Robot Navigation Taking Advantage of Moving Agents

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

***Abstract*— A crucial requirement for service robots is to be able to move in dynamic environments shared with humans as well as interact with them. Navigation in such environments is a challenging task, as the environment is constantly changing, future states have to be predicted and planning and execution must be carried on-line.**

However, even in very complex situations, humans can easily find a path that avoid both dynamic agents and static obstacles. This paper proposes a technique to take advantage of the human movement in such populated environments, selecting a leader to be followed in a probabilistic fashion, according to the robot's desired destination.

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

I. INTRODUCTION

With advances in mobile robotics and lowering costs of computers, it is becoming more and more common for us to find robots among groups of people. Service robots (home care, hospital, museum guides) are real example cases where robots have to be able to move and interact with humans in an ever changing environment. The success of interactions and human acceptance of service robots is directly related to the way they behave and approach others, as well as their capability to adapt to the environment.

However, navigation in dynamic environments is still an open and challenging issue for the robotic community. In such environments, sensor's measurements are prone to noise and the measurements have a short lifespan, being valid only for small time periods. Static features, that could guide a navigation algorithm, might not be detected. There are also limitations in the time spent by the navigation algorithm to provide solutions, and optimal approaches are unsuited for this task, as generated paths might be valid only for few time-steps.

Despite these difficulties, humans can easily navigate in dynamic environments. Moreover, their movement can provide indirect information of the environment, as they usually do not move at random. Instead of that, they 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 [1].

This research has been funded by the INRIA PAL Project and by Fundação para a Ciência e a Tecnologia SFRH/BD/46604/2008.

[1] Institut National de Recherche en Informatique et Automatique, INRIA Rhône Alpes, France anne.spalanzani@inria.fr, christian.laugier@inria.fr

[2] Department of Mechanical Engineering, Universidade de Aveiro, Portugal procopiostein@gmail.com, vitor@ua.pt

[3] UPMF, Grenoble, France

Taking advantage of that, several works propose to model those typical paths with probabilistic approaches as Gaussian Processes [2] [3] [4] or Hidden Markov Models [5], to address the problem of navigating in populated environments [6], [7]. Correctly detecting and learning a likely path permit the robot to avoid trajectories that have a risk of future collision with a pedestrian, as well as avoiding social disturbances [8], [9].

Those approaches, however, do not take into account changes people perform in their typical paths to avoid and adapt to other moving people [10]. These conditions allied to excessive future uncertainty may lead to situations where every generated path lead to colisions or frozen situations, as shown by [11].

The key insight of this paper is to present a navigation technique for dynamic environments that takes advantage of the typical movement of humans. This approach relies on the fact that people try to guarantee their safety and the safety of other people in the same environment, avoiding obstacles and avoiding hitting other persons.

In other words, people walking in populated scenarios can provide rich indirect information about their surroundings, as they are constantly dealing with large amount of high-level information and reacting to it, while following a goal.

A robot moving to a certain destination could identify behaviors of humans and detect a leader, someone moving along a typical pattern that would pass close to the destination point. After identification, the robot could follow people along that path, as "moving with the flow", relying more and more on people in front of it, as leaders or as part of a swarm.

A similar approach has been developed by [12]. With the difference that the main goal of that work is to implement a human-like motion behavior, and the choice of a leader is deterministic, based on the motion direction of the subject regarding the trajectory planned by the robot. Although such technique can be used for the same objectives that are proposed in this work, in some environments the algorithm may fail to find a leader as the extrapolation of the initial movement of a candidate may not match his/her actual goal, as illustrated in Fig. 1.

In this paper, however, the choice of a leader is performed probabilistically, using the Growing Hidden Markov Models (GHMM) technique, an extension to the HMM capable of learning the models parameters and structure in an incremental fashion. This technique not only provides a prediction of the future states of a moving agent, but also its goal, which allows the robot to plan further ahead, with a probabilistic knowledge of the goals pursued by the moving persons.

The global path planning is implemented with the

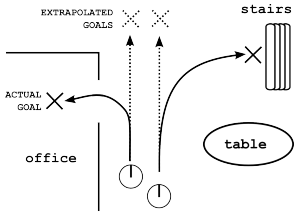

Fig. 1. Comparison of goal prediction using simple extrapolation and the actual goals of two moving agents

RiskRRT algorithm [7], which takes into account the risk of collision with dynamic obstacles while generating the tree. The use of this algorithm guarantees that the robot can find path solutions in dynamic environments when a leader is not found.

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

II. CHOOSING THE LEADER

As the proposition of this work is to take advantage of the movements of a person, the method used to choose which person to follow plays a major role. Here, the choice of a leader is implemented based on the distance between the goal given to the robot and the predicted goal of the leader candidate.

The prediction of the motion and goal of a leader is not an easy task in dynamic environments. A simplistic approach may extrapolate the current orientation and speed of a moving person in an attempt to determine his/her likely goal. But due to environment or dynamic restrictions, the subjects can completely change their trajectory in the subsequent time steps, invalidating the extrapolation, as shown in Fig. 1. Here the dashed lines represent the goal and motion prediction using simple extrapolation or a Kalman Filter. Note however that 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.

Knowing that the movement of humans is highly dependent on the environment structure and interest points, recent approaches take advantage of the typical paths in order to make predictions of motions and goals of humans. This approach can overcome the limitations posed by simple extrapolation techniques, as it allows to take into account the structure of the environment as well as the most common motion patterns.

In the current work, in order to predict the goal of moving agents, the Growing Hidden Markov Models (GHMM) algorithm is used [5]. 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, updating the model, as seen in Fig. 2.

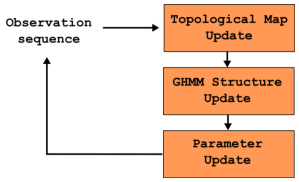

Fig. 2. GHMM learning algorithm overview (adapted from [5])

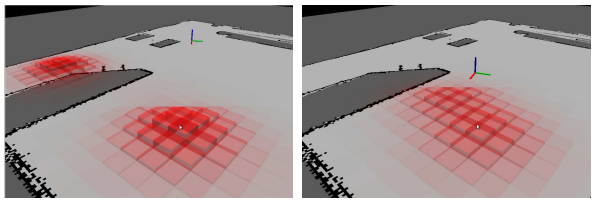

Fig. 3. Two instants in the goal prediction of a dynamic object. The height of the bars is proportional to the probability of a cell to be the final goal

The GHMM algorithm consists of the use of the Growing Neural Gas (GNG) algorithm [13], 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.

One very important aspect of the GHMM algorithm is that it is based on the hypothesis that moving agents always try to reach a goal in the environment. Therefore, each goal in the scenario has a different HMM associated to it. As the intention of the current work is to predict the most likely goal of a moving agent, the GHMM inherently provides a direct solution to that problem.

III. 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. Simplistic approaches that try to use the current leader position as a subgoal may bring the robot to situations where undetected obstacles are present between the robot and the leader.

Following the leader path, generating subgoals along the tracked trajectory can overcome situations were obstacles that were not detected appear between the robot and the leader. However, if the robot is not close enough to the leader, it can lose track from the person or different agents that are not leaders may appear in the scene, blocking the tracked path.

Besides that, the motion algorithm has to be able to maintain a navigation solution even in situations where no leader is found, or after a leader is lost due to sensor occlusions or scene exit by the leader being followed. These constraints are addressed with the use of a variation of the RRT algorithm that efficiently explores the free space while at the same time takes into account the risk of collision with moving agents in a scenario.

A. Risk Rapid-exploring Random Tree

The Risk Rapid-exploring Random Tree (RiskRRT) is a variation of the classic RRT algorithm presented by [14] developed for navigation in dynamic environments. It takes into account the risk of traveling along generated paths according to predicted objects' motion. It combines a part dedicated to perception (of static and moving obstacles) with another for planning trajectories. Navigation and planning are done in parallel.

The configuration-time space is searched randomly, and a tree T is grown from the initial configuration all over the configuration space. The algorithm chooses a point P in the configuration space and tries to extend the current search tree toward that point.

The points P are randomly sampled on the map, but at the beginning, and then once every 100 times, the goal itself is chosen; this bias, which has been empirically set, speeds up the exploration toward the goal. The node chosen for extension is the most *promising* node: all the nodes in T are weighted taking into account the risk of collision and the estimated length of the total path:

$$\tilde{w}(q_N) = \frac{L_\pi(q_N)}{dist(q_0, q_N, P)} \quad (1)$$

$$w(q_N) = \frac{\tilde{w}_(q_N)}{\sum_q \tilde{w}_q} \quad (2)$$

At numerator, the likelihood of $\pi(q_N)$ is normalized with respect to the length of the path N; at denominator, $dist(.)$ is the sum between the length of the path from the root q_0 to the node q_N (which is known) and an estimation of the length of the path to P.

The weights are normalized over the set of nodes in the tree (2). The node to grow next is then chosen taking the maximum over the weights or drawing a random node proportionally to the weight. The new node q_+ is obtained applying an admissible control from the chosen node q toward P. The weight of q_+ is computed. If $w(q_+) \geq w(q)$ the tree is grown again from q_+ toward P otherwise another point is sampled from the space.

The likelihood of each partial path can also be expressed as the multiplication of the independent probability of collision with static (P_{cs}) and dynamic components (P_{cd}):

$$L_\pi(q_N) = \prod_{n=0}^{N}(1 - P_{cs}(q_N)) \cdot \prod_{n=0}^{N}(1 - P_{cd}(q_N)) \quad (3)$$

The prediction approach for forecasting the position of moving obstacles in the near future is done using the GHMM predictor. With the information of probable occupied positions in the future, the robot can anticipate the behavior of the agents. The selection of the best trajectories is done by taking into account the probability of collision for each path.

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. Therefore, the risk function must rely on safety but also in human friendly navigation.

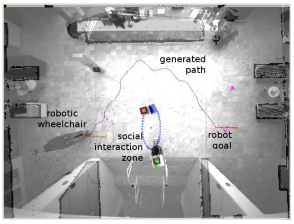

Fig. 4. RiskRRT algorithm avoiding a social interaction zone

The current version of the RiskRRT algorithm also takes into account the interactions among humans in order to behave in a socially acceptable way. The robot behavior should follow social conventions, respecting proximity constraints, avoiding people interacting or joining a group engaged in conversation without disturbing them. Therefore, the risk function must rely on safety but also in human friendly navigation.

After these extensions the "probability of success" calculated for every partial path is given by the probability of not encountering a collision along the path and not entering a social interaction zone. For more details about this method, refer to [15]. Fig. 4 depicts a situation where the RiskRRT algorithm generates a path that avoids the social interaction between two persons.

B. Algorithm

The developed program to follow a leader is shown in III-B. The program starts after receiving a desired goal for the robot, which is used to initiate the RiskRRT algorithm. At the same time, the tracking program starts, in order to detect moving agents in the scenario. The detected agents are fed into the GHMM predictor, which outputs the prediction of a goal for each person.

After that, the Euclidean distance between the robot's current goal and each subject's predicted goal is computed. If this distance falls inside a threshold, the corresponding subject is chosen as a leader.

In the case a leader is found, the robot starts to track its path, but does not immediately starts to follow the leader. This only takes place once the elected leader is closer to the goal than the robot. The reason for this criteria is to avoid situations where the robot would have to move away from the goal in order to follow the path of a leader.

Once the criteria to start following a leader is satisfied, the robot calls an $update$ routine in RiskRRT planning algorithm. This routine 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, which lies over the leader path. The $getSubGoal$ routine decides when to pass a new subgoal of the path to the RiskRRT algorithm, based on the robot

Leader Follower

```
1: procedure leader_follower
2:     goal ← readGoal()
3:     RiskRRT.init(goal)
4:     while goal not reached do
5:         agents ← Tracker()
6:         goalPred ← GHMM(agents)
7:         for i = 1 → agents.size() do
8:             d ← Distance(goal, goalPred[i])
9:             if d < thresh then
10:                foundLeader = true
11:                leader = i
12:                return
13:            elsefoundLeader = false
14:            end if
15:        end for
16:        if foundLeader = true then
17:            path ← trackPath(leader)
18:            if leader is closer then
19:                subgoal ← getSubGoal(path)
20:                RiskRRT.update(subgoal)
21:            end if
22:        else
23:            RiskRRT.update(goal)
24:        end if
25:    end while
26: end procedure
```

distance to the previously passed subgoal. This sequence of steps makes the robot follow the tracked leader's path.

This sequence of steps makes the robot follow the tracked leader path. The use of the Risk-RRT algorithm to reach and follow a leader's path has two main advantages. Firstly, it provides a reliable method to navigate until the leader's path start, 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 the leader path, 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, reduce 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.

IV. EXPERIMENTS

The experiments were performed using several independent modules using the Robot Operating System (ROS) [16]. The modular architecture provided by ROS gives a series of advantages when implementing experiments, as modules can be added or removed, allowing the test of different techniques without the need of modification of the remaining modules.

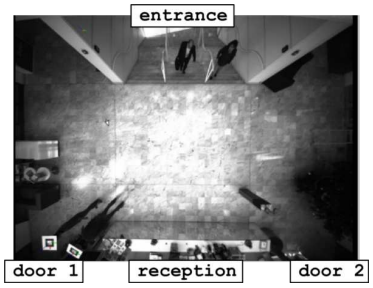

Fig. 5. INRIA Rhône Alpes entrance hall

Fig. 6. Subjects wearing hats with markers and camera with wide angle lens at INRIA's hall

To conduct the experiments, the main hall of INRIA Rhône Alpes has been chosen (Fig. 5). It is an interesting choice as it has a large flow of people during different times of the day, entering and leaving the building during lunch hours and at the beginning and the end of a working day. These conditions allow an easy understanding of the typical paths present in the scenario.

A. Real Data Acquisition

The trajectories used in the experiments to test the robot capability to chose and follow a leader are real human trajectories, that were previously tracked and recorded. An overhanging camera with wide angle lens provides an overview of the test area. The implemented tracker is based on the work of [17]. 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. 6.

The GHMM is trained using a set of the real data acquired with the tracking system. Volunteers were asked to move naturally among interest points in the environment, as the entrance of the hall and the two doors. Fig. 7 shows a sample of the these trajectories.

B. Test Scenario

Two types of tests were conducted, one that evaluates the leader detection technique when several subjects move close to each other (Fig. 8), and another test that evaluates the advantage of the proposed technique to avoid moving agents (Fig. 9). The human perception and detection are performed by a second computer, using the techniques mentioned in the previous subsection.

The tests were conducted using a simulated robot, while the scenario agents represent real data recorded from the motion of humans. The scenario is simulated using

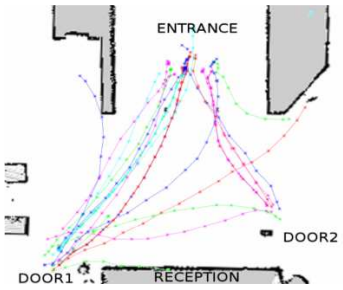

Fig. 7. Real data used in the GHMM initial training

PLAYER/STAGE and the robot has a copy of the environment map and localizes itself based on its odometry and a simulated laser range finder. The robot is represented as a light gray rectangle, and starts in the upper 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 tree nodes and the solid line is the best path found by the planning algorithm.

C. Leader Detection Test

In the first test, shown in Fig. 8, 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 of following the red subject, as its predicted goal lies within a distance threshold from the robot's goal.

Once that the leader is closer to the goal than the robot, by an empiric factor, the planned trajectory is computed from the robot to the subgoal. This corresponds to the first position of the chosen leader's path, and the robot starts to move along the trajectory taken by the human, in the direction of its desired goal.

D. Dynamic Agents Avoidance Test

The objective of this test is to evaluate the benefits of following a leader in order to avoid other dynamic agents and is shown in Fig. 9. 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, using the same approach as in the previous test, the robot starts to follow him/her. As the leader approaches the two humans moving in the opposite direction, they naturally give room for him/her to pass. As the robot is closely following the path taken by the leader, it is able to continue to move without the need to take evasive

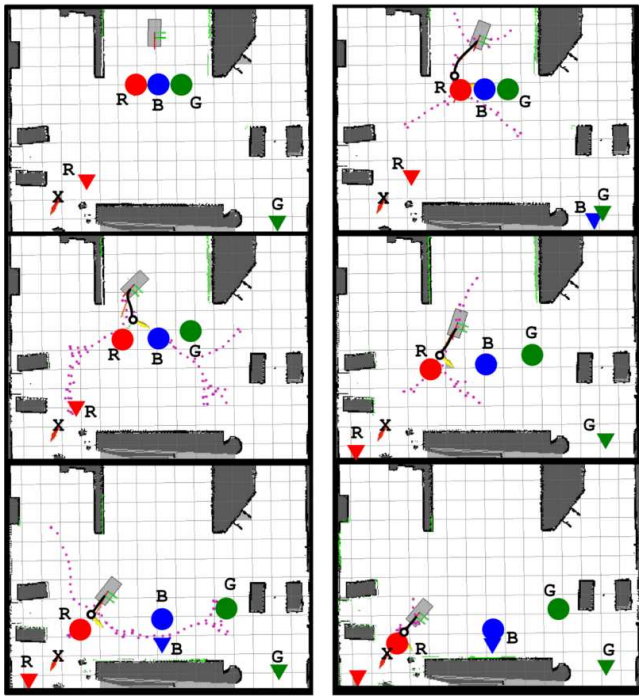

Fig. 8. Results of a typical test of leader detection and following

measures to avoid the two incoming persons. As a result, the robot benefits from a straight trajectory toward its goal.

E. 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 navigation technique employed to follow the leader's path continuously explore the surrounding space for alternative trajectories. In the case the leader is lost of the path being followed becomes blocked, the branches that were generated in different directions can be used to find a new path, without the need of replanning from scratch. This is a very important characteristic while navigating in dynamic environments, specially due to time constraints.

The resulting behavior is that the robot is able to follow the leader while at the same time exploring the open spaces to accommodate new and unpredicted situations.

In the second test scenario, the advantages of following a leader in a dynamic environment becomes evident. 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.

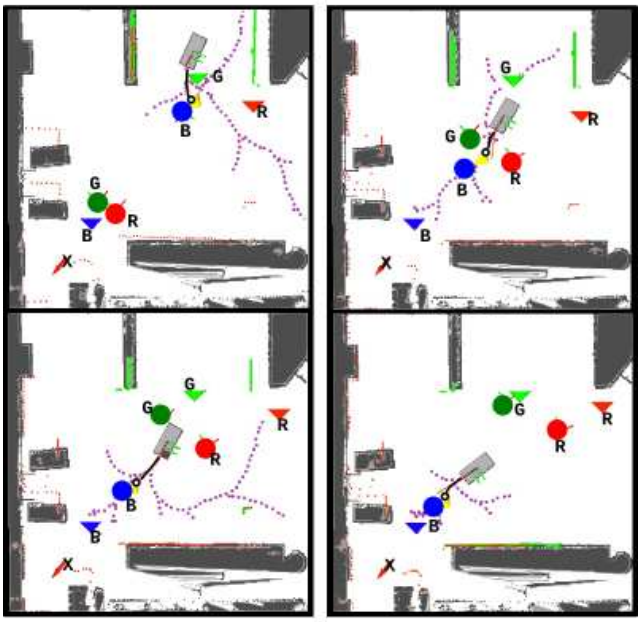

Fig. 9. Leader following allows the robot to avoid two incoming persons

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

V. CONCLUSIONS

This work presents a method to take advantage of human motion in dynamic environments by selecting and following a leader. Its has two main contributions. The first is the methodology used to leader selection, which takes into account the typical paths in an environment and provides a probabilistic inference of subject's goal. The second is the modification of an algorithm designed for motion planning in dynamic environment, in order to adapt it to the task of leader following, while maintaining its original characteristics.

Tests used real data for the leader selection part, while the leader following algorithm was tested in a simulated and real environment. 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. It is our belief that the prediction algorithm can be further improved if it takes into account not only the position of the agents but also their speed and orientation, making it able to anticipate even more the future motion and goals of persons.

New ideas arose throughout this work, as the possibility to find leader that help the robot in portions of its path, which would required a more refined technique of leader selection, with the robot alternating between aided and unaided navigation.

Experiments will continue in different scenarios, with more tests in specific situations as leader obstruction/loss and also crowded environments. Currently a crowd simulator framework is in development, where extensive testing can take place and the impact of the robot behavior in the humans around can be taken into account.

ACKNOWLEDGMENT

We thank Dizan Vazques, Jorge Rios-Martinez and Arturo Escobedo for very useful insights and discussions.

REFERENCES

[1] 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/

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] C. Fulgenzi, A. Spalanzani, and C. Laugier, "Probabilistic motion planning among moving obstacles following typical motion patterns," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 2009, pp. 4027–4033.

[8] 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.

[9] 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.

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

[11] 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.

[12] 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.

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

[14] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, p. 378, 2001.

[15] 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.

[16] 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.

[17] 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.