Say-REAPEx: An LLM-Modulo UAV Online Planning Framework for Search and Rescue

Björn Döschl[1] and Jane Jean Kiam[2]

Abstract— **While unmanned aerial vehicles (UAVs) are proven beneficial in search and rescue (SAR) missions, the scalability of their deployment is in practice still challenging as high-level decision-making capabilities for UAVs still lack, and the natural human-in-the-loop command and communications in a SAR mission are rarely tackled. Some promising large-language-model- (LLM-)modulo planning frameworks have been developed for general robotics, combining the strengths of LLMs given their vast training data, but complementing them with domain-specific knowledge and reasoning capabilities for more robust planning. However, adopting the existing frameworks for online planning in a SAR mission requires further adaptations to scale for larger problems, while assuring the real-time planning capability. We introduce Say-REAPEx, an LLM-modulo online planning framework that discards irrelevant or lowly feasible actions based on domain-specific knowledge in order to reduce the size of the planning problem, while leveraging online heuristic search to reduce uncertainty of future rewards. Results of validation tests based on realistic SAR missions show that Say-REAPEx is 70 % more efficient compared to existing frameworks, while maintaining better and comparable success rate.**

I. INTRODUCTION

Unmanned area vehicles (UAVs) are increasingly utilized for search and rescue (SAR) missions due to their promising capabilities, such as rapid deployment, ability to access dangerous areas, and cost-effectiveness compared to conventional methods [21]. Furthermore, they have shown great potential in minimizing risks for both rescuers and victims. Their deployment spans various environments, from man-made to natural disasters, from maritime to mountainous areas. In all cases, response time is a critical factor [12], [33].

In current SAR operations, a single UAV is often operated by an entire team [3]. Reducing crew size remains a challenge, largely due to the limited modifiability of commercially available UAVs. Additionally, integrating command-and-control communications into UAVs is still uncommon, as these systems are often tasked with simple skills like navigating from point A to point B or tracking objects [16], although in practice, UAV teams are issued complex commands in natural language, e.g. *"Search area A for a missing person P, a male with reported cardiac issues and dementia. He was last seen half an hour ago wearing a red backpack and blue jacket in the open area A and may be heading towards the highway. Send images of potential sightings to the K9 unit."* Executing such tasks requires advanced natural language processing (NLP) and symbolic decision-making to adapt to the dynamic, open-world environment [17]. In order to use natural language as instruction, recent research suggests that NLP can be integrated as an extension of a user interface (UI), enabling more intuitive commands that closely mirror human communication patterns, thereby streamlining human-drone collaboration. Although current efforts aim to enhance UIs for better usability and interoperability in SAR missions, they remain insufficient for handling highly dynamic and complex rescue scenarios [9].

Recent studies in robotics have explored the use of large language models (LLMs) to bridge this gap [5], [11], venturing thereby into leveraging LLMs for flexible decision-making. While many argue that LLMs significantly enhance explainable and decision-making capabilities by processing natural language inputs and generating contextually appropriate responses, their capabilities in decision-making remain unreliable. According to studies performed by planning experts, are only usable in non-ergodic environments[1] if integrated with specialized modules [18].

The SayCan framework demonstrated how combining LLMs with domain-specific knowledge can empower robots to perform high-level tasks with a better understanding of contextual details [1]. However, SayCan is hardly scalable for larger problems, resulting in the extended framework SayCanPay that includes also heuristics to guide the search [13], but is limited to only offline planning. Despite these advancements, the application of these frameworks for UAVs in SAR operations is not without limitations, due to their reliance on training data. Furthermore, while SayCanPay can tackle larger problems thanks to the use of a heuristic to guide the search, it is limited to only offline planning.

Building on SayCan and SayCanPay, we propose *Say-REAPEx*[2], an online planning framework for our SAR domain. The main contributions of our work can be summarized as follows:

- Unlike SayCan and SayCanPay, the score-models of *Say-REAPEx* do not over-rely on scoring pre-trained in a static world using expert-generated plan trajectories.
- *Say-REAPEx* can handle significantly more different actions, while keeping the computation time for determining the next best action consistent.

[1]Author is with the Department of Aerospace Engineering, University of the Bundeswehr Munich, 85579 Neubiberg, Germany `bjoern.doeschl@unibw.de`

[2]Author is with the Department of Aerospace Engineering, University of the Bundeswehr Munich, 85579 Neubiberg, Germany `jane.kiam@unibw.de`

[1]An ergodic environment implies that an agent acting in it can reach any state at any given moment [18]

[2]*REAP* is a "**R**eal-world **E**nvironment for **A**I-**P**lanning proposed in [19], while *REAPEx* extends it with **Ex**ecution capabilities. *Say* extends *REAPEx* with the ability to plan for problems instantiated using natural language.

- We use AEMS2 [24] for online POMDP planning as a heuristic to guide the update of the Q-values, which we use in lieu of the "Pay"-model in SayCanPay that is only for offline planning.

In this work, we start by defining the planning problem as a Partially Observable Markov Decision Process (POMDP). Subsequently, we provide details on the "Say-", "Can-" and "Pay-" models of *Say-REAPEx*, followed by a pseudocode that describes how these models are used within a framework for determining in an online manner the next best action to be undertaken by the SAR-UAV. Validation tests show that *Say-REAPEx* outperforms SayCan and SayCanPay in the overall computation time despite the substantial number of actions in our SAR use case, and reduces extensively the number of calls to the LLM-API.

II. Related Work

A. SAR Missions with UAVs

The use of UAVs in SAR operations is becoming more popular. The survey paper by Lyu et al. [21] highlights that the flexibility, mobility, and real-time data processing capabilities offered by UAVs improve the success rates significantly in disaster response by reaching otherwise inaccessible areas. The study also elaborates on the integration of various sensors such as thermal imaging and LiDAR for enhanced target perception and localization. Similarly, Vincent-Lambert et al. offer a scoping review in [33], on the use of UAVs in wilderness search and rescue (WSAR) operations, identifying thereby UAVs as valuable tools in locating victims, mitigating risks, and improving the efficiency of SAR missions, particularly in remote and hazardous environments. However, there has not been research work focusing on high-level human-in-the-loop decision-making in SAR missions using UAVs.

B. Planning with LLMs

In the context of planning with LLMs, a distinction can be made between pure LLM planners and LLM-supported hybrid planners [22]. The majority of LLM-only methods focuses on natural language prompts, which emulate dialogues by providing new instructions in textual form and by generating textual response [23], to be either parsed into parameters and actions or is given directly in code format to be executed [28], [32]. Other works in this regard learn interactions with LLMs to obtain more grounded task models [14], to model the symbolic representation of subgoal plans [35], to compute smooth plan trajectories by combining LLM and hand-sketched trajectories [36] or to pre-train robots on specific tasks [6].

However, Kambhampati et al. argue that LLMs' planning abilities are limited to only generating plausible responses [18]. Furthermore, LLMs appear successful in planning for tasks only when the user already knows the correct answer and even hint it in the prompts. Therefore, in an non-ergodic environment, it is more beneficial to use LLMs as a "helper" for planning, for example by combining them with model-based verifiers. In line with this concept are planning frameworks such as SayCan by Ahn et al. [1] and SayCanPay by Hazra et al. [13]. In SayCan, an LM is used to suggest potential candidate actions; these suggestions are validated against an affordance scoring to determine whether the actions are feasible in the environment the robot finds itself in. This idea of a grounded model-based verifier was adopted and extended in SayCanPay, by adding a heuristic to accelerate the search process in an offline planning approach.

C. Online Planning and Online Algorithms to Solve POMDPs

Online planning involves selecting actions in real-time (i.e. during execution), allowing thereby decision-making based on feedback signals arising from the interaction with the environment. In SAR missions, due to the incomplete information of the goal state[3], dynamics of the environment, and new incoming information of the problem instance (e.g. new last-seen location of the missing person, additional task assignment to the UAV), decisions must be made during execution within a *planning and acting* framework [10].

While [12] treats the SAR mission as a multi-UAV offline multi-objective optimization problem, other similar target detection and recognition problems using UAVs are treated as online optimization problem to account for uncertainties arising from the observation of states [4], [34]. The solution to this class of planning problem relies mostly on approximate online algorithms. Frequently used approaches are for example POMCP [27] that includes Monte Carlo tree search in expanding the current belief state, while [26] improves POMCP by adding safety requirements to be adopted for safety-critical problems. Other works use look-ahead heuristics such as the AEMS2 heuristics to minimize uncertainty in future rewards [24], while [8] enhances AEMS2 with a selection mechanism to improve plan quality.

III. Problem Statement of a SAR Mission

Given the uncertainties arising from the actions and observations, we model our planning problem as a Partially Observable Markov Decision Process (POMDP) with the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{O}, b_0, \mathbf{T}, \mathbf{O}, \mathbf{R}, \gamma)$, where $\mathcal{S}$ is the state space, with a state $s \in \mathcal{S}$ being represented as (l_{UAV}, p_{UAV}, s_{UAV}, l_1, c_1, m_1, sz_1, d_1 ..., l_O, c_O, m_O, sz_O, d_O, ms), where l_{UAV}, p_{UAV} and s_{UAV} denote the location, position and the type of sensor payload of the SAR-UAV, while l_o, c_o, m_o, sz_o and d_o denote the location, color, mobility, size of the object o to be searched and its state of being detected or not. ms denotes the current meteorological season. $\mathcal{A}$ is the action set, comprising all skills that the SAR-UAV is capable to perform. $\mathcal{O}$ is the observation space, b_0 is the initial belief state, $\mathbf{T}(s, a, s') : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition probability function to the next state s' by applying action a at the current state s, $\mathbf{O}(o, a, s') : \mathcal{O} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ maps a probability to an observation o, and an action a applied leading to the next state s', $\mathbf{R}(s, a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ returns the immediate payoff for applying action a at state s, and γ

[3]We only know what to search for, but not where to search.

is the discount factor. At each time step t, after applying the action a_t, the belief state b_{t+1} is updated with the following equation:

$$b_{t+1}(s_{t+1}) = \mu \mathbf{O}(o_t, a_t, s_{t+1}) \cdot \sum_{s_t \in \mathcal{S}} \mathbf{T}(s_t, a_t, s_{t+1}) b_t(s_t), \tag{1}$$

with μ being a normalizer variable.
Planning Problem Instance In practice, the SAR planning problem is provided by the user (i.e. drone operator) in form of a natural language instruction $\mathcal{L}$, that contains information on the goal state s_g, objects to the searched, transition function $\mathbf{T}$ (e.g. the mobility of the missing person/object), reward function $\mathbf{R}$, and the (initial) belief state b_0. Once a goal state is believed to be reached, the human operator will check and confirm or deny. Therefore, the reachability of the goal state is fully observable.
Solution to Online Planning Problem As we intend to solve the planning problem in real-time: at each time instant, given the current belief state b_t, the next best action $a^*_{t+1} \in \mathcal{A}$ will be selected according to a pre-defined scoring f:

$$a^*_{t+1} = \underset{a_{t+1} \in \mathcal{A}}{\operatorname{argmax}} \; f(a_{t+1}, s_g, h_t), \tag{2}$$

where h_t is a sequence of past action-observation pairs, i.e. $h_t = ((a_0, o_0), ..., (a_t, o_t))$.

IV. *Say-REAPEx*: An Online Planning for SAR Missions

Solving a planning problem given a problem instance defined by a natural language instruction $\mathcal{L}$ has been dealt with by several frameworks, with the most promising being the SayCan-Framework by Anh et al. [1] and its extension for offline and larger planning problems, the SayCanPay-Framework by Hazra et al. [13], which considers a heuristic to accelerate the search. We adopt the fundamental ideas of these frameworks as they leverage language models to help planning, specifically with planning problems that are instantiated with natural language instruction(s) in an open-world environment. The solution can come in handy for problems in which the definition of an exact planning domain and problem instance can be very challenging.

However, as mentioned in Section I, for solving our SAR online planning problem, we need to adapt SayCan and SayCanPay for the following reasons:

1) SayCan is not guided by heuristics, and can therefore fail in larger planning problems.
2) SayCanPay includes heuristics to accelerate the search, but is design for use in offline planning.

A. Say-REAPEx*: Say-Score*

Given a language instruction $\mathcal{L}$ and the action-observation trace h_t, a prompt $m_a = (\ell_a, \ell_{s_g}, h_t)$ is constructed, which comprises the natural-language description of an action $a \in \mathcal{A}$, the natural-language description of the goal state s_g extracted from $\mathcal{L}$, and the action-observation trace h_t. The prompt is then evaluated by an LLM to determine the scoring of the action a as a relevant candidate next action a_{t+1} for reaching the goal state in form of a probability:

$$f^{\text{Say}}_{a_{t+1}}(a, s_g, h_t) = f^{\text{LLM}}(m_a) = p(a_{t+1} = a | s_g, h_t). \tag{3}$$

Note the linear latency of promptings with LLMs, they do not scale well with the number of actions in $\mathcal{A}$. In the SayCan and SayCanPay frameworks, the number of actions considered in the benchmarking problems are relatively small, while in our SAR use case, due to the parameters each action can take on, i.e. the high-level action "fly to", once parameterized with the L-locations in the assigned operation area, generate altogether L different actions $a_1, ..., a_L \in \mathcal{A}$.

Therefore, we consider an additional Say-Score for high-level non-parameterized actions. Equation 3 can be rewritten as

$$f^{\text{Say}}_{\hat{a}_{t+1}}(\hat{a}, s_g, h_t) = f^{\text{LLM}}(m_{\hat{a}}) = p(\hat{a}_{t+1} = \hat{a} | s_g, h_t), \tag{4}$$

where $\hat{a} \in \hat{\mathcal{A}}$ denotes the action without parameter, and for each $\hat{a}$, there is a list of associated parameters $p_{\hat{a}} = \{p_1, ..., p_n, ..., p_N\}_{\hat{a}}$, such that $\hat{a}(p_n) \in \mathcal{A}$, for all $p_n \in p_{\hat{a}}$, and $\hat{a}(p_n) \neq \hat{a}(p_m)$, if $n \neq m$.

B. Say-REAPEx*: Can-Score*

A Can-Score typically evaluates the feasibility of an action. In *Say-REAPEx*, we define two Can-scorings: $f^{\text{Can-pre}_a}$ and $f^{\text{Can-comp}_a}$. $f^{\text{Can-pre}}_a$ only considers the preconditions of the action when evaluating its feasibility, and is defined by

$$\begin{aligned} f^{\text{Can-pre}}_{a_{t+1}}(a, h_t) =& p(pre(a)|h_t) && (5)\\ =& \sum_{s_t \in \mathcal{S}} p(pre(a)|s_t, h_t) \cdot \underbrace{p(s_t|h_t)}_{b(s_t)} && (6)\\ =& \sum_{s_t \in \mathcal{S}} \delta(s_t \models pre(a)) \cdot b(s_t), && (7) \end{aligned}$$

where $p(pre(a)|h_t)$ denotes the probability of the preconditions of a being satisfied given the action-observation trace h_t. Using the Theorem of Total Probability, we obtain Equation 6. $\delta(s_t \models pre(a)) = 1$, if state s satisfies the preconditions of a (i.e. $pre(a)$), otherwise, $\delta(s_t \models pre(a)) = 0$.

$f^{\text{Can-comp}}_a$ evaluates the feasibility of an action based on the success rate of completing the action given a state:

$$\begin{aligned} f^{\text{Can-comp}}_{a_{t+1}}(a, h_t) =& p(comp(a)|h_t) && (8)\\ =& \sum_{s_t \in \mathcal{S}} p(comp(a)|s_t, h_t) \cdot \underbrace{p(s_t|h_t)}_{b(s_t)} && (9)\\ =& \sum_{s_t \in \mathcal{S}} p(comp(a)|s_t) \cdot b(s_t), && (10) \end{aligned}$$

where $p(comp(a)|h_t)$ denotes the probability of completing action a given the action-observation trace h_t. We assume that $p(comp(a)|s, h_t) = p(comp(a)|s)$, as the success probability of completing action a depends only on the true state s instead of on the action-observation trace h_t.

The probability distribution of $p(comp(a)|s)$ can be learned in a domain-specific manner [15]. This is particularly relevant for modeling the ability of a sensor payload to detect and recognize an object when scanning an area in a SAR mission, as the probability of completing the action with success depends on the color and size of the object, the payload used by the drone, its altitude, and the meteorological season coupled with the type of location[4] being scanned.

$$p(comp(a)|s) = \frac{p(s|comp(a)) \cdot p(comp(a))}{p(o_t)} \tag{11}$$

We learn the distribution models using the Bayesian theorem above in a simulated realistic environment (shown in Figure 2) built with Unreal Engine and AirSim [19].

Note that, similar to the Say-Score, the Can-Scores described by Equation 7 and 10 can be defined for non-parameterised actions as well.

C. Say-REAPEx*: Pay-Score*

While *Say-REAPEx* offers flexibility to plan with incomplete problem model, LLMs must be complemented with domain-specific knowledge using Can-Scores to eliminate infeasible actions or actions with low probability to be completed. While this concept was also considered in SayCanPay [13] (with different details in the implementation and scoring), an additional Pay-Model was also introduced to look-ahead, in order to find shorter plans leading to the goal state, hence minimizing execution time, which is critical for a SAR mission. However, SayCanPay introduces learned heuristics (from static planning environment) that are only applicable for offline planning. Given the dynamics of a SAR mission, the heuristic used in SayCanPay cannot be exploited here. Instead, we resort to the Anytime Error Minimization Search (AEMS2) heuristic introduced by Ross and Chaib-draa [24] for approximate online algorithm to search for solutions of large POMDPs. The general idea is, given a current belief state, a fringe belief state of the search tree will be chosen based on the AEMS2 heuristic that leads towards exploring nodes with the most substantial uncertainty regarding future rewards, while being a node on a path that carries the most substantial cumulative reward. The chosen fringe node will be expanded with applicable actions to calculate their Q-values, followed by a backtracking to update Q values of all ancestor nodes. These steps, (i.e. select fringe node, expand tree, update ancestor nodes), are repeated until either the allocated planning time is over, or for a given number of iterations.

The Pay-Score used in *Say-REAPEx* is defined by

$$f^{\text{Pay}}_{a_{t+1}}(a, b_t) = \underline{Q}(b_t, a), \tag{12}$$

where $\underline{Q}(b, a)$ is the lower bound of

$$Q(b,a) = R(b,a) + \gamma \sum_{s \in \mathcal{S}} b(s) \sum_{s' \in \mathcal{S}} T(s,a,s') \cdot \sum_{o \in \mathcal{O}} O(o,a,s') V(b(s'|a,o)), \tag{13}$$

where V is the maximum Q-value with respect to a.

[4]As an intuitive example, an electro-optical camera is less effective in recognizing a missing person in a green jacket on a open field than in an urban area.

D. Say-REAPEx

Algorithm 1 describes the pseudo code of *Say-REAPEx.*

Algorithm 1 Say-REAPEx

```
Given: A high level natural language instruction $\mathcal{L}$, an initial
    belief state $b_0$, an observation function $\mathcal{O}$, and the set of
    action $\mathcal{A}$ and their language description $l_a$.
 1: Define the transition function $\mathbf{T}$, the reward function $\mathbf{R}$
    and the goal state $s_g$ from $\mathcal{L}$
 2: $t \leftarrow 0$, $\Sigma \leftarrow \emptyset$
 3: while $l_{a_t} \neq$ "land" || $\tau(t) < \tau_{\text{max}}$ do
 4:     for $\hat{a} \in \hat{\mathcal{A}}$ and $l_{\hat{a}} \in l_{\hat{\mathcal{A}}}$ do                ▷ Select Action
 5:         Compute $f^{\text{Say}}_{\hat{a}_{t+1}}(\hat{a}, s_g, h_t)$ based on Eq. 4
 6:         Compute $f^{\text{Can-pre}}_{\hat{a}_{t+1}}(\hat{a}, h_t)$ with Eq. 7
 7:         $f^{\text{Say-Can}}(\hat{a}) = f^{\text{Say}}_{\hat{a}_{t+1}}(\hat{a}, s_g, h_t) \cdot f^{\text{Can-pre}}_{\hat{a}_{t+1}}(\hat{a}, h_t)$
 8:         $\Sigma \leftarrow \Sigma \cup \{f^{\text{Say-Can}}(\hat{a})\}$
 9:     end for
10:     $\hat{a} \leftarrow \operatorname{argmax}_{\hat{a} \in \hat{\mathcal{A}}} \Sigma$
11:     $\Sigma \leftarrow \emptyset$
12:     $\chi_{\hat{a}} \leftarrow \hat{a}(p_{\hat{a}})$            ▷ Find all parameterized actions
13:     for $a \in \chi_{\hat{a}}$ do                ▷ Select likely Parameters
14:         Compute $f^{\text{Can-comp}}_{a_{t+1}}(a, h_t)$ based on Eq. 10
15:         $\Sigma \leftarrow \Sigma \cup \{f^{\text{Can-comp}}_{a_{t+1}}(a, h_t)\}$
16:     end for
17:     $\chi_{\hat{a},k} \leftarrow \texttt{get\_top\_k}(\chi_{\hat{a}}, \Sigma, k)$    ▷ Get top-k actions
18:     $\Sigma \leftarrow \emptyset$
19:     for $a \in \chi_{\hat{a},k}$ and $l_a \in l_{\chi_{\text{mod}}}$ do  ▷ Select Parameters
20:         Compute $f^{\text{Say}}_{a_{t+1}}(a, s_g, h_t)$ based on Eq. 3
21:         Compute $f^{\text{Pay}}_{a_{t+1}}(a, b_t)$ based on Eq. 12
22:         $f^{\text{Combined}}_{a} \leftarrow f^{\text{Say}}_{a_{t+1}}(a, s_g, h_t) \cdot f^{\text{Pay}}_{a_{t+1}}(a, b_t)$
23:         $\Sigma \leftarrow \Sigma \cup \{f^{\text{Combined}}_{a}\}$
24:     end for
25:     $a* \leftarrow \operatorname{argmax}_{a \in \mathcal{A}} \Sigma$
26:     EXECUTE($a*$)
27:     update belief state $b_{t+1}$ with Eq. 1
28:     $h_{t+1} \leftarrow (h_t, (a, o))$
29:     $t \leftarrow t + 1$
30: end while
```

While the UAV is not landing, or if the maximum flight time τ_{max} is not exceeded, a next action will be selected. The selection of the next action is carried out in three main *for*-loops. Figure 1 shows the schematic system diagram of Say-REAPEx, depicting Algorithm 1. In the first *for*-loop (from Line 4 to 9), the aim is to select a non-parameterized action $\hat{a}$ using a Say-Score evaluated by LLM and a Can-Score that considers only the preconditions of a non-parameterized action. The first *for*-loop is to limit the number of prompts to be processed by the LLM and is indicated as "Select Action" in the figure. After $\hat{a}$ is selected in Line 10, the list of parameterized actions $\chi_{\hat{a}}$ associated to $\hat{a}$ will be determined, alongside with their Can-scores that are based on the success rate of the action and computed as in Equation 10, (see Line 13 to Line 16). Subsequently, another selection is performed to select the top-k parameterized actions, while

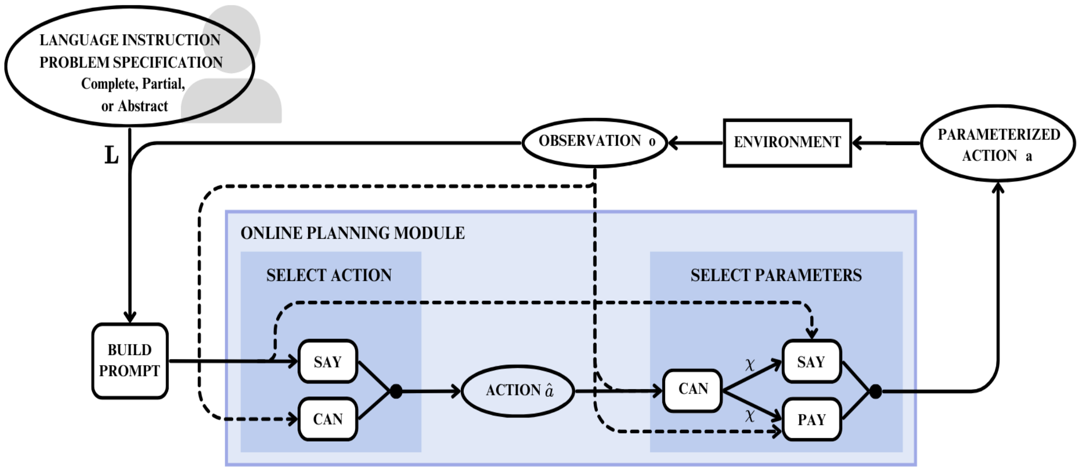

Fig. 1. System diagram of *Say-REAPEx*. Two key steps are distinguished: first, a feasible action is selected, followed by determination of its respective parameters. This action is then sent to the execution platform, which processes it and provides a new observation based on the updated environment.

others are discarded. This first discarding of lowly-scored parameterized actions is done in the CAN-Module of "Select Parameters" in the system diagram in Figure 1. This is to filter out action parameters that have low (Can-)scores, so that the prompts to be processed in the third *for*-loop can be further reduced. The third *for*-loop spanning from Line 19 to Line 24, as illustrated in the figure by the Say- and Pay-Module, follows the concept of the SayCanPay framework, using scorings that are described in Equations 3 and 12 respectively. The new action a_{t+1} is selected based on the combined scoring f_a^{Combined}. After the execution of the action, the belief state, as well as the action-observation trace are updated.

V. Experiments & Results

A. Experimental Setup

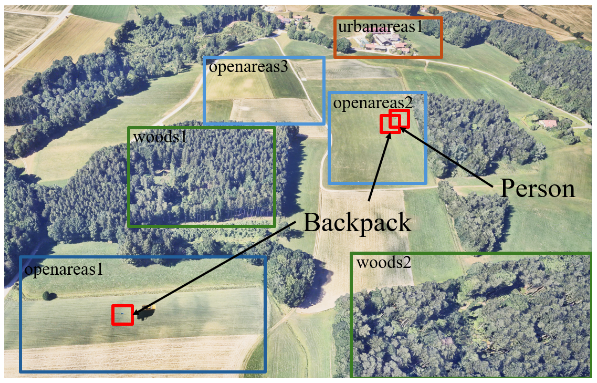

Fig. 2. Simulation environment in Unreal Engine with the operation areas and the location of the person indicated.

To assess the efficacy of *Say-REAPEx*, we use the REAP-Framework [19] as the validation environment, within which our planner was developed. This framework uses ROS2 [30] as a middleware and implements a range of predefined actions for UAVs. These actions are selected and sent to the UAV by a planning and execution mechanism, while an offboard controller node receives the commands and directs the flight controller. The simulation is set up in Unreal Engine 5.2 using AirSim [25] as quad-copter simulator and physics engine. The combination of the Cesium plugin and the Google Photorealistic tileset allows for the deployment of drones in a realistic scenario. A screenshot from the simulation, with marked operation areas and to be searched objects is shown in Figure 2. For object detection, a YOLOv8 [31] model that has been pre-trained on the COCO dataset [20], and is fine-tuned on a smaller dataset, with the objective of reducing the number of classes to only those relevant in a SAR mission, is deployed. As LLM, we mainly used the gpt-4o-mini, but other LLMs as Llama3 [7], Claude3 [2] and Gemini [29] were also tested. For the validation, language instructions similar to "*Search areas open fields. If you find a person, confirm it, return home, and land.*" is utilized. This text is initially provided by the human-operator and subsequently parsed by the planner to be inserted as a goal in the prompt of the LLM model. In this scenario, 9 non-parameterized actions $\hat{\mathcal{A}} = \{$"Return to Home and Land", "Take off", "Search an area", "Hover", "Confirm an object", "Ascend", "Descend", "Perform mapping", "Take one single image"$\}$ are being considered. The resulting number of parameterized actions $\mathcal{A}$ differs depending on the number of parameters, e.g. $\hat{a} =$"Ascend" can take different altitudes as parameters to describe how much the UAV should ascend $\{\hat{a}(30), \hat{a}(60), \hat{a}(90)\}$.

B. Results

We evaluate the performance of *Say-REAPEx* to that of SayCan (i.e. by omitting the first and second *for*-loops and also by omitting f^{Pay} in the combined scoring in Line 22 of Algorithm 1) and to that of an online SayCanPay implementation (i.e. by omitting only the first and second *for*-loops in Algorithm 1) by comparing the computation times for a given number of (parameterized) actions (5, 25, 50, 75, 100, 125) (see Figure 3).

As the number of available actions increases, the compu-

TABLE I

COMPARING THE MEAN COMPUTATION TIME OF THE NEXT ACTION, THE MEAN NUMBER OF PROCESSED ACTIONS IN EACH SELECTION STEP, THE MEAN PRECISION OF ALL SELECTED ACTIONS AND THE SUCCESS RATE WITH DIFFERENT MODELS. HERE THE NUMBER OF ACTIONS IS $|\mathcal{A}| = 48$.

	SayCan	SayCanPay	SayCan-SayCanPay	Say-REAPEx
∅ Action Computation Time	35.675 s	37.586 s	2.832 s	**2.104 s**
∅ Number of Processed Actions	48	48	21.4	**13.2**
∅ Precision in %	45.2	49.5	49.3	52.5
Success in %	47.5	52.5	50.0	55.0

tation time for the action selection in the SayCan and the SayCanPay model exponentially rises. This is due to the

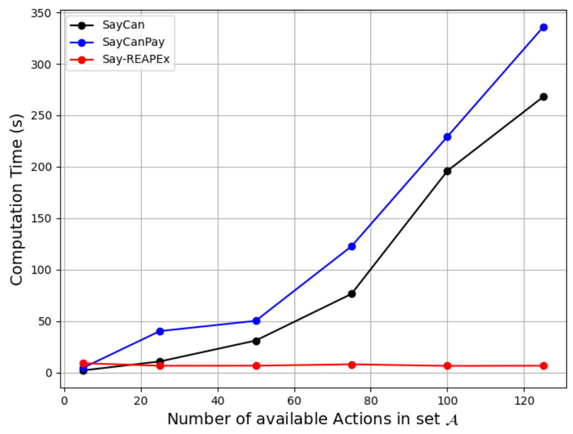

Fig. 3. Comparing the Action Selection Computation Time of different Number of Actions for the SayCan, SayCanPay and Say-REAPEx.

synchronous LLM evaluation of all possible actions in set $\mathcal{A}$. Both approaches take more than 250 seconds if 125 actions are to be considered, which is too long as an online UAV planning approach in a SAR mission. In comparison, the *Say-REAPEx model* reduces the action set to only the promising and highly feasible parameterized actions. Consequently, the computation time remains relatively consistent despite the increasing number of parameterized actions to be considered. Moreover, since APIs such as gpt-4o-mini have restrictions on their call rate limit, it is beneficial to discard non-relevant of infeasible actions.

In the second validation, we constructed 20 test runs, with each test run limited to at most 16 actions to emulate the battery capacity of the SAR-UAV since we do not include a high-fidelity energy consumption model in the simulation environment. Table I displays the mean computation time, the mean number of processed actions, the mean precision and the success rate for SayCan, SayCanPay, SayCan-SayCanPay (i.e. by omitting the second *for*-loops in Algorithm 1) and Say-REAPEx. The mean action computation time reveals that the high number of evaluated actions in SayCan and SayCanPay causes the much greater computation time. Considering the 48 given actions, our approach only computes up to 13.2 actions on average in each action selection step, which is $\approx 72\%$ fewer than the SayCan approach in our scenario.

Although the mean action computation time and the mean number of processed actions are greatly reduced with Say-REAPEx, no vast improvement is seen with the precision and success rate. Object recognition is apparently the main hindrance to improving precision and success rate. In the test runs, either objects are not recognized or wrong objects were recognized, causing unnecessary follow-up actions to "Hover", "Confirm" and "Take an image".

VI. CONCLUSION

In this work, we introduced *Say-REAPEx*, an automated high-level UAV online planning framework for SAR-operations in the open world, in which the complete modelling of a planning problem (domain and instance) can be impossible. The LLM-module planning framework is also able to understand instructions formulated in natural language by the human UAV-operator in a SAR mission.

In Say-*Say-REAPEx*, we adapted the existing SayCan and SayCanPay approaches by:

1) developing a two-step action selection process consisting of an action selection step (of non-parameterised actions) and an action parameter selection step;
2) adopting new scorings so that sensor capabilities can be considered in the Can-Model, and that a heuristic for online planning guides the exploration of actions more efficiently in the Pay-Model.

These improvements allow us to employ the method for SAR operations, in which reduced planning time and the efficiency of a plan are critical, given the limited battery capacity of SAR-UAVs. Moving forward, we plan to

1) extend Say-REAPEx for use by multiple UAVs, as multi-UAV systems benefit more from automated decision-making functions without increasing the size of the drone team in a SAR mission while reducing response time;
2) incorporate more flexibility into the existing framework to accept additional incoming instructions or information which alter the goal states, the transition, and the reward functions.

VII. ACKNOWLEDGMENT

This work was supported the Federal Ministry of Education and Research (Bundesministerium für Bildung und Forschung, BMBF). The problem modeling benefited from valuable input provided by domain experts of the Penzberg Mountain Rescuers (Bergwacht Penzberg).

References

[1] Michael Ahn et al. "Do As I Can and Not As I Say: Grounding Language in Robotic Affordances". In: *arXiv preprint arXiv:2204.01691*. 2022.

[2] Anthropic. *Claude 3: A Conversational AI by Anthropic*. https://www.anthropic.com. Accessed: 2024-09-13. 2023.

[3] Bundesamt für Bevölkerungsschutz und Katastrophenhilfe. *EGRED: Empfehlungen für die Planung von Schutzräumen und Evakuierung im Zivilschutz*. 2022.

[4] Caroline Carvalho Chanel, Florent Teichteil-Königsbuch, and Charles Lesire. "Multi-Target Detection and Recognition by UAVs Using Online POMDPs". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 27.1 (2013), pp. 1381–1387.

[5] Daniel S. Drew. "Multi-Agent Systems for Search and Rescue Applications". In: *Current Robotics Reports* 2.2 (June 2021), pp. 189–200.

[6] Danny Driess et al. *PaLM-E: An Embodied Multimodal Language Model*. 2023. arXiv: 2303.03378.

[7] Abhimanyu Dubey et al. *The Llama 3 Herd of Models*. 2024. eprint: 2407.21783.

[8] Adam Eck and L.-K Son. "Online heuristic planning for highly uncertain domains". In: *13th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2014* 1 (Jan. 2014), pp. 741–748.

[9] Max Friedrich et al. "RESPONDRONE - A Multi-UAS Platform to Support Situation Assessment and Decision Making for First Responders". In: *Advances in Human Factors in Robots, Unmanned Systems and Cybersecurity*. Ed. by Matteo Zallio, Carlos Raymundo Ibañez, and Jesus Hechavarria Hernandez. Cham: Springer International Publishing, 2021, pp. 110–117.

[10] Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning and Acting*. Cambridge University Press, 2016.

[11] Shashank Govindaraj et al. "Command and Control Systems for Search and Rescue Robots". In: *Search and Rescue Robotics*. Rijeka: IntechOpen, 2017. Chap. 8.

[12] Samira Hayat et al. "Multi-objective UAV path planning for search and rescue". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 5569–5574.

[13] Rishi Hazra, Pedro Zuidberg Dos Martires, and Luc De Raedt. "SayCanPay: Heuristic Planning with Large Language Models using Learnable Domain Knowledge". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 18. 2024, pp. 20123–20133.

[14] Wenlong Huang et al. "Grounded Decoding: Guiding Text Generation with Grounded Models for Embodied Agents". In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023.

[15] Wenlong Huang et al. "Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents". In: *CoRR* abs/2201.07207 (2022).

[16] Shumaila Javaid et al. "Communication and Control in Collaborative UAVs: Recent Advances and Future Trends". In: *IEEE Transactions on Intelligent Transportation Systems* 24.6 (2023), pp. 5719–5739.

[17] Shumaila Javaid et al. "Large Language Models for UAVs: Current State and Pathways to the Future". In: *IEEE Open Journal of Vehicular Technology* 5 (2024), pp. 1166–1192.

[18] Subbarao Kambhampati et al. *LLMs Can't Plan, But Can Help Planning in LLM-Modulo Frameworks*. 2024. arXiv: 2402.01817.

[19] Oliver Kraus, Lucas Mair, and Jane Jean Kiam. "REAP: A Flexible Real-World Simulation Framework for AI-Planning of UAVs". In: *ICAPS 2023 System Demonstrations*. July 2023.

[20] Tsung-Yi Lin et al. "Microsoft COCO: Common Objects in Context". In: *Computer Vision – ECCV 2014*. Ed. by David Fleet et al. Springer International Publishing, 2014, pp. 740–755.

[21] Mingyang Lyu et al. "Unmanned Aerial Vehicles for Search and Rescue: A Survey". In: *Remote Sensing* 15.13 (2023).

[22] Yueen Ma et al. *A Survey on Vision-Language-Action Models for Embodied AI*. 2024. arXiv: 2405.14093.

[23] Yao Mu et al. *EmbodiedGPT: Vision-Language Pre-Training via Embodied Chain of Thought*. 2023. arXiv: 2305.15021.

[24] Stéphane Ross and Brahim Chaib-draa. "AEMS: An Anytime Online Search Algorithm for Approximate Policy Refinement in Large POMDPs". In: *International Joint Conference on Artificial Intelligence*. 2007.

[25] Shital Shah et al. "AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles". In: *Field and Service Robotics*. 2017.

[26] Shili Sheng, David Parker, and Lu Feng. "Safe POMDP Online Planning via Shielding". In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. 2024, pp. 126–132.

[27] David Silver and Joel Veness. "Monte-Carlo Planning in Large POMDPs". In: *Advances in Neural Information Processing Systems*. Ed. by J. Lafferty et al. Vol. 23. Curran Associates, Inc., 2010.

[28] Ishika Singh et al. "ProgPrompt: Generating Situated Robot Task Plans using Large Language Models". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. 2023, pp. 11523–11530.

[29] Gemini Team et al. *Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context*. 2024. arXiv: 2403.05530.

[30] Dirk Thomas, William Woodall, and Esteve Fernandez. "Next-generation ROS: Building on DDS". In: *ROSCon Chicago 2014*. Mountain View, CA: Open Robotics, Sept. 2014.

[31] Rejin Varghese and Sambath M. “YOLOv8: A Novel Object Detection Algorithm with Enhanced Performance and Robustness”. In: *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*. 2024, pp. 1–6.

[32] Sai H. Vemprala et al. “ChatGPT for Robotics: Design Principles and Model Abilities”. In: *IEEE Access* 12 (2024), pp. 55682–55696.

[33] Craig Vincent-Lambert, Anje Pretorius, and Bernard Van Tonder. “Use of Unmanned Aerial Vehicles in Wilderness Search and Rescue Operations: A Scoping Review”. In: *Wilderness and Environmental Medicine* 34.4 (2023), pp. 580–588.

[34] Bo Wang et al. “UAV autonomous path optimization simulation based on radar tracking prediction”. In: *EURASIP Journal on Wireless Communications and Networking* 2018.1 (Oct. 2018), p. 239.

[35] Kaizhi Zheng et al. *JARVIS: A Neuro-Symbolic Commonsense Reasoning Framework for Conversational Embodied Agents*. 2022. arXiv: `2208.13266`.

[36] Weiqin Zu et al. *Language and Sketching: An LLM-driven Interactive Multimodal Multitask Robot Navigation Framework*. 2024. arXiv: `2311.08244`.