Abstract—This work presents BUM: a Bayesian User Model able to learn and estimate user characteristics in a distributed manner using heterogeneous information. A unified user representation is obtained from an inference process, receiving a set of independently estimated user characteristics from different sources. The independence of characteristic models enables the system to be modular, with each module estimating one characteristic. The proposed model is iterative, fusing new observations and measurements with previous information in a process regulated entropy. The system allows diverse implementations, such as the combination of multiple robots with a cloud infrastructure or distributed ambient sensors. This work aims to enable the system to perform on-line learning while interacting with users. The system is also able to obtain a correct user representation from heterogeneous information, even when some user characteristics cannot be computed. To demonstrate its functionality, the system is tested on two experimental datasets, obtained from simulated experiments and with real users. This technique advances the state of the art in the areas of AAL and user-adaptive systems, and in cloud-connected robots and Internet of Things (IoT), allowing for these heterogeneous and naturally-distributed teams of devices to better model their users, potentially achieving higher interaction autonomy.

Index Terms—Social Robots, Robot Perception, Multimodal Human-Robot Interaction, User Modeling, User Profiling, User-Adaptive Interaction, Internet of Things

I. INTRODUCTION

User-adaptive systems are able to automatically adapt to their user’s characteristics, effectively compensating for their preferences [10], skill levels [9] and impairments [11][16]. These systems have been widely explored in the field of Human-Computer Interaction (HCI), outperforming non-adaptive systems in multiple metrics and scenarios [4]. These systems enjoy widespread attention in HCI [18][16], and have reached a state of maturity and commercial use e.g. in systems such as Netflix or Google Search. There exist several surveys on the usage of these techniques in several fields, such as social media [1], the web [4][7] and intrusion detection [19].

User-adaptive systems rely on a user model [16][12], which acts as a repository of information about a user to be used for adapting the system’s behavior. This model can be as simple as a single attribute, e.g. the user’s proficiency in using the system, or as complex as the user’s personality [24].

In this work we present a Bayesian User Model (BUM) for determining a unified user representation, by learning user characteristics from heterogeneous information gathered by distributed devices. A fusion mechanism is proposed, using independent data-gathering devices as weak learners (and estimators) to continuously increase the global knowledge of the system after each individual estimation step. Although these estimators can be interchangeable, in this work we use Bayesian estimators [6] to infer user characteristics due to their ability to integrate updated likelihood models and recursively compute new estimates, enabling the testing of the fusion mechanism in such conditions. Unified user representations (encoded as characteristics vectors) which result from this estimation process are clustered, yielding user profile models. As a consequence of the clustering process, the profile models are resilient to missing user characteristics. In cases where a user characteristic is not available, it can be derived from the user profile model, endowing the system with robustness.

This work is structured as follows. The remainder of Section I introduces the related work, our contribution to

1 Gonçalo Martins and Luís Santos are with the Institute of Systems and Robotics, University of Coimbra, 3030-290 Coimbra, Portugal, gmartins@isr.uc.pt, luis@isr.uc.pt.
2 Jorge Dias is with the Khalifa University of Science, Technology & Research (KUSTAR), Abu Dhabi, UAE, jorge.dias@kustar.ac.ae
the field and the main goals of the work. Section II presents the BUM model, split into the estimation and fusion sub-processes. Section III presents the user profile component of the system. Section IV presents the experimental goals, methodology and results obtained for our proof-of-concept, wherein we perform a mix of simulated and real interactions with various users. Section V presents our discussion on the results obtained, validating our claims. Lastly, Section VI presents our conclusions.

A. Related Work

User Modelling systems can be split into two main categories: those meant to support Human-Robot Interaction (HRI) and those applied to Human-Computer Interaction (HCI). These systems have been under development in the HCI community for a very long time [16], and are a relatively infant field in HRI. Information about the user can be obtained in two main ways: by asking the user for input (explicit), without user intervention (implicit) [4]. These systems are interlinked by the concepts of interoperability, shell systems and user model servers, which we describe below.

Model interoperability consists of the degree to which different user modeling systems are able to integrate their information, aiming for a holistic representation of the user [3]. Interoperable systems become very interesting when the adaptive system is decentralized or composed of heterogeneous devices, since high interoperability allows for a more efficient use of information.

Shell systems [12] consist of “empty” user modeling systems, that have to be populated with domain knowledge, such as the model’s structure and update rules. These systems are very versatile, with the very same system being able to be used in a variety of applications.

User modeling servers [7][3] consist of services operating on the client-server model, wherein a user model is constructed by a server that receives information from one or more clients. This model can then be requested by clients, which can use them to adapt in the context of their specific application. User modeling servers avoid the issue of interoperability by centralizing the construction of the user model. This implies the usual disadvantages of centralized systems, such as the increased local workload and the possibility of catastrophic centralized failure.

Despite its relative infancy, the field of user modeling in robotics has already seen some developments in user modelling. Some works rely on static user models, which are given to the system before starting its operation. One for of these can be seen in [5], where generalized user profiles, dubbed Personas, are used to adapt to users that are expected to display similar characteristics. Similarly, the authors of [8] present a companion robot that is adapted to the user by performing an initial questionnaire for configuration. Systems employing static user models can be very effective at adaptation, and can count on advantages such as relative simplicity and lower processing power needs. However, these systems are unable to deal with the natural changes that users undergo, and are also unable to correct their own perspective of the user’s characteristics.

There are also works that rely on implicit data gathering and dynamic user models. For instance, the authors of [17] present a robot able to measure how easily the user changes their task plan to accommodate the robot’s own plan, showing increased efficiency in executing the task.

There are several ways in which these user models can be used for adaptation using robots. [10] adapts to the user based on implicit information, by making use of a history of interactions to adapt future actions, similarly to [14], showing advantages when compared to non-adaptive systems. [13] presents a robot tutoring system, akin to those found in HCI, which relies on the user’s preferences and skill level. Lastly, [21] presents a robot guide that learns the user’s preferences regarding its actions, ultimately achieving full autonomy, being able to act without any command.

Some works take their adaptive facilities to a level closer to the user, by adapting to their personality traits. Works such as [23][24][20] employ a model of the user’s personality, and use this information to synthesize a personality of their own, aiming to achieve a higher level compatibility with the user. These works stem from research in the field of Personality Computing [25], wherein systems are produced with the goal of estimating the user’s personality traits from multiple sources of information, a small minority of which including robots.

B. Key Goals and Contributions

We could not find any systems able to seamlessly combine information gathered from heterogeneous sources, and able to be implemented in multiple topologies, including combinations of Social Robots with external sensor networks. We aim to advance the state of the art by proposing a distributed estimation framework that is able to learn by fusing information from heterogeneous sources. Furthermore, since User Modelling techniques have not fully transferred into HRI, we also advance the state if the art by proposing a technique that can be seamlessly used with robots.

Specifically, we propose a system that:

1) Performs on-line learning of user characteristics, fusing heterogeneous data from distributed sources;
2) Estimates and groups characteristics into profile models at runtime;
3) Is tolerant and robust to erroneous inputs, i.e. missing evidence and/or characteristics.

These system features constitute the claims we aim to support with our experiments and discussion.

II. ON-LINE LEARNING OF USER CHARACTERISTICS

A. Estimation of User Characteristics

The main goal of the model is to infer a unified user representation (vector of the user’s characteristics), \( \mathbf{C} \in \mathbb{R}^n \), where \( n \) is the number of user characteristics under study. This process, called Estimation, is illustrated in Fig. 2. The process takes as input a vector of evidence \( \mathbf{E} \in \mathbb{R}^m \), where
\( n \) is the number of evidence variables, and the user’s identity \( Id \in \mathbb{N} \). The main output of the system is the distribution
\[
P(C|E, Id) \propto P(C)P(E, Id|C),
\]
which encodes the \( Id \) user’s characteristics revealed by the evidence. By using the user’s identity as an additional evidence variable, the model is able to represent a population of users while still allowing for personalization. The likelihood \( P(E, Id|C) \) is iteratively computed via our fusion mechanism, detailed in Section II-B (Eq. 8).

This estimation process is split into modules, with each module inferring one of the characteristics in the \( C \) vector, assuming that all characteristics are statistically independent from one another. For each module, a Bayesian Program [6] is used to infer the characteristic, solving Eq. 1. Once estimated, the characteristics can be used by the underlying system, e.g. for user-adaptive interaction or to generate new labels for learning. Furthermore, for the purposes of implementation, we also assume that all evidence variables are independent, allowing us to decompose Eq. 1 into
\[
P(C|E, Id) \propto P(C) \prod_{i \in E} P(E_i, Id|C).
\]

A characteristics-space representation of all users is obtained by performing maximum a posteriori estimation for all characteristics of a user, producing a point \( U_u \) in characteristics space for each user, in a process illustrated in Fig. 2. Each point in defined as
\[
U_u = [C_1, C_2, \ldots, C_n],
\]
where \( n \) is the number of characteristics being inferred, and \( u \) the identity of the user, and \( U \) is the matrix containing all users. In this formulation, different users can show the same characteristic for different inputs and, conversely, different users can show different characteristics for the same inputs. Furthermore, there is no limitation in how many users can display exactly the same characteristics, which can lead to superimposed users in the characteristics-space representation. There is no limit to how many devices can contribute to the global knowledge of the population, through the fusion mechanism described below. No assumptions are made as to the source of the data, as long as it can be transformed to fit the structure of the evidence vector.

B. Information Fusion and Learning

As illustrated in Fig. 3, the system uses the result of the estimation step to update the common representation of the users in the system, and is able to combine both soft and hard labels.

For each estimation of the system, a tuple is generated:
\[
T_i = (L_i, E, h_i),
\]
where \( L_i \in \mathbb{N} \) is the label obtained for characteristic \( C_i \), via Maximum a Posteriori estimation:
\[
L_i = \arg\max_X P(C_i|E, Id).
\]
\( h_i \) is the entropy of the distribution \( P(C_i) \), which we approximate by the posterior obtained from the estimation step, \( P(C_i|E, Id) \):
\[
h_i = H(P(C_i)) \approx H(P(C_i|E, Id)).
\]
\( H \) is the entropy function, as defined in [22]:
\[
H(X) = - \int_X P(x) \log P(x) \, dx
\]
Each user characteristic model is stored in a likelihood of the form \( P(E, Id|C) \), which represents the aggregate knowledge of the whole system, and is used for Estimation, as seen in Section II-A. This distribution is iteratively computed as a Gaussian kernel\(^1\) by performing
\[
P(E, Id|C_i = L_i)_{k+1} = \frac{1}{\psi} (P(E, Id|C_i = L_i)_k + D)
\]
where \( D \) is a learning factor, function of \( T_i \), defined according to the label received and \( \psi \) is a normalization factor, ensuring that the resulting probability distribution is valid. This sub-process is performed every time a new tuple \( T_i \) is received by the system.

\(^1\)For the purposes of this research, Gaussian distributions are used. Other distribution types can be used.
By employing this fusion mechanism, and in addition to the benefits presented in Section I-B, the system is able to learn faster through the parallelization of the learning procedure. Additionally, by employing individual devices as soft learners, the system can operate without the express need for supervision. This is achieved by the fact that the distributed system can integrate information from various devices asynchronously, thus allowing for the integration of distributed information about each user without the limitation of one-on-one interaction.

An ensemble of Bayesian estimators is applied due to their ability to integrate updated likelihood models and recursively compute new estimations, transparently integrating the fusion mechanism and whose properties are deemed appropriate for validating it. Different estimator types could be used, as long as the structure of their output is adapted to cope with the entropy they contain:

\[ D = P(E, I| C = c)_{observed} = \mathcal{N}(\mu, \Sigma) \]  

(9)

where \( \mu \) is defined according to the evidence received:

\[ \mu = E \]  

(10)

and \( \Sigma \) is a covariance matrix where each diagonal element is defined by entropy:

\[ \Sigma_{i,i} = F(h_i), \]  

(11)

which is calculated in Eq. 6.

1) Soft Labels: Soft labels, i.e. labels generated by the system’s distributed classifiers, are fused according to the entropy they contain:

\[ D = P(E, I| C = c)_{observed} = \mathcal{N}(\mu, \Sigma) \]  

where \( \mu \) is defined according to the evidence received:

\[ \mu = E \]  

(10)

and \( \Sigma \) is a covariance matrix where each diagonal element is defined by entropy:

\[ \Sigma_{i,i} = F(h_i), \]  

(11)

which is calculated in Eq. 6.

2) Hard Labels: Hard labels, i.e. labels received from an external labeler, which encode ground-truth data, are also fused through the Gaussian kernel principle:

\[ D = \mathcal{N}(\mu, \Sigma) \]  

(12)

where \( \mu \) is defined as before, according to Eq. 10. \( \sigma \) is defined according to Eq. 11, where \( h_i \) is set to a very low value, conveying the high certainty of information received from the external labeler.

These can be obtained, for instance, by directly asking the user a question, in an approach resembling active learning [2]. Thus, the system is unable to deal with users who purposefully mislead the system, a problem which we consider out of the scope of this work.

III. OBTAINING USER PROFILES

User profiles are obtained from the characteristics-space representation of the users, as illustrated in Fig. 2. The Expectation-Maximization [6] algorithm is used for clustering the users, producing a Gaussian mixture on characteristics space. This results in a set of n-dimensional Gaussian distributions, which are the main output of this sub-process:

\[ \theta = (\mu, \Sigma) = EM(U), \]  

(13)

where \( \mu \) contains the means of the clusters, and \( \Sigma \) the respective covariance matrices. \( EM \) denotes the Expectation-Maximization algorithm, where \( Z \) is a variable indicating cluster membership of \( U \), and \( \theta \) the cluster Gaussian parameters.

The algorithm consists of alternating the expectation step:

\[ Q(\theta(\theta^{(t)})) = E_{Z|U, \theta^{(t)}} [\log L(\theta; U, Z)] \]  

(14)

and the maximization step:

\[ \theta^{(t+1)} = \arg \max_{\theta} Q(\theta|\theta^{(t)}) \]  

(15)

until a convergence condition, defined by the difference of the model obtained in subsequent iterations, is reached.

Each of the resulting clusters represents a user profile model. A profile represents a common “type” of user found by the system when combining its knowledge on their characteristics. Thus, the system operates on the whole population, revealing inter-identity relationships among users.

As with any unsupervised learning technique, these profiles present two key challenges: the determination of the number of clusters to optimize and the attribution of semantic value to the clusters. Information-theoretic techniques can be used to determine the number of clusters that best fits the data [6]. In our case, we opted for the exploitation of a priori knowledge for the definition of the number of profiles, as discussed in Section IV. Attributing semantic value to the profiles can be achieved by superimposing a semantic sectioning of the characteristics space. For instance, if 3D space is split in 8 octants, each octant can be attributed to a certain semantic value, and each cluster “named” according to the octant they fall in.
A. User Profile Classification

If one or more of user’s characteristics can no longer be estimated due to an input failure, the missing characteristics can be derived from the estimated cluster (the cluster that minimizes the distance metric to the user unified representation), allowing the system to gracefully degrade and continue to operate, as illustrated in Fig. 4.

Assuming a user given by a vector with incomplete characteristics:

\[ U_{inc} = [C_0, \ldots, C_{k-1}, C_{k+1}, \ldots, C_n], \]  \hspace{1cm} (16)

i.e. as defined in Eq. 3 except for the missing \( k \)-th component, \( C_k \).

The distance from this user to each of the clusters is obtained through

\[ d_i(C, \Sigma) = \sqrt{\sum_{C_j \in U_{inc}} (C_j - \Sigma_j)^2}, \]  \hspace{1cm} (17)

where \( \Sigma_j \) is the element of cluster \( i \) corresponding to the characteristic \( C_j \) of the user. \( d_i \) corresponds to the Euclidean distance over the projections of clusters in \( n-1 \)-dimensional space, i.e. excluding the dimension that could not be estimated due to a fault. The closest cluster to the user is selected by minimizing the distance:

\[ \Sigma^* = \arg\min_{\Sigma} d_i(C, \Sigma) \]  \hspace{1cm} (18)

Once the user is matched to a cluster, each of the missing characteristics would then be obtained from the means of the clusters by performing

\[ C_j = \Sigma^*_j \]  \hspace{1cm} (19)

for every missing characteristic \( C_i \).

IV. Experiments

A. Goals and Metrics

The goal of our experiments is to evaluate the system’s performance and support the claims presented in Section I-B. In order to evaluate the system’s performance, we will employ the following performance metrics:

- Estimation error, \( \epsilon \);
- K-L Divergence, \( D_{KL} \).

The estimation error \( \epsilon \) is defined as:

\[ \epsilon = |c_{est} - c_{real}| \]  \hspace{1cm} (20)

where \( c_{est} \) and \( c_{real} \) represent the characteristic estimation and the ground-truth value, respectively. This measurement provides the accuracy of the characteristic model, when compared to the ground truth. It is also an indicative measure of how much new information about the user the system was able to integrate.

The Kullback–Leibler divergence is defined for continuous distributions as

\[ D_{KL}(P||Q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} \, dx, \]  \hspace{1cm} (21)

being equal to

\[ D_{KL} = E_p[\log(p(x)/q(x))], \]  \hspace{1cm} (22)

which we compute numerically with a Monte Carlo [6] approximation based on \( 10^3 \) samples of the Gaussian mixture. This measurement allows us to evaluate the user profile models estimated by the system, comparing them with the reference population.

Experiments are divided into two categories:

- Experiments with synthetic data (Section IV-B);
- Experiments with real users (Section IV-C).
B. Experiments with Synthetic Data

We developed an implementation involving a simulated team of social robots and cloud infrastructure\(^2\). The fusion sub-process takes place on the cloud, with the remaining sub-processes taking place on each robot. Label vectors \(T\) are fused with previous information on the cloud, resulting in refined likelihood distributions which are propagated to the robots, as described in Section II.

A user population of 100 users was randomly generated from a set of 4 profiles. The system operated for 3000 iterations. Each user was sampled from a random profile and uniformly-distributed random noise was added to the sample. We consider 3 characteristics \(C_i\), 10 evidence variables \(E_j\), representing different types of information. Each \(E_j\) has 2 states. Each \(C_i\) is dependent on a different set of 5 randomly selected \(E_j\), resulting in the joint distribution \(P(C_i|E_j=1:5, Id)\). No assumptions are made as to the sensory input of \(E_j\), only that in this case, for simplicity, it is represented by two states. The system was operated iteratively, as illustrated in Alg. 1.

Randomly selected evidence was sampled from the population and fed to the Estimation process, generating soft labels for fusion. For the first iteration of each combination of input variables, the label generator was given the corresponding ground truth value. Thus we could allow the system to start with uniform likelihoods, converging by iterating on the seed model formed by the initial hard evidence provided.

To test the robustness of the model to missing evidence/user characteristics, we emulate a failure in a device that estimates a specific characteristic by deactivating one of the system’s modules after a number of iterations. Then, a number of new users are added to the population, to demonstrate the system’s ability to tolerate the fault by allowing it to operate with one of its modules disabled.

C. Experiments with Real Data

The purpose of the experiments with real data is to execute a feasible subset of the simulation experiments, demonstrating the system’s applicability to a realistic scenario. This allows for proof-of-concept interactions to occur in a feasible timespan, while allowing us to demonstrate that the system exhibits results indicating a strong similarity with the simulation experiments. It is not the purpose of these experiments to evaluate the different characteristics per se, but rather to demonstrate that user characteristics converge by fusing data from distributed weak classifiers providing asynchronous evidence. We employ 3 characteristic models:

- User’s preferred speech volume (\(C_1\)).
- User’s preferred distance to the robot (\(C_2\)).
- User’s talkativeness (\(C_3\)).

As person regularly speaks on average 180 words per minute [26], we derive a maximum of 5 words per second as the maximum speech speed, which we use to formulate a seed model for the estimation of \(C_3\).

Each characteristic had a source of information (evidence):

- \(C_1\): The robot asks the user if it is talking too loudly;
- \(C_2\): The robot asks the user if it is too far from them;
- \(C_3\): The user’s mean speech speed for the past 5 replies.

Characteristics are estimated, according to the formulation of Section II-A, using a Bayesian program (Fig. 6). Our experiments use a predetermined the number of user profile models (clusters). A social robot, developed in the GrowMeUp project [15], was employed as the main interactive device for these tests. A user-adaptive decision-making technique was developed for this purpose, described in Alg. 2.

A total of 7 participants participated in the study by interacting with the robot, with ages ranging between 20 and 35 years. The main scenario, from the perspective of the robot, consists of the following: The robot engages the user in casual conversation on various topics. While taking turns speaking, the robot sometimes asks the user if they are happy with the current volume and speaking distance. The user’s answers are used to adapt the robot’s operation to the user. Each trial in the session lasted for 5 to 10 minutes, representing ~20 interaction cycles between the robot and user. The procedure is as follows:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCDS.2018.2878451, IEEE Transactions on Cognitive and Developmental Systems

Input: Q: Normal questions to be asked,
V: Questions about volume,
D: Questions about distance.

Output: T: Tuples used for training the BUM system.
E: Evidence for the BUM system.

while Q ≠ {} do
    // Remove random question from Q
    q = Q.pop();
    // Ask the question via robot.
    reply, time = ask(q);
    // Generate evidence from response.
    e = reply.n_words() / time;
    send_evidence(e);
    // Ask about volume
    if not asked_vol and rand() < 0.25 then
        asked_vol = True;
        q_v = V.pop();
        reply, time = ask(q);
        adjust_volume(reply);
        send_tuple(reply);
    end
    // Ask about distance
    if not asked_dist and rand() < 0.25 then
        asked_dist = True;
        q_d = D.pop();
        reply, time = ask(q);
        adjust_distance(reply);
        send_tuple(reply);
    end
end

Algorithm 2: Algorithm implemented on the user-adaptive decision-making module used for data gathering. The system asks questions from a pool, also asking about the distance or volume that it is using to communicate, using the replies to generate hard evidence.

1) Users were introduced to the research being performed and to what data would be gathered;
2) Users were asked about each individual characteristic;
3) The users interacted freely with the system;
4) The session was concluded.

The sessions took place in a laboratory setting, illustrated in Fig 5. The sessions with users resulted in a dataset that, given the flexibility of the system, is analyzed and processed in several ways, allowing for various tests to be performed. Namely, the dataset was used for performing tests with the data from a single user, as well as for the full population, by selectively playing back and interleaving the information obtained during the tests. The baseline measurements performed also allowed for the comparison between the robot’s results and the user’s self-evaluation.

D. Results

Fig 7 summarizes the results obtained from the system in both the simulated and real experiments. Fig. 7a shows the total error $e$. We can observe an error decay over time, where the characteristic models are updated after receiving different types of evidence $E_j$ asynchronously. This occurrence is justified by the correct integration of new evidence into the characteristic model through the fusion mechanism.

Fig. 7b shows the evolution of the characteristic estimation error and the characteristic states of a single user characteristics over time. It can be observed that after ~15 iterations, the characteristic estimations converge and the error decays to zero. Despite the fact that different types of evidence $E_j$ are fused independently and asynchronously into the characteristic model, results show a continuous error decay on posterior estimations, suggesting an improvement on each of the characteristic models.

In Fig. 7c we show the results for all users on the real dataset. Similarly to the simulated dataset results on Fig 7, the error rate tends to decay as the system gains information about the users, despite the progressive addition of more users to the system. The error rate converges to a near-zero value, yet unable to reach a null error. This is due to the discrepancies observed between the characteristics reported by the users and the hard evidence gathered by the robot during interaction, illustrated in Fig. 7d.

Fig. 8 illustrates the evolution of the user profile models, as well as of the K-L divergence of the associated Gaussian mixture models when compared to the reference user profiles. We can observe that, as the number number of interactions increases, and the system integrates new evidence, the characteristics of all users converge to the ground-truth. The clusters also converge, thus illustrating the inter-identity relationships present in the data.

Fig. 9 illustrates the results obtained for the failure scenario. Fig. 9a corresponds to the failure scenario running on the simulated population. We can observe, on iteration 1000, that the estimation error stagnates, a sign of the fault in the system, but that by making use of the clusters, the system is able to maintain roughly constant error. We can also observe a large error spike on iteration 2000, where a new population of users is added. While still operating with a failed module, the system is able to learn these new users, achieving a decreasing trend in estimation error comparable to that observed before system failure. Fig. 9b illustrates the same behavior of the system when one of the modules fails when operating on the real dataset.

V. Discussion

The results of a single run of the system, illustrated in
that the system can be transferred to realistic scenarios. Similarity in the results from simulated and real experiments, as illustrated in Fig. 1. It is also demonstrated, through the modules potentially operating on different sensor modalities, heterogeneous information, which is provided by different that the system is able to learn new characteristics from performance for a single or multiple users. These results show it was received. In both tests, convergence to a low estimation error was achieved, demonstrating that the model can learn and estimate the user’s characteristics, as well as to incorporate new information as it was received. In both tests, convergence to a low estimation error was achieved, demonstrating that the model can learn and estimate the user’s characteristics with comparable performance for a single or multiple users. These results show that the system is able to learn new characteristics from heterogeneous information, which is provided by different modules potentially operating on different sensor modalities, as illustrated in Fig. 1. It is also demonstrated, through the similarity in the results from simulated and real experiments, that the system can be transferred to a realistic scenarios.

Thus, our results support claim 1.

It is interesting to note that the user’s self-assessment different significantly from the measurements gathered by the robot during direct interaction, as illustrated in Figs 7b and 7c. In these figures, we observe a remainder of estimation error, i.e. a lower boundary of error below which the system does not fall. This boundary corresponds to the deviation between the user’s self-assessment and the characteristics estimated by the robot during direct interaction, as illustrated in Fig. 7d. This deviation occurs despite the fact that the users indicated to the robot that they felt comfortable with its actions and, as thus, ensured that that was their preference. These discrepancies can be attributed to several sources, such as small discrepancies in the operation of the robot while gathering ground truth, or the user changing their mind when they start interacting with the system.

By observing the evolution of the population and clusters presented in Fig. 8, we can see that the system is able to
cluster users into user profiles. As indicated by the low K-L divergence achieved upon convergence, the user profiles obtained match the profiles obtained from the reference population. The decreasing nature of the K-L divergence observed can also be taken as evidence that the system is learning from interactions, thus further supporting claim 2.

The results of the failure scenarios, presented in Fig. 9, illustrate how the system is able to continue operation despite a failure in one of the modules, both in the synthetic and real datasets. We can observe that the system is able to learn a new population of users, albeit at a slower pace, by using the obtained clusters for the estimation of characteristics when the corresponding modules fail. To compensate for the induced fault in the estimation module, it uses a matching mechanism described in Section III, thus supporting claim 3. As a consequence, the proposed system exhibits fault-tolerant behaviour, as our results demonstrate the system’s ability to gracefully degrade its operation.

A. Edge Cases

Despite supporting our claims, the system presents some theoretical particularities that warrant discussion. Firstly, the

Fig. 8: The evolution of the user profile models (top), user population (middle) and K-L divergence with respect to the reference population (bottom). As tuples are fused into the likelihood, the Estimation process becomes more accurate, and the K-L divergence decreases.

Fig. 9: The results obtained for the failure scenario when run on synthetic and real data.
system will have a tendency to reinforce its seed model while it is executing. As such, if the system receives as input the same evidence repeatedly, its likelihood may deteriorate to the point of mis-estimating neighboring evidence values, reaching a performance plateau as in Fig. 7c, i.e.,

The current fusion mechanism takes into account all evidences equally. If there is a mismatch in the discriminatory power of each evidence value, the system may be integrating unnecessary information into the model. This allows the learning of tuples even if the label is not consonant with all evidence, potentially learning erroneous information.

VI. CONCLUSION

We have presented BUM, a Bayesian User Model for learning the characteristics of a user population by fusing information from distributed sources. We have presented its formulation, materialization and implementation.

We have shown that the system is able to learn and accurately estimate the users’ characteristics, supporting our initial claims. We have also shown that a our distributed system can use user profiles for fault-tolerance, allowing for greater flexibility in deployment, supporting a single robot or a team of robots operating under a cloud infrastructure. Furthermore, we have shown that it performs well in both simulated and real trials, demonstrating its ability to be transferred to real use cases.

In the future, we intend to apply this technique on a larger number of real user characteristics, such as psychological traits and well-being. It would also be interesting to couple this technique with a decision-making technique to achieve user-adaptiveness.

ACKNOWLEDGMENTS

This work was developed in the context of the GrowMeUp project, funded by the European Union’s Horizon 2020 Research and Innovation Programme - Societal Challenge 1 (DG CONNECT/H) under grant agreement No 643647.

The authors would like to acknowledge the assistance of all those who participated in our experiments.

REFERENCES


