Towards Intelligent Companion Systems in General Aviation using Hierarchical Plan and Goal Recognition

Prakash Jamakatel
University of Bundeswehr Munich
Neubiberg, Bavaria, Germany
prakash.jamakatel@unibw.de

Pascal Bercher
The Australian National University
Canberra, ACT, Australia
pascal.bercher@anu.edu.au

Axel Schulte
University of Bundeswehr Munich
Neubiberg, Bavaria, Germany
axel.schulte@unibw.de

Jane Jean Kiam
University of Bundeswehr Munich
Neubiberg, Bavaria, Germany
jane.kiam@unibw.de

ABSTRACT

Modern ultralight aircraft in general aviation are equipped with an onboard Pilot Assistance System (PAS) as a companion system, meant to guide the pilot in decision-making, e.g. with plan suggestions, especially in critical situations. For more meaningful guidance, the PAS must possess a continuous understanding of the context, i.e. the pilot's intention, so that decision-making support is relevant. However, in realistic settings, the pilot's intention is not communicated manually, but can only be proactively monitored by the PAS. This paper explores the possibility of embedding domain expertise using Hierarchical Task Network (HTN) planning to track the pilot's intention, by recognising the pilot's current goal task judging from the pilot's actions. Furthermore, by leveraging probability theory for state estimation, we derive belief values to be associated with the recognised goal task, inferred from already executed actions which are in turn inferred from in-cockpit observable measurement data. Statistical evaluation using data collected from human-in-the-loop tests shows that our method for tracking the pilot's intention is reliable enough to provide the PAS with a contextual understanding in real time.

CCS CONCEPTS

• **Human-centered computing → User models**.

KEYWORDS

Plan and goal recognition, Pilot assistance system, Hierarchical Task Network planning, Action recognition, State estimation

ACM Reference Format:
Prakash Jamakatel, Pascal Bercher, Axel Schulte, and Jane Jean Kiam. 2023. Towards Intelligent Companion Systems in General Aviation using Hierarchical Plan and Goal Recognition. In *International Conference on Human-Agent Interaction (HAI '23), December 4–7, 2023, Gothenburg, Sweden.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3623809.3623877

1 INTRODUCTION

Piloting an airplane involves perceiving and processing a wide array of information and subsequently, acting (as optimally as possible) by performing sub-tasks to achieve the intended goal tasks, e.g. land airplane, fly to a location, cruise at a fixed altitude, etc. Due to the abundance of information to be processed, performing tasks can lead to mental overload [38]. This in turn increases the number of human-induced errors during the flight, or worse, during emergency situations. Pilot Assistance Systems (PAS) have been developed as onboard companion systems to help mitigate these errors and make flying safer [33]. Some examples are CHAP-E [3] and LNAS [1]. CHAP-E guides the pilot by providing action checklists in different stages of the flight according to the state of the environment and the airplane. LNAS is a landing assistance system that suggests optimized flight plans based on fuel consumption and noise reduction. These systems are developed for larger aircraft, typically for airliners. Although adaptable for other types of aircraft, e.g. Ultralight (UL) aircraft, little work in this regard has been done.

UL aircraft belongs to the class of smaller and lighter aircraft geared mainly towards amateur pilots. Therefore, legal requirements on flight training for UL-pilots are in general less demanding. While this makes flying a UL more accessible for the general public, studies have shown that most accidents in general aviation result from human (pilot) error [10], which are often fatal. Furthermore, studies have sustained the proof that the integration of PAS as onboard companion systems can reduce the number of fatalities [6]. In line with this report, companion systems should be built into future UL cockpits to mitigate pilot errors by providing contextual help so as to assist the pilot by complementing their skills according to flight situations, and by intervening whenever necessary, thus timely reducing the risk of fatal accidents. For this, it is essential for the PAS to understand the context, or more concretely, the pilot's intention, in order to assist the pilot in a meaningful manner.

In this work, we conceptualise and develop an automated *intention tracking module* of the PAS by exploring a Plan and Goal Recognition (PGR) method based on Hierarchical Task Network (HTN) planning. The motivation to explore HTN-planning for PGR is twofold. Hierarchical planning is often used for providing flight guidance to pilots, since tasks in pilots' handbooks are often formulated in a hierarchical manner. Furthermore, more HTN planning techniques are becoming available as open-source tools, and can therefore be exploited for HTN-based PGR as well. Using HTN-based PGR to devise the current goal task requires a list of already executed actions (also referred to as "action prefix") [17][1], which can only be inferred from observable measurement data. Therefore, we develop a method to determine the belief value of the recognised goal task, which also represents the extent to which the PAS *thinks* the goal task in question is intended by the pilot.

We first provide an overview of related works, followed by the problem statement. Subsequently, we describe how HTN-based PGR is used, followed by a method to determine the belief value of the recognised goal task, given the observable measurement data. For validation purposes, we use the Ultralight Cockpit planning domain [25] to perform our human-in-the-loop tests and thus demonstrate that HTN backed-systems are suitable for this use case.

2 RELATED WORK

Besides PAS such as CHAP-E and LNAS described in Section 1, various *companion systems*, i.e. technical systems that appear as companions to their users, were developed [7]. They are meant for executing daily chores, such as Do It Yourself, in which the user is guided through home improvement tasks by detailed instructions generated according to preferences [4], as well as interactive robotic companions for shopping [5] or for elderly care [37]. Besides, a surge of companion technologies can be seen in Real-Time Safety-Critical (RTSC) systems, such as Advanced Driver Assistant Systems (ADAS), which are indispensable companion systems in many modern automotive cockpits [41].

The companion technology Do It Yourself generates contextual guidance by considering the user's preferences provided manually by the user [4]. In RTSC systems, it is unlikely for the PAS (or ADAS) to gain awareness of the context using methods based on manual input. PAS (or ADAS) usually rely on different paradigms, for example by recognising the pilot's (or driver's) current activity or intention. Where data is abundant, machine learning approaches using video or other sensor data are popular for activity recognition [12, 29, 40]. Although these approaches have gained huge popularity in activity recognition in the automotive domain, the usability in PAS is relatively low because data is scarce in this setting. On the other hand, many methods inspired by Belief-desire-intention models were developed, such as the knowledge-based intent recognition system for PAS [35], as well as other approaches based on probability theory [36] and evidence theory [19] that were also proven usable to detect pilot's intention. Grammar-based representation for plan recognition was described by Geib and Steedman in [13]; the application of grammar-based plan recognition was proven successful in a pedagogical context in [2] and in an exploratory domain in [27]. [22] provides an overview of various approaches for achieving activity recognition. However, these methods do not draw benefits from the emerging automated planning technologies and are often decoupled from decision-making support mechanisms.

In automated planning, "planning" is considered the process of "choosing and organizing the actions that can achieve the given objective" [15]. Determining the sequence of actions (or the objective) the agent is pursuing is referred to as *plan* (or *goal*) recognition. In classical planning, PGR as planning was introduced by Ramirez and Geffner [30], where modified planning systems were used, and was later extended to work with off-the-shelf planners [31].

Hierarchical planning is another planning paradigm, that is more expressive than classical planning and can include more extensive domain expert knowledge to decompose a high-level task (network) into low-level executable actions. Therefore, the use of hierarchical planning has been explored in a wide range of applications, from web services [34], robotics [20, 39], to driver activity recognition [11], and even mission or flight planning in aerospace [3, 24].

Analogous work on PGR was also developed using the HTN planning paradigm. Höller et al. developed a PGR method that requires only the transforming of plan recognition problems into HTN planning problems [17], which, different from the goal recognition technique discussed in [8] that uses the transformation technique, no special solver is required. It benefits entirely from the advancement made in HTN planning. Since HTN is more expressive than their counterparts in classical planning and allows to model the tasks in a more human-like nature [14, 16], PGR as HTN planning problem is therefore highly applicable in PAS, where expert knowledge is often also hierarchical in nature.

3 PROBLEM STATEMENT

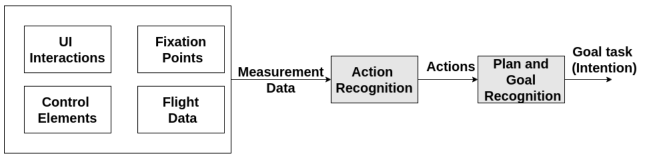

Figure 1: Intention tracking in PAS.

Figure 1 depicts the architecture of a PAS to track the pilot's intention and consists of measurement data, an action recognition module and a PGR module. A main concern in a companion system used for RTSC systems is to provide *appropriate decision-making support at the right moment*. However, the *appropriate support* depends largely on the pilot's needs, which vary over time, depending on the pilot's intention, which, in a realistic cockpit configuration, *is not communicated explicitly*. It is therefore essential to infer the pilot's intention from observable measurement data, such as flight data, fixation points[2], as well as interactions with the User Interface (UI) of the cockpit and with the control elements.

In this paper, we assume that the pilot's intention is represented by a goal task (e.g. `land`*) inferred from actions, which are in turn inferred from observed measurement data.* Each goal task $G_l \in \mathcal{G} = \{G_1, \ldots, G_L\}$ can be mapped to an initial task network tn_I (composed of only one abstract root task) described by Höller et al. [17]. Formally, we consider the set of observation data $\mathcal{X} = \mathcal{X}^d \cup \mathcal{X}^c$, where $\mathcal{X}^d = \{X_1^d, \ldots, X_{I_d}^d\}$ denotes the set of different types of discrete raw observation data (e.g. `button_state`), and $X_i^d = x_i^d \in \{0, 1\}$, while $\mathcal{X}^c = \{X_1^c, \ldots, X_{I_c}^c\}$ denotes the set of different types of continuous raw observation data (e.g. 'flight altitude'), and $X_i^c = x_i^c \in \mathbb{R}$. The solution to our intention recognition

[1]For example, to perform the goal task "land airplane", the pilot needs to execute actions to "align airplane to the landing stripe", "decrease airspeed", etc.

[2]Wearable eye trackers are becoming more miniaturised to be integrated into sunglasses or helmets worn by pilots.

problem is a set of belief values associated to all goal tasks , i.e. $bel(\mathcal{G}) = \{bel(G_1 = 1), \ldots, bel(G_L = 1)\}$ being true. The goal task with the most substantial belief value will be considered the pilot's intention by the PAS; subsequent decision-making support will be based on the recognised intention.

4 USING HTN PLANNING FOR PLAN RECOGNITION

A widely used formalism in hierarchical planning is HTN planning [16]. An *HTN planning problem* can be defined as a tuple $P = (F, C, A, M, s_0, tn_I, g, prec, add, del)$. F is a set of propositional state features. A state s is defined by the subset of state features that hold true. The initial state of the problem is $s_0 \in 2^F$, while $g \subseteq F$ is the state-based goal description. A is the set of symbols for actions (or primitive tasks)[3] directly executable by the actor in the domain, while C is the set of symbols for compound tasks (to be decomposed into actions). The decomposition is defined using the methods defined in M. The preconditions, add-, and delete-effects of action are defined in $prec, add, del$ respectively. tn_I is the initial task network. A solution to an HTN planning problem is a task network[4] that can transform the initial task network to a plan (a sequence of actions) that is executable at the initial state s_0.

4.1 Plan and goal recognition as HTN planning

Plan and goal recognition as HTN planning compiles the PGR problem into an HTN planning problem. Let the sequence of observed actions be $O_k = (o_1, o_2, \ldots, o_i)\,, \forall o \in O_k, o \in A$. Then new propositional symbols and duplicates of the actions in O_k are introduced as a plan prefix, to be placed correctly into the generated plan [17]. New decomposition methods are defined so that newly defined actions with altered preconditions, state features, and effects can be reached. The PGR problem is then reduced to an HTN planning problem that can be solved using off-the-shelf planners, such as the PANDA planning framework[5] [17]. For brevity, specific PGR-methods for the definition of new actions, propositions, and composition methods are not discussed here but can be found in [17]. The solution of a PGR-problem is a goal task (i.e. an initial task network tn_I) recognised as being performed, given the observed actions O_k.

Algorithm 1 summarizes how the PANDA planning framework can be used to recognise a goal task. The lines in black describe the conventional method, while the lines in blue take into account the partial observability of the problem, which will be detailed in the following section. Conventionally, observed actions are parsed into an observation list (see for loop from Line 5) and included in an *observation file* to be used together with the domain and problem files[6] by PANDA for PGR (see Line 12). Initial task networks of problem files that obtain a successful plan given the observation file will be parsed as goal tasks (see Line 13). Algorithm 1 is performed periodically at a pre-defined time interval.

[3]In an HTN, the terms *primitive tasks* and *actions* can be used interchangeably to describe tasks of the lowest abstraction-level that can be executed by the agent.
[4]A task network is a set of compound tasks and actions, governed by their ordering relations.
[5]https://github.com/panda-planner-dev/pandaPIpgrRepairVerify
[6]The domain file describes the compound tasks and actions, as well as the methods to decompose a compound task in a totally or partially ordered manner. A problem file for PGR contains the initial states, and the initial task network.

Algorithm 1 Algorithm for plan recognition with belief value

```
1:  Collect all available measurement data
    $X_1^c, \ldots, X_{M_X}^c, X_1^d, \ldots, X_{N_X}^d$ at time $t$
2:  Determine the belief values of the evidences using Equation 5
3:  Determine the belief values of the actions using Equation 7
4:  observations_list ← []
5:  for each observed action $o$ do
6:      if $bel(o) \geq b_\epsilon$ then
7:          observations_list.append($o$)
8:      end if
9:  end for
10: write observations_list to the observation file
11: for each problem_file do
12:     run PGR using domain, problem and observation files
13:     parse goal task, if a plan exists
14: end for
15: Determine the belief value for each detected goal task using
    Equation 8
```

However, the conventional method of using PANDA for PGR assumes that observation of actions is deterministic, which is possible only if the human agent provides exact information on the actions being executed. As discussed in the previous section, manual communication of actions in an RTSC system is almost impossible, and can thus only be inferred from observed measurement data.

5 CONSIDERING PARTIAL OBSERVABILITY IN PGR

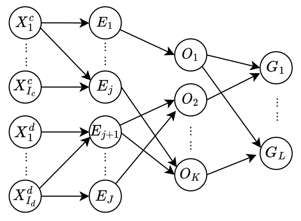

Figure 2: Conditional dependencies between measurement data and evidences, and between evidences and actions.

Many cognitive architectures on skilled human workers (e.g. a pilot) adopt the cognitive Skill, Rule and Knowledge (SRK) architecture proposed by Rasmussen [32]. It is important to note that on the *Skill* level, the human worker relies on the sensorimotor capabilities to execute actions. While the methods for decomposing abstract tasks in the HTN model can be compared to the contents at the *Knowledge*-level in the SRK-model, actions in the HTN model are comparable to the rules of the *Rule*-level. However, as sensors used for monitoring the human agent cannot directly indicate their actions (or "rules" in Rasmussen's model), but only the effects the human's sensorimotors have on the world state, the human actor's current actions can only be inferred from raw measurement data (see Figure 1 for measurement data available in an ultralight cockpit). However, a direct inference of a human's deliberate action from the mixed discrete-continuous measurements may not be always possible. We therefore introduce *evidences* as semantic evidences to be inferred from raw measurement data. As an intuitive example, "being near an airport" is an evidence inferred from the measured position of the airplane. Together with the evidence "aligned with airport" inferred from the measured heading, the action "fly towards airport" can be inferred. Figure 2 depicts the conditional dependencies between measurement data X_i^c and X_i^d, evidences E_j, actions O_k and goal tasks G_l using causal arrows.

Note that E_j, O_k and G_l are all discrete random variables with binary outcomes (i.e. `true` = 1 or `false` = 0), of which a probabilistic estimation from the discrete (and continuous) measurement data X_i^d or X_i^c is possible. The causal arrows represent dependency: if there is no arrow pointing from node B to node A, $P(A|B) = P(A)$.

5.1 Inferring a discrete random variable from discrete or continuous data

Let A denote a random event that assumes *discrete* values a to be inferred using N given (*conditionally independent*) events $\{B_1 = b_1, B_2 = b_2, \ldots, B_N = b_N\}$, with b_n assuming continuous or discrete values. The *posterior probability* of $A = a$ given the events $B_1, \ldots, B_N$, i.e. $P(A \mid B_1, \ldots, B_N)$ can be expressed as

$$P(A \mid B_1, \ldots, B_N) = \frac{1}{1 + \frac{\prod_n P(B_n|\bar{A})P(\bar{A})}{\prod_n P(B_n|A)P(A)}}. \tag{1}$$

Equation 1 can be obtained using Bayes theorem:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}, \tag{2}$$

and the total probability rule (T.P.R.):

$$P(B) = P(B \mid A)P(A) + P(B \mid \bar{A})P(\bar{A}), \tag{3}$$

while considering conditional independence (C.I.) of the events $B_1 \ldots, B_N$. More concretely,

$$\begin{aligned}
&P(A \mid B_1, \ldots, B_N) \\
&\overset{\text{Bayes}}{=} \frac{P(B_1, \ldots, B_N \mid A)\, P(A)}{P(B_1, \ldots, B_N)} \\
&\overset{\text{C.I.}}{=} \frac{P(B_1 \mid A) \cdots P(B_N \mid A)\, P(A)}{P(B_1, \ldots, B_n, B_N)} \\
&\underset{\text{T.P.R.}}{\overset{\text{C.I.}}{=}} \frac{\prod_{n=1}^{n=N} P(B_n \mid A)\, P(A)}{\prod_{n=1}^{n=N} P(B_n \mid A)\, P(A) + \prod_{n=1}^{n=N} P\left(B_n \mid \bar{A}\right) P(\bar{A})} \\
&= \frac{1}{1 + \frac{\prod_n P(B|\bar{A})P(\bar{A})}{\prod_n P(B|A)P(A)}}.
\end{aligned} \tag{4}$$

Note that Equation 1 is tractable, as $P(B|A)$ and $P(B|\bar{A})$ can be obtained using modelled likelihood functions, while the prior probabilities can be assumed in practice as $P(A) = P(\bar{A}) = 0.5$. Table 1 shows a non-extensive list of distributions functions we use to model the likelihood functions required for calculating the following belief values.

5.2 Belief of an evidence

We define the belief distribution of an evidence E = `true` as a conditional probability formally represented as

$$bel(E = 1) = P(E = 1 \mid X_1 = x_1, \ldots, X_I = x_I), \tag{5}$$

where $X_i = x_i$, with $i = 1, \ldots, I$ are the observed measurement data. Using Equation 1 , the belief value $bel(E = 1)$ can be determined if the likelihood functions $f(X_i \mid E = 1)$ are known.

5.3 Belief of an action being performed

Similarly, we define the belief distribution of an action $O = 1$ being performed as a probability conditioned on the measurement data $X_1 = x_1, \ldots, X_I = x_I$:

$$bel(O = 1) = P(O = 1 \mid X_1 = x_1, \ldots, X_I = x_I). \tag{6}$$

Using the *total probability rule*, the above equation can be extended as[7]

$$\begin{aligned}
&bel(O = 1) \\
&= P(O \mid X_1, \ldots, X_I) \\
&= \sum_{E_1 \times \ldots \times E_J \in \{0,1\}^J} P(O \mid E_1, \ldots, E_J) \cdot P(E_1, \ldots, E_J \mid X_1, \ldots, X_I) \\
&\overset{\text{C.I.}}{=} \sum_{E_1 \times \ldots \times E_J \in \{0,1\}^J} P(O \mid E_1, \ldots, E_J) \cdot \prod_{E_j, j \in \{1,\ldots,J\}} P(E_j \mid X_1, \ldots, X_I) \\
&= \sum_{E_1 \times \ldots \times E_J \in \mathcal{E}} P(O \mid E_1, \ldots, E_J) \cdot \prod_{E_j, j \in \{0,\ldots,J\}} P(E_j \mid X_1, \ldots, X_I),
\end{aligned} \tag{7}$$

where $\mathcal{E}$ in the last line represents the set of all possible combinations of outcomes of $E_1 \times \ldots \times E_J$, with $E_j = 1$, for all j having a causal arrow pointing to O, since $P(O \mid E_1, \ldots, E_J) = 0$, if there is a j with a causal arrow pointing from E_j to O, such that $E_j = 0$. We assume that $P(O \mid E_1, \ldots, E_J, X_1, \ldots, X_I)$ is equal to $P(O \mid E_1, \ldots, E_J)$ since the measurement data $X_1, \ldots, X_I$ do not convey more information on the conditional probability $P(O \mid E_1, \ldots, E_J)$, if the evidences are given.

To determine the belief value, Equations 1 can be used, provided the probabilities of $P(E_j \mid O_k)$ and the likelihood functions $f(X_i \mid E_j = e)$ are given.

5.4 Belief of a goal task

The PGR in Algorithm 1 (including the lines in blue) can be used to determine the goal task(s), given the probabilistically inferred actions. First, belief values of the evidences conditioned on the measurement data are determined in Line 2, followed by the belief values of the actions in Line 3. Subsequently, actions with belief values greater than a given threshold will be selected as a pre-filtering step (see for loop from Line 5) and included in an *observation file* to be used together with the domain and problem files by PANDA for recognising the goal task(s) currently executed (see Line 12).

Lastly, the recognised goal task(s) will be attributed belief values computed using Equation 8 (see Line 15). The belief values are

[7]For brevity of notation, discrete random variables Y with binary outcomes take the outcome value $Y = 1$ if it is not specified. By definition, the compliment $\bar{Y}$ takes the outcome value 0.

name	$f(x)$, continuous distribution	parameters	remarks
Normal	$\frac{1}{\sigma_x\sqrt{2\pi}}\exp\left(-\frac{1}{2}\left(\frac{x-\bar{x}}{\sigma_x}\right)^2\right)$	mean, $\bar{x}$; standard deviation, σ_x	outcomes defined in $x \in \mathbb{R}$
Exponential	$\begin{cases}\lambda e^{-\lambda x}, & \text{for} \quad x \geq 0;\\ 0, & \text{for} \quad x < 0,\end{cases}$	$\lambda > 0$, rate parameter	for outcomes defined in $x \in \mathbb{R}_{\geq 0}$
Kumaraswamy	$abx^{a-1}(1-x^a)^{b-1}$	shape parameters, $a, b > 0$	outcomes defined in $x \in [0,1]$;
Uniform	$\begin{cases}\frac{1}{b-a} & \text{for } a \leq x \leq b\\ 0 & \text{for } x < a \text{ or } x > b\end{cases}$	lower bound, a; upper bound, b	has constant likelihood in given range
U-quadratic	$\alpha(x-\beta)^2$	$\beta = \frac{b+a}{2}$, where a and b are lower and upper bounds	outcomes defined in $x \in [a,b]$; represents double-bounded continuous likelihood; can represent variables with similar likelihood at extremities
name	**$P(k)$, discrete distribution**	**parameters**	**remarks**
Bernoulli	$\begin{cases}p & \text{if } k=1\\ q=1-p & \text{if } k=0\end{cases}$	$k = \{0,1\}$	used for discrete random with binary outcomes

Table 1: Likelihood functions modelled using Probability Density Functions (PDF) from [26, 28].

useful to guide the PAS on which goal task is most likely the pilot's intention.

$$bel(G=1) = P(G \mid X_1,\ldots,X_I) \cdot \delta_{PGR}(G), \tag{8}$$

The determination of $P(G \mid X_1,\ldots,X_I)$ is similar to the determination of the belief value of an action, i.e. $bel(O=1)$. The Knonecker-delta $\delta_{PGR}(G)$ takes the value 1 if G is recognised as a goal task by Algorithm 1, otherwise $\delta_{PGR}(G) = 0$.

6 EMPIRICAL EVALUATION

The planning domain used for the empirical evaluation is adopted from the Ultralight Cockpit domain[8] [25].

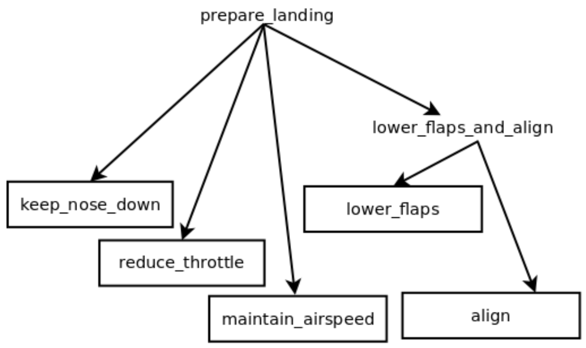

Figure 3: Hierarchical task model for the sub-task "prepare landing" of task "landing.

Figure 3 depicts an example HTN model, with `prepare_landing` (i.e. prepare for landing) being the initial task network. Task names in boxes are actions, i.e. primitive tasks that do not need further decomposition and are directly executable by the human agent

```
(:method m_prepare_landing
  :parameters (?noseDown - NoseDown
    ?throttleThirtyPercent - ThrottleThirtyPercent
    ?correctAirSpeed - CorrectAirSpeed ?airplane - Airplane
    ?landableLocation - LandableLocation ?flapLevelOne - FlapLevelOne
    ?alignedFinalState - AlignedFinalState)
  :task (prepare_landing noseUp throttleThirtyPercent correctAirSpeed)
  :precondition ( )
  :subtasks(and
     (task1 (keep_nose_down ?noseDown ))
     (task2 (reduce_throttle ?throttleThirtyPercent ?noseDown))
     (task3 (maintain_airspeed ?correctAirSpeed))
     (task4 (lower_flaps_and_align ?flapLevelOne  ?alignedFinalState)))
  :ordering(and
     (< task1 task2)
     (< task2 task3)))
```

Figure 4: HDDL-encoding of the method to decompose pilot's task to prepare for landing.

(i.e. the pilot). Other tasks are compound tasks to be decomposed with a *method*, of which the encoding in the Hierarchical Domain Definition Language (HDDL) [18] is shown in Figure 4. HDDL is expressive and easy to comprehend. From Figure 4, it is directly comprehensible that various subtasks are to be performed by pilots while preparing for landing. These are to keep nose down, to reduce throttle, to maintain current airspeed, and to lower flaps while aligning to the landing stripe, as required in the pilot's operating handbook [9]. The subtask `lower_flaps_and_align` is not a primitive task, of which the method(s) to decompose is specified separately in the HDDL file. Furthermore, the ordering of the subtasks can be clearly formulated: the subtasks to be performed are either totally ordered, partially ordered, or without order.

Table 2 shows the selected possible goal tasks (i.e. initial task networks) for the empirical study in this work, and are grouped into tasks performed under nominal and non-nominal flight conditions. Nominal conditions denote normal flight conditions, while non-nominal flight conditions denote adverse flight conditions. The total- or partial-orderings of the tasks, as well as the (maximum) number of actions are listed accordingly. These tasks were selected

[8]https://github.com/UniBwM-IFS-AILab/ValidationTests

type	ordering	goal task	max number of actions
nominal	partial	land_airplane	9
	none	cruise	4
	partial	take_off	4
	total	fly_over_landable_location	3
	none	check_instruments	6
non-nominal	partial	react to engine failure on ground	4
	partial	react_to_engine_failure_during_take_off	5
	partial	react_to_engine_fire_on_ground	4
	partial	react_to_engine_fire_during_take_off	5
	partial	react_to_engine_fire_during_flight	6

Table 2: Selected tasks for empirical evaluation

discrete data	continuous 1-d data	2-d data
lower flaps button	current altitude	fixation points
apply brakes button	latitude/longitude	
release brakes button	throttle ratio	
turn off ignition button	roll ratio	
turn on ignition button	heading	
master switch on button	pitch ratio	
switch fuel tanks button	yaw ratio	
turn off the fuel button	airspeed	

Table 3: Collected sensor data to generate evidence

to cover all the major tasks that a pilot might have to perform in both nominal and non-nominal conditions, but are not exhaustive. With the list of goal tasks in Table 2, there are more than 500 possible plans (i.e. sequences of actions).

6.1 Collecting measurement data

Table 3 shows the measurement data collected from various sensors and cockpit instruments to recognise the pilot's goal task. As described in Section 5, and also depicted in Figure 2, actions of the HTN model often cannot be inferred directly from the collected measurement data. Therefore, we define evidences E_* as intermediate random variables, of which the likelihood functions $P(x \mid E_*)$ can be modelled. For intuitive illustrations, multiple examples will be provided here. The likelihood function of a button being pressed in the event the pilot presses on the button can be modelled by:

$$P(\texttt{button\_state} \mid \texttt{button\_pressed}) = \begin{cases} 0, & \text{if } \texttt{button\_pressed} = 0 \\ 1, & \text{if } \texttt{button\_pressed} = 1 \end{cases} \quad (9)$$

Similarly, the likelihood function can also be defined for continuous measurement data. Consider the evidence E =`getting_near_landable_location` (i.e. to get near a landing location). E has two outcomes (i.e. $e = 1$ or 0). The likelihood function of obtaining the measured distance x given the evidence of getting near the landable location $f(X = x \mid E = 1)$ can be modelled using the Kumaraswamy distribution (see Figure 5). The modelled likelihood functions will subsequently be used to compute the belief values according to Equations 5, 7 and 8. In this work, we consider only likelihood functions selected by domain-experts. However, there exists a possibility that these likelihood functions be obtained using learning algorithms to customise for individual behavior patterns.

The likelihood of the event the pilot is looking at a planar object in the cockpit (e.g. the altimeter, the map display) given the two-dimensional distances of the fixation points (obtained from

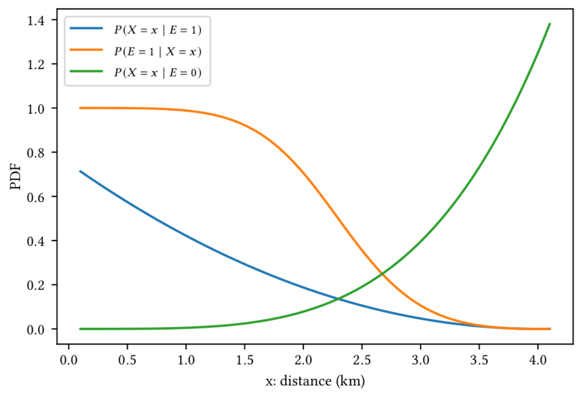

Figure 5: Likelihood functions and posterior probability for the evidence getting near a landable location (in orange) against the measured distance. The Kumaraswamy distribution is used, with 3.5 km being the cut-off distance.

the eye-tracker) to the centroid of the region of interest (ROI) delimiting the object, can be modelled using a multi-variate normal distribution (see Table 1). The belief value of the evidence E =`looking_at_panel` is symmetric around the centroid and has a value of 1, while the value decreases as the fixation point lies farther from the centroid. The PDF is parametrized in such a way that all the points with a distance greater than 2.5 cm from the centroid of the ROI have belief value smaller than 0.5 (size of ROI is 10 cm^2).

6.2 Experiment setup

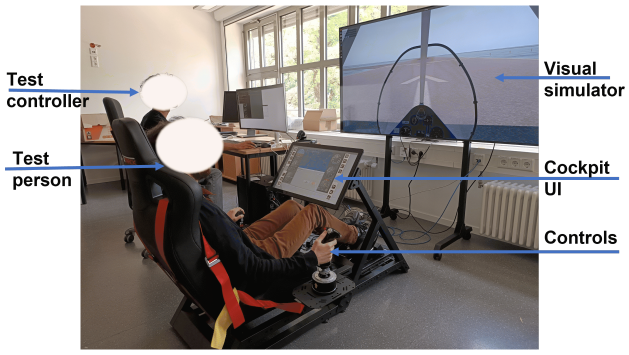

Figure 6: Experiment setup.

Figure 6 shows the setup of the experiment, which is similar to the setup in [21]. X-Plane, a widely used flight simulator with realistic flight dynamics and an immersive simulation environment, is used to simulate the Aerolite 103 aircraft model. Flight data from X-Plane is communicated using ExtPlane[9]. The flight control data, such as the throttle and yoke, are first sent to X-Plane and then read through ExtPlane. Pilot fixation data is generated using

[9]https://github.com/vranki/ExtPlane

Pupil Core[10]. The collected (time-stamped) data is saved in an SQL database.

In each experiment, the test person is instructed to carry out: *i.)* just an action and *ii.)* a goal task from Table 2 by the test controller, and the instruction serves as the ground truth. *It is assumed that the test person is rational and goal-directed.* The experiment is carried out with two different test persons: the first test person is experienced with the simulator, while the second is a novice. The various data sources are collected during the experiments and then processed later[11]. The data are processed to evaluate the inference of actions and goal tasks from raw measurement data.

6.3 Evaluation

All evaluations are carried out using an Intel i7-5820K CPU @ 3.30 GHz with 32 gigabytes of RAM.

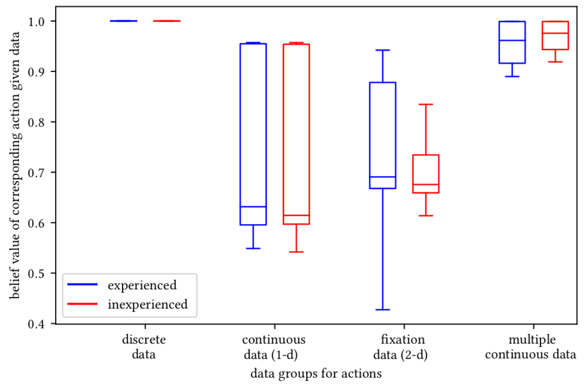

Figure 7: Action recognition comparison between experienced and inexperienced test subjects.

First, the action inference is tested. For this, each test person is instructed to carry out the actions included in the domain. This test is carried out to evaluate the action recognition module. No time limit is set, and the action to be performed is selected randomly. The experiments and data are synchronised using an external time stamp. This experiment is repeated 10 times for each action and each test person. Figure 7 shows the comparison of statistically obtained belief values for the instructed actions grouped according to the respective type of measurement data the actions depend on (i.e. there exists a causal link that can be traced back to at least one type of measurement to the action). In the figure, the data group "multiple-continuous data" refers to the actions, that are inferred by combining multiple continuous data. The developed model outputs sufficiently substantial belief values for the instructed actions.

Next, belief values of actions given the measurement data in Table 3 are evaluated with a temporal resolution of 100 ms. The choice of this temporal resolution is to ensure the real-time capability of the designed system. Figure 8 shows the belief values of three representative detected actions in this experiment (i.e.

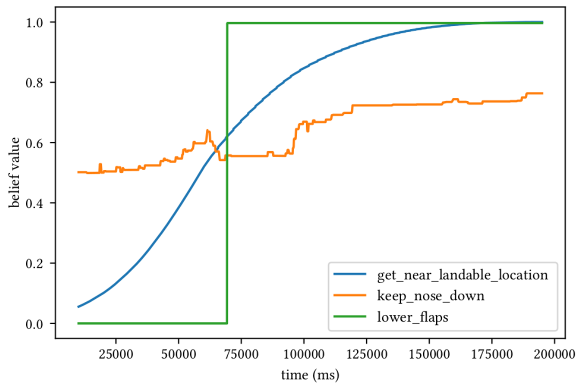

Figure 8: Belief value of selected actions relevant for task "land airplane".

`get_near_landable_location`, `keep_nose_down` and `lower_flaps`), and demonstrates that the belief value for an action varies dynamically over the course of the experiment. The action `lower_flaps` is very clear-cut as the belief value increases immediately to 1 once the button to lower flaps is pressed (i.e. the evidence), which is inferred only from a single measurement data (i.e. state of the button). Similarly, the action `get_near_landable_location`, although follows a continuous variation, also provides a clear increase in value as it depends (indirectly) on two measurement data (i.e. distance to the landable location, and height relative to the landable location). However, the action `keep_nose_down` is associated with a belief value that is more "inconclusive" than the other two actions. This is due to the fact that the pilot needs to regulate the pitch angle of the aircraft continuously and in incremental steps in the course of the action to keep the nose of the aircraft down. While calculating the belief value for the corresponding goal task, the highest yet observed belief value for an action is taken.

The evaluation of goal recognition follows. This workflow is explained using the result of the goal task "land airplane". First, the test person is instructed to carry out this task. At any given time, actions from the list of recognised actions, with a belief value greater than the pre-defined threshold are selected. This observation sequence serves as the input for the goal task recognition. Figure 9 shows the belief value of the task "land airplane". The belief value for the goal task increases when a relevant action for the task is carried out. These are seen as sudden jumps in belief value in Figure 9.

Statistical analysis was carried out to determine the maximum belief value obtained over the course of execution of each goal task in Table 2. A mean maximum belief value of 0.73 and a median belief value of 0.70 were obtained, indicating that the probability model is consistent for recognising the various goal tasks.

Since there are tasks that share the same actions, some tasks are misidentified until the observed action prefix is unique to the task being carried out. This is illustrated in Figure 10. Two task `react_to_engine_failure_on_ground` and `react_to_fire_on_ground` have the same belief value for prefix length of three since they share three actions between them, but once the prefix becomes unique, the belief value of the former task increases while that

[10]https://pupil-labs.com/products/core/
[11]However, the implementation is compatible for use in real-time.

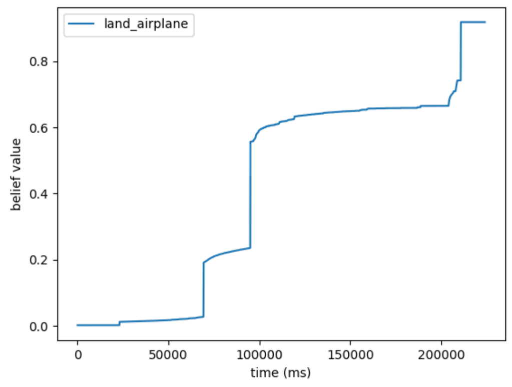

Figure 9: Belief value over time for the goal task "land airplane".

of later stagnates. This relates to the worst-case distinctiveness measure explained in [23] and should be considered in future work.

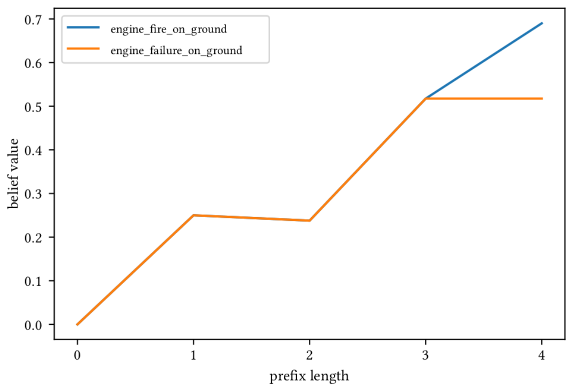

Figure 10: Belief value for goal task `react_to_fire_on_ground` and `react_to_engine_failure_on_ground`, while the pilot was instructed to execute the goal task `react_to_fire_on_ground`).

Figure 11 shows the time required by the plan recognition system given the length of action sequence (prefix length) to detect the corresponding task. For all totally-ordered, partially-ordered, and tasks with no order, the PGR system is able to recognize the task within 10^{-2} seconds. This time includes the I/O operation required to interact with the PGR system. This is in line with the requirement so that the system can be used in an online scenario. Note that the dash in the box plot represents that only one data point is taken. This is limited by the number of actions in goal tasks considered.

7 CONCLUSION AND FURTHER WORK

UL cockpits need to be more secure in the future, and for this, the integration of PAS capable of providing contextual decision-making support is essential. This paper presents a method to leverage HTN models for PGR. The advantage of doing so is to exploit the already

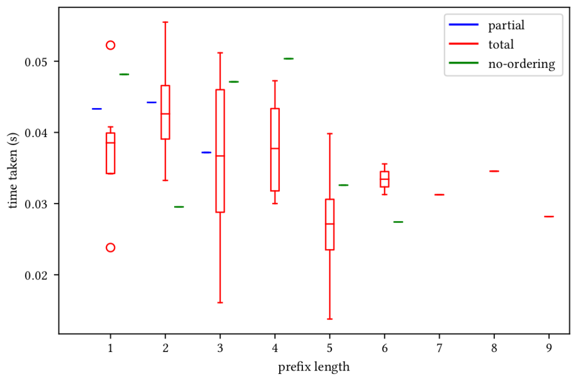

Figure 11: Time taken for PGR with respect to prefix length for different task ordering.

existing HTN models, which are predominantly used in providing guidance (in form of plan suggestions) to the pilot.

Since the PGR requires a prefix of executed actions, which can only be partially observed, we also developed a method to derive the belief value of a recognised goal task. In the event a prefix of executed actions leads to the recognition of multiple goal tasks, the PAS will assume that the goal task with the most substantial belief value *is* indeed the intended goal task. Although the method was validated for the ultralight PAS use case, its transferability to other RTSC systems is highly likely.

The presented method was empirically validated with human-in-the-loop tests. Three key assumptions are made during the development of the system:

- While executing a goal task, the pilot does not miss any action in the HTN model.
- No multitasking is considered, i.e. the pilots do not execute more than one goal task at the same time.
- The pilot is perfectly rational, i.e. the pilot performs the best course of possible actions without error.

Especially in non-nominal situations, due to mental overload or time criticality, the pilot may deviate from the behavior of a perfectly rational agent. Since non-nominal situations are associated with more errors, lifting the above assumptions is essential for future works. A small sample size was used in this paper and should also be increased in future work. Besides, this work only derives belief values of the recognised goal tasks, but not the belief values of the decomposition methods. The belief propagation from the lower abstraction levels to the higher abstraction levels can be included in the future into the HTN-planning framework.

ACKNOWLEDGMENTS

This work is done under the framework of Project MOREALIS, a research and innovation project funded by the German Federal Ministry of Economic Affairs and Climate Action.

REFERENCES

[1] Fethi Abdelmoula and Marco Scholz. 2018. LNAS—a pilot assistance system for low-noise approaches with minimal fuel consumption. In *Proceedings of the 31st congress of the international council of the aeronautical sciences*, Vol. 96. Belo Horizonte, Brazil, 1–14.

[2] Ofra Amir and Ya'akov Gal. 2013. Plan recognition and visualization in exploratory learning environments. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 3, 3 (2013), 1–23.

[3] J. Benton, David Smith, John Kaneshige, Leslie Keely, and Thomas Stucky. 2018. CHAP-E: A Plan Execution Assistant for Pilots. *Proceedings of the International Conference on Automated Planning and Scheduling* 28, 1 (2018), 303–311.

[4] Pascal Bercher, Gregor Behnke, Matthias Kraus, Marvin Schiller, Dietrich Manstetten, Michael Dambier, Michael Dorna, Wolfgang Minker, Birte Glimm, and Susanne Biundo. 2021. Do it yourself, but not alone: companion-technology for home improvement—bringing a planning-based interactive DIY assistant to life. *KI-Künstliche Intelligenz* 35, 3-4 (2021), 367–375.

[5] Francesca Bertacchini, Eleonora Bilotta, and Pietro Pantano. 2017. Shopping with a robotic companion. *Computers in Human Behavior* 77 (2017), 382–395.

[6] BFU. 2022. *Studie zur Flugsicherheit von Luftsportgeräten - Analyse von Unfällen und Störungen mit Luftsportgeräten in Deutschland in den Jahren 2000-2019*. Technical Report BFU22-803.1. Bundesstelle für Flugunfalluntersuchung.

[7] Biundo Biundo and Andreas Wendemuth. 2017. *Companion Technology - A Paradigm Shift in Human-Technology Interaction*. Springer International Publishing.

[8] Nate Blaylock and James Allen. 2006. Hierarchical instantiated goal recognition. In *Proceedings of the AAAI Workshop on Modeling Others from Observations*. 8–15.

[9] Cessna Aircraft Company. 1972. *Cessna 172 1974 Skyhawk Owner's Manual: Pilot Operating Handbook (POH) / Aircraft Flight Manual (AFM)*. Independently Published.

[10] Alex De Voogt, Filipe Chaves, Erik Harden, Miguel Silvestre, and Pedro Gamboa. 2018. Ultralight Accidents in the US, UK, and Portugal. *Safety* 4, 2 (2018), 23.

[11] Juan Fernandez-Olivares and Raul Perez. 2020. Driver Activity Recognition by Means of Temporal HTN Planning. *Proceedings of the International Conference on Automated Planning and Scheduling* 30 (2020), 375–383.

[12] Richard G Freedman and Shlomo Zilberstein. 2019. A unifying perspective of plan, activity, and intent recognition. In *Proceedings of the AAAI Workshops: Plan, Activity, Intent Recognition (Honolulu, HI)*. 1–8.

[13] Christopher W Geib and Mark Steedman. 2007. On Natural Language Processing and Plan Recognition.. In *IJCAI*, Vol. 7. Citeseer, 1612–1617.

[14] Thomas Geier and Pascal Bercher. 2011. On the Decidability of HTN Planning with Task Insertion. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Three (IJCAI'11)*. AAAI Press, Barcelona, Spain, 1955–1961.

[15] M. Ghallab, D. Nau, and P. Traverso. 2016. *Automated Planning and Acting*. Cambridge University Press.

[16] Daniel Höller, Gregor Behnke, Pascal Bercher, and Susanne Biundo. 2016. Assessing the Expressivity of Planning Formalisms through the Comparison to Formal Languages. *Proceedings of the International Conference on Automated Planning and Scheduling* 26, 1 (Mar. 2016), 158–165.

[17] Daniel Höller, Gregor Behnke, Pascal Bercher, and Susanne Biundo. 2018. Plan and Goal Recognition as HTN Planning. In *Proceedings of the 30th IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE Computer Society, Volos, Greece, 466–473.

[18] Daniel Höller, Gregor Behnke, Pascal Bercher, Susanne Biundo, Humbert Fiorino, Damien Pellier, and Ron Alford. 2020. HDDL: An extension to PDDL for expressing hierarchical planning problems. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 9883–9891.

[19] Fabian Honecker and Axel Schulte. 2017. Automated Online Determination of Pilot Activity Under Uncertainty by Using Evidential Reasoning. In *Engineering Psychology and Cognitive Ergonomics: Cognition and Design*, Don Harris (Ed.). Springer International Publishing, Cham, Switzerland, 231–250.

[20] Ajinkya Jain and Scott Niekum. 2018. Efficient hierarchical robot motion planning under uncertainty and hybrid dynamics. In *Conference on Robot Learning*. PMLR, Zürich, Switzerland, 757–766.

[21] Prakash Jamakatel, Sondes Morchedi, and Jane Jean Kiam. 2023. FRICO: An AI-Enabled Friendly Cockpit Assistance System. In *ICAPS 2023 System Demonstrations*.

[22] Charmi Jobanputra, Jatna Bavishi, and Nishant Doshi. 2019. Human activity recognition: A survey. *Procedia Computer Science* 155 (2019), 698–703.

[23] Sarah Keren, Avigdor Gal, and Erez Karpas. 2016. Goal recognition design with non-observable actions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 30.

[24] Jane Jean Kiam, Valerie Hehtke, Eva Besada-Portas, and Axel Schulte. 2019. Hierarchical Planning Guided by Genetic Algorithms for Multiple HAPS in a Time-Varying Environment. In *Intelligent Human Systems Integration 2019*, Waldemar Karwowski and Tareq Ahram (Eds.). Springer International Publishing, Cham, Switzerland, 719–724.

[25] Jane Jean Kiam and Prakash Jamakatel. 2023. Can HTN Planning Make Flying Alone Safer?. In *Proceedings of the 6th ICAPS Workshop on Hierarchical Planning (HPlan 2023)*. 44–48.

[26] Ponnambalam Kumaraswamy. 1980. A generalized probability density function for double-bounded random processes. *Journal of hydrology* 46, 1-2 (1980), 79–88.

[27] Reuth Mirsky, Ya'akov Gal, and Stuart M Shieber. 2017. CRADLE: an online plan recognition algorithm for exploratory domains. *ACM Transactions on Intelligent Systems and Technology (TIST)* 8, 3 (2017), 1–22.

[28] A. Papoulis and S.U. Pillai. 2002. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill.

[29] Kunyu Peng, Alina Roitberg, Kailun Yang, Jiaming Zhang, and Rainer Stiefelhagen. 2022. TransDARC: Transformer-based Driver Activity Recognition with Latent Space Feature Calibration. *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2022), 278–285.

[30] Miquel Ramırez and Hector Geffner. 2009. Plan recognition as planning. In *Proceedings of the 21st international joint conference on Artifical intelligence*. Citeseer, Morgan Kaufmann Publishers Inc, 1778–1783.

[31] Miguel Ramírez and Hector Geffner. 2010. Probabilistic Plan Recognition Using Off-the-Shelf Classical Planners. *Proceedings of the AAAI Conference on Artificial Intelligence* 24, 1 (2010), 1121–1126. https://doi.org/10.1609/aaai.v24i1.7745

[32] Jens Rasmussen. 1983. Skills, rules, and knowledge; signals, signs, and symbols, and other distinctions in human performance models. *IEEE Transactions on Systems, Man, and Cybernetics* SMC-13, 3 (1983), 257–266.

[33] Ian Savage. 2013. Comparing the fatality risks in United States transportation across modes and over time. *Research in transportation economics* 43, 1 (2013), 9–22.

[34] Evren Sirin, Bijan Parsia, Dan Wu, James Hendler, and Dana Nau. 2004. HTN planning for web service composition using SHOP2. *Journal of Web Semantics* 1, 4 (2004), 377–396.

[35] Michael Strohal and Reiner Onken. 1998. Intent and error recognition as part of a knowledge-based cockpit assistant. In *Applications and Science of Computational Intelligence*, Steven K. Rogers, David B. Fogel, James C. Bezdek, and Bruno Bosacchi (Eds.), Vol. 3390. International Society for Optics and Photonics, SPIE, 287 – 299.

[36] Stefan Suck and Florian Fortmann. 2016. Aircraft Pilot Intention Recognition for Advanced Cockpit Assistance Systems. In *Foundations of Augmented Cognition: Neuroergonomics and Operational Neuroscience*, Dylan D. Schmorrow and Cali M. Fidopiastis (Eds.). Springer International Publishing, Cham, Switzerland, 231–240.

[37] Rong Wang, Huiguo Zhang, and Cyril Leung. 2015. Follow me: A personal robotic companion system for the elderly. *International Journal of Information Technology (IJIT)* 21, 1 (2015).

[38] Christopher D Wickens. 2008. Multiple resources and mental workload. *Human factors* 50, 3 (2008), 449–455.

[39] Jason R. Wilson, Seongsik Kim, Ulyana Kurylo, Joseph Cummings, and Eshan Tarneja. 2019. Developing Computational Models of Social Assistance to Guide Socially Assistive Robots. *CoRR* abs/1909.06510 (2019).

[40] Yang Xing, Chen Lv, Huaji Wang, Dongpu Cao, Efstathios Velenis, and Fei-Yue Wang. 2019. Driver activity recognition for intelligent vehicles: A deep learning approach. *IEEE transactions on Vehicular Technology* 68, 6 (2019), 5379–5390.

[41] Yueru Xu, Zhirui Ye, and Chao Wang. 2021. Modeling commercial vehicle drivers' acceptance of advanced driving assistance system (ADAS). *Journal of intelligent and connected vehicles* 4, 3 (2021), 125–135.