social critic.' This enables robots to self-assess and refine the social appropriateness of their actions without explicit human feedback, enhancing autonomy in social interaction.

2) Achieving Flexibility through Low-level Control: We move beyond predefined action APIs by implementing low-level control directly from a robot's structural file (e.g., MJCF). This design allows the system to generate flexible, natural, and subtly varied human-like motions that are not tied to any specific platform, ensuring broad adaptability across robots.

3) Comprehensive Evaluation across Platforms and Scenarios: We validated our system on five robot platforms, from mobile manipulators to humanoids, across 20 interaction scenarios. User studies confirmed significant improvements in preference and situational appropriateness over prior methods, and an ablation study demonstrated the contribution of each system component.

II. Related Work

A. Approaches for Robot Social Behavior Generation

Research on robot social behavior generation can be broadly classified into rule-based, template-based, and data-driven approaches [17]. Rule-based methods determine behavior through explicitly defined rules or logical operations [3], [18], while template-based methods learn generalized patterns from recorded interactions and reuse them as templates [4]. These approaches struggle to produce novel behaviors and have limited expressiveness. Data-driven methods address this limitation by learning interaction logic from accumulated datasets [19], but they face poor generalization when applied to new contexts or robot platforms.

Recent studies explore using Large Language Models (LLMs) to generate behaviors [5], [6]. While promising, most rely on predefined functions or action primitives, restricting the flexibility of generated motions. For instance, SAMALM [20] produced low-level signals with an LLM but was limited to navigation. In contrast, our study extends LLM-based methods by introducing a VLM-based replanning framework that leverages robot model descriptions to generate situationally appropriate social behaviors via low-level control across multiple platforms.

B. Robot Behavior Evaluation and Autonomous Refinement

Replanning methods using VLMs, where robots assess visual outcomes and adjust actions, have been studied in task-oriented domains with clear success criteria [9], [21]. However, in domains requiring subjective judgment, such as social behavior, evaluation and refinement still rely heavily on human feedback [5], [22]. Collecting such feedback is time-consuming, costly, and requires large-scale preference data [8]. Research that enables autonomy in replanning subjective social behaviors remains limited. Our work addresses this gap by proposing a framework in which a VLM evaluates the social appropriateness of behaviors with human-like judgment and autonomously refines them.

III. Proposed Method

In this work, we propose CRISP (Critique-and-Replan for Interactive Social Presence), a framework for autonomously

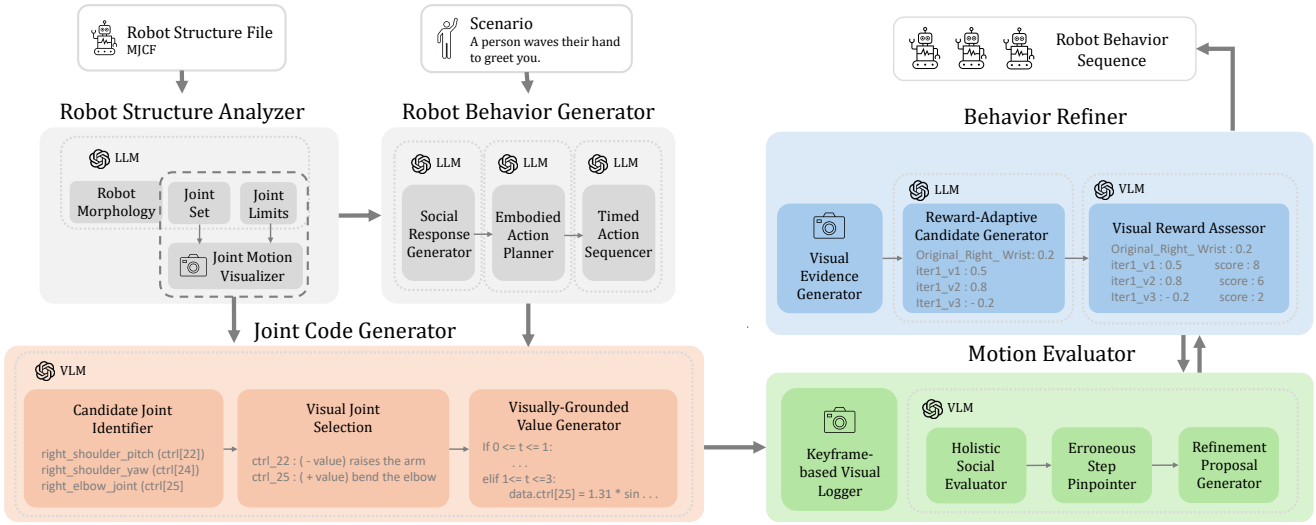

Fig. 2. Overview of the proposed social behavior generation framework. The framework parses robot morphology, plans social behaviors, generates low-level joint commands, and uses a VLM critic to iteratively evaluate and replan actions.

generating and refining flexible social behaviors that are not dependent on a specific robot. Fig. 2 illustrates the overall system architecture. The system consists of five core modules: a Robot Structure Analyzer, a Robot Behavior Generator, a Joint Code Generator, a Motion Evaluator, and a Behavior Refiner. In this section, we will describe the technical details of each module in sequence.

A. Robot Structure Analyzer

The Robot Structure Analyzer is responsible for automatically analyzing the robot's physical structure and kinematic constraints, and extracting key information for subsequent modules. It is designed to automate the manual analysis and motion function design process for each robot, which was required in previous studies, thereby ensuring the system's general applicability. The analysis process consists of two stages.

Structural Specification Analysis using LLM. First, the specification file containing the robot's structural information (in this study, MJCF, as we use the MuJoCo simulator) is taken as input. The LLM parses this file to automatically extract the following three pieces of information:

- Joint Set (J): It identifies the set of all actively controllable joint axes of the robot, $J = \{j_1, j_2, ..., j_n\}$.
- Joint Limits: For each joint $j_i \in J$, it extracts the minimum and maximum permissible angle range $[L_i^{\min}, L_i^{\max}]$.
- Robot Morphology: It provides a textual summary of the robot's overall degrees of freedom and the connectivity of its body parts. For example, for the G1 robot shown in Fig. 5, it generates information like "Each arm has 7-DOF, and the waist can move along the pitch, roll, and yaw axes." This morphological information is later used by the Robot Behavior Generator to devise actions appropriate for that specific robot.

Simulation-based Visualization of Joint Movements. The extracted joint information (J and $[L_i^{\min}, L_i^{\max}]$) is used to build a visual dataset D_{visual}. This is to overcome the limitation that LLMs struggle to predict intuitive spatial relationships or the results of movements from text information alone. Fig. 3 shows an example for a wrist joint. For each joint j_i, we capture three configurations: the simulator default (center panel) and two samples at $\pm 50\%$ of the joint's half-range around the default value (side panels). Each panel includes both a full-body view and a zoomed-in crop of the actuated region to reveal subtle local motion. The resulting set $D_{\text{visual}} = \{I_1, \ldots, I_n\}$, where each I_i is the set of visualization images for joint j_i, serves as a visual dictionary that maps joint-value changes to spatial movements, which the VLM in the Joint Code Generator consults to produce spatially plausible code.

B. Robot Behavior Generator

This module translates a human social action in a given context into a concrete sequence of robot behaviors. Inspired by the work of Mahadevan et al. [5], we adopted the idea of having an LLM generate actions while considering

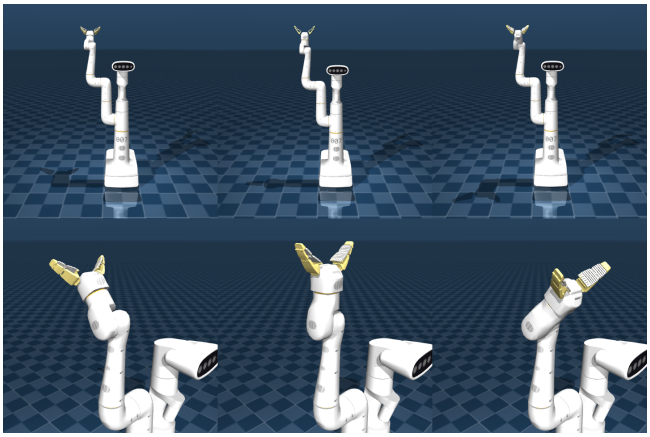

Fig. 3. Visualization of robot joint range of motion. Full-body and zoomed-in images for positive, zero, and negative values of the Everyday robot's wrist joint.

the robot's physical constraints. We further enhanced this by adding the capability to plan the estimated execution time for each action step, thereby improving the realism and flexibility of the generated behavior. First, the current interaction context (e.g., "A person waves their hand to greet you.") and the robot's morphology and capabilities, provided by the Robot Structure Analyzer, are input to the LLM. Based on this, the LLM translates the human social action (e.g., "Smile warmly and wave back to acknowledge their greeting.") into a form that the robot can actually perform. For example, for a robot incapable of smiling or turning its head, the action is specified as "Turn the torso to face the person, raise one arm, and use the wrist joint to wave." This translated robot action is then fed back into the LLM, which breaks down the entire behavior into smaller steps and assigns an estimated execution time to each. This results in a plan with temporal information, such as "Step 1: Rotate the waist slightly to the left to create a welcoming posture (0-1 s)," ensuring a natural flow of motion. Thus, this module generates a flexible social behavior plan that considers the robot's characteristics and temporal flow, and passes it to the next module.

C. Joint Code Generator

This module converts the step-by-step behavior plan (denoted as S_k, where k is the index of each behavior step) from the Robot Behavior Generator into executable code driving joint movement. It uses the set of movable joints (J), their limits ($[L_i^{\min}, L_i^{\max}]$), and the visual dataset (D_{visual}) from the Robot Structure Analyzer as input. The module employs a Vision-Language Model (VLM) that uses Chain-of-Thought (CoT) reasoning [23] to determine the appropriate joint position value v^* for each behavior step S_k.

Identify Relevant Joints. The VLM first interprets the behavior description S_k (e.g., "Tilt the head upward.") to identify a set of candidate joints $J_{candidate} \subseteq J$ relevant to the action. For this example, 'head yaw' and 'head pitch' would be selected as candidates.

Joint Selection through Visual Analysis. The VLM then consults the visual data $\{I_m | j_m \in J_{candidate}\}$ for the candidate joints. By visually analyzing the movements depicted in these images, it selects the joint j^* that best matches the description S_k. For instance, to "Tilt the head upward," the VLM chooses the 'head pitch' joint, whose image ($I_{HeadPitch}$) shows vertical motion, over the 'head yaw' joint ($I_{HeadYaw}$), which shows horizontal motion.

Value Generation based on Visual Reference. Using the visual data I_{j^*} for the selected joint, the VLM infers the final value v^* that matches the required intensity and direction from S_k. It references the positive and negative movement examples within I_{j^*}. For example, if the positive sample in $I_{HeadPitch}$ shows a moderate tilt (an angular position of 0.5 radians), but the command requires tilting further, the VLM will generate a value greater than 0.5. All generated values v^* are constrained within the joint's limits, $[L_{j^*}^{\min}, L_{j^*}^{\max}]$.

This process is repeated for all behavior steps, generating a set of joint control commands $C_k = \{(j^*, v^*), \dots\}$ for each step. Finally, the complete control sequence for the entire behavior, $C = \{C_1, C_2, \dots, C_m\}$, is passed to the next module.

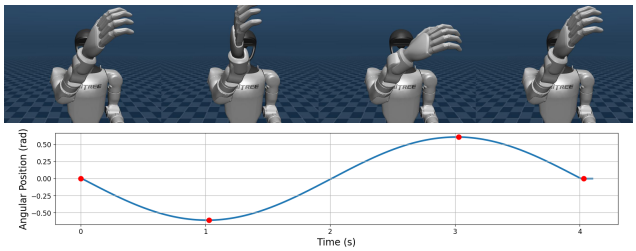

Fig. 4. Example of keyframe capture for continuous movement. Keyframes are captured where the angular velocity is zero. The graph below shows the change in wrist angle over time, with red dots indicating the captured moments.

D. Motion Evaluator

This module evaluates the social appropriateness of the generated joint code and proposes refinements if necessary. The process involves two main stages: visual log generation and a three-step evaluation.

Visual Log Generation. First, the generated joint code is executed in a simulator to create a visual log V, composed of keyframe images for each behavior step (S_k). Keyframes are captured as follows:

- For a single target pose (e.g., moving a shoulder to 0.7), one keyframe is captured upon completion.
- For continuous motion (e.g., a wave), keyframes are captured at points of zero angular velocity to represent the motion's flow. (See Fig. 4)

Each keyframe includes both full and zoomed-in views, similar to the Robot Structure Analyzer. This visual log V is then passed to the VLM for evaluation.

Holistic Social Appropriateness Evaluation. First, the VLM performs a holistic evaluation using only the context and visual log (V). It assesses the motion based on social appropriateness, human-likeness, and gesture completeness as guided in the prompt (please refer to the supplementary webpage for the full prompt). For instance, if a 'hand wave' lacks wrist motion, the VLM might critique it for not being human-like: "The wrist needs to be waved for a natural greeting." If no issues are found, the behavior is deemed "highly appropriate," and the process concludes.

Pinpointing the Erroneous Step. If a critique is generated, the VLM then pinpoints the erroneous step. It semantically compares the critique with each step in the behavior plan $\{S_1, \dots, S_m\}$ to find the most relevant step index, k^*. If multiple steps are relevant, the earliest one is chosen to efficiently manage cascading effects on subsequent actions. We empirically validate this design choice by comparing against a variant that refines the entire plan at once in Section IV-C.

Generating Specific Refinement Proposals. Finally, the VLM generates a specific refinement plan, A_{refine}, for the problematic step S_{k^*} based on the critique. It first identifies all related joints as candidates. By referencing their visual

data (D_{visual}), it determines the best way to modify the motion. The resulting plan, A_{refine}, takes one of three forms:

- *Adjust*: Modify the value of an existing joint command.
- *Delete*: Remove an unnecessary joint command.
- *Add*: Add a new, missing joint command.

This refinement plan is then recorded in the Behavior Artifact file, making it available to the next module.

E. Behavior Refiner

This module explores the optimal robot behavior by modifying the joint code based on refinement proposals (A_{refine}) from the Motion Evaluator and iteratively evaluating the results. This process operates similarly to concepts in Reinforcement Learning, and we formalize it as the Reward-based Adaptive Search (RAS) algorithm. We define each iteration of the refinement process as t.

Initialization and Candidate Value Generation (Initialization: $t = 0$). When the Behavior Artifact file and the refinement instruction (A_{refine}) are received, the LLM generates an initial set of candidates $V_0 = \{v_{0,1}, v_{0,2}, v_{0,3}\}$ for the erroneous step (S_k*) where the subscript 0 denotes the initialization iteration ($t = 0$). In the first iteration, values are explored with a wide range (σ_0) within the joint's limits $[L_{j^*}^{\min}, L_{j^*}^{\max}]$. Two of the three candidates follow the direction from A_{refine}, while the third is generated in the opposite direction to avoid local optima. Each value is saved to the Step Log file for that step.

Evaluation and Iterative Exploration ($t \geq 1$).

1) Visual Evidence Generation and Reward Evaluation: The generated candidates are passed to the capture module to create a visual log. For fluid motions, keyframes are captured when the angular velocity is zero. Each image is fed into the VLM, which assesses the movement by comparing it to the given goal and assigns a reward score $R(v_{t-1,i})$ between 1 and 10.

- $R_i \in [8, 10]$: Successful movements.
- $R_i \in [5, 7]$: Correct direction but incomplete.
- $R_i \in [3, 4]$: Incorrect but not opposite direction.
- $R_i \in [1, 2]$: Opposite direction.

The value with the highest reward and the reward itself are defined as v_{t-1}^* and R_{t-1}^*, respectively:

$$R_{t-1}^* = \max_i R(v_{t-1,i}),$$

$$v_{t-1}^* = \arg\max_{v \in V_{t-1}} R(v).$$

2) Termination Condition Check & Reward-based Search Width Update: The algorithm terminates if $R_{t-1}^* \geq \tau$ (we set τ to 8 in our experiment), at which point v_{t-1}^* is finalized in the Behavior Artifact file and returned to the Motion Evaluator.

If the reward is less than τ, the VLM adaptively adjusts the search width σ_t based on the maximum reward R_{t-1}^*.

$$\sigma_t = \begin{cases} \alpha \cdot \sigma_{\text{base}} & \text{if } 5 \leq R_{t-1}^* < \tau \quad \text{(fine-grained search)} \\ \beta \cdot \sigma_{\text{base}} & \text{if } R_{t-1}^* < 5 \quad \text{(broad search)} \end{cases}$$

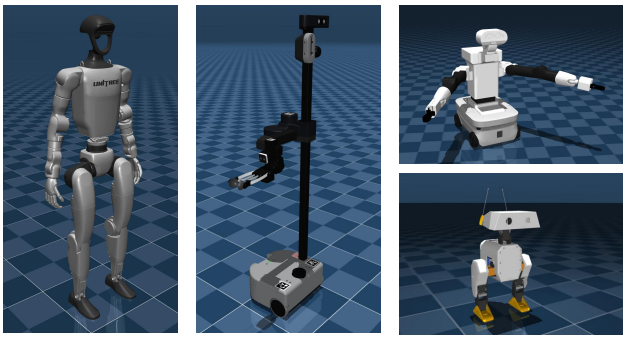

Fig. 5. Types of robots used in the experiment: Unitree G1, Stretch 3 (left), TIAGo (top right), Open Mini Duck (bottom right). The Everyday robot is shown in Fig. 3. The proposed method can generate social behaviors for a wide range of robots, from simple ones to humanoids.

Here, σ_{base} is the base search width, and α and β are scaling factors for fine-grained and broad searches (e.g., $0 < \alpha < 1, \beta > 1$). For instance, if a reward of 5 or higher is received, the search width is reduced for fine-tuning. Conversely, if the reward is low, the width is increased for a broader search. The parameters α_{base}, α and β are empirically determined for each robot, considering the general characteristics of its joints' ranges of motion. for G1 robot whose key joints operate in a wide range (≈4 rad on average), we set σ_{base}=0.6, α=0.4, and β=1.5.

3) New Candidate Value Generation and Iteration: In each iteration, we sample three new candidates—a number chosen to balance exploration diversity against evaluation cost—from a normal distribution $\mathcal{N}(v_{t-1}^*, \sigma_t^2)$ and clipped to be within the joint's limits $[L^{\min}, L^{\max}]$.

$$\tilde{v}_{t,i} \sim \mathcal{N}(v_{t-1}^*, \sigma_t^2) \quad \text{for } i = 1, 2, 3$$

$$v_{t,i} = \text{clip}(\tilde{v}_{t,i}, L^{\min}, L^{\max})$$

This new candidate set V_t is then sent back for repeated evaluation.

4) Search Failure Diagnosis: If rewards are consistently low ($R_{t-1}^* \leq 2$), the process returns to the Motion Evaluator to search for a new joint. All refinement processes and reward histories are recorded in the Behavior Artifact file to manage the search history and prevent infinite loops.

IV. Experiments

To verify whether our proposed VLM-based replanning system, CRISP, provides a superior user experience, we conducted a human-subject evaluation. For the experiments in this paper, we used OpenAI's GPT-4o (Accessed in August 2025) as both the VLM and LLM, with the temperature set to 0.

A. Experiment Setup

The user evaluation was conducted by having participants watch video clips of robot behaviors generated by different systems for specific scenarios and then rate their quality. To assess the generalization performance of the proposed system across robots with different morphologies, we used the following five types of robots: Google Research's Everyday Robot (mobile manipulator), Hello Robot Stretch 3 (mobile

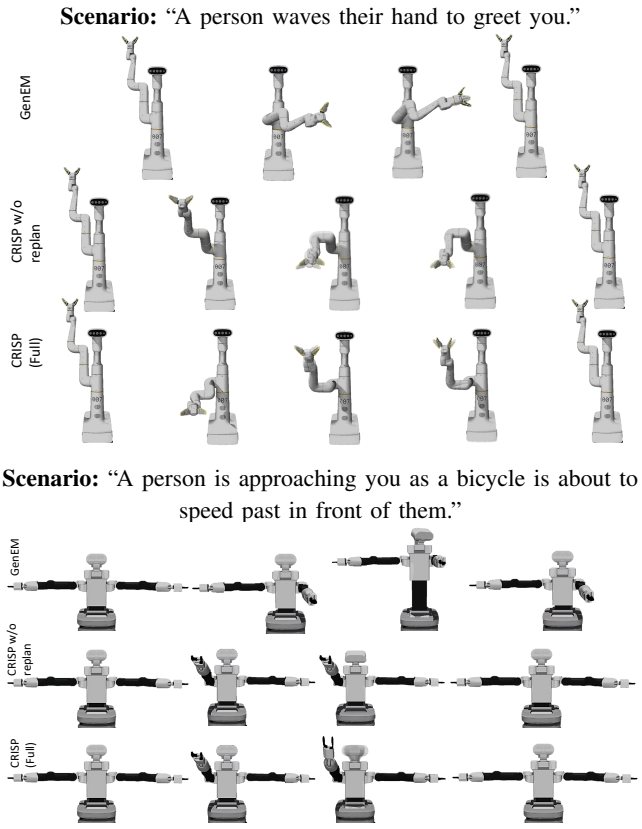

Fig. 6. Sample results of generated motions for Everyday robot and TIAGo for given scenarios. The first row shows the result from GenEM, the second row from CRISP w/o replan, and the third row from CRISP (Full). See Supplementary Video.

manipulator), Open Mini Duck (an open-source project based on Disney Research's BDX Droid, a simple duck-shaped robot), PAL Robotics TIAGo (dual-arm manipulator), and Unitree G1 (humanoid robot) (see Fig. 5 and 3).

For each robot, we designed four types of scenarios to evaluate behavior generation capabilities from multiple perspectives: 1. Simple, well-defined scenarios (e.g., "A person waves their hand to greet you."). 2. Simple, but open-ended scenarios related to emotional responses (e.g., "A person sighs deeply in disappointment in front of you."). 3. Scenarios where a person gives a clear command (e.g., "A person commands you to move your torso up and down and swing your arms."). 4. Complex, open-ended scenarios involving ambivalent emotions or difficult actions (e.g., "A person waits for an important online result, fidgeting excitedly with a hopeful expression. After seeing the result on the screen, their face falls, their shoulders slump, and they sink into their chair."). In these scenarios, we evaluated the performance of our complete system, CRISP with replan, against two baselines. The first baseline was the GenEM model [5], which we modified for a fair comparison. Specifically, instead of using its native Robot API to generate code, we adapted the model to produce direct joint-level commands using the joint information (Joint Set and Joint Limits) provided by our Robot Structure Analyzer. The second baseline was a version of our system without its core components: the Motion

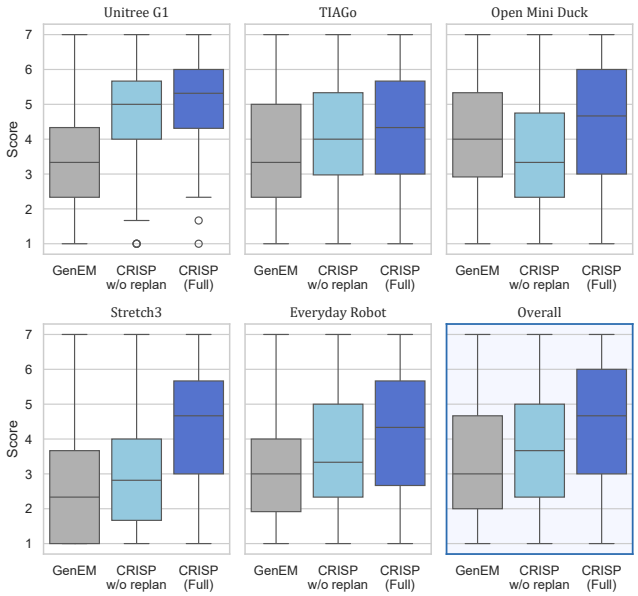

Fig. 7. User study results comparing the baseline GenEM, our method without replanning (CRISP without replan), and the full proposed method (CRISP). The box plot labeled 'Overall' (bottom right) represents the overall average for each system. Scores are aggregated across Q1, Q2, and Q3.

Evaluator and the replanning loop (CRISP w/o replan).

To present the evaluation materials to participants, all behaviors corresponding to each robot, scenario, and system combination were recorded as simulation video clips. To ensure consistency, all videos were filmed with the same background and from the same position. After watching each video clip, participants rated the robot's behavior on a 7-point Likert scale across three metrics: (Q1) ease of understanding, (Q2) situational appropriateness, and (Q3) likeability. To prevent fatigue and loss of concentration that could arise from one participant evaluating all five robots, each participant was randomly assigned to evaluate three of the five robots. We used a balanced incomplete block design to ensure fair distribution of robots among participants and a balanced Latin square design to minimize bias from the order of video presentation. Consequently, each participant completed 108 evaluations in total (3 robots $\times$ 4 scenarios $\times$ 3 systems $\times$ 3 questions).

B. Human Study Results

We collected responses through the crowdsourcing platform Prolific. The survey was conducted with 50 participants from English-native countries (28 male, 22 female; mean age 42.2 $\pm$ 11.5 years). A simple attention check was included to exclude inattentive participants. Fig. 7 presents the mean scores aggregated across all scenarios and metrics (Q1–Q3) for each robot. Given the high consistency across Q1–Q3 (Cronbach's $\alpha > .92$), results are presented as an aggregate score of the three metrics. The 'Overall' plot represents the overall average for each system. The analysis revealed that CRISP had a clear advantage in user preference over the other two systems. The mean preference scores were highest for CRISP at 4.5 ($\pm$1.11), followed by CRISP w/o replan at 3.79 ($\pm$1.16), and GenEM (modified version) at

Scenario: "A person is looking for a seat, and you guide them to an empty one on your right."

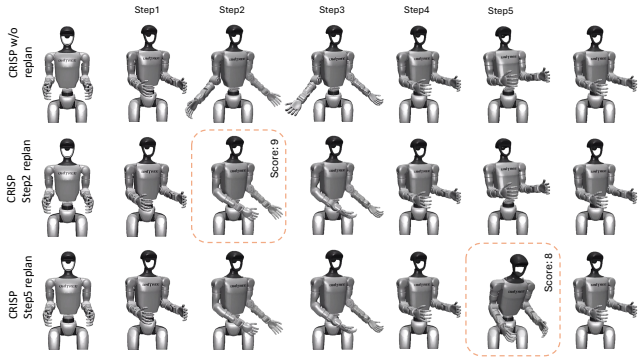

Fig. 8. Evolution of a plan through replanning. The first row is the initial plan (CRISP w/o replan). The second and third rows show subsequent revisions. The boxed steps are the actions modified during replanning (at steps 2 and 5), with the score in each box indicating the reward assigned by the VLM for the new action.

3.4 ($\pm$1.13). To verify the significance of the difference in preference among the three systems, a Wilcoxon signed-rank test with Holm-Bonferroni correction was performed for pairwise comparisons, which confirmed statistically significant differences between all pairs ($p < .001$).

Question-wise analysis showed that the mean scores across the three questions were broadly similar (GenEM: Q1=3.31, Q2=3.46, Q3=3.41; CRISP w/o replan: Q1=3.77, Q2=3.84, Q3=3.75; CRISP: Q1=4.50, Q2=4.64, Q3=4.35). Importantly, the same ranking (CRISP $>$ CRISP w/o replan $>$ GenEM) was consistently observed for all three questions, underscoring the robustness of the improvement.

Please see Fig. 6 and supplementary video for the sample results. Fig. 8 shows how the behavior evolves through replanning.

C. Ablation Study

To verify the impact of each component of our proposed system on its overall performance, we conducted an ablation study. We compared CRISP with four variant models to evaluate the contribution and necessity of each component: (M1) a model where the Behavior Refiner generates only a single action, (M2) a model that refines the entire plan at once, (M3) a model that takes multi-view images (Fig. 9) as input, and (M4) a model that excludes the zoomed-in shot from the standard VLM input (Fig. 4). For a fair comparison, the number of replan iterations for all models was limited to 10, and the same Unitree G1 robot was used. Ten evaluators watched five generated videos per model for the scenario "A person waves their hand to greet you" and rated the task success as 'Success', 'Unsure', or 'Failure'. For analysis, 'Unsure' was treated as 'Failure'. The success count in Table I is calculated by summing all 'Success' ratings from the ten evaluators and dividing by the number of runs (five).

Table I shows the results of the ablation study. The proposed system achieved an average success count of 4.6 out of 5 runs, using an average of 10.2 VLM input images and 3.4 replan iterations to achieve its goal, demonstrating high efficiency. The utility of each module in the proposed

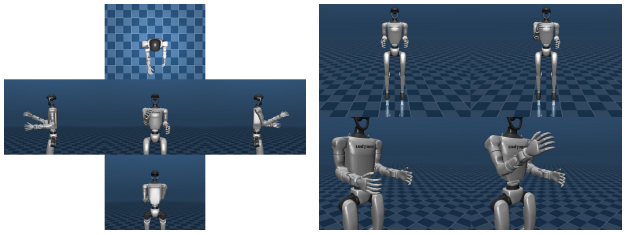

Fig. 9. Example VLM input images. (Left) Temporally-sampled multi-view input for the M3 model. (Right) Our proposed keyframe-based, dual-view (full and zoom shot) input.

TABLE I

ABLATION STUDY RESULTS. AVERAGE VALUES OVER 5 RUNS.

Model	Success Count	# VLM Input Images	# Replans
Proposed	4.6	10.2	3.4
M1	1.7	9.4	9.4
M2	0.9	140.0	10.0
M3	3.7	41.2	3.4
M4	2.6	13.8	4.4

system can be confirmed by comparing it with the variant models. M1, which only refines a single action, had a similar number of input images (9.4) to the proposed model but required 9.4 replan iterations and achieved only 1.7 successes. This suggests it fell into a local optimum and performed inefficient searches. M2, which refined the entire plan at once, had the lowest success count of 0.9, used 140.0 input images, and reached the maximum of 10.0 replan iterations, demonstrating its significant inefficiency. M3, using multi-view images, showed a relatively high success rate of 3.7 but was inefficient, requiring 41.2 images to achieve its goal–more than four times the visual information needed by the proposed model. Finally, M4, which excluded the crucial zoomed-in shots, saw its success count drop to 2.6, clearly showing that accurately perceiving the robot's fine movements is vital for task success.

V. DISCUSSION

The proposed replan model demonstrated its superiority by achieving the highest user preference in most scenarios. CRISP's performance improvement was especially prominent with the physically complex Unitree G1, suggesting that our use of visual information plays a key role in planning subtle postures and movements for complex robots.

Despite the overall superiority of the replan model (average user rating CRISP: 4.5, GenEM: 3.4), exceptional results were observed in scenario 2 for the Open Mini Duck (CRISP: 3.9, GenEM: 4.4) and TIAGo robots (Open Mini Duck: "A person is in front of you with a sad expression.", TIAGo: "A person cowers in fear in front of you."). These results are likely attributable to the unique nature of scenario 2, which involves open-ended social interactions. This scenario evaluates the robot's response to user emotions, and human expectations for a response to emotions like sadness are subjective and varied. For example, some users might expect empathy and comfort, while others might prefer a humorous action to change the mood. In such situations where there is

no single correct answer, it is difficult for a VLM-generated behavior plan to satisfy every user's individual expectations. This contrasts with the universally high performance of the models in scenario 3, which had a clear objective. This indicates that while the proposed system is highly effective for tasks with objective goals, adapting to the subjective and complex social contexts of humans is a challenge that needs to be addressed in future research.

Additionally, several technical aspects suggest directions for future work. First, as current VLMs evaluate spatial accuracy via static images, they lack temporal awareness (e.g., speed or smoothness). Future research should incorporate video processing to refine these temporal elements. Second, while this study focused on upper-body interaction, whole-body dynamics like walking remain beyond the current system's scope. Implementing such behaviors would likely require a hybrid approach that integrates our framework with physics-informed control or existing API-based methods.

Regarding implementation, the current system takes $\sim$3 min for initial motion generation (Joint Structure Analyzer: $\sim$1 min, Robot Behavior Generator: $\sim$20 s, Joint Code Generator: $\sim$1.5 min). The replanning process requires 10–20 min depending on the complexity and iterations (avg. $\sim$2.5 min per replan), making it ideally suited for offline robot behavior authoring. This allows developers to pre-generate high-quality social motions without manual key-framing. Moreover, the framework establishes an autonomous pipeline for creating large-scale training datasets, which can be applied to Vision-Language-Action (VLA) models to facilitate real-time generation. Ultimately, this research serves as a vital foundation for securing the real-time social interaction capabilities of robots, advancing VLM-based behavior generation toward practical, real-world deployment.

VI. Conclusion

In this paper, we presented CRISP, an autonomous replanning framework that employs a VLM as a social critic to enable robotic self-correction of social behaviors. By leveraging low-level control based on robot structure files, the framework achieves high flexibility and generalizability. Our user studies with five distinct robot types validated that CRISP generates more appropriate and socially preferred behaviors than existing methods, and an ablation study confirmed the efficacy of each component. We believe the proposed framework marks a step forward toward socially aware robots that autonomously adapt and refine their behaviors in human-centric environments, reducing reliance on costly human feedback and broadening cross-platform applicability. Remaining challenges include handling subjective social contexts, improving temporal expressiveness, and achieving real-time performance. Addressing these limitations will further advance the development of interactive robots capable of rich, autonomous social presence.

Acknowledgments

Illustrations in Fig. 1 were generated by Google Gemini.

References

[1] H. B. Mohammadi, N. Xirakia, F. Abawi, I. Barykina, K. Chandran, G. Nair, C. Nguyen, D. Speck, T. Alpay, S. Griffiths *et al.*, "Designing a personality-driven robot for a human-robot interaction scenario," in *ICRA*. IEEE, 2019.

[2] D. Porfirio, A. Sauppé, A. Albarghouthi, and B. Mutlu, "Transforming robot programs based on social context," in *CHI*, 2020.

[3] A. Aly and A. Tapus, "A model for synthesizing a combined verbal and nonverbal behavior based on personality traits in human-robot interaction," in *HRI*. IEEE, 2013.

[4] P. David, M. Cakmak, A. Sauppé, A. Albarghouthi, and B. Mutlu, "Interaction templates: a data-driven approach for authoring robot programs," in *PLATEAU: 12th Annual Workshop at theIntersection of PL and HCI*, 2022.

[5] K. Mahadevan, J. Chien, N. Brown, Z. Xu, C. Parada, F. Xia, A. Zeng, L. Takayama, and D. Sadigh, "Generative expressive robot behaviors using large language models," in *HRI*, 2024.

[6] J. Park, T. Jeong, H. Kim, T. Byun, S. Shin, K. Choi, J. Kwon, T. Lee, M. Pan, and S. Choi, "Towards embedding dynamic personas in interactive robots: Masquerading animated social kinematic (mask)," *IEEE Robotics and Automation Letters*, 2024.

[7] K. Weber, H. Ritschel, I. Aslan, F. Lingenfelser, and E. André, "How to shape the humor of a robot-social behavior adaptation based on reinforcement learning," in *ICMI*, 2018.

[8] R. Tian, Y. Wu, C. Xu, M. Tomizuka, J. Malik, and A. Bajcsy, "Maximizing alignment with minimal feedback: Efficiently learning rewards for visuomotor robot policy alignment," *arXiv:2412.04835*, 2024.

[9] A. Mei, G.-N. Zhu, H. Zhang, and Z. Gan, "Replanvlm: Replanning robotic tasks with visual language models," *IEEE Robotics and Automation Letters*, 2024.

[10] Y. Chen, W. Cui, Y. Chen, M. Tan, X. Zhang, J. Liu, H. Li, D. Zhao, and H. Wang, "Robogpt: an llm-based long-term decision-making embodied agent for instruction following tasks," *IEEE Transactions on Cognitive and Developmental Systems*, 2025.

[11] M. Destephe, M. Brandao, T. Kishi, M. Zecca, K. Hashimoto, and A. Takanishi, "Walking in the uncanny valley: Importance of the attractiveness on the acceptance of a robot as a working partner," *Frontiers in psychology*, vol. 6, p. 204, 2015.

[12] M. Suguitan, R. Gomez, and G. Hoffman, "Moveae: modifying affective robot movements using classifying variational autoencoders," in *HRI*, 2020.

[13] Y. Shentu, P. Wu, A. Rajeswaran, and P. Abbeel, "From llms to actions: Latent codes as bridges in hierarchical robot control," in *IROS*, 2024.

[14] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou, "Information theoretic mpc for model-based reinforcement learning," in *ICRA*, 2017.

[15] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. P. Foster, P. R. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn, "Openvla: An open-source vision-language-action model," in *CoRL*, 2025.

[16] Q. Dong, L. Li, D. Dai, C. Zheng, J. Ma, R. Li, H. Xia, J. Xu, Z. Wu, B. Chang, X. Sun, L. Li, and Z. Sui, "A survey on in-context learning," in *EMNLP*, 2024.

[17] N. Oralbayeva, A. Aly, A. Sandygulova, and T. Belpaeme, "Data-driven communicative behaviour generation: A survey," *ACM Transactions on Human-Robot Interaction*, vol. 13, no. 1, pp. 1–39, 2024.

[18] C.-M. Huang and B. Mutlu, "Robot behavior toolkit: generating effective social behaviors for robots," in *HRI*, 2012.

[19] Y. Yoon, W.-R. Ko, M. Jang, J. Lee, J. Kim, and G. Lee, "Robots learn social skills: End-to-end learning of co-speech gesture generation for humanoid robots," in *ICRA*, 2019.

[20] W. Wang, I. Obi, and B.-C. Min, "Multi-agent llm actor-critic framework for social robot navigation," *arXiv:2503.09758*, 2025.

[21] M. Skreta, Z. Zhou, J. L. Yuan, K. Darvish, A. Aspuru-Guzik, and A. Garg, "Replan: Robotic replanning with perception and language models," *arXiv:2401.04157*, 2024.

[22] P. Huang, Y. Hu, N. Nechyporenko, D. Kim, W. Talbott, and J. Zhang, "Emotion: Expressive motion sequence generation for humanoid robots with in-context learning," *IEEE Robotics and Automation Letters*, 2025.

[23] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," in *NIPS*, 2022.