An Intelligent Human Simulation (InHuS) for developing and experimenting human-aware and interactive robot abilities

Anthony Favier, Phani-Teja Singamaneni, Rachid Alami

An Intelligent Human Simulation (InHuS) for developing and experimenting human-aware and interactive robot abilities

Anthony Favier[1,2], Phani Teja Singamaneni[1] and Rachid Alami[1,2]*

***Abstract*— Testing and experimenting software which synthesize human-aware and interactive robot abilities is a challenging task. To answer the lack of individual intelligent agents in simulation, this project aims to develop a generic autonomous simulated human both reactive and rational specially designed to act and interact in a simulated environment where a robot acts and interacts. A generic architecture called *InHuS* has been designed for the deployment of such a system. A first implementation has then been implemented in order to investigate more deeply its pertinence and usability. The chosen context is social navigation since our team has contribution in this specific topic and also because there exist today a number of available human-aware robot navigation algorithms. Several experiments have been conducted with different robot navigation planners showing that the system is able to create pertinent situations and also long-term runs situations for testing, challenging and measuring the performance of such planners. These experiments also allowed to exhibit a set of tools and metrics for the analysis of robot behavior over time. Hence, we claim that *InHuS* is a pertinent approach to provide a suitable framework for the first steps of debugging, experimenting and tuning human-aware and interactive robot abilities before their final evaluation with real naive users.**

I. INTRODUCTION

Significant efforts are dedicated today towards the development of robots which interact, assist humans or work side-by-side with them. To be accepted by the society human-robot interactions (HRI) need to be and evaluated as pertinent, safe, predictable, consistent and pleasant. There is already much work under way to try to endow a robot with such properties and hence get closer to a robot acting as an assistant or a partner for humans. However, people working in the HRI field are facing a constraining issue which is testing and evaluating their systems. It is easy to evaluate the effectiveness of completing a task [24] (e.g. time to complete). But evaluating an interaction is much more challenging since there are no defined standards or common metrics yet. Potential metrics or standards are discussed and proposed by some research teams but they have yet to be accepted and used by everyone [11], [29], [2]. User studies with naive people and realistic situations are undoubtedly essential to evaluate and validate the pertinence of such systems. Such studies are generally conducted as a preliminary step to guide future implementations where, quite often, Wizard of Oz schemes [26] are used (e.g. [14]) or when the developed system is sufficiently mature, tuned and robust (e.g. [9]). But, experiments using real humans and robots are burdensome for many reasons : slow, hardly repeatable, expensive, etc. And on the other hand, simulating realistic interactions is difficult which makes simulation also problematic. Now, the robot developers faces a big difficulty: how to test repeatedly and intensively their planners even when they are not sufficiently robust and also how to challenge their systems in a sufficiently large variety of environments and situations which are very costly to build. There is definitely a need for an "intelligent artificial human" who would accept to dedicate a much time as necessary to help the HRI robot developer and serve as a "guinea pig" which challenges the robot interactive and decisional abilities.

Our main contribution is the generic architecture *InHuS* which is able to provide an autonomous reactive and rational simulated human. This architecture also proposes a way to easily send goals to all agents present in the simulation. An implementation of this architecture has been made in a case study on social navigation. The relevant results of this case study validate the architecture and the overall approach of creating an intelligent human simulation. The appropriate human's behavior is discussed through different experiments with different robot reactive planners. The system's ability to describe and evaluate the interactions is shown as well in these experiments.

The rest of the paper is structured as follows. First, section II outlines the motivations behind this project by exposing the constraints of real life experiments and the limitations in simulation. Section III presents the generic architecture with the rational behaviors it should grant, its portability and the components structuring it. Next, section IV proposes an implementation of this architecture in a case study on navigation. The integration is detailed and experiments with different robot planners are discussed. Finally, section V presents discussions and possible future work.

II. WHY AN INTELLIGENT HUMAN SIMULATION

Testing and evaluating performance of human-aware robot software is challenging[33]. The current ways to evaluate such systems are often either user studies with real humans and real robots, or in simulation with virtual robots and operator-controlled human avatars. Both have pros and cons discussed here below.

A. User Studies for HRI abilities

User studies[5] provide very relevant data since real-life situations with end-users are created. This is often a way to

[1] With LAAS-CNRS, Universite de Toulouse, CNRS, Toulouse, France
[2] With Artificial and Natural Intelligence Toulouse Institute (ANITI)
* Contacts : {ptsingaman, anthony.favier, rachid.alami}@laas.fr

validate or demonstrate the final system in the conditions it has been designed for.

However, such experiments are burdensome. First, it's hard to recruit people for tests because they are time-consuming and, a selection of the candidates has to be made. Indeed, there is a huge difference between specialist and non-specialist users. Having technical knowledge about robotics influences how one interacts with a robot.

In addition, the experiments are slow, they cannot run faster than real-time and neither be parallelized. They are also limited by human fatigue. Thus, it's impossible to run the tests in high number or for a long period of time. Experiments in real world are also hardly controllable and reproducible. Therefore, gathering statistics with this method is almost impossible. Even for specialist users, debugging in such conditions is tiresome and extends the development duration.

Moreover, it has been shown that the answers fulfilled in questionnaires afterwards can differ from the actual feelings felt during the interaction. This can create biases on the results, and thus, not reflect the reality.

Lastly, more than needing real humans, such method requires an exclusive physical access to the robot and a place to run the experiment. This can be constraining, expensive and also prevent others from conducting experiments meanwhile.

B. Simulations

One of the main benefits of simulations [6] is the fact that there is no need to access the real robot. Hence, several experiments can be run at the same time. As many robots as wanted can be added and, the experiment environment can easily be changed in contrast to user studies. Thus, simulations are often quicker and easier to conduct. It makes them ideal for testing and debugging during development.

However, simulations have limitations. Robots need humans to interact with, but adding humans in a simulation is a complex task. There are different possibilities to control the human avatar and simulate interactions.

First, the avatar can be scripted. It only executes a series of predefined actions without being reactive to its environment. This is easy to set up but interactions will be very limited. Nevertheless, scripting is a decent solution to quickly debug a system in its early stages.

Secondly, the avatar can be controlled by a real human [10], [17]. This can be done by using a game controller or even motion capture. Augmented or Virtual Reality can also be used to improve immersion and thus have even more realistic reactions from the human avatar [7], [21], [3], [30]. Using both Virtual Reality and motion capture mostly benefits manipulation scenarios because the human will be able to control the avatar with high fidelity. However, navigation scenarios require a game controller or a keyboard to move the avatar. Creating realistic and accurate movements with such devices is way harder. Moreover, having a controlled avatar requires a real human only focused on controlling it. This bring us back to some limitations discussed about user studies like the difficulty in recruiting people, running speed and the limitations due to human fatigue.

Finally, in order to have a reactive autonomous avatar, a proper controller can be developed for it. Two main aspects have to be realized: On one hand the human avatar has to be animated and on the other hand the avatar has to be controlled to generate realistic behavior [12]. Some works already present how to generate realistic motions [22]. But controlling the avatar autonomously to generate realistic behavior represents a large amount of additional work, so most of the time only simple behaviors are implemented. The avatar can be made reactive with some efforts, yet, making it also able to make rational choices requires too much work and is often aborted. MIT proposed an interesting work called VirtualHome [25], but interactions between agents are limited. On the other hand, crowd or pedestrian simulations can be found like MengeROS [4] or PedSim[1]. These are useful and effective but as individual their behavior is simple.

C. InHuS

Given the limitations mentioned above, this project aims to develop from scratch a generic autonomous simulated human avatar both reactive and rational to answer the lack of individual intelligent agents. This avatar has to be adaptive and reactive to what happens in its environment. It has to be rational as well, that is to say, to be able to make relevant decisions when trying to achieve its goal. This implies being able to choose its own goals and change them itself when required or desired. Moreover, the avatar must have a consistent perception of its environment, for example, be able to isolate the robot from other objects present around it. This system's main purpose is to help testing and debugging HRI systems with simulation by providing an intelligent human avatar. Thanks to the saved execution data, computing metrics can allow to compare or even evaluate different systems. Nevertheless, *InHuS* doesn't claim to provide any standards yet, and thus, neither to perfectly evaluate a planner nor to be able to validate a system.

III. GENERIC *InHuS* ARCHITECTURE

The system's architecture needs to be both generic, flexible and able to generate rational behaviors. By generic we mean that the architecture isn't dependent on the type of tasks and activities handled. But note that a generic implementation is impossible. Thus, when instantiated, some components of *InHuS* will be specific to the application. The architecture is inspired by common architectures from autonomous robotics. However, this system distinguishes itself for different reasons. First, the system will always remain in a simulated environment, in contrast with robotic systems whose purpose is to be run on real robots. Secondly, a human user can take control and send goals at any time to the avatar. Finally, about perception, the human avatar takes advantages of the simulation by directly receiving exact data from the simulator, later altered, instead of using simulated sensors as robots usually do.

[1]https://github.com/srl-freiburg/pedsim_ros

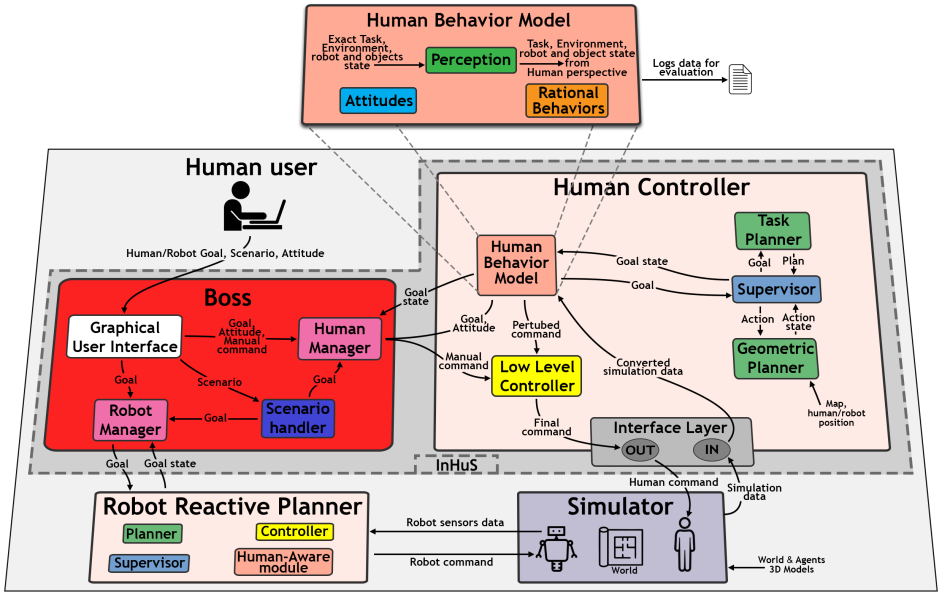

Fig. 1. Generic *InHuS* architecture showing the *Boss* and the *Human Controller* and their interactions. The *Robot Reactive Planner* is the system being tested and challenged by *InHuS*. Notice that the *Human Controller* does not interact directly with the *Robot Reactive Planner* but through the Simulator.

A. Rational Behaviors for the simulated Human avatar

In order to provide an intelligent avatar, the system has to have a relevant goal manipulation. First, the avatar is persistent and will try to reach its goal despite the disturbances. However, if the disturbances become too annoying, the system can decide to reach its goal in another way, or also to suspend it for a moment and even choose a new one. Thus, the system is persistent but, in some cases, can offer other solutions instead of being too persistent.

Moreover, the simulated human can be set in different states which could be compared to moods, behaviors or attitudes. Indeed, these states affect the goal decision and/or the reactions regarding other agents in the environment. This is useful to create a lot of different situations to watch the agents behavior.

It is also completely possible to control the avatar manually at anytime during the execution. This can be useful to create a very specific situation and see if other agents react as expected.

B. Portability with respect to simulators

During the development, robotic systems often use a chosen simulator for test. There are various possibilities for such simulator, sometimes it can even be developed specifically for the named system. In order to improve the usability of the simulated human, an interface layer between *InHuS* and the simulator is available in order to adapt to a given simulator. This is to ensure portability and to allow to test different robot planners on different simulators with limited effort.

C. The Interactive HR simulation Framework

The architecture is aimed to be as generic and flexible as possible. It is composed of two main macro components : the **Boss** and the **Human Controller**. The *Boss* is an interface for the human operator to easily define mission scenarios and send goals to all agents (Human(s) and Robot(s)) in the simulation. It helps to repeatedly run defined scenarios in the same conditions. The second component, *Human Controller*, controls the simulated human. To ensure the rational behaviors described above, its internal architecture has been designed following general guidelines for reactive intelligent agents[1], [18]. Each component has a role more or less generic. Hence, some components might need an implementation that is specific to the application. The architecture and the internal communications can be seen in Fig. 1. Each component's role is described below :

- **Simulator interface** : This is the interface layer mentioned in the previous section. It can convert the data going through it and adapt their frequency.
- **Task Planner** : Its purpose is to elaborate a plan to achieve a given goal of the human avatar. High level goals can be hierarchically decomposed into sub-goals until reaching atomic elementary actions.
- **Geometric Planner** : This component role is to refine at geometric level and execute the elementary actions instantiated by the *Task Planner*. Note that it can have multiple instances, one for each action type e.g. navigation planner, manipulation planner, etc. The commands are sent to the *Human Behavior Model*, where it can be perturbed.
- **Human Behavior Model** : As a core component, this is where most of the avatar behavior is defined and dictated. First, received data from the simulator can be altered to adjust to what the human should know (the so-called human perspective). In the same way, generated commands can be perturbed to add some noise. Goals are either transmitted from the *Boss*, or chosen here

from a predefined known list of goals. This is also where one can define and emulate a mood or an attitude which affect the goal decision or reactions regarding other agents in the environment.

- **Supervisor** : This is the second core component. It receives a goal to achieve from the *Human Behavior Model*. Once received, the goal is sent to the *Task Planner*, which will send back a plan to follow. Then, its job will be to supervise the plan execution and to deal with contingencies. At each step, the *Supervisor* sends the action to execute to the corresponding *Geometric Planner*. The concerned *Geometric Planner* informs the *Supervisor* about the action state e.g. done, failed, etc.
- **Low Level Controller** : This component allows the human operator to take manual control of the avatar. It takes into account the generated command and the potential manual command to compute the final command to send.

D. Perception: Building the human perspective

Even if it has already been mentioned, it is interesting to clarify how the perception of the avatar works. Indeed, Human can gather as many information as needed from the simulator e.g. the exact position of another agent at any given time. After being processed by the *Interface Layer*, these information are sent to the *Human Behavior Model* to be limited to what the human should know from its perspective[28], [23]. Like the case study, the visibility of the avatar can be taken into account to make it consider the robot position only if this last one is visible. In that way, the perception is built inside the *Human Behavior Model* component.

IV. A CASE STUDY: SOCIAL NAVIGATION

The human-aware navigation *InHuS* implementation [16], [27] has been specifically designed and tuned to provide a context where humans are navigating and adapting their behaviour in a usual "standard" environment (office, public space, home). They navigate and adapt their navigation in a rational goal-based manner based on their perspective of the environment. They can "encounter" robots which share the same environment and have some positions to reach.

In this section, we present the specificities of this navigation case by first exposing some integration choices. Then, we explain the different specific behaviors given to the avatar. And finally, experiments in several scenarios with different robot planners will be presented and their results discussed.

A. InHuS taylored to social navigation

The decision has been taken to limit the goals to positions to reach. Thus, every goal can be achieved by executing the elementary action "move to the goal position". Having more complex goals is discussed later.

It has also been decided to focus this case study on having only two agents in the environment : the human avatar and the robot. However, the possibility to add another robot and even another avatar is also discussed later in the paper.

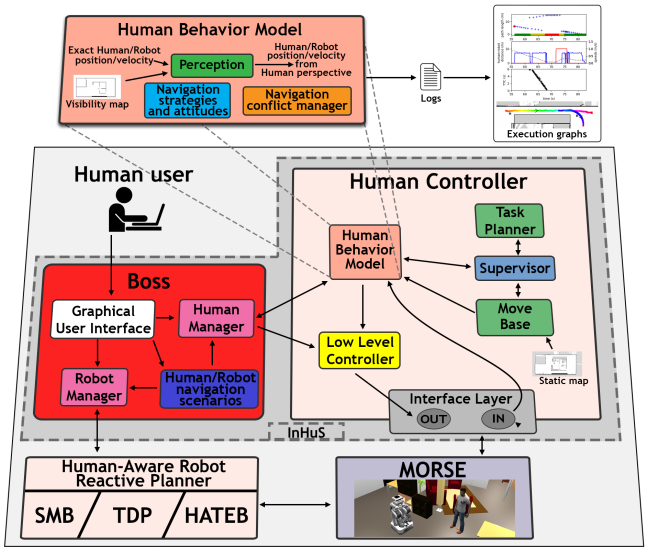

Fig. 2. The architecture proposed for the case study on social navigation: ROS Navigation Stack as a *Geometric Planner* - MORSE as Simulator - Three Robot Reactive Planners have been integrated (see section IV-C) - Visibility is taken into account for the perception - *Human Behavior Model* dealing with specific strategies and attitudes towards encountered robots - Navigation conflict manager - Activity analysis Displays.

To handle navigation actions a navigation planner has been chosen as *Geometric Planner*. We used the *ROS Navigation Stack (Move Base)* since the whole system has been implemented using *ROS Melodic*. The avatar periodically replans to its navigation goal which make it quite reactive. In addition, we used the MORSE Simulator [10] for the integration and the experiments. The implemented architecture can be seen in figure 2 and summarize the specificities of the implementation compared to the generic architecture.

B. Specific human avatar behaviors

For this case study the human avatar has been endowed with some rational behaviors specific to navigation. They are implemented in the *Navigation conflict manager*.

1) Avatar blocked: In order to give some persistence to the avatar, it will keep trying to reach a given goal and even detect if its path is blocked. Indeed, the main reasoning behind detecting if the robot is blocking or not the avatar's way is based on one strong hypothesis made about goals. Any goal given to the system has to be reachable if the avatar is alone in the simulated environment. Thus, if it has trouble to reach its goal, it is obviously because of an other agent's presence. The blocking behavior is articulated in three phases presented just below.

Checking : Initially the avatar computes a path to its goal without considering the robot. Next, the human follows the found path and starts replanning periodically this time considering the robot. If at some point no path is found, according to the hypothesis above, it means the robot is blocking the only possible way to reach the goal. The other case is when the new computed path is too much longer than

the previous one. Instead of following this detour the choice has been made to consider that the robot is blocking the way.

Approach : If during the checking phase the avatar has been considered as blocked, it switches to an approach state. In this state, the avatar alternatively either plan without the robot to get close to the blocking spot and give its intention to pass through, or checks if the robot is still blocking the way by computing a path with the robot back. This path is only used for this check, it is not followed. If, after checking, the robot is not blocking the way anymore then the avatar switches back to its nominal plan execution state. However, if the way is still blocked and the avatar is too close to the robot then it stops and switches to the blocked state.

Blocked : In this state, the avatar stops and checks if the robot is still blocking the way as explained in the *Checking* paragraph. If the way is clear the avatar switches back to the plan execution state. However, if the robot is still blocking the way but moving away, above a certain distance the avatar goes back into the *Approach* state to get closer.

2) Attitudes relative to encountered situations: They are implemented in the *Human Behavior Model* and affect the decisions and reactions of the human avatar regarding the encountered robot. They can be set at any moment through the *Boss* interface. Five attitudes have been implemented to give an idea of what is possible.

The default one is *NONE*. No additional reactions are added. Thus, the avatar executes its plan and checks as explained above if the robot is blocking its way.

The first one affecting goals is *NON_STOP*. The avatar is always active in this mode. As soon as the current goal is done, a new one is picked from a list of predefined known goals. This attitude allows to easily run long scenarios and create unintended situations.

The second one manipulating goals is *RANDOM*. The avatar periodically attempts, with random draws, to choose a new random goal from the list. Thus, its goal can randomly change before being reached.

Another attitude, *STOP_LOOK*, affects the reaction regarding the robot. If the robot gets close enough, the avatar suspends its goal and stops to look in the robot direction for a small amount of time. Its goal is resumed, and the attitude is reset later when the robot is far enough.

The last attitude is *HARRAS*. This state has been made to mimic a non cooperative human. The avatar tries to disturb the robot as much as possible by always going in front of it.

C. Experiments with different scenarios and three planners

InHuS has been tested using three different robot planners and in several scenarios. The first planner is a very simple one, it is in fact just a navigation geometric planner, a *Move Base* node. It has no human-aware features and just tries to find a path to reach its goal. In this paper it will be referred as *Simple Move Base (SMB)* planner. The second planner will be referred as *Human Aware Timed Elastic Band (HATEB)*. It has been developed at LAAS-CNRS by Harmish Khambhaita [13] and is still being improved by Phani-Teja Singamaneni

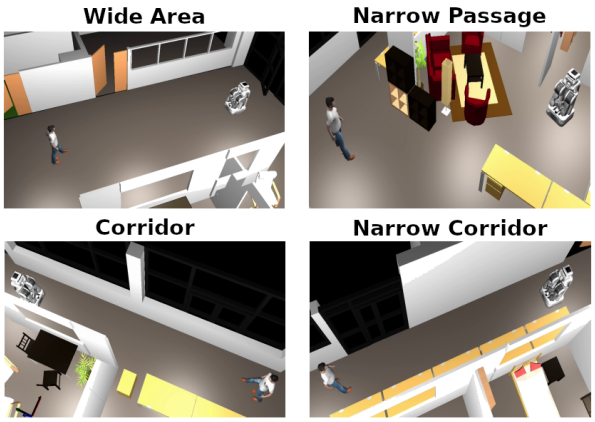

Fig. 3. Four repeatable scenarios. The endless fifth scenario isn't shown.

[31], [32]. The last one is a planner from Kollmitz et al. [15]. It uses time dependent planning on a layered social cost map. It will be referred as *TDP*.

First, as an important warning, the experiments presented here only have as purpose to validate the *InHuS* system and not to evaluate the planners themselves. Indeed, the planners used are neither correctly configured nor in optimal conditions. Thus, their performance in this paper are not representative of their real potential.

In order to have some variety, five scenarios have been designed and implemented in the *Boss*. According to the wanted scenario, the *Boss* sends goals to both agents at the right times. The main four scenario are shown on Fig. 3. The Wide Area scenario is simple but interesting, where the avatar and the robot cross each other in the middle of a large obstacle-free area. It shows if the human has to deviate from its initial trajectory or if the robot takes all the avoidance effort. The Narrow Passage makes both agents cross a small passage at the same time. Typically, a conflict happens at the narrow spot and one agent should wait for the other to cross before passing through. In the next one, they cross in a Corridor just large enough for both of them. This means that if one stays in the middle, the other is blocked. It requires an adaptation of both agents to be successful. Eventually, the fourth scenario is an even Narrower Corridor where the agents cannot cross each other. Hence, one must go out of the corridor to clear the way for the other. The last scenario isn't shown in Fig. 3. It is a long routine scenario that can be ran forever, where the avatar and the robot have loops. The paths followed have been chosen to create many conflicts situations. Besides showing if the robot planner is robust over time, it can create situations that would never have been considered without *InHuS*.

D. Discussion about the challenging scenarios

Before discussing the results of the scenarios we will present the data format produced from the logs during execution. The example taken happens in the narrow corridor, the robot blocks the avatar's way and then clears it by backing off out of the corridor. First, an execution graph is created and can be seen in Fig. 4. It is divided into three

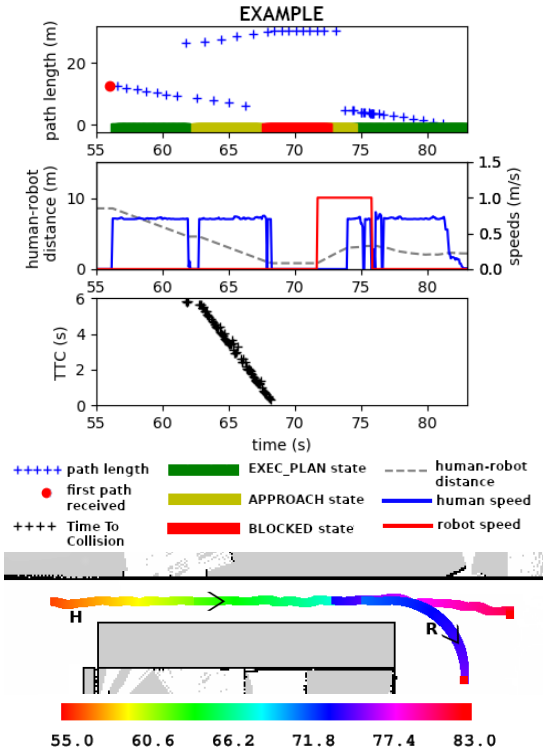

Fig. 4. Example of an execution graph and a colored path (time is seconds)

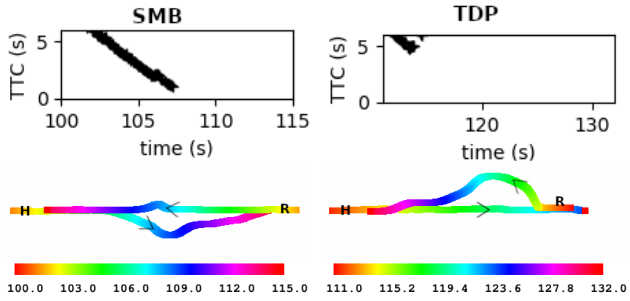

Fig. 5. Scenario Wide Area with SMB and TDP

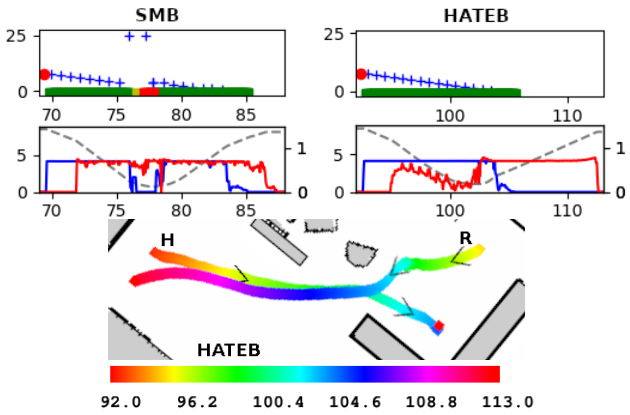

Fig. 6. Scenario Narrow Passage with SMB and HATEB

subgraphs. The first one shows the path length found by the avatar according to time as well as the states of the avatar. This graph illustrates well the periodic replanning and the blocked behavior explained in the section IV-B.1. The second subgraph plots the distance between the avatar and the robot and their speeds over time. The last one draws a metric computed by the avatar called Time To Collision. This estimates the time before both agents collide regarding their current position and speed. Along with the execution graph, the path followed by both agents is shown on a map and colored according to time to know their position at any given time. It can be seen at the bottom in Fig. 4.

For all scenarios the avatar only considers the robot if it is in its field of view and less than $5m$ far.

In each execution of the presented scenarios, all metrics from the graphs as well as the colored path give interesting information. However, due to space limitations, only the most relevant ones are shown and discussed below. Again, these experiments are discussed here only to show that our system is able to put robot planners into interesting situations and its ability to provide relevant metrics condensed in visual graphs to analyse the execution. It is not our goal here to compare or evaluate the planners performance.

1) Wide Area: In this first scenario, thanks to the colored path and the TTC computed, we can see in Fig. 5 that SMB quite threatened the avatar with a TTC that went down to $1s$. The avatar had to deviate from its straight planned trajectory to avoid the robot. On the other hand, TDP anticipated the collision and quickly moved away from the avatar's trajectory in order to not bother or threaten it. In this case the TTC didn't went below $5s$. HATEB provided quite similar results to TDP.

2) Narrow Passage: Thanks to the metrics plotted on the graphs in Fig. 6, we can see at first glance that the avatar has been blocked by the SMB planner. Moreover, by looking at the distance and speeds subgraph we can say SMB was quite threatening because it passed just in front of the avatar pretty fast. On the other hand, we can see that HATEB did way better because the avatar hasn't been blocked nor was put in approach mode. Also, we can clearly see the robot slowing down as it gets close to the avatar and accelerating again only when the collision is avoided, which is way less threatening. TDP results look like the HATEB ones.

3) Corridor: Graphs in Fig. 7 from SMB and HATEB are interesting. First, they show that HATEB was a bit late to put itself on one side, thus it blocked the avatar's way for some time. It switched to the approach state to get close anyway until the robot got close enough to the wall to clear the way for the avatar. The velocity graph also indicates that the robot slowed down when the avatar was close. However, we can see in the figure that SMB stayed in the middle of the corridor and blocked the way. When very close to the avatar, the robot replanned and started to back-off to follow a whole new path not going through the corridor. The velocity graph shows that SMB didn't slow down proactively. It just stopped and maneuvered to back-off once very close. We can see the approach and blocked state as the robot comes close to the avatar and the last approach state corresponds to when the robot is backing-off. The avatar starts following the robot

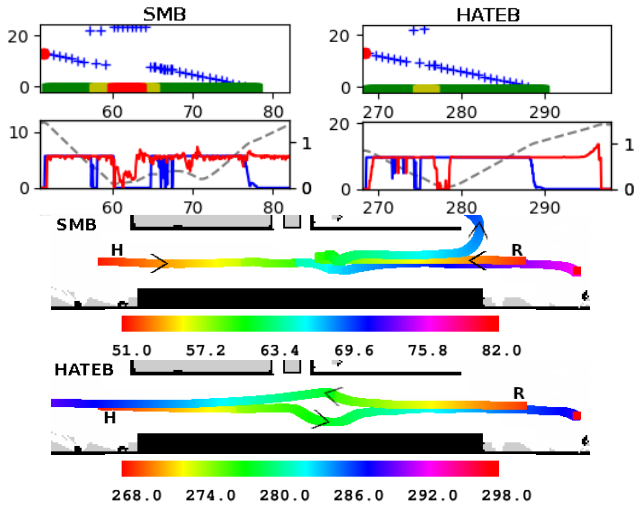

Fig. 7. Scenario Corridor with SMB and HATEB

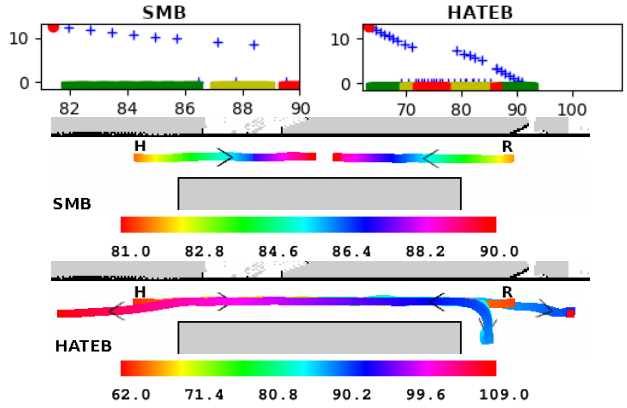

Fig. 8. Scenario Narrow Corridor with SMB and HATEB

until it's out of the corridor. For its part, TDP did very well by going and staying close the wall early. The avatar finds a path and crosses the robot without any trouble.

4) Narrow Corridor: This scenario is very challenging, agents can't cross each other and there is no other path. The graph in Fig. 8 shows well how SMB got stuck in the corridor with no solution. TDP also failed in this scenario. The avatar stays blocked forever. However, we can see on the figure that HATEB handles this case. Both agents come close to each other and soon after being blocked the robot starts to back off to free the way. The avatar follows it in approach mode and eventually reaches its goal. Once the avatar has passed the robot, this last one goes back in the corridor to reach its own goal.

5) Long Runs: An example of this scenario running with HATEB can be seen at the end of the attached video[2]. It lasts more than 30min and creates a significant number of conflict and challenging situations. In order to analyse it we can make a process to look at the distance evolution. We consider that if the avatar and the robot stay too close for a certain amount of time then it means there is a conflict taking some time to be resolved. Thus, the timestamp is saved in order to generate a list of every conflict with their corresponding timestamp to look at afterwards.

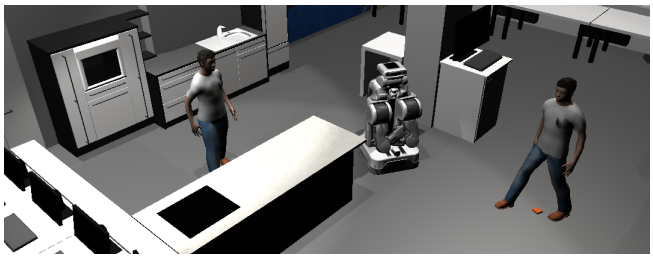

Fig. 9. New environment with 2 avatars both running *InHuS*.

Thanks to the unintended situations created with this routine and the other scenarios some improvements have already been made to the HATEB planner.

E. Simulating Multiple intelligent humans

The current implementation is focused on supporting only one intelligent human acting in the simulated environment. In a prototype version of the system, we have been able to introduce a second "intelligent" avatar in the same scene also running within *InHuS*. For the moment, this comes with some limitations due to the need to take into account potential conflicts and interactions between the simulated humans. However, we are confident on the fact that the architecture can support multiple instances in the same scene. Fig. 9 shows an environment where a second simulated human is added. This also shows the ability to easily switch to different environments.

V. DISCUSSION AND FUTURE WORK

The purpose of this work was to create a simulated human both reactive and rational in order to offer a reliable and pertinent solution to experiments and evaluations in simulation. Thus, a generic and modular architecture has been found and detailed. This architecture is able to cover many kind of goals thanks to a supervisor, a task planner and different geometric planners to execute elementary actions. The *Human Behavior Model* component can affect the perception, output command, goal execution and adds reactions regarding other agents. This component's purpose is to endow the avatar with rational behaviors.

As a proof of concept, the architecture has been implemented in a case study on social navigation. The conducted experiments showed that the system is able to create pertinent, long and repeatable situations to test any robot planners. Moreover, the experiments demonstrated that the visual generated execution data, showing relevant metrics over time, were an effective way to describe and evaluate the robot behavior. In addition, *InHuS* already helped to improve the HATEB planner thanks to the situations and results created during the experiment phase. It was indeed able to repeatably put the planner into situations rarely encountered

[2]https://cloud.laas.fr/index.php/s/MIHahWL1Fsk8RZV

before. Therefore, this case study validates the approach and the usefulness of the system.

As future work, many possibilities can be considered. First, the navigation behavior can be improved by using another navigation planner or different parameters. Also, it can be improved simply by adding some behaviors like making the human take long detours after being blocked for a while to offer a solution to the robot. However, having an human avatar too intelligent that always offers solutions to the robot isn't pertinent and won't put enough stress on robots to have relevant results. Furthermore, the *Task Planner* component could be improved to treat higher level goals e.g. use a Hierarchical Task Network. Moreover, adding a component running *MoveIt!* as an additional *Geometric Planner* could make the system able to handle manipulation actions. Also, adding an immersive manual control of the avatar for instance with a Virtual Reality headset could capture some natural and spontaneous reactions of the human user. Using another simulator than MORSE can also be considered to either go towards more realistic simulations e.g. SimTwo Realistic Simulator[8], USARSim[19][20], or go towards more efficient simulations e.g. Unreal Engine 4. We can also imagine integrating a default simulator with its own interface layer as part of *InHuS*. That way, we could provide a simulated environment to test robot planners not meant for simulation.

We are considering the possibility to make *InHuS* Social Navigation Stack publicly available as a tool for human-aware robot navigation developers to test their one contribution.We can imagine to also invite contributors to provide challenging environments and scenarios. There is also room for developing better automatic analysis tools to assess the quality of robot navigation behavior with respect to social criteria.

REFERENCES

[1] R. Alami, R. Chatila, S. Fleury, M. Ghallab, and F. Ingrand. An Architecture for Autonomy. *The International Journal of Robotics Research*, 17:pp.315–337, Apr. 1998.

[2] A. Aly, S. Griffiths, and F. Stramandinoli. Metrics and benchmarks in human-robot interaction: Recent advances in cognitive robotics. *Cognitive Systems Research*, 43:313–323, June 2017.

[3] B. Armstrong, D. Gronau, P. Ikonomov, A. Choudhury, and B. Aller. Using Virtual Reality Simulation for Safe Human-Robot Interaction. *American Society for Engineering Education*, page 9, Mar. 2006.

[4] A. Aroor, S. L. Epstein, and R. Korpan. MengeROS: a Crowd Simulation Tool for Autonomous Robot Navigation. *arXiv:1801.08823 [cs]*, Jan. 2018. arXiv: 1801.08823.

[5] K. Belhassein, G. Buisan, A. Clodic, and R. Alami. Towards methodological principles for user studies in Human-Robot Interaction. page 7.

[6] H. Choi, C. Crump, C. Duriez, and E. A. Elmquist. On the use of simulation in robotics: Opportunities, challenges, and suggestions for moving forward. *Proceedings of the National Academy of Sciences*, 118(1):e1907856118, Jan. 2021.

[7] A. Cleaver, D. Tang, V. Chen, and J. Sinapov. HAVEN: A Unity-based Virtual Robot Environment to Showcase HRI-based Augmented Reality. *arXiv:2011.03464 [cs]*, Nov. 2020. arXiv: 2011.03464.

[8] P. Costa, J. Goncalves, J. Lima, and P. Malheiros. SimTwo Realistic Simulator: A Tool for the Development and Validation of Robot Software. *Computer Science*, page 18, 2011.

[9] F. Dehais, E. A. Sisbot, R. Alami, and M. Causse. Physiological and subjective evaluation of a human–robot object hand-over task. *Applied Ergonomics*, 42(6):785–791, Nov. 2011.

[10] G. Echeverria, S. Lemaignan, and A. e. A. Degroote. Simulating Complex Robotic Scenarios with MORSE. In *Simulation, Modeling, and Programming for Autonomous Robots*, volume 7628, pages 197–208. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. Series Editors: _:n367 Series Title: Lecture Notes in Computer Science.

[11] D. Feil-Seifer, K. Skinner, and M. J. Matarić. Benchmarks for Evaluating Socially Assistive Robotics. page 18.

[12] J. Hirth, S. A. Mehdi, N. Schmitz, and K. Berns. Development of a Simulated Environment for Human-Robot Interaction. *TELKOMNIKA*, 9(3):465, Dec. 2011.

[13] H. Khambhaita and R. Alami. Viewing Robot Navigation in Human Environment as a Cooperative Activity. In N. M. Amato, G. Hager, S. Thomas, and M. Torres-Torriti, editors, *Robotics Research*, volume 10. Springer International Publishing, 2020.

[14] K. L. Koay, E. A. Sisbot, D. S. Syrdal, M. L. Walters, K. Dautenhahn, and R. Alami. Exploratory Study of a Robot Approaching a Person in the Context of Handing Over an Object. In *AAAI spring symposium: multidisciplinary collaboration for socially assistive robotics*, Stanford, United States, Mar. 2007.

[15] M. Kollmitz, K. Hsiao, J. Gaa, and W. Burgard. Time dependent planning on a layered social cost map for human-aware robot navigation. In *2015 European Conference on Mobile Robots (ECMR)*, pages 1–6, Lincoln, United Kingdom, Sept. 2015. IEEE.

[16] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch. Human-aware robot navigation: A survey. *Robotics Auton. Syst.*, 61(12):1726–1743, 2013.

[17] S. Lemaignan, G. Echeverria, M. Karg, J. Mainprice, A. Kirsch, and R. Alami. Human-robot interaction in the MORSE simulator. In *Proceedings of the seventh annual ACM/IEEE HRI*, 2012.

[18] S. Lemaignan, M. Warnier, E. A. Sisbot, A. Clodic, and R. Alami. Artificial Cognition for Social Human-Robot Interaction: An Implementation. *Artificial Intelligence*, 247:45–69, June 2017.

[19] M. Lewis, S. Hughes, J. Wang, and M. Koes. Validating Usarsim for Use in HRI Research. *th ANNUAL MEETING*, page 5.

[20] M. Lewis, J. Wang, and S. Hughes. USARSim: Simulation for the Study of Human-Robot Interaction. *Journal of Cognitive Engineering and Decision Making*, 1(1):98–120, Mar. 2007.

[21] O. Liu, D. Rakita, B. Mutlu, and M. Gleicher. Understanding human-robot interaction in virtual reality. In *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 751–757, Lisbon, Aug. 2017. IEEE.

[22] P. Maurice, V. Padois, Y. Measson, and P. Bidaud. Digital human modeling for collaborative robotics. In *DHM and Posturography*, pages 771–779. Elsevier, 2019.

[23] G. Milliez, M. Warnier, A. Clodic, and R. Alami. A framework for endowing an interactive robot with reasoning capabilities about perspective-taking and belief management. In *IEEE RO-MAN*, 2014.

[24] D. Perille, A. Truong, X. Xiao, and P. Stone. Benchmarking Metric Ground Navigation. *arXiv:2008.13315*, Nov. 2020.

[25] X. Puig, K. Ra, M. Boben, J. Li, T. Wang, S. Fidler, and A. Torralba. VirtualHome: Simulating Household Activities Via Programs. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8494–8502, Salt Lake City, UT, June 2018. IEEE.

[26] L. D. Riek. Wizard of oz studies in hri: A systematic review and new reporting guidelines. *J. Hum.-Robot Interact.*, 1(1):119–136, July 2012.

[27] J. Rios-Martinez, A. Spalanzani, and C. Laugier. From proxemics theory to socially-aware navigation: A survey. *Int. J. Soc. Robotics*, 7(2):137–153, 2015.

[28] A. C. Schultz and J. G. Trafton. Towards collaboration with robots in shared space: spatial perspective and frames of reference. *Interactions*, 12(2):22–24, 2005.

[29] A. Steinfeld, T. Fong, D. Kaber, M. Lewis, J. Scholtz, A. Schultz, and M. Goodrich. Common metrics for human-robot interaction. In *Proceeding of the 1st ACM SIGCHI/SIGART HRI Conference*, 2006.

[30] M. Suomalainen, A. Q. Nilles, and S. M. LaValle. Virtual reality for robots. In *IEEE/RSJ IROS, Las Vegas, NV, USA*, 2020.

[31] P. Teja S. and R. Alami. HATEB-2: Reactive Planning and Decision making in Human-Robot Co-navigation. In *29th IEEE RO-MAN*, 2020.

[32] P. Teja S., A. Favier, and R. Alami. Human-aware navigation planner for diverse human-robot contexts. *Under review (submitted to IROS 2021)*.

[33] A. Thomaz, G. Hoffman, and M. Çakmak. Computational human-robot interaction. *Found. Trends Robotics*, 4(2-3):105–223, 2016.