Run-time Reasoning from Uncertain Observations with Subjective Logic in Multi-Agent Self-Adaptive Cyber-Physical Systems

Ana Petrovska, Malte Neuss
Technical University of Munich
Munich, Germany
petrovsk@in.tum.de, neuss@in.tum.de

Ilias Gerostathopoulos
Vrije Universiteit Amsterdam
Amsterdam, Netherlands
i.g.gerostathopoulos@vu.nl

Alexander Pretschner
Technical University of Munich
Munich, Germany
pretschn@in.tum.de

***Abstract*—Modern society has become increasingly reliant on the omnipresent cyber-physical systems (CPSs), making it paramount that the contemporary autonomous and decentralized CPSs (e. g., robots, drones and self-driving cars) act reliably despite their exposure to a variety of run-time uncertainties. The sources of uncertainties could be internal, i. e., originating from the systems themselves, or external—unpredictable environments. Self-adaptive CPSs (SACPSs) modify their behavior or structure at run-time in response to the uncertainties mentioned above. The adaptation relies on gained knowledge from the observations that the SACPSs make during their operation. As a result, to build the knowledge, the need for run-time observations aggregation and reasoning emerges since the observations made by decentralized CPSs are uncertain, partial, and potentially conflicting. In response, in this paper, we propose a novel methodological approach for deriving or aggregating knowledge from uncertain observations in SACPSs utilizing the Subjective Logic. The effectiveness of the proposed approach is demonstrated through extensive evaluation on an in-house, multi-agent system from the robotics domain.**

***Index Terms*—uncertainties, subjective logic, knowledge aggregation, self-adaptive systems, cyber-physical systems**

I. Introduction

Cyber-physical systems (CPSs) are software-intensive systems that control, communicate, and coordinate various processes in the physical and the digital worlds. Over the past decades, the boundary between the digital and physical has become increasingly blurry, accompanied by the rising prominence of CPSs. This process was further accelerated by the rapidly decreasing costs of embedded systems, which ultimately led to the omnipresence of CPSs as seen today. Nowadays, CPSs come in all shapes and sizes with applications as diverse as self-driving cars, smart homes, and entire robotic fleets, including autonomous robots and UAVs [6, 18].

As modern society increasingly relies on CPSs, these systems must act reliably even when faced with different run-time uncertainties. For instance, internal system uncertainties, i. e., different software or hardware faults and failures, as well as external uncertainties, e. g., uncertain and changing environments or *execution contexts* [21, 24, 28] in which the systems operate. Further, due to the limited range of their sensors, CPSs can only observe their context partially, which introduces another uncertainty dimension in the systems. A common approach to deal with uncertain and changing conditions at run-time while preserving the system's performance is to make CPSs self-adaptive.

In architecture-based self-adaptation [8, 12, 17, 40], a self-adaptive system—including self-adaptive CPS (SACPS)—is comprised of a *managed element* and an *adaptation logic*. The managed element can be either a software system or a CPS. If the SACPS consists of multiple autonomous and decentralized CPSs (i. e., managed elements), we refer to it as a Multi-Agent SACPS (MA-SACPS). On the other hand, the adaptation logic gives the managed element the ability to self-adapt and is commonly implemented using the MAPE-K feedback loop [17, 31, 32]. MAPE-K relies on four separate modules that *M*onitor, *A*nalyse, *P*lan and *E*xecute adaptations based on a shared *K*nowledge element. Furthermore, every system operates in a context. The context is the relevant part of the environment that can be observed, but not controlled. The knowledge (*K*) should keep models of both (1) the context in which the CPSs are operating, and (2) the CPSs themselves. These models should be continuously updated at run-time to reflect the dynamic changes in both the context and the CPSs; therefore, they need to be models at run-time (models@RT).

Models@RT [3, 4, 10, 38] are a promising approach for managing complexity at run-time, also considered as adaptation mechanisms for realizing self-adaptive systems. In past efforts, Floch et al. [10] and Bennaceur et al. [3] have identified the need for reasoning mechanisms as part of the models@RT, based on which the models are updated; however, no concrete solutions have been proposed. Concretely, the authors recognize as open research challenges: 1) the need for creating run-time models, and updating them in response to changes in the system and system's context; and 2) the need for reasoning (i. e., information or knowledge aggregation) based on which the models are updated.

Problem. MA-SACPSs are exposed to a variety of run-time uncertainties resulting in inaccurate and partial observations, which potentially lead to conflicting observations made by different CPSs. This can impact the overall performance of the adaptive system. Consequently, there is a need for reasoning

by effectively aggregating the different observations before updating *K*. This can be seen as an uncertainty resolution strategy applied at run-time.

Gaps. Although knowledge representation, aggregation and reasoning are essential for building MA-SACPSs [41], there is a scarcity of approaches for modeling *K* that allow capturing uncertain and conflicting observations from multiple, decentralized CPSs, to effectively aggregate the observations and eventually update the *K*. Additionally, in prior works, knowledge modeling has been typically treated as a domain-specific task [11], leading to ad-hoc solutions to its aggregation.

Solution. In this paper, we present a methodological approach for knowledge aggregation and reasoning in MA-SACPSs that is domain-independent and can deal with reasoning on uncertain, partial, and conflicting observations. Concretely, our approach uses Subjective Logic (SL) [14, 15] to update the knowledge in the adaptation logic at run-time by aggregating observations of the context made by each CPSs in a MA-SACPSs.

Contribution. Building on our previous work in which we introduced the idea of knowledge aggregation via Subjective Logic [25], in this paper we detail and extensively evaluate our fully developed approach. Succinctly, we make the following contributions:

(i) We present our fully developed SL-based approach for knowledge aggregation and reasoning in MA-SACPSs.
(ii) We provide an open-source implementation of a in-house ROS-based multi-robot system, which acts as a testbed for different scenarios involving uncertain and conflicting observations in the robotics domain.
(iii) We evaluate the effectiveness and sensitivity of the proposed SL-based approach through extensive controlled experiments.

Organization. In Section II we describe the class of use cases from the CPSs domain to which our approach is applicable, and a use case instantiation used as a running example throughout the rest of the paper. Section III summarizes the necessary background, before giving an overview of the approach in Section IV. In Section V we briefly describe the implementation of the approach, followed by the evaluation in Section VI. Section VII discusses related work, before concluding the paper in Section VIII.

II. Use Case

A. Class of Use Cases

Our proposed solution applies to any use case where multiple CPSs need to collaborate and coordinate processes assuming decentralized, partial and uncertain monitoring and centralized analysis. For example, mobile CPSs that autonomously traverse an environment to discover and attain tasks (e. g., robots or drones), or stationary agents that might have overlapping ranges of sensing (e. g., radio antennas). The CPSs could operate in two-dimensional (e. g., robots, self-driving cars) or three-dimensional environments (e. g., drones, UAVs). The dimension of the environment consequently defines how the context is modeled.

The complex and heterogeneous nature of CPSs often requires the SACPS to consist of multiple MAPE-K loops [25, 27, 41]. The use of multiple, interconnected MAPE-K loops leads to numerous challenges, one of which is the coordination of the control loops [39]. One possible solution is the use of design patterns [41] for distributed self-adaptive systems. In our paper, we assume that the considered CPSs are capable of independently monitoring the context in which they operate and simultaneously execute actions, i. e., performing the M and E phases of the MAPE-K on a *local level*. The decentralized phases of the MAPE-K inside the managed elements (the CPSs) are controlled by a single, centralized instance of planning (P), analysis (A), and knowledge (K) in the adaptation logic. Concretely, the MAPE-K loops inside the adaptation logic are structured according to the Master-Slave pattern (see Figure 1), previously proposed by Weyns [41]. To motivate the need for reasoning or knowledge aggregation, the centralized MAPE-K loop needs to reason on the uncertain, partial and potentially conflicting observations made by the decentralized monitoring, which, once aggregated, become knowledge.

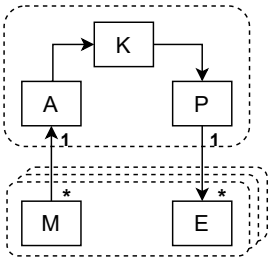

Fig. 1. Master-Slave pattern of the MAPE-K loop (updated from [41]).

B. Running example

The running example comprises of one or multiple robots operating in a room, in which dirt patches are continuously spawned with unknown location patterns and frequencies. The dirt patches represent cleaning tasks for the robots. Each robot is able to autonomously move to its destinations (i. e., the tasks' locations) while avoiding 1) static obstacles (e.g., walls, furniture, etc.), and 2) dynamic obstacles (e.g., other robots, humans) along its way. The robots fulfill their mission by discovering and cleaning the dirt i. e., completing the tasks without collision. They explore the room and detect new tasks in a distributed manner with a scanner, e. g., a LiDAR sensor. In addition to simply keeping the room clean, we also want to improve the performance or the quality of this cleaning process, e. g., cleaning the room in the shortest possible time.

However, the robots are subjected to external and internal uncertainties, manifested via the continuous appearance of tasks in the room—with unknown location patterns and different sensor uncertainties that cannot be anticipated during the design of the system. Due to the technical limitations of their sensors, the robots only monitor a limited range around them and can discover the newly appearing tasks only if they are within their range of observation. The partiality of the robots' observations introduces inefficiency to the overall system performance. Additionally, due to the sensor technology being imprecise and faulty, each robot might mistakenly sense a dirt task when there is none, and on the contrary, fail to sense an actual task. This potentially results in different robots holding

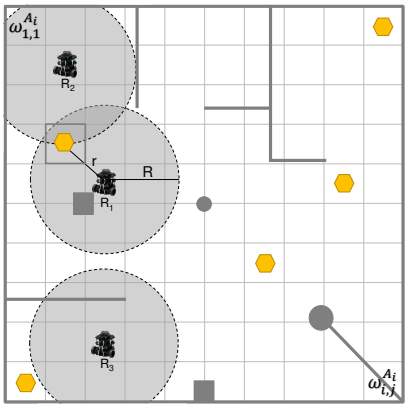

Fig. 2. Grid map and partial robots' observations, adopted from [25]

different opinions regarding the space they observe, which requires appropriate conflict resolution during the aggregation process.

The partial observations made by each robot are sent periodically to the centralized part of the adaptation logic, for instance, a `Cleaning Controller` (see Figure 3), in which the received observations are aggregated. The resulting aggregated knowledge, in terms of the presence of dirt tasks in the room, is the "best", most complete representation of the dynamic and uncertain context (i. e., the room) at a specific time. Optimizing the collective monitoring and analysis by obtaining the best representation of the state of the room, allows the performance of the full MAPE-K loop to also be improved and potentially optimized.

We model the context as a *grid map* with a size equal to the room's size (see Figure 2). A *context variable* ($w_x^{A_i}$) captures whether a dirt task occupies a specific cell of the grid map or not; therefore, there are as many context variables as cells in the map of the room. The figure also shows the robots' observation ranges and the tasks for the robots depicted in yellow hexagons.

III. Background

A. Run-time Uncertainties

In dynamic and adaptive systems, such as SACPSs, sources of uncertainty occur in one of the following three phases: requirements, design and run-time phase [30]. In this paper, we only consider uncertainty sources that occur in the run-time phase. We also classify the sources of run-time uncertainties as 1) *internal*—originating from the self-adaptive system itself, i. e., sensor failure, sensor imprecision, sensor noise and effectors, and 2) *external*, e. g., unpredictable environment.

In this work, we specifically focus on inaccuracy and inconsistency (stemming from sensor failure, imprecision, and sensor noise), and unpredictable environment. Ramirez et al. [30] define sensor failure as "a sensor inability to measure or report the value of a property", while inconsistency is defined as "two or more values of the same property that disagree with each other". Additionally, it is often infeasible 1) for the CPSs to observe the complete environment (i. e., context) in which they operate, due to the technical limitations of their sensors, e. g., limited sensor range, and 2) for the developers of the MA-SACPS to anticipate at system's design all the possible states of the context, which the systems will encounter during its operation. Consequently, the unpredictability of the environment will ultimately impart some partiality and uncertainty onto the MA-SACPS through its monitoring architecture.

B. Subjective Logic

When we assume an objective world, we can use binary logic to assert propositions about a state of the world to be either false or true. Nonetheless, it is practically impossible to determine with absolute certainty whether a given proposition is true or false. Through probability calculus, which takes argument probabilities in the range [0,1], we are able to reflect subjectivity by allowing propositions to be partially true. However, due to the lack of sufficient evidence, we are often unable to estimate probabilities with confidence. Furthermore, whenever the truth of a proposition is assessed, it is always done by an individual, and it cannot be considered to represent a general and objective belief. In order to reflect as faithfully as possible the perceived world in which we are immersed, a formalism to express degrees of uncertainty about beliefs is needed; said formalism shall also include belief ownership to reflect the subjective nature of beliefs [14, 15].

Subjective Logic (SL) [14, 15] is a framework for artificial reasoning, based on probabilistic logic and Dempster-Shafer theory of evidence [9, 33]. In recent years, SL has gained prominence because of its capability to deal with the degree of (un)certainty of propositions, inherently allowing 1) uncertainties representation as part of the fundamental building block of SL, called *Subjective Opinions* (see Section III-B1)), and 2) reasoning about the uncertainties through a process of *Belief Fusion* in which multiple Subjective Opinions are aggregated based on the selected fusion operator (see Section III-B2). For further explanation on SL please refer to [15, 25].

1) Subjective Opinions: A subjective opinion expresses a belief about a state variable X which takes its value from a domain $\mathbb{X}$. This domain represents all the possible states X can be. A binary domain $\mathbb{X} = \{x, \overline{x}\}$ is a type of domain that consists of two complementary states x and $\overline{x}$. In our running example, the state-space is a binary domain where the complementary states correspond to context variables (i. e., cells) in the grid map being occupied or unoccupied $\{x = \text{occupied}, \overline{x} = \text{unoccupied}\}$ by a task. If the domain of a subjective opinion is binomial, it is called a *binomial subjective opinion*. For the sake of brevity, the theoretical discussion of subjective logic is limited to binomial subjective opinions, as they are the only relevant in our case.

Definition 1 (Binomial Opinion [15])**.** Let $\mathbb{X} = \{x, \overline{x}\}$ be a binary domain with binomial random variable $X \in \mathbb{X}$. A binomial opinion about the truth of value X is the ordered quadruplet $w_x = (b_x, d_x, u_x, a_x)$, where the additivity requirement $b_x + d_x + u_x = 1$ is satisfied, and where the respective

parameters are defined as

b_x: belief mass in support of x being true (i.e. $X = x$),

d_x: disbelief mass in support of x being false (i.e. $X = \overline{x}$),

u_x: uncertainty mass representing the vacuity of evidence,

a_x: base rate, i. e., probability of x being true without any evidence.

Binomial opinions that have $u_x = 1$ and $u_x = 0$ are referred to as a vacuous and dogmatic opinions, respectively. The projected probability of a binomial opinion about value x is defined by: $P(x) = b_x + u_x a_x$.

2) Belief Fusion: Through a process of belief fusion, multiple opinions regarding the same proposition are merged or aggregated into a single, collective opinion. For instance, in our running example, multiple robots $R_1, R_2, \ldots, R_N$ issue opinions $w_x^{R_1}, w_x^{R_2}, \ldots, w_x^{A_N}$ for the same cell x, and these opinions need to be conveniently aggregated in a single opinion. Belief fusion can be realized using different operators [15, 37]: *averaging belief fusion*, *cumulative belief fusion*, *weighted belief fusion*, *consensus & compromise fusion*, and *belief constraint fusion operator*. Each of these fusion operators emphasizes different aspects when fusing multiple opinions. Subsequently, the choice of the operator depends on the aggregation's objective. In our use-case, the observations are made independently by multiple agents; thus, their opinions can be treated as independent pieces of evidence. Furthermore, compromises between said opinions are desired, such that the aggregated opinion is as accurate as possible. As a result, we choose the Cumulative Belief Fusion (CBF) and Consensus & Compromise Fusion (CCF) operators, although the implementation of the approach (further explained in Section V) supports all the other operators. In the following, we briefly summarize the two selected fusing operators.

Cumulative Belief Fusion (CBF) treats the individual opinions that are aggregated as independent pieces of evidence for the same proposition. This cumulatively increases the belief and/or disbelief value of the aggregated opinion while reducing its uncertainty. It is most suitable for combining multiple non-conflicting opinions. Namely, applying CBF to non-conflicting, uncertain opinions reduces the uncertainty of the resulting opinion.

Consensus & Compromise Fusion (CCF) maintains the shared belief masses between all the aggregated opinions. For conflicting opinions, a compromise is found, which has increased uncertainty. This operator is helpful for identifying a set of shared beliefs among all agents. The fused opinion would represent the set of causes that all opinions agree on.

The belief fusion aims to solve potential conflicting observations while increasing the observations' confidence according to the chosen fusion operator. To demonstrate this point, we provide a sample calculation for a pair of agreeing and disagreeing opinions w_x^1, w_x^2 on a same context variable in Table I.[1] We can see that when the opinions that are being aggregated agree, CBF creates an aggregated opinion whose belief increases that of the sources. CCF tries to find compromises between opinions, which is why their belief and disbelief values are in between the belief masses of ω_x^1 and ω_x^2. Additionally, with both fusion operators, we can observe improvement in the confidence of the aggregated observations, i. e., the uncertainty mass in the aggregated opinion decreases compared to the individual uncertainty masses of each robot's opinions. When the opinions are in conflict, CBF tries to maximize their belief masses. It is important to note that this process of aggregating conflicting opinions reduces the uncertainty of the aggregated opinion when using CBF, which can be undesirable. In contrast, CCF actively increases the uncertainty when opinions are conflicting. As a result, the resulting projected probability $P(x)$ is reduced when using CCF compared to CBF. This example showcases the different ways the two operators handle conflicts: whereas CBF essentially follows the strong opinion, CCF strives for a compromise. As we will see in our robotic use case, this difference manifests in the tradeoff between taking decisions fast (about whether a dirt is detected and should be cleaned) and taking decisions that are sound (since more than one robot agrees on the matter).

[1]For the mathematical definition of CBF and CCF see [16, 37].

TABLE I
AGGREGATING TWO OPINIONS w_x^1, w_x^2.

	Agreeing Opinions				Conflicting Opinions			
Param.	w_x^1	w_x^2	CBF	CCF	w_x^1	w_x^2	CBF	CCF
b_x	0.85	0.52	0.84	0.80	0.85	0.18	0.72	0.35
d_x	0.10	0.18	0.11	0.11	0.10	0.52	0.24	0.14
u_x	0.05	0.30	0.05	0.09	0.05	0.30	0.04	0.51
a_x	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
$P(x)$	0.88	0.28	0.86	0.84	0.88	0.28	0.74	0.61

IV. APPROACH

In our approach, the MA-SACPS consists of several MAPE-K loops structured according to the master-slave pattern [41] As shown in Figure 3, the Monitor and Executor elements of the MAPE-K loops are distributed among the decentralized agents, controlled by centralized Analyzer, Planner and Knowledge elements, each element explained in the following.

A. Monitor

Each managed element (i. e., CPS) in the MA-SACPS comprises a Monitor component. The Monitor observes the context in which the CPS operates, and additionally, it does (a minimal) monitoring of the CPS itself, e. g., it reports on the current position of the system in the context. Since the CPS cannot observe the entire context, it only provides partial observations that are subjected to uncertainties. Based on these observations, the Subjective Opinion Creator—contained within the Monitor—creates *binomial subjective opinions* (see Section III-B1) about the context variables that are within the agent's range. These binomial subjective opinions are then independently forwarded to the centralized Analyzer.

In our running example each robot contains a Monitor component. It periodically senses the position of the robot, and detects the presence of dirt tasks in the room. As explained

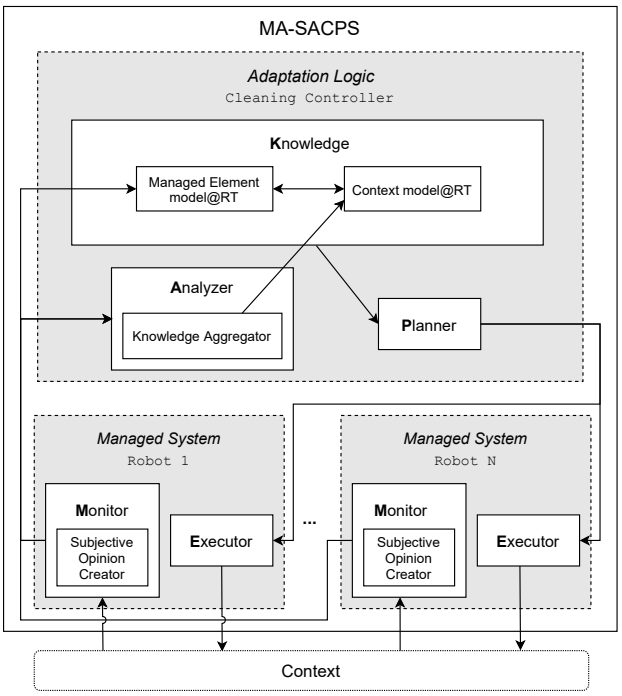

Fig. 3. High-level overview of the approach for reasoning in MA-SACPSs, adopted from [25].

above, each Monitor creates binomial subjective opinions about the context variables (is a cell occupied or not), for all the cells that are within the observation range of the robot at a specific point in time. The process of creating the subjective opinions is described in the following.

1) Subjective Opinion Creator: Every agent (A_i) creates a subjective opinion $w_x^{A_i}$ for every context variable x (i. e., grid cell) that is within its vision R. x can be either occupied or unoccupied by a task, and according to Definition 1 we define an opinion $w_x = (b_x, d_x, u_x, a_x)$ as following:

$$b_x = \begin{cases} 1 - u_X, \text{ for } x = occupied \\ 0.0, otherwise \end{cases} \qquad u_x = min\left(\frac{r}{R}, 0.99\right)$$

$$d_x = \begin{cases} 1 - u_X, \text{ for } x = unoccupied \\ 0.0, otherwise \end{cases} \qquad a_x = 0.5$$

where r is the distance between the agent and the cell and R is the agent's sensor range. This definition facilitates an uncertainty that increases linearly with the measurement distance up to an maximum value of 0.99. This value was chosen deliberately, since the observations at the edge of sensor range can be considered highly uncertain, but can still contribute towards the knowledge aggregation; hence, we assign an uncertainty value of 0.99 instead of a totally uncertain opinion (i. e., u_x = 1). Moreover, the polar nature between belief b_x and disbelief d_x was chosen to reflect the fact that agent can either detect a grid cell to be occupied or unoccupied by a task, and not a combination of both. The base rate a_x is always considered to be the default base rate for a binary domain, i. e., $a_x = 0.5$. Finally, no subjective opinions are issued for (i) cells which are occupied by static obstacles (e. g., walls), (ii) cells that lie outside of the detection range of the agent.

B. Analyzer

The Analyzer is a centralized component in which the reasoning takes place, with aim to solve any potential conflicting observations, while increasing the confidence of the observations according to the chosen subjective logic operator. The Analyzer contains the Knowledge Aggregator, which collects and fuses all the subjective opinions that are issued by the Subjective Opinion Creators in the decentralized agents. The aggregated opinions are used to update the run-time context model in K.

In our running example the Analyzer is a component in the `Cleaning Controller`. It combines the partial observations by aggregating the multiple binomial subjective opinions from different robot about the context variables that are within their sensing range. In particular, it is responsible for combining the opinions made by all the robots for all the cells that they are observing using the subjective logic operators. The process of aggregating multiple binomial subjective opinions is discussed next.

1) Knowledge Aggregator: This is a centralized node that aggregates the observations of the individual agents $(A_1, A_2, \dots A_N)$. The aggregated observations from all the agents are stored in the context model in the Knowledge, modelled as a grid map (see Section II-B). Upon system initialization, every context variable corresponding to each cell in the grid map is initialized with a vacuous subjective opinion: $w_x = \left(0, 0, 1, \frac{1}{2}\right)$. This initialization has the following two purposes: (i) a vacuous opinion will inform the analyzer that there is no previous knowledge about the context variables, and (ii) when the first knowledge aggregation is executed, the existing vacuous opinions do not influence the aggregation result.

The knowledge aggregation takes place every time the Subjective Opinion Creator of an agent publishes new subjective opinions for the context variables of the cells observed by the agent. Namely, the Knowledge Aggregator extracts the opinions that were previously stored for each context variable in the grid (w_x^g), and fuses them with the newly created opinion from each agent $(w_x^{A_i})$, resulting in a new aggregated opinion (w_x^{agg}) per context variable. The context variable x in the grid map is then updated based on aggregated opinion (w_x^{agg}), and this depicts is the overall process for updating the context model@RT (i. e., the grid map) in K. The process is executed with the same frequency for all the agents, and if multiple agents issue new opinions for the same context variable $(w_x^{A_1}, w_x^{A_2}, w_x^{A_3}, \dots)$ at the same time, then all of those opinion are together fused with the opinion from the grid (w_x^g).

The opinions are combined according to the following three schemes: CBF Scheme, CCF Scheme and Combination Scheme. The first two schemes are based on the CBF and CCF fusing operators, previously explained in Section III-B2:

$$\text{CBF Scheme}: \; w_x^{agg} = \text{CBF}(w_x^g, w_x^{A_i})$$
$$\text{CCF Scheme}: \; w_x^{agg} = \text{CCF}(w_x^g, w_x^{A_i})$$

However, preliminary testing revealed that these two SL operators cannot support long-term knowledge aggregation (see Section VI-B). In response, as part of this work, we have proposed a third scheme—Combination (Comb.) Scheme, defined as following:

$$w_x^{agg} = \begin{cases} \text{CCF}(w_x^g, w_x^{A_i}), & \text{if } u(w_x^g) < k \cap OT(w_x^g) \neq OT(w_x^{A_i}) \\ \text{CBF}(w_x^g, w_x^{A_i}), & \text{otherwise.} \end{cases}$$

where $u(w_x^g)$ is the uncertainty of context variable, k is a constant and $OT(w_x^{A_i})$ is the opinion type of the new opinion, defined as:

$$OT(w_x^{A_i}) = \begin{cases} \text{occupied}, & \text{if } b(w_x^{A_i}) \geq d(w_x^{A_i}) \\ \text{unoccupied}, & \text{if } b(w_x^{A_i}) < d(w_x^{A_i}), \end{cases}$$

where $b(w_x^{A_i})$ and $d(w_x^{A_i})$ are the belief and disbelief of the new opinion. In a nutshell, the combination scheme will always use the CBF operator, except when the new and grid opinions are conflicting and the uncertainty of the grid opinion is less than a given constant $k = 0.1$. We derived k numerically based on how many opinions need to be aggregated to change the opinion from unoccupied to occupied while still respecting the real-time capabilities of the knowledge aggregation (KA). Due to space limitations, the numerical calculation is not part of this paper.

C. Planner and Executor

The Planner is a centralized component in our approach that is responsible for selection of the adaptation actions or plans, which are later executed by the Executor components housed in every agent. The Planner relies on the aggregated knowledge, i. e., it uses the (updated) context model@RT to determine the actions for all the agents, in order for the MA-SACPS to adapt and accomplish the adaptation goals in the most efficient way. We do not prescribe how to perform the planning: any planning approach (e.g., rule-based, goal-oriented) can be used within our approach. Finally, each decentralized agent has an Executor component responsible for executing the actions previously determined by the Planner.

In our running example the Planner is part of the centralized `Cleaning Controller`, which receives as input the aggregated knowledge of the context. The aggregated knowledge, in terms of the presence of dirt tasks in the room, is the "best", most complete representation of the dynamic and uncertain context (i. e., the room) that is only partially observed at a specific time. Accordingly to it, the Planner assigns the discovered tasks to the robots. The outcome of the Planner is finally sent to the Executor in each robot. Each Executor contains an adaptive priority queue of the locations of the dirt tasks assigned to the specific robot. In our implementation, the queues of the robots are modified at run-time, based on the distance of the robot closest to the newly appeared task. Each robot picks the next task in its queue, and navigates autonomously to the corresponding cell.

V. Implementation

As part of this work, to investigate the usefulness of knowledge aggregation (KA) in MA-SACPSs, and to assess the correctness and the effectiveness of our proposed solution, we have implemented a testbed motivated by the running example from Section II-B. Since robotics is an intrinsically heterogeneous domain, setting up a *multi-robot* system is a challenge in itself. In response, in this paper, we provide a ROS-based, multi-robot system simulated in Gazebo [1, 19]. Namely, our implementation enables simulation of 1) a custom number of robots, and 2) the context in which they operate. Our running example concretely includes simulating a room with static (e. g., walls, corridors, furniture, etc.) and dynamic object i. e., the tasks that continuously appear for the robots in the room. The robots that we simulate are *TurtleBot 3 Burger*[2]. Furthermore, Gazebo relies on well-established physics engines, which enable simulations with high fidelity that closely resemble real-world robotics systems and their environments, with a physically correct representation of the robots, including their size and volume, frictions, as well as their sensors and actuators. Finally, the robots use Adaptive Monte Carlo Localization (AMCL) for localization and navigation.

Although we built a simulated robotics system, implementing realistic sensing (just how a real robot would sense its surroundings) was a prerequisite to constructing the subjective logic observations aggregation. As a result, the realistic sensing required more complex implementation to add different types of sensor uncertainties compared with, for example, "mocked" sensing in which modelling and adding the sensor noise would have been more simplistic. In the initial implementation of the robotic system, previously presented in [25], the sensor noise was sampled from a Gaussian distribution and only affected the *border* of the sensor beam. Although the former implementation did introduce some sensor noise in the system, it did not support the uncertainties from our running example (see Section II-B and Section III-A). Concretely, it was insufficient to support and prove the need for knowledge aggregation since it was incapable of creating conflicting observations within the sensor range. Consequently, we implemented a more sophisticated sensing model, in which the conflicting observations result from false positive (FP) tasks (tasks being observed by a robot that do not exist in reality) or false negative (FN) tasks (tasks not being observed when they in reality exist). Tasks that exist and the robots can observe are true positives (TP) in this setting.

The source code of the complete implementation and the installation instructions are available on the following link: https://github.com/tum-i4/Aggregatio. For the SL-based calculations, we used an open-source Java implementation[3].

[2]https://www.turtlebot.com/

[3]https://github.com/vs-uulm/subjective-logic-java

VI. Evaluation

In this section, we first explain the evaluation setup before discussing the obtained results.

We identified the following three research questions that guided our evaluation:

RQ1. How do the different SL aggregation schemes influence the KA in MA-SACPSs?

RQ2. Can SL-based KA in MA-SACPSs correct faulty measurements?

RQ3. How does the value of the threshold impact the SL-based KA in MA-SACPSs?

In RQ1, we investigate the *feasibility* of different SL aggregation schemes for KA in our approach. In RQ2, we evaluate the *effectiveness* of our approach's best aggregation scheme (derived from RQ1) on the core problem, i. e., reducing uncertainties and resolving conflicts introduced by inconsistent and faulty measurements. In RQ3, we investigate the *sensitivity* of our approach to the main parameter for decision making, i.e. the expected probability $P(x)$ of the context variables.

A. Experimental Setup

To answer the identified research questions, we have created three main experiments (see Table II), conducted based on 240 different simulation runs. Each experiment addressed one research question, and all the simulation runs lasted for 45 minutes in real-time. In all experiments, the dimension and the layout (e. g., corridors, walls, etc.) of the room in which the robots operate, along with the robots' initial positions and their sensor range, are held constant. The dimensions of the room and the sensor range of the robots were chosen in a way that at any given time the robots can only partially observe the room there are in. In particular, we chose a room size of $10 \times 10m$, since the robots can only detect obstacles up to a distance of $3.5m$.

Furthermore, we ran simulations under different settings, controlling: (i) the SL aggregation schemes for KA (part of the adaptation logic), (ii) the number and the properties of the robots (i. e., the managed elements), and (iii) the appearance of the tasks in the room (i. e., the context). With respect to the first point, we experimented with different *subjective logic operators* and *thresholds* values. The threshold is the projected probability $P(x)$ value for each detected task, necessary to be reached in order for the task to be promoted to a goal for the robots. With respect to the second point, we varied the *number of robots* and their sensing capabilities controlled by the *false positive (FP)* probability of each robot. The FP probability captures the percentage of faulty observations per robot. Finally, with respect to the environment, we varied the *rate of appearance of true positives (TP)* (in seconds) and the *location of appearance* of the tasks. A seed value controls the location of the appearance of the tasks, used to replicate the same distribution of tasks within the room in different experiments. To increase the validity of the results, we have run each experiment five times with different seed values. Using these variables, we explored the capabilities of SL-based KA and its impact on the performance of the MA-SACPS.

TABLE II
Design of the three experiments of the evaluation.

Parameter		Topic of Investigation RQ1	RQ2	RQ3
Experiment		1	2	3
KA		CBF, CCF, Comb.	[No, Comb., Comb.]	Comb.
Threshold		0.8	0.8	0.2, 0.4, 0.6, 0.8
No. of Robots		2	[1, 2, 5]	2
FP Prob.	R_1	0	0.1, 0.2, ... 0.9	[0.2, 0.5, 0.8]
	R_2	0	0.1	[0.2, 0.5, 0.8]
Rate of TP (s)		15, 60	60	60
Location (seed)		71, 72, ... 75	71, 72, ... 75	71, 72 ... 75

The rows in Table II represent the values of the variables used for a specific experiment. When multiple values are given per variable, then every possible combination is tested individually. For example, in Experiment 1 the CBF, CCF and Comb. operators are individually tested for all seed values. On the contrary, the square brackets indicate that these parameters are varied simultaneously. For example, in Experiment 2, when one robot is used, no knowledge aggregation is performed, whereas for two and five robots, the Comb. operator is used. Similarly, in Experiment 3 both robots R_1 and R_2 will have a FP probability of 0.2 in the first experiment, 0.5 in the next, etc. When using more then two robots, the additional robots will have the same FP probabilities as robot two (R_2). Finally, when Knowledge Aggregation is 'No', a task is classified as a goal after a single measurement, i. e., after it has been initially observed by one of the robots.

B. Experiment 1: Feasibility of different SL aggregation schemes

We initially aimed at the evaluation to use either CBF or CCF as operators for KA. Since CCF is desired when opinions are conflicting (see Section III-B2), the preliminary experiments quickly showed that it is hard to achieve extreme beliefs or disbeliefs when opinions are agreeing. This made it very difficult to surpass the required threshold needed for a task to be propagated as a goal and resulted in only a few tasks being completed. Subsequently, the focus shifted towards the CBF operator, which performed quite well. The cumulative aggregation method of CBF is ideal when opinions agree, and it is even capable of handling conflicting opinions to some degree. However, further analysis revealed that the task completion rate using CBF slowly deteriorates and eventually stops when running long simulations. We first explain this phenomenon via an analytical discussion and then show the results of the experiment.

1) Analytical Discussion: As discussed in Section III-B2, when fusing two opinions using CBF, the resulting opinion will have an uncertainty that is lower than that of the source opinions independent of whether the opinions agree or disagree. This property means that consecutive applications of CBF to fuse opinions issued by the robots will continuously decrease the uncertainty of the opinions of the context vari-

ables stored in the grid map. As the uncertainty decreases, the impact of a new opinion on the opinion in the grid map decreases as well, and after some time the grid opinions become too entrenched to change regardless of measurements. This demonstrates that this property is crucial when designing MA-SACPSs that need a support of a long-term KA.

The process that was described above can be demonstrated analytically with a few simple steps. First, consider three opinions: $w_x^{vac} = (0, 0, 1, 0.5)$, $w_x^{oc} = (0.7, 0, 0.3, 0.5)$, and $w_x^{unoc} = (0, 0.7, 0.3, 0.5)$ where the occupied and unoccupied opinions are in conflict with one another, and w_x^{agg} is the opinion of a context variable stored in the grid map. Initially, upon system initialization, $w_x^{agg} = w_x^{vac}$. Subsequently, the occupied opinion is aggregated with the base opinion, resulting in a new base opinion. This process is repeated 30 times. Afterward, the unoccupied opinion is aggregated with the base opinion for 30 times. These two steps of aggregating 30 occupied and 30 unoccupied opinions are repeated four times. Figure 4 plots the projected probability $P(x)$ of the w_x^{agg} opinion after each single aggregation (blue line). It can be seen from the figure that as the number of aggregated opinions increases, the effect of aggregating 30 occupied opinions diminishes significantly. The second peak occurs only 60 aggregations after the first peak and yet has a projected probability of about 40% less than the first peak after 30 aggregations. In our experiments, we consider a threshold of 80%, and opinions are issued every second. Thus, if a task is spawned at the first simulation minute (after 60 aggregations took place), even if the grid opinion has a projected probability of 50% and the robot measures the task for 30 seconds, it would still not reach the threshold and become a goal. As the figure shows, this gets even worse with time.

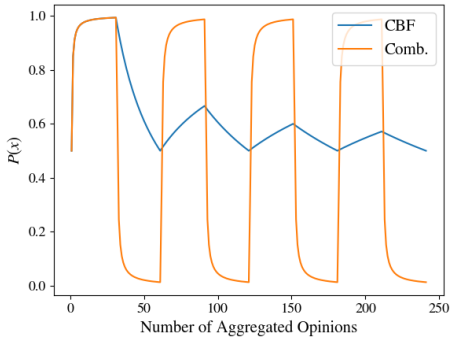

Fig. 4. The projected probability $P(x)$ shows that the effect of an individual opinion diminishes with time when using the CBF operator, and how the Comb. scheme overcomes this problem.

To address this issue, we have derived a new aggregation scheme (Comb.) using a combination of the CBF and CCF operators, which maintains the CBF operator's cumulative property but facilitates a faster switch in opinion when faced with contradicting opinions, and does not deteriorate with time. The proposed aggregation scheme was previously discussed in Section IV. Figure 4 shows that with the Comb.

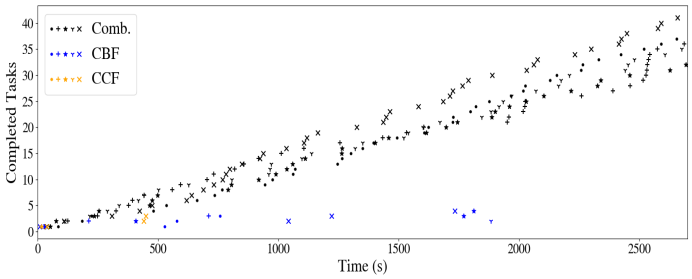

Fig. 5. Number of tasks completed using CBF, CCF and Comb. Schemes. The different symbols correspond to the different seed values.

scheme, the occupied and unoccupied opinions have equal impact on the opinion in the grid map and that their impact does not diminish over time.

2) Experimental Results: In this section, we experimentally assess RQ1. For that purpose, we set the FP probabilities to zero to demonstrate that the observed behaviors are solely related to the length of the simulation time and not to any uncertainty sources. The CBF, CCF, and Comb. aggregation schemes were each tested using the same five seeds with a threshold of 80% and a TP spawn interval of 15 and 60 seconds. The time at which the tasks have been completed was measured—Figure 5 shows the cumulative count of completed tasks over time. The different symbols correspond to the different seeds that have been tested at the spawning interval of 60 seconds. The results clearly show that independent of the location of the tasks, we can observe the same behavior: Comb. completes tasks at a relatively steady rate throughout the simulation. In contrast, the CBF operator only works well for the first few minutes before completely stopping. Lastly, the CCF operator performs the worst, which is expected as it is designed to find compromises and not aggregate observations cumulatively. The same behavior has been observed with a spawning interval of 15 seconds (plot not shown).

In summary, while addressing RQ1, we came up with the interesting finding that long-term KA is not feasible by using the original SL operators in isolation, based on which we proposed the new scheme—Comb. These experiments enabled us to prove some of the theoretical limitations of the CBF and CCF operators (Section III-B2) in an application from practice.

C. Experiment 2: Effectiveness of the KA

In this experiment, we investigate the effectiveness of the proposed combination scheme for KA by evaluating if the approach can correct the behavior of a robot with a faulty sensor. In this experiment, R_1 is considered to be the faulty robot subjected to a FP probability that ranges from 10% to 90% with a 10% step. Our baseline case is a single robot (R_1) that does not use KA. The other two cases consist of two and five robots and use the Comb. aggregation scheme for KA. All robots except R_1 are assumed to operate nominally with a FP probability of 10%. The measured metric is the number of FP and TP tasks that are completed by R_1.[4]

[4]By "completing" a FP task, we mean that the robot actually navigated to the place where the dirt was supposed to be.

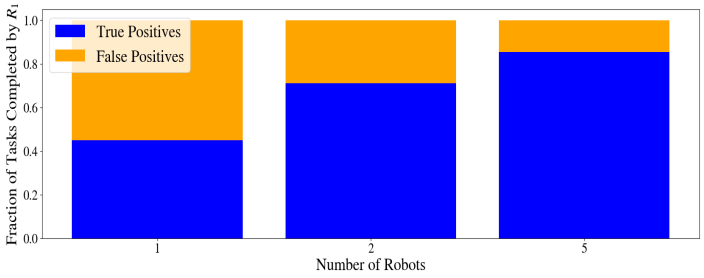

Fig. 6. The TP fraction of the tasks completed by robot 1 for a FP probability of 50%. The one robot case does not use KA whereas the combination aggregation scheme is used for two and five robot case. The results have been averaged over the five seeded runs.

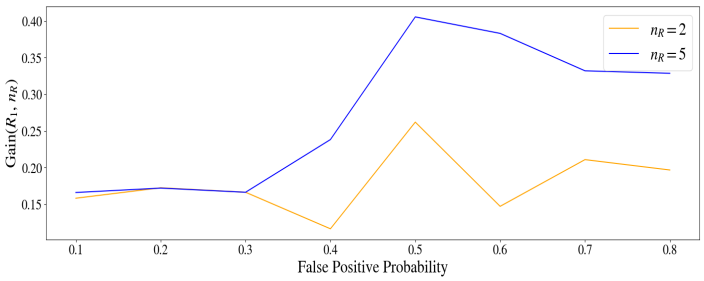

Fig. 7. The variation of the gain of robot 1 with FP probability when using two or five robots. The gain increases with FP probability, which demonstrates the effectiveness of the KA in correcting faulty observations.

Lets us first consider the tests when the FP probability of R_1 is 50%. Figure 6 plots the fraction of tasks completed by R_1 for the three different cases. As one would expect, about half of the completed tasks are TPs without KA, when only one robot is present. This fraction is significantly increased to about 70% when adding an additional robot and using KA. Adding even more robots further raises the TP fraction of tasks completed by R_1. This trend is expected since more robots make more independent observations, making it easier to correct faulty measurements. The bar charts further indicate that the gain in TP fraction per added robot is diminishing as the number of robots increases, which also makes sense as the extra vision gained per robot also decreases.

Instead of plotting a bar-chart for every FP probability case, we calculate the gain in TP fraction of robot R_i as follows:

$$\text{Gain}(R_i, n_R) = \frac{n_{TP}(R_i)}{n_{FP}(R_i)+n_{TP}(R_i)}\Big|_{n_R} - \frac{n_{TP}(R_i)}{n_{FP}(R_i)+n_{TP}(R_i)}\Big|_{1R}$$

where n_{TP} and n_{FP} is the number of TP and FP tasks completed by robot R_i and n_R is the number of robots in a multi-robot case. In particular, we calculate $\text{Gain}(R_1, n_R)$ for $n_R = 2$ and $n_R = 5$. The results in Figure 7 show that the gain in the TP fraction increases together with the FP probability for both cases—with two and five robots.

In summary, with respect to RQ2, we can conclude that SL-based KA enables the correction of faulty measurements made by a robot. In particular, even a single well-functioning robot can rectify the wrong measurements of another, faulty robot to a significantly extent. We have also observed that (i) increasing the number of well-functioning robots improves the accuracy of collective sensing; (ii) KA becomes increasingly effective as the faulty observations increase.

D. Experiment 3: Sensitivity analysis of the impact of the threshold value

The third experiment explores the impact of the threshold value on the behavior of the MA-SACPS. All simulations use two robots, and the FP probabilities are varied between 0.2, 0.5, and 0.8. Each of these variations is tested with four different thresholds ranging from 0.2 to 0.8 with a step of 0.2. As a metric, the number of completed TP and FP tasks are measured and averaged across the different seed values, based on which the average TP and FP fraction of completed tasks is calculated as follows:

$$\overline{TPF} = \frac{\overline{n}_{TP}}{\overline{n}_{FP} + \overline{n}_{TP}}, \quad \overline{FPF} = 1 - \overline{TPF}$$

where $\overline{n}_{TP}$ and $\overline{n}_{FP}$ are the average number of completed TP and FP tasks, and $N = n_{TP} + n_{FP}$ is the total number of tasks completed in a particular test. Moreover, the average number of completed tasks ($\overline{N}$) and the range in the number of completed tasks ($N_{max} - N_{min}$) are determined over five runs. The results from the experiment are depicted in Table III.

First, the table shows that increasing the threshold leads to an increased $\overline{TPF}$ for all the FP probabilities that have been tested. This indicates that the accuracy of the KA in MA-SACPS can be tuned using the threshold. This is expected, since the threshold dictates the minimum certainty the MA-SACPS must have to pursue the completion of a task. Nonetheless, the impact of the threshold on $\overline{TPF}$ is relatively small: changing it from 0.2 to 0.8 induces an increase of only 3.4% in $\overline{TPF}$ in the 0.2 probability case; 12.3% in the 0.5 probability case; and 3.8% in the 0.8 probability case. Furthermore, we can also observe another trend in the average number of completed tasks. Unsurprisingly, as the threshold increases, the number of completed tasks decreases.

In summary, with respect to RQ3, we conclude that the value of the threshold has an impact to SL-based KA, but a relative small one, and in any case a smaller impact than the number of collaborating robots. We also observed a clear trade-off between the accuracy of KA and the number of completed tasks.

E. Threads to validity

In order to have more realistic and comparable experiments, we kept the appearance rate of TP constant between runs since we wanted to explore how different FP probabilities and aggregation schemes respond to the same context (same ground truth of TP in the room). As a result, different FP probabilities result in a different number of tasks, i. e., higher FP probabilities generate more tasks in the room. This can be problematic for high FP probabilities as a high density of tasks restricts robots' observations, which severely limits the KA. These effects limit the validity of experiments at very high FP probabilities.

TABLE III
RESULTS FROM EXPERIMENT 3.

FP Prob.	Threshold	$\overline{TPF}$	$\overline{FPF}$	$\overline{N}$	$N_{max} - N_{min}$
0.2	0.2	0.769	0.231	53.6	7
	0.4	0.773	0.226	49.4	6
	0.6	0.794	0.206	45.6	5
	0.8	0.803	0.197	40.6	11
0.5	0.2	0.455	0.545	79.6	31
	0.4	0.485	0.515	41.8	64
	0.6	0.532	0.468	50.0	41
	0.8	0.578	0.422	25.6	57
0.8	0.2	0.205	0.795	93.6	68
	0.4	0.200	0.800	43.5	36
	0.6	0.219	0.781	47.4	86
	0.8	0.243	0.757	29.6	67

For example, in Experiment 2, the slight decreases at the end of Figure 7 are probably due to the aforementioned drop in the experiments' validity as the FP probability increases. In Experiment 3, we observed that the impact of the threshold is smaller than we have initially anticipated. Namely, at small FP probabilities (Table III), one would expect only a small improvement due to KA as faults rarely occur. On the contrary, at larger FP probabilities, one would expect a more considerable increase in the $\overline{TPF}$ than what is exhibited by the results. The same effect can be observed in the last column of Table III, which shows the difference between the maximum and the minimum number of tasks completed for the five different seed values. This clearly demonstrates that the results fluctuate a lot more as the FP probability increases. For example, for a threshold of 0.8 at a FP probability of 0.2 the range in N ($N_{max} - N_{min}$) is 27% of $\overline{N}$, whereas a FP probability of 0.8 with the same threshold has a range in N that is 226% of its $\overline{N}$. Subsequently, high FP probabilities require a lot more testing for accurate results. Furthermore, in Experiment 3, we concluded that there is a trade-off between the accuracy of KA and the number of tasks it completes. Based on this trade-off, one can conjecture that there is an optimal threshold that maximizes the number of completed TP tasks and varies with the FP and FN probabilities. In our experiment, the increase in $\overline{TPF}$ is too small to facilitate such a movement of the maximum; however, more extensive and statistically significant tests might prove this hypothesis.

VII. RELATED WORK

Knowledge/information aggregation. The need for knowledge aggregation arises in fields as varied as sensor fusion, expert system development and most prominently in multi-agent systems [26]. Across the field of multi-agent systems, knowledge is the information gained by agents' observations, often referred to as belief or belief base [26]. A belief is represented as propositional logic-based formalism [13, 26]. Grégoire and Konieczny in [13] present a survey about the approaches dealing with logic-based information fusion and discuss its relationship to multi-agent negotiation. Methods for belief merging are discussed in [20, 26, 34, 36]. These methods provide a general basis for knowledge aggregation; however, they have been only studied in the field of information systems. To the best of our knowledge, no knowledge aggregation technique has previously been used in the frame of SACPSs.

Sensor fusion. Munz and Dietmayer [22] proposes an approach to enhance the detection performance of a sensor fusion system measured in terms of detection rate versus false alarm rate. For that purpose, an algorithm is used which directly incorporates the DST-based sensory information. In [35], DST is also used to model the sources of uncertainty before applying the evidence fusion. However, the past approaches are mainly concerned with the aggregation of data in a single system, where the set of sensors are the multi-sources; rather than the aggregation of information or knowledge across multiple independent systems in a multi-agent system setup for self-adaptation purposes. Sensor fusion merges concrete sensor information, whereas, in our paper, we aggregate knowledge—which are two fundamentally different methods. Additionally, SL has not been used as a framework for fusing sensor information before, both generally and in the frame of self-adaptive systems.

Mitigating uncertainties in self-adaptive systems. As discussed in Section III-A, in this work, we focus on the following run-time uncertainties: sensor inconsistency, sensor failure, and unpredictable environment. Although the past literature proposes different mitigation strategies for sensor failure [5, 11] and unpredictable environment [2, 5, 7, 11, 23, 29], to the best of our knowledge, no solutions have previously tackled sensor inconsistency. Almost all existing uncertainties mitigation strategies are based upon one or multiple feedback loops, i. e., the different MAPE-K phases [5] that are not modified beyond their design-time specifications. Also, they are often designed for a specific application [23]. The most prominent framework for mitigating uncertainties is Rainbow [11], which focuses on architectural reusability. Rainbow focuses on isolating the feedback loop from the managed system as much as possible, enabling system-independent but knowledge-specific infrastructure. Furthermore, the framework does not support run-time modification of the adaptation logic, which disables the framework to deal with uncertain situations and changing conditions that were not anticipated during its design, as we do in our paper. In comparison to the past solutions, our approach presents a run-time uncertainly resolution strategy that tackles sensor inconsistency, as well as sensor failure and unpredictable environments.

VIII. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a domain-independent methodological approach for knowledge aggregation and reasoning of decentralized monitoring in MA-SACPS, which inherently produce partial, faulty and potentially conflicting context observations. The proposed approach allows capturing uncertainty at run-time on a local level, and effective reasoning and knowledge aggregation for global decision-making. The conducted experiments revealed that i) no single SL operator

is capable of providing long-term real-time KA capabilities, ii) the proposed SL-based KA approach is capable of correcting the observations of a faulty agent, and iii) there is a trade-off between the number of tasks that are completed and the accuracy of the agents, and the threshold can be used to tune the accuracy of KA to a desired level.

As future work, we want to find a threshold, if one exists, that maximizes the number of completed TP tasks, which varies with the FP probabilities. Since the FP probabilities result from the agents' interaction with the environment, the existence of a maximum would render the threshold as another parameter that can be optimized in real-time, ultimately posing another possibility for self-adaptation. Alternatively, one can also evaluate different and more advanced combinations of SL operators and different room sizes and layouts to improve the performance of KA even further, or investigate how KA with SL differs from reasoning with other non-monotonic logics.

REFERENCES

[1] Carlos E. Aguero, Nate Koenig, Ian Chen, Hugo Boyer, Steven Peters, John Hsu, Brian Gerkey, Steffi Paepcke, Jose L. Rivero, Justin Manzo, Eric Krotkov, and Gill Pratt. Inside the virtual robotics challenge. 12:494–506, 2015. ISSN 1545-5955. doi: 10.1109/TASE.2014.2368997.

[2] Luciano Baresi, Liliana Pasquale, and Paola Spoletini. Fuzzy goals for requirements-driven adaptation. In *2010 18th IEEE International Requirements Engineering Conference*, pages 125–134. IEEE, 2010.

[3] Amel Bennaceur, Robert France, Giordano Tamburrelli, Thomas Vogel, Pieter J Mosterman, Walter Cazzola, Fabio M Costa, Alfonso Pierantonio, Matthias Tichy, Mehmet Akşit, et al. Mechanisms for leveraging models at runtime in self-adaptive software. In *Models@ run. time*, pages 19–46. Springer, 2014.

[4] Gordon Blair, Nelly Bencomo, and Robert B France. Models@ run. time. *Computer*, 42(10):22–27, 2009.

[5] Yuriy Brun, Giovanna Di Marzo Serugendo, Cristina Gacek, Holger Giese, Holger Kienle, Marin Litoiu, Hausi Müller, Mauro Pezzè, and Mary Shaw. Engineering self-adaptive systems through feedback loops. In *Software engineering for self-adaptive systems*, pages 48–70. Springer, 2009.

[6] Hon Chen. Applications of cyber-physical system: A literature review. *Journal of Industrial Integration and Management*, 2017. doi: 10.1142/S2424862217500129.

[7] Betty HC Cheng, Pete Sawyer, Nelly Bencomo, and Jon Whittle. A goal-based modeling approach to develop requirements of an adaptive system with environmental uncertainty. In *International Conference on Model Driven Engineering Languages and Systems*, pages 468–483. Springer, 2009.

[8] Shang-Wen Cheng, David Garlan, and Bradley Schmerl. Architecture-based self-adaptation in the presence of multiple objectives. In *Proceedings of the 2006 international workshop on Self-adaptation and self-managing systems*, pages 2–8, 2006.

[9] A. P. DEMPSTER. A Generalization of Bayesian Inference Author (s): A . P . Dempster Source : Journal of the Royal Statistical Society . Series B (Methodological), Vol . 30 , No . 2 Published by : Wiley for the Royal Statistical Society Stable URL : http://www.jstor.o. 30 (2):205–247, 1968.

[10] Jacqueline Floch, Svein Hallsteinsen, Erlend Stav, Frank Eliassen, Ketil Lund, and Eli Gjorven. Using architecture models for runtime adaptability. *IEEE software*, 23(2): 62–70, 2006.

[11] David Garlan, S-W Cheng, A-C Huang, Bradley Schmerl, and Peter Steenkiste. Rainbow: Architecture-based self-adaptation with reusable infrastructure. *Computer*, 37(10):46–54, 2004.

[12] David Garlan, Bradley Schmerl, and Shang-Wen Cheng. Software architecture-based self-adaptation. In *Autonomic computing and networking*, pages 31–55. Springer, 2009.

[13] Eric Grégoire and Sébastien Konieczny. Logic-based approaches to information fusion. *Information Fusion*, 7(1):4–18, 2006. ISSN 1566-2535. doi: https://doi.org/10.1016/j.inffus.2005.08.001. URL http://www.sciencedirect.com/science/article/pii/S1566253505000771.

[14] Audun Jøsang. A Logic for Uncertain Probabilities. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9(3):271–311, 2001.

[15] Audun Jøsang. *Subjective Logic*. Springer International Publishing Switzerland, 2016. ISBN 978-3-319-42335-7. doi: 10.1007/978-3-319-42337-1. URL http://link.springer.com/10.1007/978-3-319-42337-1.

[16] Audun Jøsang, Dongxia Wang, and Jie Zhang. Multi-source fusion in subjective logic. *20th International Conference on Information Fusion, Fusion 2017 - Proceedings*, 2017. doi: 10.23919/ICIF.2017.8009820.

[17] Jeffrey O. Kephart and David M. Chess. The vision ofautonomic computing. *Computer 36*, pages 43–50, 2003. doi: https://doi.org/10.1046/j.1365-2745.2002.00730.x.

[18] Junsung Kim, Hyoseung Kim, Karthik Lakshmanan, and Ragunathan Rajkumar. Parallel scheduling for cyber-physical systems: Analysis and case study on a self-driving car. *2013 ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*, 2013.

[19] Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, pages 2149–2154. IEEE, 2004. ISBN 0-7803-8463-6. doi: 10.1109/IROS.2004.1389727.

[20] Sébastien Konieczny and Ramón Pino Pérez. Merging information under constraints: A logical framework. *Journal of Logic and Computation*, 12(5):773–808, 2002. ISSN 0955792X. doi: 10.1093/logcom/12.5.773.

[21] Sara Mahdavi-Hezavehi, Paris Avgeriou, and Danny

Weyns. A classification framework of uncertainty in architecture-based self-adaptive systems with multiple quality requirements. *Managing Trade-Offs in Adaptable Software Architectures*, pages 45–77, 2017.

[22] Michael Munz and Klaus Dietmayer. Using Dempster-Shafer-based modeling of object existence evidence in sensor fusion systems for advanced driver assistance systems. *IEEE Intelligent Vehicles Symposium, Proceedings*, (Iv):776–781, 2011. doi: 10.1109/IVS.2011.5940463.

[23] Peyman Oreizy, Michael M Gorlick, Richard N Taylor, Dennis Heimhigner, Gregory Johnson, Nenad Medvidovic, Alex Quilici, David S Rosenblum, and Alexander L Wolf. An architecture-based approach to self-adaptive software. *IEEE Intelligent Systems and Their Applications*, 14(3):54–62, 1999.

[24] Ana Petrovska and Alexander Pretschner. Learning approach for smart self-adaptive cyber-physical systems. In *2019 IEEE 4th International Workshops on Foundations and Applications of Self* Systems (FAS* W)*, pages 234–236. IEEE, 2019.

[25] Ana Petrovska, Sergio Quijano, Ilias Gerostathopoulos, and Alexander Pretschner. Knowledge aggregation with subjective logic in multi-agent self-adaptive cyber-physical systems. In Shinichi Honiden, Elisabetta Di Nitto, and Radu Calinescu, editors, *SEAMS '20: IEEE/ACM 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, Seoul, Republic of Korea, 29 June - 3 July, 2020*, pages 149–155. ACM, 2020. doi: 10.1145/3387939.3391600. URL https://doi.org/10.1145/3387939.3391600.

[26] Gabriella Pigozzi and Stephan Hartmann. Aggregation in multiagent systems and the problem of truth-tracking. *Proceedings of the International Conference on Autonomous Agents*, (May 2014):219–221, 2007. doi: 10.1145/1329125.1329245.

[27] Mariachiara Puviani, Giacomo Cabri, and Franco Zambonelli. A taxonomy of architectural patterns for self-adaptive systems. In *Proceedings of the International C* Conference on Computer Science and Software Engineering*, pages 77–85. ACM, 2013.

[28] Federico Quin, Thomas Bamelis, Singh Buttar Sarpreet, and Sam Michiels. Efficient analysis of large adaptation spaces in self-adaptive systems using machine learning. *2019 IEEE/ACM 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 1–12, 2019.

[29] Andres J Ramirez, Adam C Jensen, Betty HC Cheng, and David B Knoester. Automatically exploring how uncertainty impacts behavior of dynamically adaptive systems. In *2011 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011)*, pages 568–571. IEEE, 2011.

[30] Andres J Ramirez, Adam C Jensen, and Betty HC Cheng. A taxonomy of uncertainty for dynamically adaptive systems. In *2012 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 99–108. IEEE, 2012.

[31] De Lemos Rogerio, Holger Giese, Hausi A Mller, Mary Shaw, Jesper Andersson, Marin Litoiu, Bradley Schmerl, Gabriel Tamura, Norha M Villegas, and Thomas Vogel. Software engineering for self-adaptive systems: A second research roadmap. *Software Engineering for Self-Adaptive Systems III. Assurances*, pages 2–13, 2013.

[32] De Lemos Rogerio, David Garlan, Carlo Ghezzi, Holger Giese, Jesper Andersson, Marin Litoiu, Bradley Schmerl, Danny Weyns, Luciano Baresi, and Nelly Bencomo. Software engineering for self-adaptive systems: Research challenges in the provision of assurances. *Software Engineering for Self-Adaptive Systems III. Assurances*, pages 3–30, 2017.

[33] Glenn Shafer. A mathematical theory of evidence. *Princeton University Press*, 1976.

[34] Luciano H. Tamargo, Alejandro J. García, Marcelo A. Falappa, and Guillermo R. Simari. Modeling knowledge dynamics in multi-agent systems based on informants. *Knowledge Engineering Review*, 27(1):87–114, 2012. ISSN 02698889. doi: 10.1017/S0269888912000021.

[35] Yongchuan Tang, Deyun Zhou, Zichang He, and Shuai Xu. An improved belief entropybased uncertainty management approach for sensor data fusion. *International Journal of Distributed Sensor Networks*, 13(7), 2017. ISSN 15501477. doi: 10.1177/1550147717718497.

[36] Trong Hieu Tran, Ngoc Thanh Nguyen, and Quoc Bao Vo. Axiomatic characterization of belief merging by negotiation. *Multimedia Tools and Applications*, 65 (1):133–159, 2013. ISSN 13807501. doi: 10.1007/s11042-012-1136-7.

[37] Rens W. Van Der Heijden, Henning Kopp, and Frank Kargl. Multi-Source Fusion Operations in Subjective Logic. *2018 21st International Conference on Information Fusion, FUSION 2018*, pages 1990–1997, 2018. doi: 10.23919/ICIF.2018.8455615.

[38] Thomas Vogel, Andreas Seibel, and Holger Giese. The role of models and megamodels at runtime. In *International Conference on Model Driven Engineering Languages and Systems*, pages 224–238. Springer, 2010.

[39] Danny Weyns. Software engineering of self-adaptive systems. In *Handbook of Software Engineering*, pages 399–443. Springer, 2019.

[40] Danny Weyns and Tanvir Ahmad. Claims and evidence for architecture-based self-adaptation: a systematic literature review. In *European Conference on Software Architecture*, pages 249–265. Springer, 2013.

[41] Danny Weyns, Bradley Schmerl, Vincenzo Grassi, Sam Malek, Raffaela Mirandola, Christian Prehofer, Jochen Wuttke, Jesper Andersson, Holger Giese, and Karl M Göschka. On patterns for decentralized control in self-adaptive systems. In *Software Engineering for Self-Adaptive Systems II*, pages 76–107. Springer, 2013.