Pragmatic Instruction Following and Goal Assistance via Cooperative Language-Guided Inverse Planning

Tan Zhi-Xuan
Massachusetts Institute of Technology
xuan@mit.edu

Vikash Mansinghka
Massachusetts Institute of Technology
vkm@mit.edu

Lance Ying∗
Harvard University
lanceying@seas.harvard.edu

Joshua B. Tenenbaum
Massachusetts Institute of Technology
jbt@mit.edu

ABSTRACT
People often give instructions whose meaning is ambiguous without further context, expecting that their actions or goals will disambiguate their intentions. How can we build assistive agents that follow such instructions in a flexible, context-sensitive manner? This paper introduces cooperative language-guided inverse plan search (CLIPS), a Bayesian agent architecture for pragmatic instruction following and goal assistance. Our agent assists a human by modeling them as a cooperative planner who communicates joint plans to the assistant, then performs multimodal Bayesian inference over the human’s goal from actions and language, using large language models (LLMs) to evaluate the likelihood of an instruction given a hypothesized plan. Given this posterior, our assistant acts to minimize expected goal achievement cost, enabling it to pragmatically follow ambiguous instructions and provide effective assistance even when uncertain about the goal. We evaluate these capabilities in two cooperative planning domains (Doors, Keys & Gems and VirtualHome), finding that CLIPS significantly outperforms GPT-4V, LLM-based literal instruction following and unimodal inverse planning, while correlating strongly with human raters.

KEYWORDS
Inverse Planning, Bayesian Theory-of-Mind, Instruction Following, Human-Robot Cooperation, Value Alignment

1 INTRODUCTION
Humans act upon the world through our words. We make requests, give instructions, and communicate information, so that we can better coordinate with each other [4]. In doing so, we are often parsimonious, exploiting context to convey our intentions [20]. For example, if someone says “Can you hold that door?”, it is typically obvious which door they mean, even though nothing in the sentence distinguishes the door they just walked past from the one they are heading towards. We understand these requests because we interpret them pragmatically, in light of the goals and actions of others. How might we build assistive machines that do the same?

In this paper, we introduce cooperative language-guided inverse plan search (CLIPS), a Bayesian architecture for pragmatic instruction following and goal assistance (Figure 1). Building upon prior work in inverse planning [7, 44], rational speech act theory [16, 19, 53], joint intentionality [52, 56, 63], assistance games [15, 21], and reward learning [25, 32, 43], CLIPS models humans as cooperative planners who communicate joint plans as instructions. Given this model, CLIPS performs multimodal goal inference from human actions and instructions, using a large language model (LLM) to score the likelihood of observed utterances [64], and computing a distribution over goals via sequential inverse planning [67]. This distribution then informs an assistive policy, which selects helpful actions under uncertainty about the human’s goal.

We evaluate CLIPS on a suite of multi-step goal assistance problems in a doors-and-keys gridworld [67] and the VirtualHome domain [41, 42]. In these problems, the assistant must infer the human’s goal from their actions and possibly ambiguous instructions, then decide how best to help. Even when leveraging LLMs, standard instruction following methods struggle with this setting because they disregard pragmatic context [2, 50, 60], while action-only goal inference [7, 42, 67] ignores linguistic information. Multimodal LLMs have access to all information, but they fail to ground it in a coherent theory-of-mind [11, 26]. In contrast, CLIPS is able to use observed actions and inferred goals to resolve ambiguous language, interpret joint instructions, and correct for incomplete commands, achieving much higher goal accuracy and cooperative efficiency than GPT-4V, LLM-based literal instruction following, and unimodal inverse planning, while correlating strongly with goal inference and assistance judgments provided by human raters.

2 COOPERATIVE LANGUAGE-GUIDED INVERSE PLAN SEARCH
We formulate the setting for CLIPS as a language-augmented goal assistance game, an extension of assistance games [15, 21] with linguistic utterances and uncertainty over goals (i.e. desired terminal states) [46]. We define this as a two-player Markov game between a human principal and an assistive agent, described by
the tuple \((S, U, A^h, A^r, C, G, H, P_s, P_t)\), where \(S\) is the set of environment states, \(U\) a space of utterances \(u\) that the human may use to communicate at any step, \(A^h\) the set of human actions \(a^h\), \(A^r\) the set of assistant actions \(a^r\), \(C\) a set of cost functions \(C: S \times U \times A^h \times A^r \to \mathbb{R}\) that map state-action transitions to real numbers, \(G \subseteq \mathcal{P}(S)\) a set of possible goals \(g\) where each \(g \subseteq S\) is a set of (terminal) states, \(H\) a horizon after which the game automatically terminates, \(P_t(s', t | s, a^r, a^h)\) the environment transition distribution, and \(P_i(s_t, C, g)\) a distribution over initial states of the game. As in regular assistance games, the human knows (i.e. observes) the true cost function \(C\) and goal \(g\) sampled from \(P_i\), but the assistant only observes the initial environment \(s_t\). Thus, the assistant has to infer the true goal \(g\).

Assistance games admit cooperative equilibria, but they are intractable to compute in general [15], and also assume more game-theoretic optimality from humans than may be warranted [37]. As such, our goal is to approximately solve the assistance game under reasonable modeling assumptions about how humans cooperate and communicate. By fixing a human model, the game becomes a partially observable Markov decision process (POMDP) from the assistant’s perspective [14], which can be solved through approximate methods [35, 39]. We describe this human model, then show how an assistant can perform Bayesian inference over such a model, using the acquired information to better assist humans.

### 2.1 Modeling cooperative action and communication

When humans cooperate, we direct our actions towards a shared goal while expecting that others will do the same. This capacity for joint intentionality [58] allows us to cooperate rapidly and flexibly while avoiding nested reasoning about each other’s minds [56, 63]. In CLIPS, we exploit this aspect of human cognition by modeling the human principal as a cooperative planner, who computes a joint policy \(\pi\) for both agents to achieve the goal \(g\). The human follows \(\pi\) by taking an action \(a^h_t\) at each step \(t\), and assumes the assistant will take an action \(a^r_t\) accordingly. These actions lead to a change in the state \(s_t\). The human may also speak at any time \(t\) (with the decision denoted by \(d_t\)), issuing a command \(c_t\) that summarizes their intended policy \(\pi\). This command is then rendered in natural language as an utterance \(u_t\).

The overall generative process is shown in Figures 2a–b, and can be summarized as follows:

- **Goal Prior:**
  \[
  g \sim P(g)
  \]

- **Joint Planning:**
  \[
  \pi \sim P(\pi | g)
  \]

- **Action Selection:**
  \[
  a^h_t, a^r_t \sim P(a^h_t | a^r_t, s_t, \pi)
  \]

- **Utterance Generation:**
  \[
  u_t, c_t, d_t \sim P(u_t, c_t, d_t | s_t, \pi)
  \]

- **State Transition:**
  \[
  s_{t+1} \sim P(s_{t+1} | s_t, a^h_t, a^r_t)
  \]

\(P(g)\) is the assistant’s prior over the human’s goal \(g \in G\), which we take to be uniform. We can also fold in uncertainty over the human’s cost function \(C \in C\) into this prior, treating \(g\) as a specification that includes both the goal condition and action costs.

To model joint planning, we assume that the human computes a Boltzmann policy \(\pi\) for goal \(g\), which defines an approximately rational distribution over joint actions \(a^h_t, a^r_t\):

\[
\pi(a^h_t, a^r_t | s_t) = \frac{\exp(-\beta Q^\pi(s_t, a^h_t, a^r_t))}{\sum_{a^h_t, a^r_t} \exp(-\beta Q^\pi(s_t, a^h_t, a^r_t))}
\]

Here, \(\beta\) is a rationality parameter controlling the optimality of the policy (higher \(\beta\) is more optimal), under which we may place a prior \(P(\beta)\), and \(Q^\pi(s_t, a^h_t, a^r_t)\) is an estimate of the expected cost of reaching the goal \(g\) by taking actions \(a^h_t, a^r_t\) at state \(s_t\). To estimate \(Q^\pi\) efficiently, we extend prior work in online goal inference [67], using model-based planning to compute policies on-the-fly. In particular, we use real-time heuristic search (RTHS) as an anytime planner [8, 27, 28], which estimates \(Q\)-values in a neighborhood around the current state \(s_t\) via search (RTHS-POLICY-UPDATE in Fig. 2b), while using previously estimated \(Q\)-values to guide future searches. Unlike earlier methods for inverse planning, this avoids the intractability of estimating \(Q\) for all states and actions [43, 68].

1As an additional simplification, we assume that the human and the assistant take turns while acting (i.e. \(a^r_t\) is a no-op when \(a^h_t\) is not, and vice versa). This reduces the branching factor while planning, but preserves the optimal solution [9].
model utterance-model(s_t, π)
parameters: p_speak, L, K, E

\[ d_t \sim \text{BERNOULLI}(p_{\text{speak}}) \]

if \( d_t = \text{TRUE} \)

\[ a_{t:t+L} \leftarrow \text{ROLLOUT-POLICY}(s_t, π, H) \]
\[ a_{t:m} \leftarrow \text{EXTRACT-SALIENT-ACTIONS}(a_{t:t+H}) \]
\[ c_t \sim \text{RAND-SUBSET}(a_{t:m}, K) \]
\[ u_t \sim \text{LANGUAGE-MODEL}(c_t, E) \]

else

\[ c_t, u_t \leftarrow 0, "\text{null}" \]

end if

end model

(c) Utterance model \( P(u_t, c_t, d_t | π) \) as a probabilistic program.

Command: (handover you me key1) (handover you me key2) where (iscolor key1 blue) (iscolor key2 red)

Utterance: Can you pass me the blue key?

Command: (unlock you key2 door1) where (iscolor key1 green) (iscolor door1 green)

Utterance: I’m getting the blue key, can you open the green door?

Command: (handover you me key1) (unlock you key2 door1) where (iscolor key1 blue) (iscolor door1 green)

Utterance: Can you unlock the door for me?

Command: (handover you me key2) where (iscolor key2 blue)

Utterance: Hand me the blue key.

(d) Paired examples \( E \) of commands \( c_t \) and utterances \( u_t \).

Having computed the policy \( π \), we model human communication as an approximately pragmatic process. Since the human wants to achieve their goal, they are likely to convey information about the policy they have in mind. However, long utterances are costly, and so the human is likely to convey only a salient summary of the policy, reducing the cost of communication. We capture these aspects of cooperative communication through a structured generation procedure that leverages the linguistic competence of LLMs:

1. At each step \( t \), the human decides whether to communicate \( (d_t) \) with probability \( p_{\text{speak}} \). If \( d_t \) is true, we rollout the policy \( π \) for \( L \) steps, producing a series of planned actions \( a_{t:t+H} \).

2. The planned actions \( a_{t:t+L} \) are filtered down to a domain-specific set of salient actions \( a_{t:m} \), and a random subset of up to \( K \) such actions are incorporated into a command \( c_t \).

3. Finally, the command \( c_t \) is translated into a natural language utterance \( u_t \) using an LLM prompted with few-shot examples (Figure 2d). This defines a likelihood function \( P(u_t | c_t) \).

This procedure is implemented as the probabilistic program shown in Figure 2c. While it does not capture all ways in which a pragmatic speaker might prefer some utterances over others, key features of pragmatic speech are modeled: Utterances are about plans, and hence practically useful, but are also restricted to small (\( \leq K \)) but salient subsets of those plans, ensuring they are informative without being costly. Furthermore, using an LLM as an utterance model \( P(u_t | c_t) \) provides broad coverage, enabling CLIPS to handle much richer utterances than traditional pragmatic models [19].

2.2 Goal inference via inverse planning from actions and instructions

Using this probabilistic model, our assistant can infer a posterior distribution over the human’s goals \( g \) and policies \( π \) given a sequence of observed actions \( a_{t:T} \), states \( s_{t:T} \) utterance decisions \( d_{t:T} \), and utterances \( u_{t:T} \) (where \( u_t \) is the empty string whenever \( d_t = \text{false} \)). Some care must be taken, however. Since the assistant is also taking actions \( a_{t:T} \) alongside the human principal, it is tempting to infer the human’s goal by conditioning on both agents’ actions:

\[
P(g, π | s_{t:T}, d_{1:T}, u_{1:T}, a_{t:T}, a'_{t:T})
\]

\[
\propto P(g, π, s_{t:T}, d_{1:T}, u_{1:T}, a_{t:T}, a'_{t:T})
\]

\[
= P(g, π) \prod_{t=1}^{T} P(s_t | s_{t-1}, a'_{t-1:T}) P(u_t | d_t | s_t, π) P(a_t | s_t, π) P(a'_{t:T} | s_{t:T}, π)
\]

For an external observer, this is the appropriate distribution to compute [49, 64]. However, prior work on cooperative inference also computes Equation 7 when observers are internal to the environment [56, 63], which can lead to pathologies. For example, the assistant might condition on the fact that it has not moved so far \( (a'_{t:T} \text{ are no-ops}) \), leading it to infer that the human is pursing a goal \( g \) where the assistant’s help is unnecessary. To address this, it is crucial to recognize that the assistant is intervening upon the environment through its actions (Figure 2a, double-lined nodes).²

²Alternatively, it is enough to note the assistant’s actions are selected according to different policy \( π' \) than the inferred joint policy \( π \) computed by the human. Thus it is safer for the assistant not to update its beliefs as if its actions come from \( π \).
As such, we should detach the evidential connection between the assistant’s actions \(a^*\) and any causal ancestors, which we denote using Pearl’s do-operator \([40]::
\begin{align*}
P(g, \pi | s_{1:T}, d_{1:T}, u_{1:T}, a^*_{1:T}, \text{do}(a^*_{1:T})) \\
= P(g, \pi | T = P(s_1 | s_{1:t-1}, a^*_{t-1})P(u_t, d_t | s_t, \pi)P(a^*_{t} | s_t, \pi)
\end{align*}

The difference between Equations 7 and 8 lies in the final underlined term: Whereas an external observer reweights its beliefs by incorporating the likelihood of both agents’ actions \((a^*_t, a^*_t | s_t, \pi)\), our assistive agent should only incorporate the likelihood of the human’s action \(P(a^*_t | s_t, \pi)\).

We compute the distribution in Equation 8 sequentially, as shown in Algorithm 1. An initial belief \(b_0\) is returned by BELIEF-INIT, which generates a set of \(N\) weighted samples by either sampling or enumerating over the goal prior \(P(g)\) and policy distribution \(P(\pi | g)\) (accounting e.g. for uncertainty over the rationality parameter \(\beta\)). Then at each step \(t\), the previous belief \(b_{t-1}\) is updated by adjusting the weight \(w^t\) associated with each goal \(g^t\) and policy \(\pi^t\) sample. To do this, BELIEF-UPDATE first refines the \(Q\)-value estimates that define the policy \(\pi^t\) by running more iterations of RTHS (RTHS-POLICY-UPDATE in L9). The weights are then updated to reflect the new state \(s_t\) and decision to speak \(d_t\) (L10-11). If \(d_t\) is true (i.e. \(u_t\) is observed), we also marginalize over all commands \(c_t\) that could have generated \(u_t\) from \(\pi\), and update \(w^t\) with the resulting mixture likelihood (L13-14). Finally, we condition on the human’s action \(a^*_t\), and return the resulting sample collection (L16-18).

If we enumerate over all possible goals and policy configurations in BELIEF-INIT, this is an exact Bayesian inference algorithm. In environments where there are many other hypotheses for this to be efficient, Algorithm 1 can readily be extended to a sequential Monte Carlo algorithm [13, 30] in the style of sequential inverse plan search [67]. For our experiments, however, enumeration is feasible, and we opt for this approach to avoid variance from sampling.

### 2.3 Pragmatic instruction following as goal assistance under uncertainty

With Algorithm 1, our assistive agent is able to infer a distribution over the human’s goal \(g\) and policy \(\pi\) at each step \(t\). How should it use this information to act, especially if the goal \(g\) remains uncertain? In keeping with our Bayesian approach, we minimize expected cost, acting to help the human achieve their likely goal as quickly as possible [47]. Since each inferred policy \(\pi^t\) corresponds to a \(Q^\pi_t\)-value function, the assistant can select actions by minimizing the expected \(Q^\pi\)-value given its uncertainty over \(\pi^t\):

\[
a^*_t = \arg \min_{a^*_t} \mathbb{E}_{\pi^t}[Q^\pi_t(s_t, a^*_t, a^*_t)]
\]

We implement this assistance policy with Algorithm 2, which effectively solves the \(Q_MDP\) approximation of the assistive POMDP [22, 33, 35]. At each step \(t\), we assume the assistant has already updated its belief \(b_{t-1}\) from the previous step \(t-1\), and also observes the human’s action \(s_t\). We then iterate over all samples \((w^t, g^t, \pi^t)\), update the policies \(\pi^{t}\) if necessary, and perform a weighted sum of \(Q_{s^*_t}\)-values for all assistant actions \(a^*_t\) according to their inferred weights \(w^t\). This produces a vector of assistive \(Q\)-values, which we minimize over to compute the best assistive action \(a^*_t\).

### 3 EXPERIMENTS

We evaluate our method against a variety of baselines in two domains: a cooperative gridworld puzzle called multi-agent Doors, Keys & Gems (mDKG) originally developed in [64, 67], and VirtualHome (VH), a virtual household simulator [41, 42]. In both domains, a human principal and a (robot) assistant cooperate on multi-step tasks to accomplish the human’s goals. In mDKG, the goals are four colored gems which are often secured behind doors. Doors can be unlocked by keys of the same color, and the assistant can help by collecting keys or unlocking doors (Figure 1). In VirtualHome, the goals are 6–12 household tasks, including setting up tables and preparing ingredients (Figure 3). Both domains admit encodings in the Planning Domain Definition Language (PDDL).
We designed our problems to cover a range of scenarios inspired by

where the principal and assistant collaborate to set the dinner table.

To evaluate the benefit of pragmatic instruction following over

rithm 2 can also be run in online mode, we fixed the posterior to

only one modality, we included the following baselines:

apparent that the assistant was not doing the same. While Algo-

model \[10\] or the
davinci-002

language-only baseline are equivalent.

As an additional standard for comparison, we conducted a study

is available at

https://github.com/probcomp/CLIPS.jl

Multimodal LLM (GPT-4V). For the mDKG domain, we used

GPT-4 with Vision (GPT-4V) \[1\] as a purely neural baseline, prompting

it with the same set of rules and instructions that we provided
to our human raters, along with the final frame of each animated
visual that we showed to humans (similar to Figure 1(b)). This al-

lowed us to probe the degree to which multimodal LLMs are capable

of intuitive pragmatic reasoning given spatially-grounded actions

and verbal instructions \[11\]. Due to cost and rate limits, we report

single-shot performance with a temperature of zero.

More details about our models and datasets can be found in the

Supplementary Information (https://osf.io/v8ru7/). Source code

is available at https://github.com/probcomp/CLIPS.jl.

3.3 Human judgments

As an additional standard for comparison, we conducted a study

with 100 human participants from the US through Prolific (mean

age = 39.8, 59 men, 38 women, 2 non-binary), presenting each of
them with 15 goal assistance problems from the mDKG domain.

In each problem, participants saw the actions taken by the human

agent which followed a joint policy to the true goal, unless it was

more informative that our human raters were provided.

3.2 Baselines

To evaluate the benefit of pragmatic instruction following over
either a literal interpretation of instructions, or goal inference from
only one modality, we included the following baselines:

\[36\] with some extensions \[66\], which we use as the environment
representation for our planning and inverse planning algorithms.

To systematically test the ability of assistive agents to pragmati-
cally follow a human’s instructions and assist with their goals, we
developed a dataset of goal assistance problems for each domain (30
problems in mDKG, 25 problems in VH). In each problem, the goal
is initially unknown to the assistant. While taking a sequence of
actions \(a_{1:T}^u\), the human communicates one or more instructions
\(u_{1:T}\) to the assistant. The assistant then has to make an inference
about the human’s goal \(g\), and chooses actions to best assist them.
We designed our problems to cover a range of scenarios inspired by
human communication we observed in exploratory studies, varying
the information that can be deciphered from the human’s actions
or instructions (see Figure 4 and Supplementary Information).
In mDKG, we restricted the human to only one instruction \(u_T\), since
more information would render goal inference trivial. In VH, how-
ever, the goal space was larger, allowing us to construct 10 problems
with multiple utterances such as “Can you get the plates?” followed
by “Bring the bowls too!”

3.1 Model configuration

In our experiments, we configured CLIPS to compute \(Q\)-values us-
ing the real-time adaptive \(A^\star\) variant of RTHS \[27\], and a Gamma
prior over the rationality parameter \(\beta\). For the utterance model, we
used either OpenAI’s 6.7B curie model \[10\] or the davinci-002
as our LLMs, due to the more diffuse probabilities they provided
as base models. For goal assistance, we ran Algorithm 2 in offline
mode, fixing the inferred posterior after observing \(a_{1:T}^h\) and \(u_t\), then
selecting actions by minimizing expected \(Q\)-values with respect to
the fixed posterior. This was evaluated against a simulated human
agent which followed a joint policy to the true goal, unless it was
apparent that the assistant was not doing the same. While Algo-

rithm 2 can also be run in online mode, we fixed the posterior to
better match the information that our human raters were provided.

3.2.1 Unimodal Inverse Planning. We implemented action-only and
language-only inverse planning (IP) baselines as ablations of CLIPS,
where the action-only baseline is similar to sequential inverse plan
search \[67\] for inference and Watch-and-Help \[42\] for assistance.
In these baselines, the assistive agent uses the same belief update
shown in Algorithm 1, except that it conditions on only actions \(a_{1:T}^h\)
or only the utterance \(u_t\). In cases with no actions, CLIPS and the
language-only baseline are equivalent.

3.2.2 LLM-Based Literal Listener. The literal listener baseline in-
interprets the instruction \(u_t\) in state \(s_t\) without further information
about the human’s actions or goals. To implement this baseline, we
adapt the utterance model in Figure 2c to sample from the space of
all assistant-directed commands \(c_t\) that are possible in state \(s_t\).
In particular, we enumerate over all assistant actions \(\mathcal{A}^r\), and select
a subset of up to \(K\) salient actions to form a command \(c_t\). This de-
defines a distribution over commands \(P(c_t|s_t)\) which does not depend
the principal’s policy \(\pi\) or goal \(g\). Given the command \(c_t\), we use
an LLM (text-davinci-002 for mDKG, davinci-002 for VH) as an
utterance likelihood \(P(u_t|c_t)\). Assuming a uniform prior over
commands, we can perform enumerative inference to compute the
posterior \(P(c_t|u_t,s_t)\) given an instruction \(u_t\) in state \(s_t\).

Given a command \(c_t\), the assistant still needs to generate an
assistive plan. We did this in two ways: The naive literal assis-
tant interprets a command \(c_t\) by randomly selecting one concrete
grounding (e.g. “Can you pick up the red key?” could mean pick-

ing up the red key closest to the human or some other key), then
planning to achieve that concrete goal. In contrast, the efficient
literal assistant tries to directly satisfy the command \(c_t\) in the most
efficient way4 (e.g. “Could you unlock the blue door?” is satisfied by
unlocking the blue door closest to the assistant). For both variants,
the assistant can either satisfy the most likely command \(c_t^*\) or sam-
ples a command from the full distribution \(P(c_t|s_t)\). We report results
for the latter after averaging over 10 samples, using systematic
sampling to reduce variance.

3.2.3 Multimodal LLM (GPT-4V). For the mDKG domain, we used
GPT-4 with Vision (GPT-4V) \[1\] as a purely neural baseline, prompting
it with the same set of rules and instructions that we provided
to our human raters, along with the final frame of each animated
visual that we showed to humans (similar to Figure 1(b)). This al-

4Note that the naive/efficient distinction was not meaningful in the VirtualHome
problems, where utterance ambiguity was not due to grounding ambiguity.
3.4 Performance metrics

We evaluated CLIPS and the baseline methods in terms of their goal and assistance accuracy, the helpfulness and efficiency of the generated plans, and their similarity to human judgments. To evaluate accuracy, we calculated the probability \( P(g_{true}) \) assigned to the true goal for the non-literal methods, as well as precision and recall for selecting the optimal assistance options that were consistent with the instruction. To evaluate helpfulness and efficiency, we computed the length of the generated plan and the total action cost incurred by the human principal, relativized to CLIPS. Finally, to evaluate human similarity in mDKG, we calculated the correlation between each method’s outputs with the average ratings provided by our participants. We calculated Pearson’s \( r \) as well as its variability from 1000 bootstrapped samples of the human dataset.

3.5 Results

The results of our experiments are presented in Table 1. In terms of accuracy, CLIPS assigned higher probability to the true goal than the baselines (note that 100% is not possible since goals cannot always be distinguished, as in Fig. 4d), while achieving close to perfect precision and recall in selecting the optimal assistance options. Indeed, CLIPS outperformed the average human in this regard. In contrast, unimodal inverse planning was much less accurate at inferring the true goal, and all baselines selected the correct assistance options at significantly lower rates. This affected the efficiency of assistance: CLIPS was 1.26 to 1.68 times faster in achieving the human’s goal than the baselines, and produced plans that were 1.18 to 1.58 times less costly for the human.

For human similarity in the mDKG domain, we found that both the goal inferences and the assistance options produced by CLIPS correlated highly with human ratings, achieving Pearson’s \( r \) of 0.93 (95% CI: 0.91–0.94) and 0.96 (95% CI: 0.95–0.96) respectively. In contrast, unimodal inverse planning produced goal inferences that were highly dissimilar from humans. Correlation with human-selected assistance options was also poor. These results demonstrate the importance of accounting for pragmatic context. Out of all methods, GPT-4V performed the worst, possibly because of the spatial reasoning and multi-step planning required for coherent goal inference in mDKG. Among participants themselves, we found that the median person’s \( r \) of each participant’s ratings with the mean rating was 0.85 (IQR: 0.71–0.94) for goal inferences and 0.87 (IQR: 0.65–0.95) for assistance options, indicating that our dataset of human judgments is a reliable measure of average human performance.

In Figure 4, we compare the results of CLIPS against human judgements and baselines on six illustrative goal assistance problems, providing qualitative observations in the captions. The results show how CLIPS closely mirrors human judgements for both goal inference and assistance, resolving ambiguity in predicates (Fig. 4a) and indexicals (Fig. 4b) while successfully completing partial instructions (Fig. 4c) and interpreting joint instructions (Fig. 4e). In comparison, the unimodal and literal baselines make less confident inferences or fail to assist appropriately, while GPT-4V provides incoherent answers. We also find that both humans and CLIPS are able to assist appropriately when there is significant goal uncertainty (Fig. 4d), and even when the optimal assistive plans for each goal make diverging recommendations (Fig. 4f), illustrating the importance of uncertainty-aware assistance.
(a) Ambiguous Predicates. The human player asks for a red key when three are available. CLIPS resolves this ambiguity, inferring that the human wants Key 2 to reach the yellow gem.

(b) Ambiguous Indexicals. The human player refers to a door using the indexical "this." CLIPS interprets this pragmatically, inferring that Door 3 is intended even though Door 2 is equally close to the human.

(c) Partial Instructions. The human player asks for a single red key, but requires two to reach the red gem. CLIPS infers this and picks up both keys (Key 4 and Key 5), unlike the literal listener baselines.

(d) Uncertain Goals. The human player asks for a yellow key while going to pick up a red key. CLIPS pragmatically infers their goal to be one of the three gems on the left, and hence picks up the left key (Key 3).

(e) Joint Instructions. The human player asks for a blue door to be unlocked while indicating that they will pick up a red key. The CLIPS assistant infers that it should unlock Door 4 so that the human can unlock Door 5.

(f) Safe Assistance. The human player asks the robot to unlock a red door when there are two such doors. The CLIPS assistant reasons that unlocking Door 3 is safer as it leads to both blue and green gems, unlike Door 2.

Figure 4: Goal assistance problems in Doors, Keys & Gems. Each sub-figure contains a visual (left), instruction (bottom left), goal posteriors produced by each method (top right), and the probability of a key or door appearing in the assistive plans generated by each method (bottom right). Our pragmatic goal assistance method, CLIPS, best matches the goal inferences and assistance options produced by human raters (averaged across raters). In contrast, language and action-only inverse planning (Lang. IP & Act. IP) have higher goal uncertainty, the literal baselines fail to resolve instruction ambiguity, and GPT-4V often produces incoherent responses.
4 RELATED WORK

Theory of Mind as Bayesian Inverse Planning. Bayesian Theory of Mind (BToM) posits that humans reason about the actions and mental states of others through (approximately) Bayesian inference [5, 6]. In particular, Bayesian inverse planning can model how humans infer the goals of others by assuming that agents act rationally to achieve their goals [7, 18, 24, 49]. To efficiently solve these inference problems, prior work uses sequential Monte Carlo methods [13, 30] to perform online goal inference over model-based planning agents [3, 67]. CLIPS builds upon this paradigm, extending recent work by Ying et al. [64] on multi-agent inverse planning.

Rational Speech Acts. Rational Speech Act (RSA) theory frames communication as a recursive process, where speakers make pragmatic utterances that optimize relevance to listeners, and listeners interpret speaker utterances in light of the pragmatic goals that underlie those utterances [16, 19, 53]. CLIPS instantiates the RSA framework in the context of cooperative planning.

Multimodal Goal Inference and Reward Learning. Goal inference can be framed as online inverse reinforcement learning (IRL), where the aim is to learn a reward or cost function explaining the agent’s behavior in a single episode [23]. CLIPS can thus be viewed as a language-informed IRL method, though IRL is primarily applied offline given a dataset of expert demonstrations [17, 59, 62]. Particularly relevant is reward-rational implicit choice [25], a framework for multimodal Bayesian reward learning from heterogeneous human feedback. CLIPS can be seen as an application of this framework for specific modalities, but extended to the cooperative setting.

Decentralized Cooperation and Joint Intentionality. When multiple agents cooperate on a task, they usually need to track each other’s mental states to make decisions. Common approaches to this involve recursive reasoning over each other’s goals and plans, which can quickly grow intractable [12, 14]. To address this challenge, computational cognitive scientists have developed models of joint intentionality [58], where cooperating agents conceive of themselves as a group agent with a common goal in mind [52, 55]. This approach allows groups of agents to converge on shared goals [56] and efficient task decompositions [63] in a decentralized manner with limited recursion. CLIPS uses this insight to model joint planning and pragmatic instruction generation in humans.

Value Alignment and Assistance Games. CLIPS is a solution strategy for language-augmented goal assistance games, an extension of the assistance game formalism for human-AI value alignment [21, 45]. In contrast to prior work, our framework leverages joint intentionality when modeling human principals, thereby requiring less recursion than iterated best-response [21], while avoiding the intractability of equilibrium solutions [15, 37].

Instruction Following with Language Models. CLIPS is a form of grounded instruction following from natural language [38, 51, 57], using LLMs to score the likelihood of an utterance given a grounded command. While many recent studies have employed LLMs for translating language into actions [2, 50] or task specifications [29, 34, 65], we opt for an explicitly Bayesian approach, allowing our method to integrate information from both actions and instructions in a principled, reliable, and modular manner.

5 DISCUSSION AND FUTURE WORK

In this paper, we introduced cooperative language-guided inverse plan search (CLIPS) as a Bayesian architecture for pragmatic instruction following and goal assistance. By using a structured cognitive model of how speakers produce both actions and utterances given their goals, CLIPS is able to integrate information from observed actions and ambiguous instruction, inferring a distribution over speaker’s goals and intentions. Using this distribution, a CLIPS assistant is then able to assist the speaker in achieving their goal through expected cost minimization. Through our experiments, we show that the proposed architecture produces human-like outputs on goal inference and assistance tasks. Compared with the baselines, CLIPS also makes more accurate goal inferences, and chooses actions that result in more efficient plans.

While CLIPS demonstrates compelling theoretical properties and strong empirical performance, a number of challenges remain before it can be applied to scenarios with a larger number of goals and assistance options. As noted in Section 2.2, the Bayesian inference strategy we adopt is fully enumerative, but this approach breaks down once the space of possible goals, plans, and commands grows sufficiently large. Recent advances in probabilistic programming could overcome these bottlenecks. For example, more sophisticated sequential Monte Carlo strategies could be used to flexibly propose and update hypotheses, focusing computation on only the most likely sets of latent variables [30, 67]. These strategies can be applied to perform constrained decoding from large language models [31, 61], thereby allowing LLMs to be used as sound proposal distributions over a grammar for commands. This would benefit from LLMs’ performance at few-shot semantic parsing and translation [48], while preserving the Bayesian nature of our framework.

The assistance policy we introduced in Algorithm 2 could also be extended in a number of ways. In many assistive settings, uncertainty is sufficiently high that it makes sense for the assistant to take information gathering actions. Indeed, this is what people typically do when we are confused: we ask questions. To enable this ability, assistants should perform belief-space planning over the set of possible goal beliefs [35, 54], thinking ahead about which actions reveal more information about the principal’s goals. Such planning would alleviate issues that can arise due to the QMDP approximation [22, 33] used by Algorithm 2. In the longer run, assistants could be augmented with natural language outputs, enabling them to ask clarifying questions by planning ahead in a white-box, interpretable manner. If successful, this would constitute a considerable step towards trustworthy assistive AI that effectively collaborates and communicates with humans.

ACKNOWLEDGMENTS

This work was funded in part by the DARPA Machine Common Sense, AFOSR, and ONR Science of AI programs, along with the MIT-IBM Watson AI Lab, and gifts from Reid Hoffman and the Siegel Family Foundation. Tan Zhi-Xuan is supported by an Open Philanthropy AI Fellowship.

REFERENCES

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florence Leoni Aleman, Diego Almeda, Janko Altenschmidt, Sam Altman, Shyamal


