

Occupant Activity Recognition in IoT-Enabled Buildings: A Temporal HTN Planning Approach

Ilche Georgievski¹

Service Computing Department, IAAS, University of Stuttgart, Universitaetsstrasse 38, Stuttgart, Germany

Keywords: Activity Recognition, HTN Planning, Plan and Goal Recognition, Smart Buildings, Internet of Things.

Abstract: Given that people spend most of their time indoors, it is imperative that buildings maintain optimal well-being for occupants. To achieve this, research must prioritise occupants over buildings themselves. IoT-enabled buildings can improve quality of life by understanding and responding to occupant's behaviour. This requires recognising what occupants are doing based on IoT data, particularly by considering the objects they use in specific building areas. Situated within the realm of plan and goal recognition as planning, we propose a novel knowledge-engineering approach to occupant activity recognitions leveraging temporal HTN planning. Our approach consists of two primary processes: generating problem instances from IoT data and engineering HTN domain models for activity recognition. The first ensures the representation of IoT data using planning constructs, while the second integrates knowledge about occupant activities into HTN domain models. To support our approach, we provide two HTN domain models tailored for workspaces and homes. Experimental validation with the latter domain and a real-world dataset show that the quality of our computed solutions surpasses that of baseline data-driven approaches and is comparable to more advanced, hybrid approaches.

1 INTRODUCTION


The need for indoor environments that prioritise occupant well-being has become critical, as people spend the majority of their time indoors (Klepeis et al., 2001; Matz et al., 2015). This demand is further intensified by factors such as the growing emphasis on sustainability like carbon awareness, demographic shifts like an ageing population, and the rise of hybrid working models. In response, buildings, such as homes and workspaces, are being equipped with advanced technologies, including Internet of Things (IoT) devices and Artificial Intelligence (AI) functionality. These innovations enable real-time monitoring and dynamic control of buildings. However, to truly enhance occupant well-being, the focus must shift from merely optimising the buildings themselves to supporting the occupants (Awada et al., 2021). This also includes understanding occupants' activities as informed by buildings' IoT data, introducing the challenge of *occupant activity recognition*.

Occupant activity recognition is essential for creating buildings responsive to the needs of occupants (Nguyen and Aiello, 2013). However, exist-

ing approaches to this problem have significant limitations. Data-driven methods, while powerful, demand big data and often struggle with generalisation to unseen scenarios (Chen et al., 2021). Conversely, knowledge-based techniques, such as ontological reasoning, involve complex modelling (Uschold and Gruninger, 1996), may lack adaptability and scalability (Sirin et al., 2007), and have difficulty with temporal reasoning (Riboni et al., 2011). Despite the variety of available methods, a gap persists in having approaches that generalise across buildings and offer structured and adaptable integration of activities.

One promising avenue for addressing this challenge lies in the area of Plan and Goal Recognition (PGR) (Sukthankar et al., 2014; Van-Horenbeke and Peer, 2021). PGR involves inferring an agent's plans and goals based on observed actions, typically relying on a plan library or domain theory. Notably, the approach known as *PGR as planning* (Ramírez and Geffner, 2009) has shown potential by leveraging domain theory and AI planners to solve PGR problems. This approach offers high expressiveness, support for adaptable knowledge representations, the use of off-the-shelf tools, and ability to operate without big data.

Building on the strengths of PGR as planning, we propose an approach to occupant activity recognition

¹ <https://orcid.org/0000-0001-6745-0063>

based on PGR as Hierarchical Task Network (HTN) planning (Höller et al., 2018). Unlike PGR as planning that uses action-based domain theory, our approach employs HTNs as a domain theory to capture occupant activity recognition at multiple levels of abstraction in a structured fashion. HTN planning is also known for its computational efficiency and scalability, making it well-suited for the challenges of buildings.

Specifically, we propose a novel knowledge-engineering approach to framing occupant activity recognition as a temporal HTN planning problem. This approach consists of two key processes. One involves developing HTN domain models that encapsulate a hierarchical structure of tasks that describe occupant activities via involved objects and associated IoT devices, and general mechanisms for handling IoT data. The other outlines the integration of IoT data as temporal facts in the initial state of problem instances. The solution to this planning problem is a set of activities, with start and end times, recognised across different locations within a given building.

Our main contributions are as follows:

- We propose a novel framing of occupant activity recognition as a temporal HTN planning problem. This framing focuses on addressing the complexity of building environments, including hierarchical relationships, temporal properties, and IoT data integration.
- We introduce a knowledge-engineering approach that enables the application of HTN planning for occupant activity recognition. This approach ensures guided transformation of domain knowledge into a domain model and clear definition of goals and states within problem instances, essential for the success of planning (Bhatnagar et al., 2022). These problem-formulation challenges are particularly pronounced in realistic applications, where dynamic conditions require solutions to be generated in real time, as opposed to being derived from simplified or pre-generated benchmarks.
- Our work extends the application of PGR as HTN planning beyond its previous contexts, e.g., driver and office activity recognition (Fernandez-Olivares and Perez, 2020; Georgievski, 2022), to varied buildings. This also contributes to the generalisation gap in the literature, making our work unique in its focus on cross-domain application.
- By following our approach, we develop HTN domain models designed explicitly for occupant activity recognition in two common building types: workspaces and homes. Their design allows for application in various settings, from case studies to large-scale environments, offering scalability

and adaptability. These domain models can serve as a basis for further research and development.

- We show the efficiency of our approach using the *Homes* domain, a real-world dataset, and an existing HTN planner. Our preliminary results show improvements over baseline data-driven methods and comparable performance to hybrid approaches, highlighting the potential of our work.

The rest of the paper is organised as follows. Section 2 describes the occupant activity recognition problem. Section 3 offers a brief overview of temporal HTN planning. Section 4 introduces our knowledge-engineering approach, and Section 5 gives insights into the two domain models and the preliminary evaluation. Sections 6 and 7 present related work and conclusions, respectively.

2 OCCUPANT ACTIVITY RECOGNITION

We outline key concepts for occupant activity recognition, focusing on their relevance to engineering domain models and problem instances. While this overview covers essential aspects, formalising occupant activity recognition and its correspondence to HTN planning is beyond this scope and can be found elsewhere, e.g., (Georgievski, 2022).

We consider buildings equipped with IoT sensors for real-time monitoring and data collection. Sensors report readings whenever changes occur in the observed space or at regular intervals via a publish/subscribe mechanism (Al-Masri et al., 2020). The readings can be *binary data* (e.g., on/off states) or *numeric data* (e.g., humidity values). To determine whether numeric sensors are active or not, threshold values are employed (Nguyen et al., 2014). Thresholds can be *absolute* (the minimum detectable value), or *dynamic* (varying based on spatial and usage characteristics).

Buildings are organised into *locations*, each containing IoT devices. Occupants perform various activities within these locations, with an *activity area* being a logical space where specific activities occur (Curry, 1996). Ergo, *spatial relationships* emerge, linking sensors or activities to locations. The *type of activity* is often dictated by occupants' location. That is, the type of location and objects with which occupants interact – explicitly and implicitly – can constrain their activities (Liao et al., 2005; Wu et al., 2007). When these objects are associated with sensors, each type of occupant activity can be defined in terms of the *relevant sensors deployed in a specific*

activity area. For example, the definition of “Working with a computer” might involve sensors on a computer, keyboard, mouse, and chair.

Occupant’s mere presence in a location, with or without engaging in anything specifically, represents the most fundamental occupant activity, called “Presence”. Part of this means that performing a specific activity in some location also entails the “Presence” activity. For example, the activity “Working” performed in some location also requires being present. “Working” can be further refined into sub-activities, such as “Working with a computer” and “Reading”, revealing *hierarchical relationships* among the activities. Conversely, understanding when people do not occupy some location holds significance (e.g., energy saving). We can interpret this situation as a complementary type of activity called “Absence”, which can also indicate that no activity takes place (Nguyen et al., 2014). “Absence” exists at the same hierarchical level as “Presence”, indicating that an activity in a location can be either presence-based or “Absence”.

Occupant activity recognition entails *processing temporal readings of sensors* associated with objects that occupants interact with while performing presence-based activities in specific locations at specific times. Recognised activities must conform to a predefined set of relevant objects deployed in specific locations and readings of sensors associated with those objects at specific time points. Thus, the occupant activity recognition problem can be defined as follows: *Given a building with locations and sensors characterised by thresholds and spatial properties, and a set of possible occupant activities with hierarchical relationships and described by sensors linked to relevant objects, determine the set of activities occurring in all activity areas consistent with the predefined activities and temporal readings.*

3 BACKGROUND

We approach occupant activity recognition as state-based HTN planning (Georgievski and Aiello, 2015). A central construct is a task network, which is a hierarchy of primitive and compound tasks. Primitive tasks are executable actions, whereas the latter encapsulate domain knowledge that extends beyond actions. These compound tasks are decomposable into subtasks using methods. An objective is modelled as an initial task network that should be decomposed starting from an initial state using the task hierarchy. The solution is a course of action executable in the initial state.

Building on this standard concept, temporal HTN planning integrates time into the state and tasks. We focus on this integration as captured by the Hierarchical Domain Definition Language (HPDL), as designed for the SIADEX planner (Castillo et al., 2006). HPDL builds upon the Planning Domain Definition Language (PDDL) (McDermott et al., 1998), where primitive and compound tasks along with their decomposition methods are directly mapped into HPDL elements. Beyond the standard HTN constructs and derived predicates, HPDL incorporates a temporal dimension that aligns with PDDL 2.1 level 3 (e.g., temporal facts). HPDL also introduces several special features, which are instrumental in our knowledge-engineering approach. These include the modelling of temporally constrained task, incorporation of an inference task, and use of the *bind* predicate.

- Temporal facts are timed initial literals or facts with time points. For example, the fact `(timestamp "30/11/2023 09:31:32" 0)` encodes a sensor reading’s timestamp with index 0.
- Temporally constrained task is a task in a method’s subtasks that comes with temporal constraints over its start and end points and duration. For this, three special variables are used, `?start`, `?end` and `?dur`, and a logical expression with relational operators for the constraints.
- Inference task is a task that can appear in a method’s subtasks to add or remove new facts into the current state without an explicit action invocation or capture information from the current state. This task is of the form `:inline <precondition> <effect>`, where precondition and effects are usual logical expressions as defined for an action’s preconditions and effects.
- The *bind* predicate is used to bind a variable by evaluating an expression. It is of the form `(bind <var> <expression>)`.

4 THE APPROACH

Figure 1 illustrates our approach to occupant activity recognition, which consists of two main processes: *problem instance generation* from sensor data and *engineering domain models* for activity recognition. In the first process, problem instances in HPDL are generated by converting raw sensor readings into indexed temporal observations, which are then transformed into indexed facts for the initial state of an HPDL problem instance. The concept of *indexing* draws inspiration from (Fernandez-Olivares and Perez, 2020).

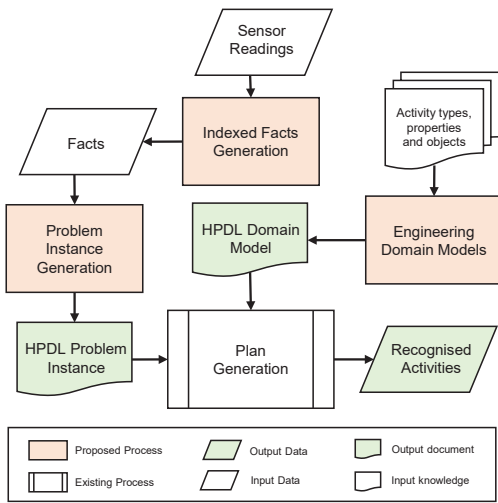


Figure 1: Flowchart representing the approach for occupant activity recognition as HTN planning.

The second process involves acquiring knowledge about relevant activities and engineering it into an HPDL domain model. This includes insights about activities, their properties, and involved objects from prior experience, existing activity models, and literature. We represent activities as primitive tasks. When a specific activity occurs at some location, the corresponding primitive task is added to the plan. Since activities can last for specific periods, the plan’s tasks are annotated with start and end times. This requires aligning the temporal points of primitive tasks with sensor timestamps (C1) and ensuring no temporal conflicts between tasks in the plan, allowing for parallel activities at different locations (C2). The HTN domain model is responsible for achieving condition C1, while plan generation ensures C2.

These processes provide the inputs for the plan generation, which is executed by a temporal HTN planner, resulting in a plan of temporally annotated activities recognised in locations within the building.

4.1 Indexed Facts Generation

We assume that raw sensor readings are gathered via a standard IoT publish/subscribe mechanism, where sensors send their readings to topics managed by a message broker, see Figure 2. These readings are stored in the order they are received, allowing for retrieval in the same sequence.

Each sensor reading includes the value and timestamp, which are translated into an indexed temporal observation. An *indexed temporal observation* is a tuple $(sID, sVal, rTimestamp, idx)$, where sID is the sensor’s unique identifier, $sVal$ is the sensor reading value, $rTimestamp$ is the reading’s timestamp, and idx

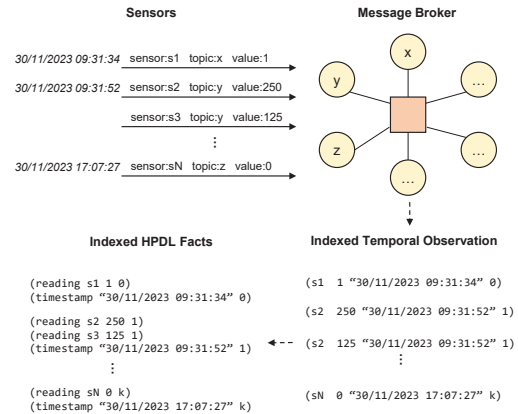


Figure 2: Process of transforming sensor readings to indexed temporal observations and generating HPDL facts.

is the reading’s position in the sequence.

Observing the correct order of sensor readings is crucial for accurate activity recognition. If readings are processed out of order, it can lead to incorrect inferences about activities. For example, if the planner first processes the PC is turned on at time t and then the monitor is on at time t' , it might incorrectly conclude the occupant is working on the PC, even if the PC was turned off between t and t' . Indexing ensures sensor readings are handled in strict sequence, enabling accurate reasoning.

To represent an indexed temporal observation, we use standard and temporal facts in HPDL. Figure 2 illustrates how sensor readings are translated into HPDL facts. For example, at time $30/11/2023\ 09:31:54$, sensors $s2$ and $s3$ report readings simultaneously on topic y , so their observations share the same timestamp and index. These are then translated into standard predicates for each reading and a temporal fact linking the timestamp to the reading index.

4.2 Engineering Domain Models

We now outline the general process for engineering HPDL domain models capable of performing occupant activity recognition. This includes the representation of relevant objects, the modelling of mechanisms for manipulating sensor readings, and the representation of occupant activities.

4.2.1 Representing Sensors, Locations, and Activities

The main types of objects within living spaces related to user activities are sensors, locations, and activities. These are represented as types at the highest level of the typing hierarchy in HPDL. We can then define subtypes using these types. Since sensors can be bi-

nary or numeric, we define their corresponding types as subtypes of the sensor type. Each sensor can be defined using these two types; for example, a monitor is a binary sensor, and a chair pressure sensor is a numeric one.

To represent sensor locations, sensor readings, and sensor status, we define a set of predicates, some of which are shown in Figure 2. Other predicates include `(sensorAt ?s - sensor ?l - location)` and `(active ?s - sensor)`.

To infer useful information from the basic predicates, we define several derived predicates. Particularly, `(isBinary ?s)` and `(isNumeric ?s)` enable determining the type of a sensor when the sensor subtype is passed as a parameter. Also, we define `(existsActiveSensor ?l - location)` to determine whether there is any active sensor at a location.

For the representation of the current ongoing activity and its start time, we employ two predicates. The first one takes the location and name of the current activity as arguments, whereas the second one takes the location and start time: `(currentAct ?l - location ?a - activity)`, `(startAct ?l - location ?time - number)`.

To infer location types, we propose a straightforward representation requiring the specification of one predicate per location in the form `(isRoomType ?l - roomType)`. For example, if the location is a *working room*, then the location type inferring predicate is `(isWorkingRoom ?l - workingRoom)`, with `workingRoom` defined under the `location` type.

4.2.2 Representing the Indexing Concept

To represent and infer the current index, we use a derived predicate, `(idx ?i - number)`. However, to get the last index in the sequence of sensor readings in some location, we need an additional derived predicate `(lastIdx ?i - number)`. Since these predicates depend on numeric variables, we need functions to generate their values, `(currentIdx)` and `(lastIdx)`. To tie the derived predicates and functions together, we employ the *bind* predicate to link *i* to the value of the current index in the sequence of sensor readings: `(:derived (idx ?i) (bind ?i (curIdx)))`. Similarly for the last index.

4.2.3 Representing the Changing Threshold Concept

To address the concept of changing threshold, we use a predicate and a function. The predicate, taking the numeric sensor, the new threshold value, and the index as parameters, represents the updated value of

the threshold for a given sensor. When a sensor's threshold gets changed, this predicate needs to be inserted at the same or before the index of the sensor readings related to this threshold. The function `(threshold ?s - numeric)` is used to associate the threshold of a numeric sensor with a numeric value.

4.2.4 Representing a Recognised Activity

For the representation of recognised activities, we propose using one primitive task, specifically the action *recognised_activity* with two parameters for the activity location and type. The action has start and endpoints but has no preconditions and effects. As this action is the only primitive task, the activities recognised in a location represent a temporally ordered sequence of `(:action recognisedActivity ?l ?a)` instances. Hence, for each *a* and *a'* at l_i , where *a* is added before *a'* in the sequence, it follows $end(a) \leq start(a')$. Consequently, a plan is a sequence of instances of this action ordered by location, i.e., the first subsequence of actions is for location l_i , the next subsequence for location l_{i+1} , and so on, aligning with the order of locations provided in the initial task network. In the final plan, for *a* at l_i and *a'* at l_j , no temporal constraints exist between *a* and *a'*.

4.2.5 Updating Sensor Status

To update the active status of sensors based on their current readings, we introduce a dedicated compound task, which distinguishes between binary and numeric sensors and has two methods for each sensor type. For binary sensors, the task activates or deactivates a sensor when the current sensor reading is 1 or 0, respectively. For numeric sensors, the logic is more complex yet the outcome remains the same. Specifically, if a numeric sensor's reading equals or exceeds its threshold, the sensor is active. Otherwise, the sensor is inactive. In all task's methods, we can employ inference tasks whose effects consist of either `(sensorActive ?s)` or `(not (sensorActive ?s))`.

4.2.6 Processing Sensor Readings per Location

Recognising occupant activities per location dictates how sensor readings should be processed to identify specific activities. We address this with a dedicated compound task, `(:task recogniseActivitiesIn ?l)`. This task handles sensor readings at location *?l* while considering dynamic thresholds and temporal information at the current index. Four methods realise

this task. The first captures new threshold values for numeric sensors at the current index. The second processes a new sensor reading by invoking the compound task for updating the sensor status and then marking the reading as processed using an inference task. The third method recognises an activity at a time point corresponding to the current index, capturing the temporal information of the current reading in its precondition and triggering the compound task (`recogniseActivity ?l ?t`) (detailed in the next subsection). Upon recognising an activity, the method increments the current index, informing the planner to look at sensor readings at the next index. These three methods lead to different decompositions via their task networks but all end with a recursive call to (`recogniseActivitiesIn ?l`). A fourth method serves as the base case to end the recursion, applied when the sequence of sensor readings for that location is complete. This method's task network includes three tasks: an inference task capturing the current activity at `?l` (including its start and end times), then the primitive task (`recognisedActivity ?a ?l`), ensuring it fits within the temporal constraints of the plan (if a temporal violation occurs, the decomposition would fail, leading to backtracking; however, since sensor readings naturally progress in time, this issue can be resolved by resetting the index), and an inference task to reset the current index.

4.2.7 Representing the Activity Hierarchy

To systematically recognise a specific activity, we propose employing a compound task (`recogniseActivity ?l ?t`), where `?l` is the location and `?t` is the timestamp. This task includes methods for different types of locations, grouped into two categories: presence-based activities and absence. Each method for a presence-based activity decomposes into a compound task for recognising activities in a specific type of location. For presence-based activities, each method decomposes into a task for recognizing activities in a specific type of location. For example, in an office building with three room types, there would be three methods, each handling presence-based activities for a particular room type. The "Absence" method checks if all sensors in the location are inactive, then decomposes into the task for recognising "Absence".

4.2.8 Recognising Presence-Based Activities

We can represent the recognition of presence-based activities at multiple levels of abstraction. At the highest level, we propose using a compound task with

at least three methods: one for recognising the basic "Presence" activity, another for continuous recognition (details in the last subsection), and one for recognising specific activities. For instance, recognising a meeting in an office might involve methods for presence, continuous presence, and conducting a meeting. If the method for recognising a specific activity fails, the process can fall back to identifying presence. The method for recognising a specific activity can also lead to further refinement if the activity can be broken down hierarchically. For example, "Meeting" might be refined into "Having a meeting" and "Giving a presentation", each represented by a compound task with its own methods, plus one for continuous recognition. This approach allows for encoding complex hierarchical relationships as needed. Each of these methods includes preconditions based on the active status of relevant sensors at the location. For example, "Giving a presentation" might require active chair pressure, projector, and acoustic sensors. If a method leads to further abstraction, it decomposes into the appropriate compound task; otherwise, it decomposes into a task that marks the beginning of the current activity and the end of the previous one, ensuring temporal constraints are met while executing (`recognisedActivity ?l ?a`).

4.2.9 Recognising "Absence"

Recognising "Absence" operates under the assumption that none of the sensors deployed at a specific location are active. This process is encapsulated in a compound task consisting of two methods. One is responsible for continuous recognition, while the other for recognising "Absence" at a given location by decomposing to the task that marks the beginning and end of activities.

4.2.10 Recognising Activities Continuously

For each type of activity, at different abstraction levels, a method is needed to handle situations where an activity is already in progress at a location. This method, while sharing similar preconditions with others, includes the predicate (`currentAct ?l ?a`). This design choice allows the planning process to continue by moving to the next index without adding the ongoing activity to the plan. When an activity `?a` is already in progress, it is stored in the predicate (`currentAct ?l ?a`). If a new activity `a'` is recognised at the same location, the current activity `?a` can then be added to the plan. The start time of `?a` is already recorded in (`startAct ?l ?timepoint`), and its end time is set to when `a'` begins. This approach allows the planning process to accurately assign start

and end times to activities.

4.3 Problem Instances Generation

For the HPDL problem instance, we define sensors and relevant locations as objects. In the initial state segment, we require initialising the sensor locations, thresholds for numeric sensors, current activity at each location to “Absence”, sensor readings and corresponding timestamps, current index to 0, and last index to k , where k is the last index in the sensor readings sequence. In the initial task network’s segment, we specify the locations where activities should be recognised using the (`recogniseActivitiesIn ?l`) task.

5 DOMAIN MODELS AND PRELIMINARY EVALUATION

We now present our results. We first give insights into the two domain models followed by the experimental setup and experimental results of our preliminary evaluation.

5.1 The Two Domain Models

Following our approach, we developed two HPDL domain models: one for recognising activities in workspaces and another for activities in homes. The *Workspace* domain can recognise nine activity types in three location types, informed by our experience with living labs and existing knowledge of office activities (Nguyen et al., 2014; Georgievski, 2022). The *Homes* domain can recognise ten activity types in five location types, drawing on our experience and existing knowledge of home activities (Naeem et al., 2007; van Kasteren et al., 2011). Both domains are designed to be generic and flexible, supporting typical activities in workspaces and homes while allowing for customisation. The two domains with sample problem instances are publicly available on GitHub.¹

5.2 Experimental Setup

Dataset To test our approach with real-world data, we used a publicly available dataset (Azkune and Almeida, 2018) of a real home (Kasteren Home A (van Kasteren et al., 2011)). The home, an apartment of a 26-year-old male, has three rooms equipped with 14 simple, non-obtrusive sensors. These sensors collected 1319 readings over 25 different days. The

¹<https://github.com/PlanX-Universe/domains>

inhabitant annotated his activities using speech recognition via a headset.

Planner. We use the publicly available version of SIADEX, which is a temporal HTN planner.²

Modelling Considerations. We use our *Homes* domain model. Since the selected dataset includes only binary sensor readings, we do not need to account for numeric sensors or dynamic thresholds. Furthermore, determining when a sensor becomes inactive (switches from 1 to 0) can be inconsistent, as some sensors deactivate immediately after activation, while others remain active for extended periods, making it difficult to interpret the status change. Thus, for this experiment, we exclude predicates related to sensor deactivation and focus solely on activation readings. Moreover, because the SIADEX’s version does not support timed initial literals, we use numeric fluents to represent time (e.g., 93347 for 09:33:47). We also split the dataset into separate problem instances for each of the 25 days as we lack timestamps to prevent temporal violations – the end of the last activity on one day may occur later than the start of the first activity on the following day.

Evaluation Metrics. We evaluate our approach using standard accuracy metrics: precision, recall, F-score, and relative gap. To obtain ground truth for comparison, we label each sensor reading in the dataset with the activity inferred from the inhabitant’s annotations, treating “None” annotations as “Absence” activities. Since our approach produces a plan with recognised activities by location, a direct comparison with the labelled ground truth is not possible. Instead, we re-label the original dataset according to the recognised activities in the plans. We calculate the metrics for each day, with the following assumptions: if the ground truth contains no activity instances for a day and our results also show no activity, we assign a score of 1 to precision, recall, and F-score for that activity (i.e., “Absence”). Otherwise, if our approach labels readings with activities that do not match the ground truth, we assign a score of 0 to each metric.

Comparison. We compare our approach with baseline data-driven techniques, including Naive Bayes (NB), Hidden Markov Model (HMM), Hidden Semi-Markov Model (HSMM), and Conditional Random

²<https://github.com/UGR-IntelligentSystemsGroup/HPDL-Planner>

Field (CRF) (van Kasteren et al., 2011). Additionally, we evaluate it against hybrid or enhanced approaches that combine data-driven and knowledge-based techniques, such as ARF (Ihianle et al., 2018), HARS (Azkune and Almeida, 2018), and USMART (Ye et al., 2014).

5.3 Results

Daily precision and recall percentages offer limited insights, as most activities show inconsistent oscillation with no clear pattern. The highest and most stable precision scores are observed for “Using toilet” and “Preparing breakfast”, while the scores for other activities fluctuate more widely. Precision for “Leaving house” and “Going to bed” also oscillates, though less extremely. This variability likely stems from the fact that each of these activities is identified by a single sensor, meaning every activation leads to recognition of the corresponding activity. Irrelevant sensor activations are common in the dataset, further adding to these fluctuations. Precision for “Absence” goes to the extreme, shifting between 1 and 0, indicating that we either recognise all or none of the non-activity readings. Recall scores also exhibit oscillation, with our approach nearly always recognising instances of “Going to bed” and “Preparing breakfast”. Other activities fluctuate between 1 and 0, primarily due to our convention of assigning 0 or 1 to metrics when they are not applicable.

Figure 3 shows the average metric percentages. The highest F-scores were for “Preparing breakfast” (0.95), “Going to bed” (0.88), and “Leaving house” (0.85). On the lower end, “Preparing dinner” and “Getting drink” scored 0.61 and 0.6, respectively, while “Taking shower” and “Absence” scored 0.51 and 0.5. Although “Using toilet” had a high precision of 0.97, its overall activity recognition rate was only 59%, leading to an F-score of 0.73.

Our approach effectively distinguishes between similar activities, such as “Preparing breakfast” and “Preparing dinner”, due to their distinct times of day, allowing the planner to check the relevance of the current timestamp. However, it struggles when time spans overlap, such as with “Getting drink” and “Preparing dinner”. Although both activities have similar F-scores, “Getting drink” has higher precision, while more instances of “Preparing dinner” are recognised.

Table 1 shows the comparison between our approach, marked with an asterisk (*), and baseline data-driven approaches, ordered by F-score. Our approach achieved 74.32% precision, 73.6% recall, and a 73.95% F-score, demonstrating that our knowledge-

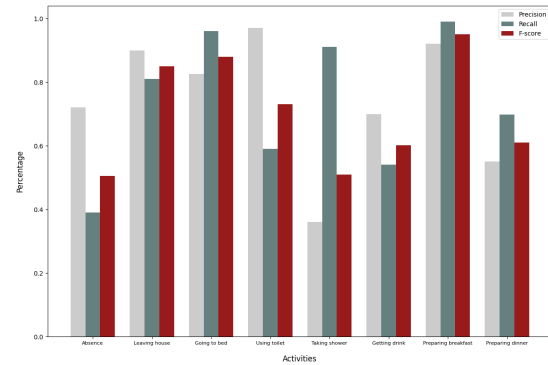


Figure 3: Average precision, recall, and F-score per activity.

Table 1: Accuracy comparison with baseline data-driven approaches on the Kasteren House A dataset.

| Approach | Precision | Recall | F-score |
|----------|--------------|--------------|--------------|
| NB | 67.3% | 64.8% | 65.8% |
| CRF | 73.5% | 68.0% | 70.4% |
| HMM | 70.3% | 74.3% | 72.0% |
| HSMM | 70.5% | 75.0% | 72.4% |
| * | 74.3% | 73.6% | 73.9% |

Table 2: Gap analysis with hybrid approaches using the Kasteren House A dataset.

| Approach | USMART | HARS | HARS |
|----------|--------|------|------|
| Gap | 0.07 | 3.97 | 19.5 |

based approach outperforms traditional data-driven baselines.

Table 2 shows the F-score gaps between our approach and more advanced, hybrid approaches. The gap between our approach and USMART is minimal, suggesting that our approach is on par with USMART, despite USMART not accounting for the “None” activity. The ability of our approach to recognise “Absence” gives it an advantage in scenarios where distinguishing between presence and absence is critical. The gap with HARS is slightly larger at 3%, indicating that HARS benefits from its combination of data-driven and knowledge-based techniques but only marginally outperforms our approach. However, the 19% gap between our approach and ARF highlights ARF’s superior performance, likely due to its effective integration of ontological reasoning with classification models. This gap points out to the potential benefits of further enhancing our approach by incorporating data-driven strategies.

6 RELATED WORK

The research on this topic can be broadly categorised into two main streams: activity recognition and plan and goal recognition (Sukthankar et al., 2014; Van-

Horenbeke and Peer, 2021). Activity recognition focuses on discovering meaningful human activities from potentially noisy, low-level sensor data. This problem has been addressed using both data-driven and knowledge-based approaches. Data-driven methods rely on large datasets for training and feature engineering while exhibiting variable prediction accuracy and poor generalisation (Jobanputra et al., 2019; Minh Dang et al., 2020; Chen et al., 2021). Knowledge-based approaches, on the other hand, formalise activity models using prior knowledge and apply reasoning to detect activities, achieving reasonable accuracy but struggle with temporal constraints, uncertainty, scalability, and adaptability (Chen et al., 2008; Riboni et al., 2011; Chen et al., 2011; Nguyen et al., 2014).

Plan and goal recognition focuses on identifying high-level goals and plans to reach goals by observing agents' primitive actions. A common approach involves using plan libraries to represent possible plans to be recognised in various domains, such as homes (Simpson et al., 2006; Bouchard et al., 2006). Some studies also consider the temporal properties of recognised plans, e.g., (Levine and Williams, 2014). A notable approach is PGR as planning, where an agent's behaviour is represented in domain theory (Ramírez and Geffner, 2009). Like our work, PGR as planning relies on domain models, typically STRIPS-based (Ramírez and Geffner, 2009; Sohrabi et al., 2016) and recently non-temporal HTNs (Höller et al., 2018). Although domain-independent, these proposals lack expressiveness needed to capture the nuances of occupant activity recognition and do tackle the knowledge-engineering aspect.

HTN planning has been applied to recognise and label sequences of divers' activities using event logs according to hours of service regulations (Fernandez-Olivares and Perez, 2020). In this approach, initial observations correspond to events in a driver's log, and service regulations are translated into a temporal HTN domain to identify driving activities, resulting in plans as interpretable labelled event logs. Also, HTN planning has been used for activity recognition in offices, though without tackling the domain-engineering problem and temporal properties (Georgievski, 2022).

Our work parallels runtime verification in planning, which studies whether observed plan executions meet intended plan semantics (Bensalem et al., 2014), and conformance checking in process mining, which studies whether business process executions, as recorded in event logs, conform to process models (van der Aalst, 2011). We go beyond verifying activities against sensor readings by identifying spe-

cific activities and generating plans that interpret sensor readings as activities with start and end times.

7 CONCLUSIONS

Enabling buildings to support occupants and their well-being requires placing occupants in the centre of their operation and recognising occupants' behaviour. We propose a novel framing of occupant activity recognition as temporal HTN planning and a new domain-engineering approach for designing HTN domain models and problem instances that effectively capture the essential aspects of occupant activities and IoT data processing. We also developed HTN domain models for workspaces and homes, demonstrating both the applicability of our approach and its ability to address real-world challenges in occupant activity recognition. Our work emphasises the importance of problem formulation in AI planning, extending beyond traditional benchmark problem generation. The efficiency of our proposal was validated using the *Homes* domain on a real dataset, showing high precision and recall for specific activities, and competitive performance compared to other approaches.

Future research will focus on refining HTN domain models to capture more nuanced activities, incorporating additional IoT data, and expanding our approach to diverse environments, such as assisted living facilities. Integrating machine learning techniques and large language models could further enhance accuracy and adaptability to occupant behaviours. For example, the integration of HTN planning as a classifier into the training process of data-driven approaches is an interesting direction for exploring performance benefits.

ACKNOWLEDGEMENTS

I thank Ivaylo Spasov for his contributions to the work that forms the foundation of this paper.

REFERENCES

- Al-Masri, E., Kalyanam, K. R., Batts, J., Kim, J., Singh, S., Vo, T., and Yan, C. (2020). Investigating Messaging Protocols for the Internet of Things (IoT). *IEEE Access*, 8:94880–94911.
- Awada, M., Becerik-Gerber, B., Hoque, S., O'Neill, Z., Pedrielli, G., Wen, J., and Wu, T. (2021). Ten questions concerning occupant health in buildings during normal operations and extreme events including

- the covid-19 pandemic. *Building and Environment*, 188:107480.
- Azkune, G. and Almeida, A. (2018). A Scalable Hybrid Activity Recognition Approach for Intelligent Environments. *IEEE Access*, 6:41745–41759.
- Bensalem, S., Havelund, K., and Orlandini, A. (2014). Verification and Validation Meet Planning and Scheduling. *International Journal on Software Tools for Technology Transfer*, 16(1):1–12.
- Bhatnagar, S., Mund, S., Scala, E., McCabe, K., McCluskey, T. L., and Vallati, M. (2022). On-the-Fly Knowledge Acquisition for Automated Planning Applications: Challenges and Lessons Learnt. In *International Conference on Agents and Artificial Intelligence*, pages 387–397.
- Bouchard, B., Giroux, S., and Bouzouane, A. (2006). A Smart Home Agent for Plan Recognition of Cognitively-impaired Patients. *Journal of Computers*, 1(5):53–62.
- Castillo, L., Fdez-Olivares, J., García-Pérez, O., and Palao, F. (2006). Efficiently Handling Temporal Knowledge in an HTN Planner. In *International Conference on Automated Planning and Scheduling*, pages 63–72.
- Chen, K., Zhang, D., Yao, L., Guo, B., Yu, Z., and Liu, Y. (2021). Deep Learning for Sensor-Based Human Activity Recognition: Overview, Challenges, and Opportunities. *ACM Computing Surveys*, 54(4).
- Chen, L., Nugent, C. D., Mulvenna, M., Finlay, D., Hong, X., and Poland, M. (2008). A Logical Framework for Behaviour Reasoning and Assistance in a Smart Home. *International Journal of Assistive Robotics and Mechatronics*, 9(4):20–34.
- Chen, L., Nugent, C. D., and Wang, H. (2011). A Knowledge-driven Approach to Activity Recognition in Smart Homes. *IEEE Transactions on Knowledge and Data Engineering*, 24(6):961–974.
- Curry, M. R. (1996). *The Work in the World: Geographical Practice and the Written Word*. University of Minnesota Press.
- Fernandez-Olivares, J. and Perez, R. (2020). Driver Activity Recognition by Means of Temporal HTN Planning. In *International Conference on Automated Planning and Scheduling*, pages 375–383.
- Georgievski, I. (2022). Office Activity Recognition Using HTN Planning. In *IEEE International Conference on Signal Image Technology & Internet-Based Systems*, pages 70–77.
- Georgievski, I. and Aiello, M. (2015). HTN Planning: Overview, Comparison, and Beyond. *Artificial Intelligence*, 222(0):124–156.
- Höller, D., Behnke, G., Bercher, P., and Biundo, S. (2018). Plan and Goal Recognition as HTN Planning. In *IEEE International Conference on Tools with Artificial Intelligence*, pages 466–473.
- Ihianle, I. K., Naeem, U., Islam, S., and Tawil, A.-R. (2018). A Hybrid Approach to Recognising Activities of Daily Living from Object Use in the Home Environment. *Informatics*, 5(1).
- Jobanputra, C., Bavishi, J., and Doshi, N. (2019). Human Activity Recognition: A Survey. *Procedia Computer Science*, 155:698–703.
- Klepeis, N. E., Nelson, W. C., Ott, W. R., Robinson, J. P., Tsang, A. M., Switzer, P., Behar, J. V., Hern, S. C., and Engelmann, W. H. (2001). The National Human Activity Pattern Survey (NHAPS): a resource for assessing exposure to environmental pollutants. *Journal of exposure science & environmental epidemiology*, 11(3):231–252.
- Levine, S. and Williams, B. (2014). Concurrent Plan Recognition and Execution for Human-Robot Teams. In *International Conference on Automated Planning and Scheduling*, pages 490–498.
- Liao, L., Fox, D., and Kautz, H. A. (2005). Location-Based Activity Recognition using Relational Markov Networks. In *International Joint Conference on Artificial Intelligence*, volume 5, pages 773–778.
- Matz, C. J., Stieb, D. M., and Brion, O. (2015). Urban-rural differences in daily time-activity patterns, occupational activity and housing characteristics. *Environmental Health*, 14:1–11.
- McDermott, D., Ghallab, M., Howe, A., Knoblock, C., Ram, A., Veloso, M., Weld, D., and Wilkins, D. (1998). PDDL - The planning domain definition language. Tech. Rep. CVC TR-98-003/DCS TR-1165, Yale Center for Computer Vision and Control.
- Minh Dang, L., Min, K., Wang, H., Jalil Piran, M., Hee Lee, C., and Moon, H. (2020). Sensor-Based and Vision-Based Human Activity Recognition: A Comprehensive Survey. *Pattern Recognition*, 108:107561.
- Naeem, U., Bigham, J., and Wang, J. (2007). Recognising activities of daily life using hierarchical plans. In *European Conference on Smart Sensing and Context*, pages 175–189.
- Nguyen, T. A. and Aiello, M. (2013). Energy intelligent buildings based on user activity: A survey. *Energy and buildings*, 56:244–257.
- Nguyen, T. A., Raspitzu, A., and Aiello, M. (2014). Ontology-Based Office Activity Recognition with Applications for Energy Savings. *Journal of Ambient Intelligence and Humanized Computing*, 5(5):667–681.
- Ramírez, M. and Geffner, H. (2009). Plan Recognition as Planning. In *International Joint Conference on Artificial Intelligence*, pages 1778–1783.
- Riboni, D., Pareschi, L., Radaelli, L., and Bettini, C. (2011). Is Ontology-Based Activity Recognition Really Effective? In *IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 427–431.
- Simpson, R., Schreckenghost, D., LoPresti, E. F., and Kirsch, N. (2006). Plans and Planning in Smart Homes. In *Designing Smart Homes: The Role of AI*, pages 71–84. Springer.
- Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A., and Katz, Y. (2007). Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics*, 5(2):51–53.
- Sohrabi, S., Riabov, A. V., and Udrea, O. (2016). Plan Recognition as Planning Revisited. In *International Joint Conference on Artificial Intelligence*, pages 3258–3264.

- Sukthankar, G., Goldman, R. P., Geib, C., Pynadath, D., and Bui, H. (2014). An Introduction to Plan, Activity, and Intent recognition. In *Plan, Activity, and Intent Recognition*, pages 1–22. Elsevier.
- Uschold, M. and Gruninger, M. (1996). Ontologies: Principles, methods and applications. *The knowledge engineering review*, 11(2):93–136.
- van der Aalst, W. M. P. (2011). *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer Publishing Company, Incorporated, 1st edition.
- Van-Horenbeke, F. A. and Peer, A. (2021). Activity, Plan, and Goal Recognition: A Review. *Frontiers in Robotics and AI*, 8.
- van Kasteren, T. L., Englebienne, G., and Kröse, B. J. (2011). Human Activity Recognition from Wireless Sensor Network Data: Benchmark and Software. In Chen, L., Nugent, C. D., Biswas, J., and Hoey, J., editors, *Activity Recognition in Pervasive Intelligent Environments*, pages 165–186. Atlantis Press.
- Wu, J., Osuntogun, A., Choudhury, T., Philipose, M., and Rehg, J. M. (2007). A Scalable Approach to Activity Recognition Based on Object Use. In *IEEE International Conference on Computer Vision*, pages 1–8.
- Ye, J., Stevenson, G., and Dobson, S. (2014). USMART: An Unsupervised Semantic Mining Activity Recognition Technique. *ACM Transactions on Interactive Intelligent Systems*, 4(4).

APPENDIX

The acronyms mentioned in this paper are listed in Table 3.

Table 3: List of acronyms.

| Acronym | Description |
|---------|---|
| AI | Artificial Intelligence |
| IoT | Internet of Things |
| PGR | Plan and Goal Recognition |
| HTN | Hierarchical Task Network |
| HPDL | Hierarchical Domain Definition Language |
| PDDL | Planning Domain Definition Language |
| NB | Naive Bayes |
| HMM | Hidden Markov Model |
| HSMM | Hidden Semi-Markov Model |
| CRF | Conditional Random Field |