Navigating in Populated Environments by Following a Leader

Procopio Silveira Stein, Vitor Santos, Anne Spalanzani, Christian Laugier

Navigating in Populated Environments by Following a Leader

Procópio Stein[1,2], Vítor Santos [1], Anne Spalanzani[2,3] and Christian Laugier[2]

Abstract— Service robots have a great potential of improving human quality of life by aiding in everyday tasks. However, robots that share an environment and interact with humans still face some challenges that limits their acceptance. One of these challenges is how to move and behave among groups of people, which is a task performed seamlessly by humans and some animals.

This is an interesting issue for the robotics community, as it is important to predict and adapt to an environment that is constantly changing, while at the same time respect social conventions. Path planning in dynamic environments has been addressed mostly by predicting future position of humans and avoiding them. However, with the increase of the number of persons in such environments, techniques that are based only on the prediction of the movement of humans can fail, as they usually ignore the human's reaction to the presence of the robot.

Instead of trying to model the complex human motion behavior, this work proposes to rely on humans to guide the robot through difficult situations, where classical approaches would fail to find a solution. This will be accomplished by a probabilistic approach for electing a human leader, according to the robot's desired destination.

In this way, the robot can take advantage of the humans' paths and behavior, effortlessly avoiding dynamic and static features as the human leader does, relieving the robot from the burden of having to generate its own path in difficult situations.

I. INTRODUCTION

Each day it is more and more common for us to find robots realizing everyday tasks and helping humans. Service robots performing tasks at homes, hospitals and museums are real example cases where robots can greatly increase human quality of life. A key feature of service robots is that they must be able to share space and interact with humans. That is also one of the most important limitation for the acceptance of robots in our everyday life, as they are expected to adapt to unpredicted situations, respect social conventions and navigate in very dynamic environments.

But navigation in dynamic environments is still a difficult task to accomplish and it is still an open and challenging issue for the robotic community. In real situations, time poses a very restrictive constraint for optimal navigation algorithms, and sensor measurements are only valid for short periods. As a result, classical path planning techniques cannot be used.

Recent techniques for robot navigation in dynamic environment are based on probabilistic and predictive approaches [1], [2], [3], [4]. These approaches take advantage of the fact that humans usually follow typical paths when moving, and explore that fact using techniques as Gaussian Processes [5] [6] [7] or Hidden Markov Models [8].

Successfully learning and detecting a typical path being followed by a person allows the robot to avoid trajectories that have a risk of future collision with a pedestrian, as well as avoiding entering personal and social spaces and causing discomfort to the persons involved.

A drawback of those approaches, however, is that they usually do not take into account changes that people perform in their typical paths to avoid and adapt to other moving people. These conditions allied to excessive future uncertainty, or crowded environments, may lead to situations where every generated path leads to collisions or frozen situations, as shown by [9].

A policy that does take into account the interactions and changes the robot generates in the environment has been proposed by [10], but although the robot manages to navigate among dense crowds, the agents moved randomly and the resulting path of the robot was suboptimal.

Humans, however, can easily navigate in dynamic environments, dealing with lots of information to accomplish an efficient navigation, even in very complex situations. The way humans move is the result of a series of information gathering and very complex decision making processes, which is not yet completely understood, although some models have been developed [11] and incorporated in planning algorithms [12], [13].

But even if not completely understood, human motion by itself is a very rich source of information and it can be used to improve robot navigation in dynamic environments. Humans move according to typical patterns, and their movements are related to features that they are interested in, such as doors, elevators, stairs or other people. Besides that, humans adapt their motion to the movement of others, cooperatively avoiding collisions and trying to guarantee their safety and the safety of other people in the same environment.

This work proposes to take advantage of the motion of humans to help robot navigation in dynamic and difficult environments, where even modern dynamic navigation algorithms would fail to provide a solution. Although one can argue that humans may also run into problems while moving among others in complex situations, it is undeniable that we are able to deal with such situations better than robots, as we can cope with a series of high-level information about the environment, the context and about other humans in a very

advanced way.

A person leading the robot is able to actively cooperate with other persons moving in the opposite direction, for example, to be able to move through them. Therefore, the robot following this leader can keep a socially accepted behavior (the same of its leader) successfully moving through difficult situations. In this way the robot can also take advantage of natural occurring patterns as line formation [14] to pass through crowds, without the need of incorporating such models in its algorithm.

An approach similar to this one has already been presented by [15]. However, some differences and improvements can be pointed out. While in that work the leader choice is deterministic, based on the apparent moving direction of a leader candidates, here this choice is made in a probabilistic fashion, based on previously learned typical paths and probable goals of the environment, enabling a more robust leader choice.

Besides that, in the work from Müller , the $A*$ planner is used for an optimal global planner which, in the case where no leader is found, is not well suited for navigating in dynamic environments. The leader following technique using the potential field method also present some problems, as obstructions can appear between the leader and the robot, which are not addressed by this approach. In the current work, for both situations, the RiskRRT [16] technique is used, which was designed for navigation in dynamic environments even in the absence of a leader and which can also manage obstructions between the robot and the leader.

To validate the proposed techniques, a crowd simulator based on the Social Forces Model [17] is implemented. In this way, all the agents involved in the experiments react to the presence of each other and, therefore, the reaction of simulated persons to the presence of the robot can be taken into account, resulting in more realistic simulations.

In section II the technique to choose a leader in dynamic environment is presented. The crowd simulator is presented in section III and the developed leader following algorithm in section IV. The experiments and their results are presented in section V, and after that, the conclusions of this work are presented in section VII.

II. CHOOSING THE LEADER

The proposed method depends greatly on the choice of a leader among moving people. Several different criteria can be used in this choice as similar paths, similar goals, distance from leader candidate and others. In this work, the leader will be chosen based on the similarity of goal. This means that a person moving to a destination close to the one of the robot is a good leader candidate.

In complex and dynamic environments, determining the goal of moving persons is not a straight forward task. In simple environments, like a single corridor, the current velocity and orientation of a candidate may be enough to determine a possible destination. However in more intricate environments, like offices, persons can perform complex movements to reach their destinations, as shown in Fig. 1. In this Figure, dashed lines represent the goal and motion
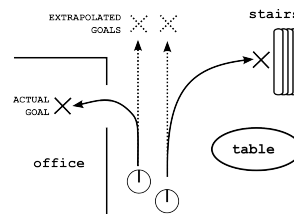

Fig. 1. Comparison of goal prediction using simple extrapolation and the actual goals of two moving agents

prediction using simple extrapolation or a Kalman Filter. However due to the environment structure and interest points, the actual goal and paths, represented by solid lines, highly differ from the ones predicted using simplistic assumptions.

A more robust method to predict future motions and goals of humans rely on typical paths, which are paths normally taken by persons to move between interest points. With this approach, the limitations posed by simple extrapolation techniques can be overcome, as it allows to take into account the structure of the environment as well as the most common motion patterns. It is necessary, however, to have previously mapped and observed an environment and how people move around it.

The technique chosen to model typical paths and goal inference is the Growing Hidden Markov Models (GHMM) algorithm [8]. It implements an approach where the *learning* and *prediction* phases are on-line concurrent processes, resulting in a *learn and predict* paradigm. The structure of the GHMMs are the same as the regular HMMs, with the difference that as new observations sequences are incorporated into the model, the transition structure and the number of states can change.

The GHMM algorithm consists in the use of the Growing Neural Gas (GNG) algorithm [18], used to estimate the model structure as well as the transition probabilities of a Hidden Markov Model (HMM). As the algorithm is adaptive, it is capable of creating or removing states to cope with new observations.

A key aspect of the GHMM is that it always associates a typical path with a goal, as it is based on the hypothesis that moving agents always try to reach a point of interest in the environment. This makes this technique well suited to be used in this work as it inherently provides an estimation of a goal, matching the motion pattern of a human with previously learned typical paths.

Figure 2 illustrates a typical learned structure of a GHMM. The sequence of connected nodes represent typical paths taken in that environment. There are three possible destinations in this example, which represent three different doors. All persons enter the environment from the same place and then, along the initial corridor, position themselves according to their intended destination. The algorithm is able to capture that motion pattern and associate them with the interest points (doors).

It is important to notice that a simple extrapolation of direction and speed at the beginning of the corridor would
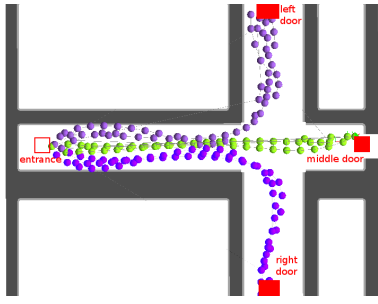
Fig. 2. GHMM structure showing typical paths leading to three different interest points (represented by solid squares) in an office building. The algorithm is able to correctly associate different goals according to the motion pattern soon after the entrance of the building.
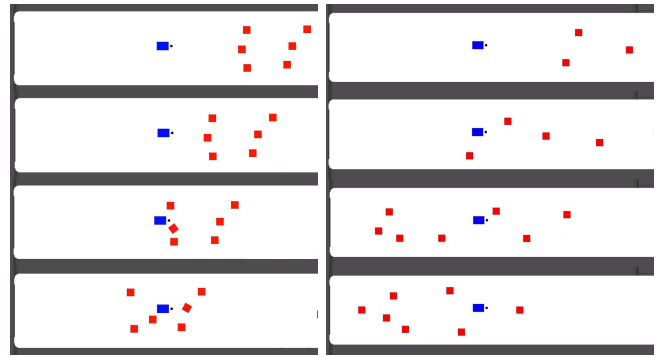

Fig. 3. Comparison between two simulators. The blue rectangle represents a static robot, while the red squares represent agents moving from the right to the left. The left column shows a simulator where agents do not interact with each other unless there is a collision risk. The right column shows a simulator with the Social Forces model implemented, notice how agents interact with each other, spreading more uniformly over the open space.

all lead to the middle door. However, this is not correct, and the GHMM algorithm is able to correctly infer the destination of persons based on the pattern of the movement they take.

III. CROWD SIMULATOR

As persons move, they constantly adapt their motion to the environment structure but also to other persons and objects. This means that even if a robot is standing still, people moving in its direction will actively avoid it.

Based on this, to be able to perform realistic experiments, it is necessary to use a simulator that implements a reactive behavior. Although very advanced simulators are available [19], in this work, a more simpler simulator is developed. It is loosely based on the Social Forces Model, from [17], which has been extensively validated in several works as a simple and efficient way of replicating pedestrian dynamics.

In this model, the resulting velocity and orientation of the robot is the resultant of the action of two types of forces:

- an attraction force that pulls the agents toward their destination and
- a repulsive force caused by other agents and by physical objects as walls and obstacles.

The attraction force was modeled as a constant value while the agents' repulsive force was modeled as a force proportional to the inverse of the squared distance to other agents. The repulsive force caused by physical objects is the normalized sum of the forces exerted by static features located within a 2×2 meters square, centered on the robot. Their value is proportional to the inverse cubic distance from the robot to these features. All the forces are summed and them normalized to a maximum value and have instantaneous effect on the robot's velocity.

The advantages of using an approach where agents implement pedestrian dynamics can be seen in Fig. 3, where it compares two different pedestrian simulators. In the left column, a simulator where agents move straight toward their goals, with a simple collision avoidance algorithm results in unrealistic pedestrian behavior, that does not take into account the reaction to other pedestrians. In the other hand, the right column shows a simulator that implements the Social Forces model and the agents actively react to the presence of the robot and of the other agents, better

replicating the real behavior that humans would have in such situation.

The result is a very useful tool to perform experiments that explore the reaction of groups of people in crowded environments, as it provides a realistic validation framework. It can be shown that even if the robot navigates following a straight line in the presence of people, it will likely succeed because, as in real life, pedestrians will adapt their motion to avoid the incoming robot, as long as they notice it. However that will be accomplished with the cost of an impolite behavior that is likely to cause discomfort for the pedestrians.

IV. FOLLOWING THE LEADER

Once a moving agent has been detected as a leader, due to a similar goal with the robot's goal, there still remains the problem on how to follow the person.

Two main methods for following someone can be applied. The robot can follow the leader's path or it can directly follow the leader. The path following approach has the advantage of guiding the robot around obstacles, even if they are not detected by its sensors, as a glass door for example, or other reasons unknown for the robot. However, as pointed out in [20], the direction-following method can results in shorter paths and yield a more human-like behavior that more closely matches the expectations of a person following robot.

Following the suggestion of that work, an hybrid solution will be implemented. The criteria for switching methods will be the distance from the robot to the leader. When this distance is above a certain threshold, the robot will engage in path-following behavior, to help it avoid undetected obstacles. Once the robot gets close to the leader, it will switch to a person-following behavior, as it will be more unlikely that obstacles will appear between the robot and the leader, besides being a more socially accepted behavior.

A. Path Planning Algorithm

Besides the following behavior, the path planning algorithm has to be able to attempt to maintain a navigation solution in dynamic environment. To accommodate

this requirement, the Risk Rapid-exploring Random Tree (RiskRRT) [21] algorithm will be used. It is a variation of the classic RRT algorithm, modified for navigation in dynamic environments. It takes into account the risk of traveling along generated paths according to predicted objects' motion.

The probability of collision, or risk, can be seen in this case as a measure of the feasibility of a path, with the maximum accepted risk specified as a threshold. The RiskRRT algorithm also takes into account the interactions among humans so the robot can behave in a socially acceptable way [22]. Therefore, the risk function is a measurement of safety and also of a human friendly navigation.

When the robot engages in a leader following behavior, the risk computation is deactivated so the robot can move closely to humans and to avoid entering frozen situations, as explained in the introduction.

B. Leader Algorithm

The developed program to follow a leader is shown in the algorithm below. The program starts after receiving a desired goal for the robot, which is used to initialize the RiskRRT algorithm. The algorithm then enters a loop until the goal is reached. The pedestrians position in the environment are sent to the GHMM predictor, which outputs a list of the predicted goal for each person.

After that, if no leader has been detected yet, a routine sweeps the agents predicted goals list and outputs an identification and Euclidean distance of the agent's predicted goal that is the closest to the robot one. If that distance is within an empiric threshold, a leader has been found and his/her identification is stored. In the case that the smallest distance between goals is still larger than the threshold, the RiskRRT algorithm computes a path until the robot's chosen goal.

If a leader has been found in a previous interaction, the program verifies that his/her predicted goal is still under the acceptable threshold. If this verification fails, the *foundLeader* flag is set to false and the loop restarts. If the verification succeeds, the leader position is stored in a structure that tracks his/her path.

The next subgoal of the RiskRRT algorithm will depend on how close the leader is from the robot, in order to determine if the robot will engage in a path-following or a person-following behavior. If the leader is close enough, according to a threshold, then the last position of his/her path (which corresponds to the leader's current location) is chosen. Otherwise, the first position of the tracked path becomes the subgoal for the RiskRRT.

The planning algorithm then calls an *update* routine, which causes the RiskRRT algorithm to use the new subgoal instead of the original goal, in order to find a path.

As a result, the algorithm explores the open space and finds a path that poses the lesser risk to bring the robot to the chosen subgoal. The $managePath$ routine manages the leader's tracked path, removing points that have been reached by the robot.

This sequence of steps makes the robot follow the leader's path or position. The use of the Risk-RRT algorithm to reach

---

Algorithm 1 Leader choice and follow.

1: **procedure** $choose\_and\_follow$
2:     $goal \leftarrow readGoal()$
3:     $RiskRRT.init(goal)$
4:     **while** goal not reached **do**
5:         $agents \leftarrow Tracker()$
6:         $goalPred \leftarrow GHMM(agents)$
7:         **if** $!foundLeader$ **then**
8:             $id, dG \leftarrow minDist(goal, goalPred)$
9:             **if** $dG < threshG$ **then**
10:                 $foundLeader = true$
11:                 $leader = id$
12:             **else**
13:                 $RiskRRT.update(goal)$
14:             **end if**
15:         **else**
16:             $dG \leftarrow Dist(goal, GHMM(leader))$
17:             **if** $dG < threshG$ **then**
18:                 $path \leftarrow trackPath(leader)$
19:                 $dL \leftarrow Distance(robot, leader)$
20:                 **if** $dL < threshL$ **then**
21:                     $subgoal \leftarrow getSubG(path[end])$
22:                 **else**
23:                     $subgoal \leftarrow getSubG(path[start])$
24:                 **end if**
25:                 $RiskRRT.update(subgoal)$
26:                 $path \leftarrow managePath(path)$
27:             **else**
28:                 $foundLeader = false$
29:             **end if**
30:         **end if**
31:     **end while**
32: **end procedure**

---

and follow subgoals has two main advantages. Firstly, it provides a reliable method to navigate between subgoals, since in a dynamic environment the space between the robot and its first subgoal may be occupied by moving agents or static obstacles.

In second place, once the robot reaches and starts to follow subgoals, the algorithm is capable of reusing nodes of its exploration tree to efficiently generate new paths for each new subgoal received by the *update* routine. The reuse of previously generated nodes, reduces the computational load of the algorithm, while still taking into account the risk of navigation.

Finally, in the case that a leader is not found, or the current leader is lost, the *update* routine sends once again to the RiskRRT algorithm its final goal, as chosen at the beginning of the program.

V. EXPERIMENTS

The experiments were performed using several independent modules of the Robot Operating System (ROS) [23]. Two types of experiments were conducted: simulating a robot
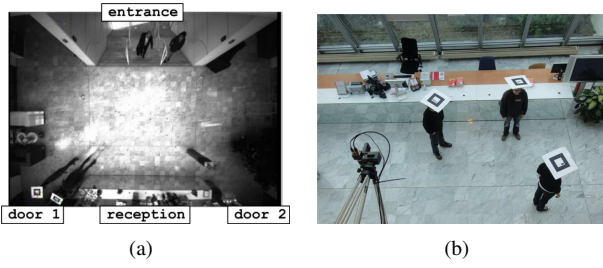
Fig. 4. (a) INRIA Rhône Alpes' entrance hall; (b) markers used for the tracking system and camera with wide angle lens at INRIA's hall
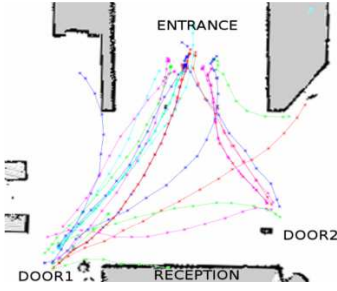

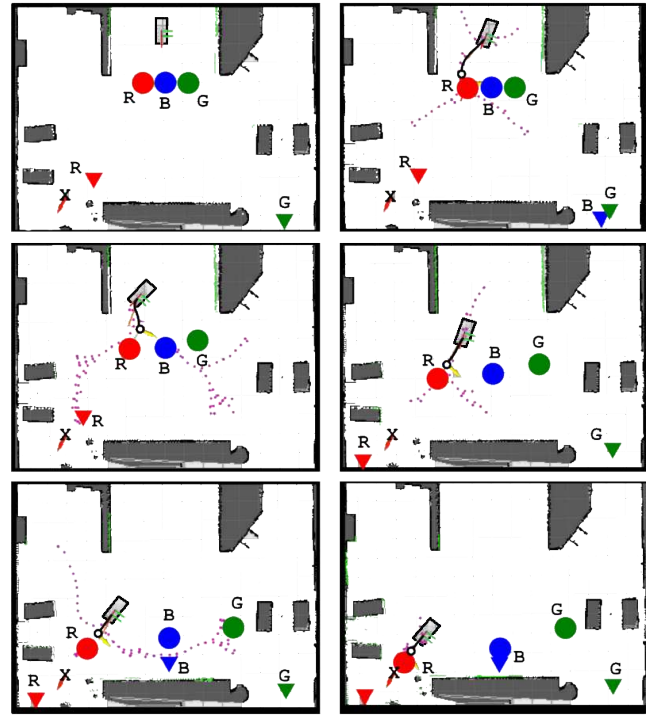Fig. 5. Typical paths acquired from real data and used in the GHMM initial training

Fig. 6. Experiments of leader detection and following, the robot is represented by the light gray rectangle and its goal by a X. Three leader candidates are represented by circles with letters R, G and B. The predicted goals are the triangles with the corresponding letters.

under a real data set and also using the crowd simulator described in section III.

A. Data Acquisition

Real data was recorded in the main hall of INRIA Rhône Alpes, which provides well defined interest points, as shown in Fig. 4(a). The implemented tracker is based on the work of [24]. In the current work, fiducial markers were worn as hats by subjects in order to provide a robust and fast deployment tracker system, as shown in Fig. 4(b). An overhanging camera with wide angle lens acquired images that were provided to the tracker algorithm.

It is important to notice that the typical paths of any environment are unlikely to change, unless some structural modification takes place. In this way, an advanced temporary setup can be assembled in order to acquire the data to train a typical path model. After this stage, the robot must rely on its own sensors or receive information from external sources to be able to track moving agents, as fixed lasers for example.

The GHMM was trained using a set of the real data acquired with the tracking system. Volunteers were asked to move naturally among predefined interest points in the environment, as the entrance of the hall and the two doors. Fig. 5 shows a sample of the trajectories used for the training.

B. Real Data + Simulation

Two types of tests were conducted, one that evaluates the leader detection technique when several subjects move close to each other, and another test that evaluates the advantage of the proposed technique to avoid agents moving in the opposite direction of the robot.

A robot was simulated using PLAYER/STAGE, while the scenario agents represent real data recorded from the motion

of humans. In the figurer, the robot is represented as a light gray rectangle, and starts in the top center of the scenario. The obstacles are colored dark gray and encompass walls, desks and sofas.

The circles are the persons detected by the overhanging camera, and the triangles are their respective predicted goals. They have a letter associated to identify their colors (Red, Green and Blue). The robot goal is marked as an X, located at the lower left of the test area. Finally, the dots represent the RiskRRT exploration nodes and the solid line is the path chosen by the algorithm.

In the first test, shown in Fig. 6, three humans start to move just in front of the robot, and pursue one different goal each. After some iterations, as the subjects start to move in the scenario, the prediction algorithm gives an estimation for two of them (red and green). Based on that estimations, the leader following algorithm makes the choice to follow the red subject, as its predicted goal lies within a distance threshold from the robot's goal.

The objective of the second test is to evaluate the benefits of following a leader in order to avoid agents moving in the opposite direction and is shown in Fig. 7. The way the robot selects and follow a leader occurs in the same fashion as in the previous test. The robot goal is again in the left bottom corner of the image, but here there are now two humans that move from the door to the stairs, in the opposite direction of the robot's desired trajectory.

After the leader is chosen, the robot starts to follow him/her. As the leader approaches the two humans moving in
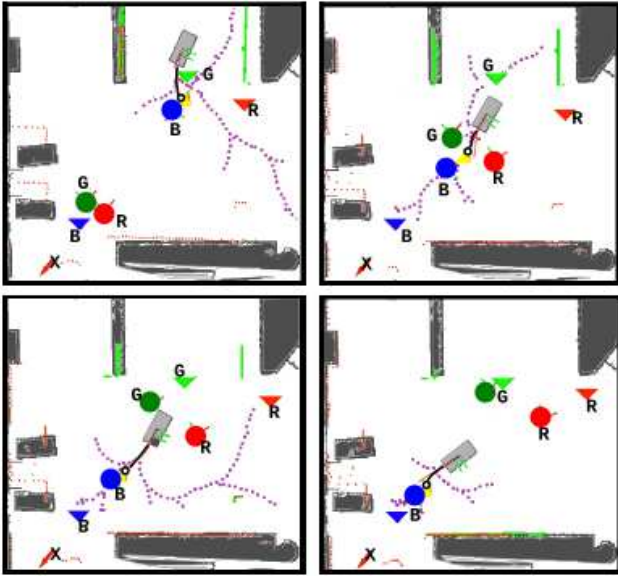
Fig. 7. Experiment of a leader guiding the robot avoid two incoming persons. The robot is represented by the light gray rectangle and its goal by a X. Three leader candidates are represented by circles with letters R, G and B. The predicted goals are the triangles with the corresponding letters.
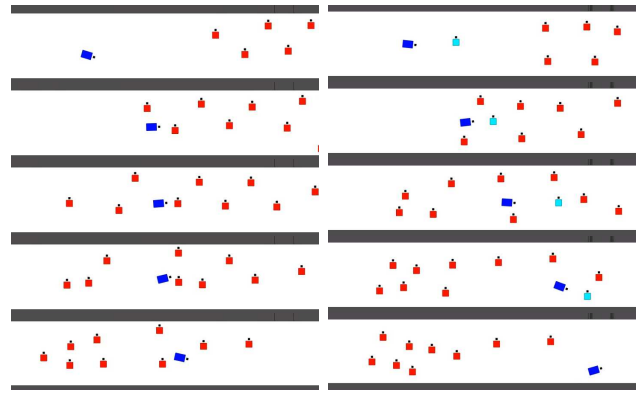

Fig. 8. Comparison between a robot trying to reach the right side of a corridor without (left) and with (right) a leader, represented by the light blue square. The blue rectangle represents the robot, while the red squares represent agents moving from the right to the left.

the opposite direction, they naturally give room for him/her to pass. The robot benefits from this space and is able to continue to move without the need to take evasive measures to avoid the two incoming persons.

C. Crowd Simulation Tests

These tests were performed using the PLAYER/STAGE software, together with ROS. Again, two types of test were carried out. The first illustrates the potential of the leader following technique to escape situations were the robot would get frozen due to incoming pedestrians in a narrow corridor. The second test brings together the leader choice and following through a group of people moving from the opposite direction in a more complex environment.

During the first experiment of this part, the GHMM algorithm is not active, as the goal is to demonstrate only the capability of a leader to guide the robot through difficult situations (Fig. 8).

This simulation shows that without the aid of a leader, even using a state-of-the-art algorithm (RiskRRT [16]) the robot is not able to find a feasible solution to its goal. The reason is that due to the number of agents and the uncertainty on the prediction of their future positions, no free space is available to the motion planning algorithm.

However, when a leader exists and the robot follows him/her, it manages to escape that situation as the leading agent knows that the others will make room for it. In the simulation case, this knowledge is recreated implementing the same algorithm in all the agents. In real situations, a human leader this knowledge is inherent to the subject as his/her is able to detect cues and behaviors on other humans that would be very hard for modern sensors and algorithms to detect, as gaze direction, context analysis, body orientation

and so on. As a quantitative result, when the robot followed a leader, the time spent to reach its goal reduced about 25%, in average.

The last experiment involves a more complex situation. The algorithm will evaluate all the persons that are within a radius from the robot as leader candidates. In the scenario there are three possible destinations (top, middle and bottom). Using the GHMM algorithm and the typical paths learned (see Fig. 2), the algorithm predicts the likely goal for each one of the candidates. The robot's goal is located in the bottom of the image (Fig. 9), and is marked with an X. The candidate that has a goal similar to the robot's is chosen, and the robot starts to follow him/her.

As in the previous experiment, the robot successfully manages to pass through the incoming group of people until reaching its destination.

D. Discussion

The tests assessed the capability of the system to predict the goal of real moving agents, as well as the ability of the designed algorithm to properly follow a chosen leader, while avoiding other dynamic agents.

Results show that the leader following algorithm makes a proper choice of a leader, based on a probabilistic approach for goal prediction, even when the initial movement and is not directed toward his/her goal. This is an important advantage of a probabilistic approach for goal detection, based on previous knowledge of the most common trajectories in the environment.

The advantages of following a leader in a dynamic environment become evident in the experiments. Classical approaches that would attempt to plan a trajectory taking into account the predicted motion of the incoming humans would fail to find an optimal solution, as a straight line to the robot's goal would be blocked.

However, as the robot follows a human that is able to correctly assume that the persons moving in the opposite direction will adapt their movement to avoid a collision, it is able to follow a straight trajectory to the goal. The
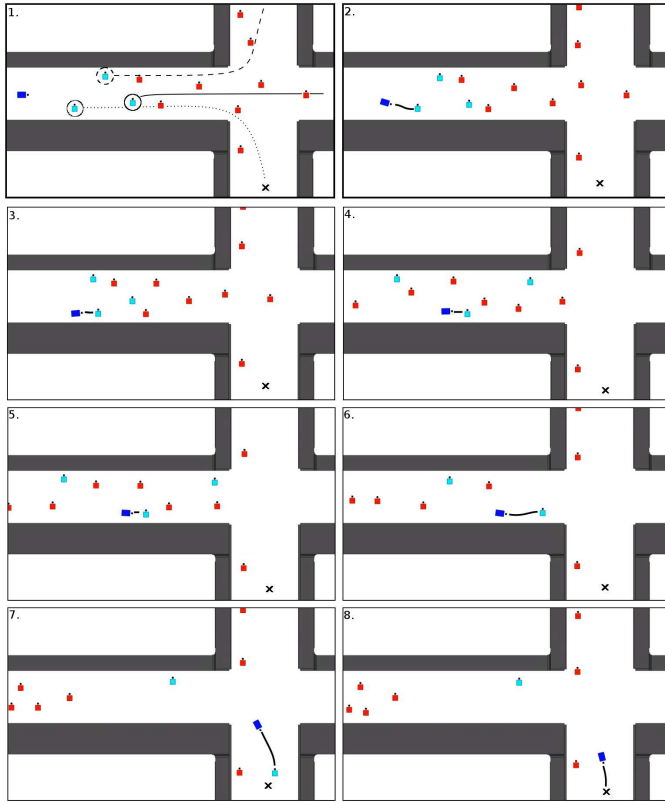
Fig. 9. The robot is represented by the blue rectangle, the leader candidates are light blue squares and the incoming pedestrians are represented by red squares. The first figure shows the predicted typical path and goal for each evaluated candidate. The leader is then chosen based on goal likeliness.

Fig. 10. Two different instants while following a group of persons. In the first image, the person at the right leaves the corridor and in the second image, the subject in the middle is about to do the same.
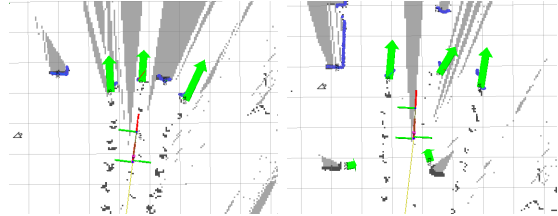

Fig. 11. The same two different instants while following a group of persons as in the previous figure. In this group of figures, the laser measurements and tracked targets (blue groups) can be seen, together with the estimated targets velocities (green arrows)

result of this experiment shows the benefit of the proposed technique. The robot is able to follow an optimal trajectory as a consequence of following a leader that has a better understanding on how to behave in such situations.

VI. IMPROVING LEADER SELECTION

Leader following has already been demonstrated as an useful way to solve typical problems regarding navigation in human environments. In simple scenarios, the choice of a leader is obvious, but that task can be more complex where various leader candidates are present and several factors must be taken into account.

In the previous experiments, the choice of a leader was based only on goal similarity, which was predicted based on the typical paths usually taken by agents in an environment. Although the results were promising, there is still room for improvements on leader selection.

The next step on this research is to have a system that can receive information about leader candidates and give each one of these candidates a score, representing how good it would be to follow each person, given the objectives of the robot itself and the current state of the environment.

In order to accomplish these objectives, a dataset with situations of a robot following humans has been built. Data collection was performed with a small car-like robot and the sensors used were a LIDAR and a wide-angle lens camera,

so videos can be taken during tests and associated with the laser scans, as shown in Figure 10 and Figure 11. The robot was always telecommanded by a person that was hidden from the subjects being followed.

In total 47 tests of the robot following persons or group of persons were recorded, with the medium duration of about 20 seconds each. Out of the total, in the end of 10 tests, interviews were conducted with the leaders in an attempt to understand how the robot affected the leaders and how the leaders reacted to the presence of the robot.

The study of this dataset and the development of an improved leader selection algorithm based on these tests will be discussed in further works.

VII. CONCLUSIONS

This work presents a method to take advantage of human motion in dynamic environments by selecting and following a leader. An important contribution is the probabilistic approach used to select a leader, which takes into account the typical paths in an environment and provides a probabilistic inference of the subject's goal.

Tests used real and simulated data. The results validated the proposed approach, with the robot being able to properly identify leaders among several subjects and follow him/her until its desired goal.

Experiments were conducted in a crowd simulator framework, that implements pedestrian dynamics, allowing a realistic evaluation of the proposed approaches, due to the capacity of agents to adapt their movement to the motion of the robot and other agents.

Future work will explore different forms of electing and taking advantage of a leader, based on the newly created dataset of situations where a robot follow humans. This will allow a better understanding of leader reactions as well as an

improved framework to decide among several leader candidates. Experiments will continue in different scenarios, with more tests in specific situations as leader obstruction/loss. Once the studied techniques are robust enough, tests will be performed with a real robot, for real world validation.

ACKNOWLEDGMENT

REFERENCES

[1] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun, "Learning motion patterns of people for compliant robot motion," *The International Journal of Robotics Research*, vol. 24, no. 1, p. 31, 2005.

[2] S. Sehestedt, S. Kodagoda, and G. Dissanayake, "Robot path planning in a social context," in *2010 IEEE Conference on Robotics Automation and Mechatronics (RAM)*. IEEE, June 2010, pp. 206–211.

[3] S. T. O'Callaghan, S. P. N. Singh, A. Alempijevic, and F. T. Ramos, "Learning navigational maps by observing human motion patterns," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2011, pp. 4333–4340.

[4] D. Vasquez, P. Stein, J. Rios-Martinez, A. Escobedo, A. Spalanzani, and C. Laugier, "Human aware navigation for assistive robotics," in *International Symposium on Experimental Robotics*, June 2012.

[5] C. Tay and C. Laugier, "Modelling smooth paths using gaussian processes," in *Proc. of the Int. Conf. on Field and Service Robotics*, Chamonix, France, 2007.

[6] K. Kim, D. Lee, and I. Essa, "Gaussian process regression flow for analysis of motion trajectories," in *2011 IEEE International Conference on Computer Vision (ICCV)*, 2011.

[7] D. Ellis, E. Sommerlade, and I. Reid, "Modelling pedestrian trajectory patterns with gaussian processes," in *2009 IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops)*. IEEE, Oct. 2009, pp. 1229–1234.

[8] D. A. Vasquez Govea, T. Fraichard, and C. Laugier, "Growing hidden markov models: An incremental tool for learning and predicting human and vehicle motion," *International Journal of Robotics Research*, vol. 28, no. 11-12, pp. 1486–1506, Nov. 2009.

[9] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct. 2010, pp. 797–803.

[10] B. Neuman and A. Stentz, "Anytime policy planning in large dynamic environments with interactive uncertainty," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2012, pp. 2670 –2677.

[11] K. Mombaur, A. Truong, and J. Laumond, "From human to humanoid locomotionan inverse optimal control approach," *Autonomous Robots*, vol. 28, pp. 369–383, Dec. 2009. [Online]. Available: http://www.springerlink.com/content/c571474575ux1t52/

[12] D. Althoff, D. Wollherr, and M. Buss, "Safety assessment of trajectories for navigation in uncertain and dynamic environments," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2011, pp. 5407–5412.

[13] P. Henry, C. Vollmer, B. Ferris, and D. Fox, "Learning to navigate through crowded environments," in *2010 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2010, pp. 981–986.

[14] D. Helbing, I. Farkas, and T. Vicsek, "Simulating dynamical features of escape panic," *Nature, Vol. 407, pp. 487-490, 2000*, 2000.

[15] J. Müller, C. Stachniss, K. Arras, and W. Burgard, "Socially inspired motion planning for mobile robots in populated environments," in *Proc. of International Conference on Cognitive Systems*, 2008.

[16] C. Fulgenzi, A. Spalanzani, and C. Laugier, "Probabilistic rapidly-exploring random trees for autonomous navigation among moving obstacles," in *Workshop on safe navigation, IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

[17] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical Review E*, vol. 51, pp. 4282–1286, 1995.

[18] B. Fritzke *et al.*, "A growing neural gas network learns topologies," *Advances in neural information processing systems*, vol. 7, pp. 625–632, 1995.

[19] A. Shigemura, Y. Ishikawa, J. Miura, and J. Satake, "An RT component for simulating people movement in public space and its application to robot motion planner development," *Journal of Robotics and Mechatronics*, vol. 24, no. 1, p. 165, 2012.

[20] R. Gockley, J. Forlizzi, and R. Simmons, "Natural person-following behavior for social robots," in *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, 2007, pp. 17–24.

[21] C. Fulgenzi, C. Tay, A. Spalanzani, and C. Laugier, "Probabilistic navigation in dynamic environment using rapidly-exploring random trees and gaussian processes," in *IEEE/RSJ 2008 International Conference on Intelligent RObots and Systems*, Nice, France, 2008.

[22] J. Rios-Martinez, A. Spalanzani, and C. Laugier, "Understanding human interaction for probabilistic autonomous navigation using Risk-RRT approach," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Sept. 2011, pp. 2014–2019.

[23] M. Quigley, B. Gerkeyy, K. Conleyy, J. Fausty, T. Footey, J. Leibsz, E. Bergery, R. Wheelery, and A. Ng, "ROS: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[24] J. Snape, J. van den Berg, S. Guy, and D. Manocha, "Independent navigation of multiple mobile robots with hybrid reciprocal velocity obstacles," in *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2009, pp. 5917–5922.