**Noname manuscript No.**
(will be inserted by the editor)

# Anticipative Kinodynamic Planning: Multi-Objective Robot Navigation in Urban and Dynamic Environments

**Gonzalo Ferrer · Alberto Sanfeliu**

Received: date / Accepted: date

**Abstract** This paper presents the Anticipative Kinodynamic Planning (AKP) approach for robot navigation in urban environments, while satisfying both dynamic and nonholonomic constraints. Our main motivation is to minimize the impact that the robot is doing to the environment, i.e. other pedestrians, while successfully achieving a navigation goal. To this end, we require a better understanding of the environment, and thus, we propose to integrate seamlessly a human motion prediction algorithm into the planning algorithm. In addition, we are able to anticipate for each of the robot's calculated paths or actions the corresponding people's future trajectories, which is essential to reduce the impact to nearby pedestrians. Multi-objective cost functions are proposed and we describe a well-posed procedure to build joint cost functions. Plenty of simulations and real experiments have been carried out to demonstrate the success of the AKP, compared to other navigation approaches.

**Keywords** Robot navigation · Motion prediction · Dynamic environments

G. Ferrer
CDISE Department, Skolkovo Institute of Science and Technology, Moscow, Russia
E-mail: g.ferrer@skoltech.ru

A. Sanfeliu
Institut de Robotica i Informatica, CSIC-UPC.
Llorens Artigas 4-6, 08028 Barcelona, Spain.
E-mail: sanfeliu@iri.upc.edu
This research was conducted at the Institut de Robòtica i Informàtica Industrial (CSIC-UPC). It was partially supported by CICYT projects DPI2007-61452 and Ingenio Consolider CSD2007-018

## 1 Introduction

It is of the greatest importance that service robots can successfully navigate in typically urban environments, and at the same time, people's behaviors should not be conditioned by the presence and the maneuvering of robots. Naturally, robots and people should interact if they are sharing the same environment, as exemplified in Fig. 1-*Top*. This issue becomes the main motivation of the present work: we aim to minimize the robot's impact towards all those nearby pedestrians, while navigating and executing tasks.

To this end, we propose an anticipative approach that is able to foresee people's trajectories and their corresponding reactions to each one of the planned actions, selecting accordingly the best robot action. The key point of our method lies in our effort to understand the scene and the implications derived from the robot's actions, as a step towards a more *intelligent* behavior.

Temporal restrictions are important in social environments: people walk and change their positions during time, and thus, we consider them as dynamic obstacles. Other approaches make the assumption that pedestrians behave as static obstacles. Under some circumstances, such as simple configurations of the environment, these approaches succeed perfectly. However, when the conditions get more complex, like a highly dynamic environment in semi-crowded scenes, understanding the consequences of robot's actions is essential to find a good navigation trajectory. In this work, we define semi-crowded as a group of up to 8 dynamic obstacles (people) that move around the robot and thus significantly alter navigation paths. On the other hand, static people or idle people are those pedestrians with velocities close to zero that do not alter navigation paths significantly. Predicting trajectories for dynamic

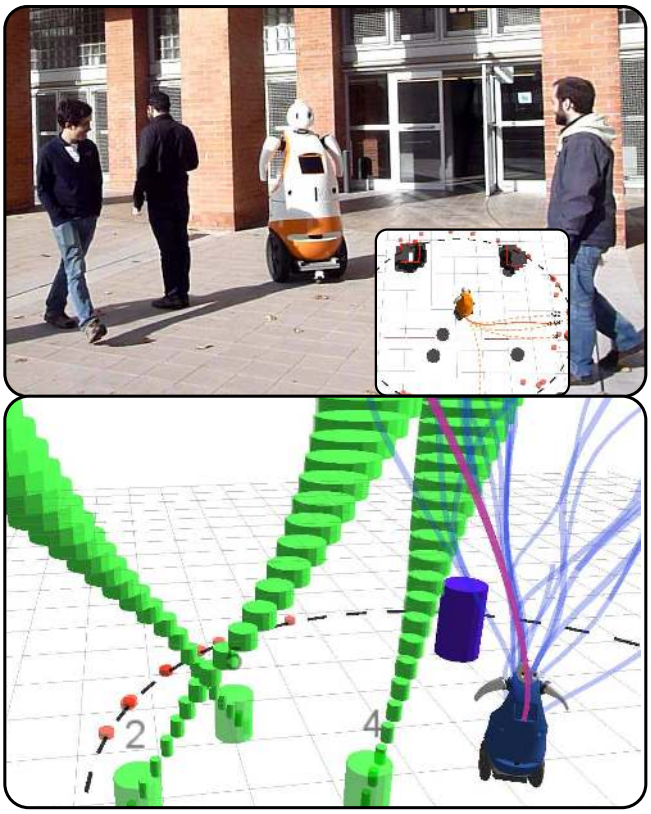


**Fig. 1** *Top*: The AKP navigating in a urban environment. *Bottom*: Robot visualization of the scene, where people are plotted as green cylinders and their predictions are drawn in $space \times time$, being the $z$ axis time.

obstacles is thus more involved and prone to error than static obstacles.

A cost-based navigation path is calculated while satisfying both dynamic and nonholonomic constraints, also referred as kinodynamic constraints. In in Fig. 1-*Bottom* is depicted an example of the temporal constraints considered by the Anticipative Kinodynamic Planning (AKP).

Our approach separates the generation of trajectories and their evaluation to achieve our objective: reducing the social-aware navigation impact and at the same time, navigating up to the goal. We have tested the validity of the algorithm first through thousands of simulations, and then in a real robotic platform comparing with widely-used methods.

The following is a list of the main contributions derived from the present work:

1. An anticipative approach, where each action of the robot corresponds to different people's predictions.
2. A well-posed multi-objective cost function to find the best trajectory among a set of candidates.
3. Experiment design seeking repeatability on a social environment.

The structure of our proposed algorithm is depicted in Fig. 2-*Left*. The white boxes represent well known methods necessary for a navigation scheme, such as a map and a global planner. Our contributions are within the blue boxes and include modifications to the local planner and an adjoint prediction module.

The remainder of this work is structured as follows. Section 2 describes the related work. In Section 3 we define the robot's and people's states considered throughout the paper. Section 4 presents the prediction framework and all its components. In Section 5, we discuss the planning algorithm used and how to integrate it with the prediction framework. The multi-objective costs are described in Section 6. Section 7 describes the simulations, as a prior step before real testings are carried out. Finally, Section 8 describes the experiments, comparing our method with other state of the art approaches in real environments.

## 2 Related work

The prediction of human motion is not an easy endeavor. The Social Force Model (SFM) by Helbing and Molnár [1] is a popular approach to describe human motion as particles geverned by external forces. There are multiple extensions of it such as heading heuristics [2], or the work of Zanlungo et al. [3], considering a collision time which determines the magnitude of the interacting force.

Arechavaleta et al. [4] predicted human motion using control criteria and human trajectories are generated in controlled environments. A real-time multi-agent navigation approach, based on the reciprocal velocity obstacle, was presented by Berg et al. [5], which works very well in simulation, while is not able to perform so well under realistic situations, where uncertainties are present. Vasquez [6] proposed a prediction system based on a joint planning problem, assuming some optimal objectives to be sough by the pedestrians.

Using heuristics and geometric criteria, Foka and Trahanias [7] proposed a geometric-based method for human motion prediction that uses human motion intentionality in terms of goals.

In the present article, we apply geometrical based predictors such as the works of [8] and [9] that infer human motion intentions and afterwards predict human motion in a continuous space, according to the Social Force Model (SFM) [1], and the Extended SFM (ESFM) [10].

The autonomous vehicles community has been interested on including an accurate prediction into the planning scheme, publishing relevant works for the problem

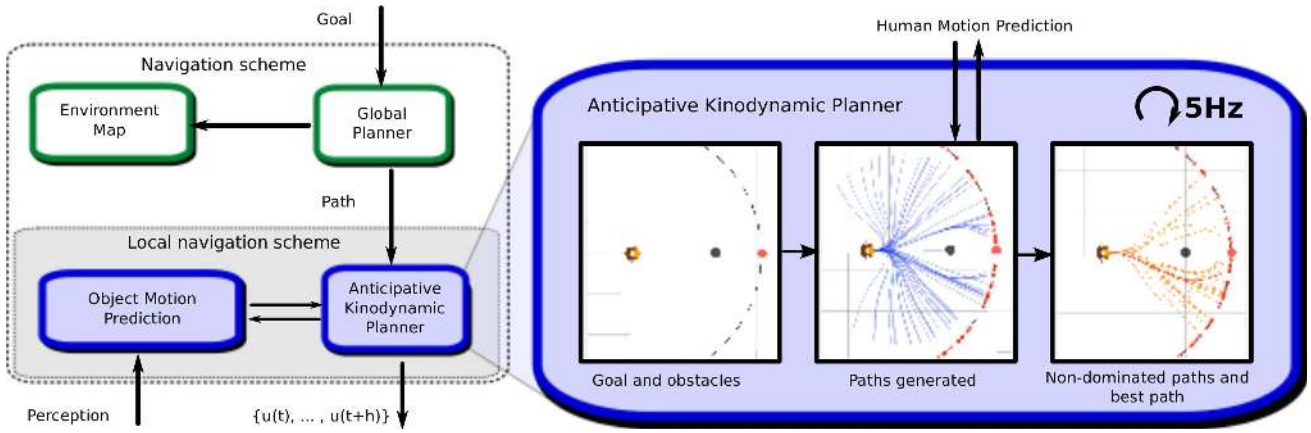


**Fig. 2** *Left*: Overview of the general AKP navigation scheme. The blue boxes are our contributions. *Right*: scheme of the procedure to calculate the best path.

studied in this paper. Gindele et al. [11] used an occupancy filter for prediction and Bahram et al. [12] proposed a discrete set of behaviors for decision making by a nonlinear model predictive approach in freeway scenarios.

The literature on planning is vast and extensive. Potential fields (PF) [13] were introduced for navigation algorithms, mostly for static environments, since a couple of decades ago. Despite their advantages, there exist many well known limitations, such as local minima or oscillations [14]. Several approaches try to overcome these limitations, such as [15], by using a randomized walking path when a local minimum is reached.

A special case of dynamic environments include robot navigation among humans. Sisbot et al. [16] proposed to model people as a summation of a PF. Svenstrup et al. [17] used a navigation PF and minimized its cost through people's Potential Fields. Fulgenzi et al. [18] calculated probabilities of collision and planned using a set of motion primitives under a limited time horizon, very similar to our approach. A pure *reactive* approach based on PF (SFM) has been proposed by Ferrer et al. [19], where their plans are socially aware in human environments. We will present in this paper a solution to overcome the well-known limitations of PFs, while maintaining their benefits, using our early work [20, 21].

The dynamic window approach by Fox et al. [22] and other velocity constrained approaches, such as the work of Simmons [23], permitted to consider obstacles and collisions. Unfortunately, they suffer from local minima as well. Approaches combining a DWA with a global planner as proposed by Brock and Khatib [24] solve the problem by introducing a global function. Our approach also relies on a global planner and repeatedly calculates a local solution.

Sampling-based techniques have become quite popular when solving planning problems. They may take into account kinematic and dynamic constraints such as [25] and [26], which is very appropriate for dynamic environments, and thus for our approach. Stachniss and Burgard [27] obtain a kinodynamic compliant trajectory by decoupling the problem into a search in space and a posterior optimization of the path satisfying the restrictions. Our approach integrates the search of a path avoiding obstacles as well as provides the inputs required to execute that trajectory considering kinodynamic constraints.

Data driven techniques have been used extensively to model human motion as well as robot planning. Bennewitz et al. [28] proposed a place-dependent prediction method in which they analyze a collection of people's trajectories in an indoor environment. Their method clusters motion trajectories by using the Expectation-Maximization algorithm, and classifies new observed trajectories.

Inverse Reinforcement Learning-based approaches [29–31] can provide good solutions by predicting social environments and planning through them. The work by Ziebart et al. [29] uses both place dependences and geometric criteria. The authors proposed to use a reward function to generate the optimal paths towards a destination. This method can also be used either for modeling of route preferences or for inferring destinations.

Kim et al. [32] formulated the problem as a Gaussian Processes Regression by low-rank matrix approximation. Chen et al. [33] proposed a Deep Reinforcement Learning architecture to predict human motion and robot collision avoidance.

A joint calculation of people's path and the robot path has been proposed by Trautman et al. [34], using Gaussian processes and a distance-based *interaction po-*

*tential* between people. This method is able to provide an approximation to an anticipative behavior finding the less occupied robot trajectory, but unlike our approach, we are able to quantify the magnitude of the alterations to nearby pedestrians and plan accordingly.

Finding a cost function to correctly characterize robot navigation among people may become an ill-posed problem, *e.g.* local minima, etc. Kuderer et al. [35] have addressed this problem and proposed a set of homotopically distinct trajectories. Sharing this same goal, we tackle this problem by proposing a multi-objective cost function where we optimize different independent criteria, such as the distance to the goal or the cost to navigate for pedestrians.

In this work, we present a multi-objective function that considers different and independent objectives, such as robot cost, nearby people cost, distance to goal, etc. Additionally, this technique permits us to correctly handle and compare different objectives into a single and well-posed function. Multi-Objective Optimization (MOO) methods [36–38] have inspired a solution to the minimization problem that selects the best path.

## 3 State-space formulation

We consider that both robots and people move in a two-dimensional space which represents urban environments. Let $\mathcal{X}$ denote the workspace and $\boldsymbol{x} \in \mathcal{X}$ describes the position $\boldsymbol{x} = [x, y]^\top$ for moving objects, including people and robots.

The configuration space $\mathcal{C}$ is defined as a pose $q \in \mathcal{C}$ in 2D, where $q = [x, y, \theta]^\top$ is the position and orientation of an object.

We define the augmented phase space $\mathcal{S}$ for the kinodynamic treatment of the states, that considers the first order derivative plus time. The action $u \in \mathcal{U}$ modifies the state $s$, and corresponds to the acceleration calculated by the the ESFM (Sec. 4.2). Similarly, $U = \{u\}$ is a set of inputs $u \in \mathcal{U}$, that characterize a trajectory.

In particular, let the state of a person be defined as $s_p = [x, y, v_x, v_y, t]^\top$. It corresponds to a double integrator, a free moving particle subject to the following differential equation in the continuous domain:

$$\dot{s}_p(s_p, u_p) = \begin{bmatrix} v_x \\ v_y \\ a_x \\ a_y \\ 1 \end{bmatrix}. \tag{1}$$

Likewise, the robot's state $s_r = [x, y, \theta, v, \omega, t]^\top$ is a unicycle model with nonholonomic constraints:

$$\dot{s}_r(s_r, u_r) = \begin{bmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \omega \\ a_v \\ a_\omega \\ 1 \end{bmatrix}. \tag{2}$$

### 3.1 Joint state-space

We define a joint state of the system that takes into account the robot state and the states for all the pedestrians in the scene. Although the dimensionality of the problem grows considerably, it is a mandatory to consider the reactions of nearby people to the robot's actions.

The joint state space $\mathcal{S}$ consists of $\mathcal{S} = \mathcal{S}_r \times \bigcup \mathcal{S}_{p_i}$, which considers the robot phase space $\mathcal{S}_r$ and the union of every person's phase space $\mathcal{S}_p$. Correspondingly, the joint state $s \in \mathcal{S}$ is defined as $s = [s_r, s_{p_1}, \ldots, s_{p_N}]^\top$.

The time variable $t = t(s)$ is equal for all the $s$ states. We will refer to $S = \{s\}$ as a set of states $s \in \mathcal{S}$ and $U = \{u\}$ a set of inputs $u \in \mathcal{U}$.

## 4 Prediction

Intuitively, prediction methods are of great importance for a better robot navigation. Unfortunately, most of the present approaches separate planning and prediction in order reduce complexity, and they do not consider the effect of moving objects in the robot navigation. In our method we take into account these effects and we will later present a simple method that combines prediction and planning in the same scheme. We analyze human trajectories under the influence of dynamic semi-crowded and heterogeneous environments, where there are moving people, moving objects, robots and also static obstacles.

In this section we explain how we compute the trajectory prediction of all the moving objects (mainly people). First we briefly explain how the human motion intentionality is predicted. Then we describe the Extended Social Force Model (ESFM) and how to apply it to the robot. Finally we explain how to compute the trajectory prediction using the ESFM.

### 4.1 Intentionality Prediction

The main purpose of the intentionality predictor is to infer which is the most expectable destination that a person is walking to.

In brief, we present the basic formulation necessary to analyze real trajectories. Let

$$\boldsymbol{X}_n(t) = \{\boldsymbol{x}_n(t_0), \boldsymbol{x}_n(t_0 + \Delta t), \ldots, \boldsymbol{x}_n(t)\} \quad (3)$$

be a set of positions (people detections) where as stated before, each point $\boldsymbol{x}_n(t) = [x(t), y(t)]_n$ is the position at time $t$ of the $n$th trajectory with respect to the world reference frame.

Moreover, we define a set of destination positions $\boldsymbol{D} = \{D_1, D_2, \ldots, D_M\}$, that indicates interesting positions in a scene that people might go to.

We infer the probability of the destination $P(\mathcal{D}_n = D_m | \ \boldsymbol{X}_n(t))$ using a naive Bayes classifier, where $\mathcal{D}_n(t)$ is the inner intention for the $n$th person to reach the destination $D_m$, and can be any of the destinations $\boldsymbol{D}$ in the scene. For further details, see [8].

## 4.2 Extended Social Force Model

In this work we use the Extended Social Force Model (ESFM) [10] for predicting human motion. This method can also be used for navigation purposes since it provides a realistic model describing interactions among humans in typical social environments [9].

The ESFM considers humans and robots as free particles in a 2D space abiding the laws of Newtonian mechanics, and is based on attractors and repulsors. The attraction forces assume that the pedestrian $n$ tries to adapt his or her velocity within a *relaxation time* $k^{-1}$,

$$\boldsymbol{f}_n^{goal}(\mathcal{D}_n) = k\big(\ \boldsymbol{v}_n^0(\mathcal{D}_n) - \ \boldsymbol{v}_n\big), \quad (4)$$

where $\boldsymbol{v}_n^0(\mathcal{D}_n)$ is the desired velocity vector to reach her goal according to the intention $\mathcal{D}_n$, and $\boldsymbol{v}_n$ is the current velocity.

The repulsive interaction forces are defined as follows:

$$\boldsymbol{f}_{n,z}^{int} = a_z e^{(d_z - d_{n,z})/b_z}\, \hat{\boldsymbol{d}}_{n,z}, \quad (5)$$

where $z$ is either a person, a robot, or a static object in the environment. For each kind of interaction force corresponds a set of force parameters $\{k, a_z, b_z, \lambda_z, d_z\}$. The distance $d_{n,z}$ from the person $n$ to the target $z$ and $\hat{\boldsymbol{d}}_{n,z}$ is the unity vector $z \to n$.

Accordingly, the resultant force is the summation

$$\boldsymbol{f}_n = \ \boldsymbol{f}_n^{goal}(\mathcal{D}_n) + \sum_{j \in P \setminus n} \boldsymbol{f}_{n,j}^{int} + \sum_{o \in O} \boldsymbol{f}_{n,o}^{int} + \sum_{r \in R} \boldsymbol{f}_{n,r}^{int}, \quad (6)$$

where each target on the scene, either a person, or an obstacle, or a robot, contributes to $\boldsymbol{f}_n$. In Fig. 3 is depicted an example of the corresponding forces actuating in a scene. This force is transformed into an acceleration, and thus, an action $u_n = \boldsymbol{f}_n / m_n$ that takes into account the mass $m_n$ of the $n$th target.

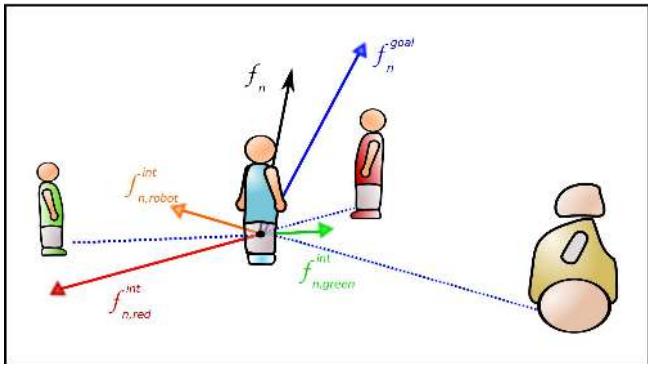


**Fig. 3** Social forces actuating in the scene for the person $n$ (in blue), which is attracted to a destination while other interactions take place with a robot and two pedestrians.

## 4.3 SFM applied to the robot

The objective is to treat the robot as a free moving particle in the space, similarly to a person as explained above. Unfortunately, nonholonomic constraints reduce the robotic platform mobility. We need to bridge the gap and provide an adjustment that permits the robot being compatible with the ESFM. The resultant robot force $\boldsymbol{f}_r = \boldsymbol{f}_{r||\theta} + \boldsymbol{f}_{r\perp\theta}$ consists of a component in the translation direction $\boldsymbol{f}_{r||\theta}$, which directly transforms into a translational acceleration and an orthogonal force $\boldsymbol{f}_{r\perp\theta}$ that does not contribute to the robot translation. In order to solve this, the robot rotation acceleration is computed considering the orthogonal force component in the following way:

$$\tau_r = \boldsymbol{r} \times \ \boldsymbol{f}_{r\perp\theta} + k_\tau \omega, \quad (7)$$

where $\boldsymbol{r}$ is the vector radius of our platform, oriented to $\theta$ and $k_\tau$ is a damping factor in order to avoid oscillations.

## 4.4 Trajectory Prediction using the ESFM

We formulate a prediction trajectory for a time horizon $t + h$. Let $\hat{\boldsymbol{s}}_n = [\hat{\boldsymbol{x}}_n, \hat{\boldsymbol{v}}_n]$ be the augmented state of position and velocity, and the continuous-time stochastic dynamics

$$d\hat{s} = \dot{\hat{s}}(\hat{s}, u)dt + g(\hat{s}, u)dW(t), \quad (8)$$

where $dW(t)$ is a Wiener process $(W(\Delta t) - W(0)) \sim \mathcal{N}(0, \Delta t I)$, $\dot{\hat{s}}(\hat{s}, u)$ is the deterministic part of the transition function, see (1), and $g(s, u)$ is the stochastic part, which becomes a Gaussian random variable $\mathcal{N}(0, \hat{\Sigma}_s)$ for the discretized problem at time interval $\Delta t$ in this particular case.

**Algorithm 1** MH trajectory prediction

1: **for** $t' = t, \dots, t+h$ **do**
2: **for** $n = 1, \dots, N$ **do**
3: **if** $\hat{\boldsymbol{x}}_n(t') \notin \mathcal{D}_n$ **then**
4: $\hat{\boldsymbol{s}}_n(t'+\Delta t) = \hat{\boldsymbol{s}}_n(t') + \int_{\Delta t} \dot{\hat{\boldsymbol{s}}}_n$
5: $\hat{\Sigma}_n(t'+\Delta t) = \hat{\Sigma}_n(t') + \hat{\Sigma}_s$
6: **end if**
7: **end for**
8: **end for**

For prediction purposes, we calculate the covariance $\hat{\Sigma}_n(t)$, however, this uncertainty diffusion is just an approximation being sometimes not consistent with reality, e.g., an abrupt change on intentionality. In practice, we chose to propagate deterministic trajectories and deal with uncertainty thought an MPC architecture (see Sec. 5.1).

Accordingly, the set of predicted states is

$$\hat{S}_n(t+h) = \{\hat{s}_n(t+\Delta t), \hat{s}_n(t+2\Delta t), \dots, \hat{s}_n(t+h)\} \quad (9)$$

$$\hat{\Sigma}_n(t+h) = \{\hat{\Sigma}_n(t+\Delta t), \hat{\Sigma}_n(t+2\Delta t), \dots, \hat{\Sigma}_n(t+h)\}. \quad (10)$$

As shown in Alg. 1, all the present targets in the scene propagate simultaneously and the expected propagations are used in the next iteration. Once a $n$th target succeeds on its intention $\mathcal{D}_n$ to reach its corresponding destination $D_m$, it remains idle waiting for the rest of the targets to complete their trajectories or until the time horizon $h$ expires.

The ODE appearing in Alg. 1 line 4 is solved using integration methods, furthermore, it is subject to a limitation in velocity $\|v\| < v_{max}$. In Fig. 4 is depicted a simple trajectory for two people, where the future position and covariance of each person is calculated simultaneously. We will discuss later how to incorporate this prediction technique into the planning scheme.

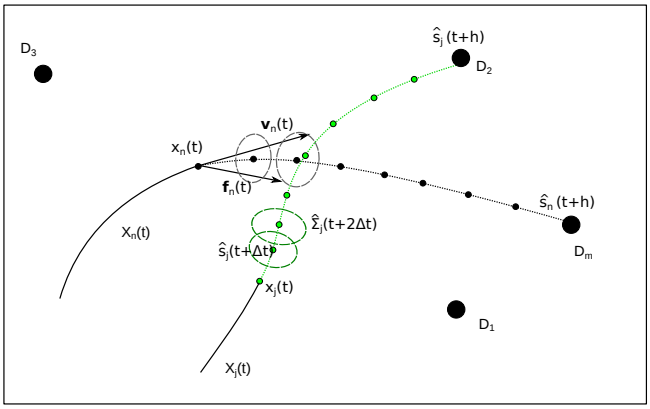


**Fig. 4** Trajectory prediction for two people at time $t + 2\Delta t$.

**Algorithm 2** AKP$(q_r^{goal}, s_{ini}, t_{horizon}, K)$

1: Initialize $\mathcal{T}(\mathcal{V}, \mathcal{E}) \leftarrow \{\emptyset\}$
2: $\mathcal{V} \leftarrow s_{ini}$
3: $\{q_{p_i}^{goal}\}$ = PEOPLE_INTENTIONALITY$(\boldsymbol{X}(t), \boldsymbol{D})$
4: **for** $j = 1$ to $K$ **do**
5: $[q_r^g, \beta]$ = SAMPLE$(q_r^{goal})$
6: $s_{parent}$ = FIND_NEAREST_VERTEX$(q_r^g, \mathcal{T}, \beta)$
7: $[U^{new}, S^{new}]$ = EXTEND$(s_{parent}, q_r^g, \{q_{p_i}^{goal}\})$
8: $\boldsymbol{J}^{new}$ = COST_TO_GO$(U^{new}, S^{new}, q_r^{goal}, \mathcal{T})$
9: **if** NO_COLLISION$(\mathcal{S}_{new})$ **then**
10: $\mathcal{V} \leftarrow \mathcal{V} \cup \{S^{new}, \boldsymbol{J}^{new}\}$
11: $\mathcal{E} \leftarrow \mathcal{E} \cup \{U^{new}\}$
12: **end if**
13: **end for**
14: **return** MINIMUM_COST_BRANCH$(\mathcal{T})$

## 5 Anticipative Kinodynamic Panning

The scheme of our planning approach was presented on the introduction section. Fig. 2-*Right* depicts the basic procedure to calculate the robot path:

- Get and track obstacles and people in the scene.
- Calculate a large set of paths by using a modified version of a kinodynamic RRT.
- Choose a set of possible path candidates and then calculate the best path.

We have chosen to use the ESFM to alleviate calculations, since other methods focus only on solving the Boundary Value Problem to connect a pair of poses and are not capable of capturing the interacting nature of the environment. An advantage of using the ESFM is that we are directly planning in the action space $u \in \mathcal{U}$, and thus, it is not necessary the use of a controller with respect to a given path.

The prediction module provides a good prior of the future trajectories of the dynamic obstacles in the scene. Our approach samples multiple paths that modify these priors and calculates the best trajectory that the robot can describe. We use the basic mechanics of the kinodynamic RRT [25] to extend paths, in order to generate a large set of feasible paths and chose the best path according to multiple minimization criteria.

Algorithm 2 requires four inputs: the goal $q_r^{goal}$, the initial state $s_{ini}$, the horizon time $t_{horizon} = t + h$, and the number of vertices $K$. The $q_r^{goal}$ provides the position and orientation of the final robot configuration. The initial state $s_{ini} \in \mathcal{S}$ contains the information of the robot state plus all people's states considered on the scene. The horizon time $t_{horizon}$ specifies the temporal window used to forecast the plan and the predictions.

The algorithm builds a tree $\mathcal{T}(\mathcal{V}, \mathcal{E})$ and returns the minimum cost branch. The edges $\mathcal{E}$ are the control inputs $u \in \mathcal{U}$, and the vertices $\mathcal{V}$ consist of the joint state

$s \in \mathcal{S}$ and the accumulated cost $\boldsymbol{J}$ to reach that vertex (see Sec. 6).

The general algorithm scheme can be seen in Alg. 2, which is quite similar to the original RRT, although we have added some modifications, that are described in the following subsections.

### 5.1 Horizon time

The horizon time parameter sets the amount of time that the planner forecasts in order to obtain a path similar to a model predictive control (MPC). Although the inputs $\{u_r(t), \ldots, u_r(t+h)\}$ are calculated, only the first input command is executed and a new set of inputs will be calculated in the next iteration. In practice, MPC is a very successfull technique to deal with unknown system modeling, as discussed in Sec. 4.4.

### 5.2 Sampling

The horizon time bounds the region of exploration $\mathcal{C}_r$ to a circle radius equal to $h \cdot v_{max}$ as depicted in Fig. 5. Since there is a strong time restriction, the robot goal $q_r^g$ is sampled using a Gaussian distribution, over the boundary of $\mathcal{C}_r$ to ensure that the paths generated indeed expand $\mathcal{T}$. If the number of nearby obstacles on the scene is high, the $q_r^g$ sampling distribution is widespread by augmenting linearly the Gaussian covariance w.r.t number of people.

### 5.3 RRT extend: ESFM steering heuristic

As explained before, the prediction algorithm used by our approach is based on the ESFM. We additionally make use of this model in order to build a steering heuristic that permits us to connect a pair of poses $q_{ini}$ and $q_f$, in a computationally fast manner.

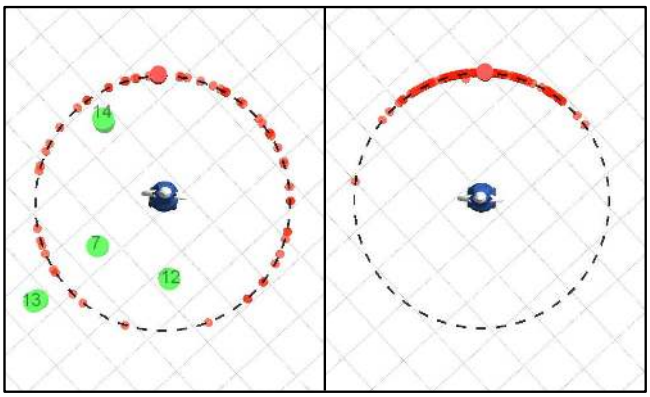


**Fig. 5** Random goals $q_r^g$ distribution, on the *right* there are no people in $\mathcal{C}_R$ and search is concentrated on the goal direction. On the *left* the density of nearby people on the scene is higher, and thus, the $q_r^g$ sampling distribution is widespread.

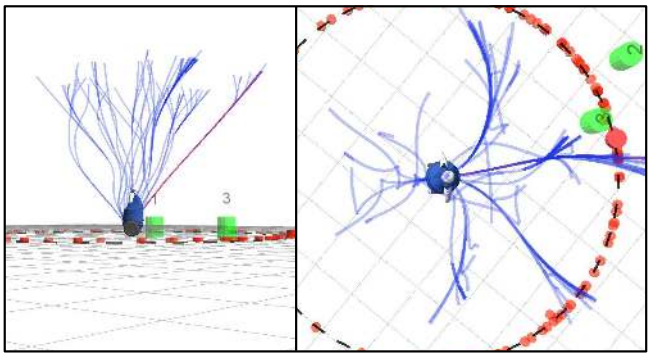


**Fig. 6** Tree $\mathcal{T}$ of paths in the space $\mathcal{X} \times time$. On the *left*, the $z$ axis represents time. On the *right* projection of $\mathcal{T}$ in $\mathcal{X}$.

We calculate the resultant robot force $\boldsymbol{f}_r$ using (6), which takes into account its environment $[s_{p_1}, \ldots, s_{p_N}]$, while at the same time tries to reach the given random goal $q_r^g$, that represents its inner intentionality $\mathcal{D}_r$. In Fig. 6 is depicted an example of depth propagation, where each path tries to reach its corresponding random goal.

Since the action space is limited to $\mathcal{U}_r$ and we cannot control the environment, we need to infer human intentions. As proposed in [8], we calculate the most expectable $q_{p_i}^{goal}$ for every person on the scene (line 3 in Alg. 2).

The EXTEND function is depicted in Alg. 3, where it firstly propagates the robot state $s_r(s_{parent})$ accordingly to $u_r^{new}$ and integrates the differential constraints (2) by using integration methods in the time interval $\Delta t$. Then, for every person on the scene, and if the person has not reached its inferred goal $q_{p_i}^{goal}$ (line 5 in Alg. 3), an action $u_{p_i}^{new}$ is calculated depending on the rest of the dynamical obstacles on the scene and the new robot state $s_r^{new}$ (line 6 in Alg. 3) in a cooperative way.

The EXTEND function calculates consecutively for each instant of time the propagation for all people, and propagates the state of the system until $t_{horizon}$ is reached or the robot has reached the random goal $q_r^g$. The EXTEND function closely resembles Alg. 1, but introducing the calculation of the new robot state $s_r^{new}$, in which the computed planning action $U^{new}$ is seamlessly integrated with the prediction algorithm in an anticipative approach.

As explained in previous sections, steering methods present excellent characteristics to drastically reduce the computational cost of kinodynamic planners, both in the EXTEND function as well as the COST-TO-GO. However, there is an important drawback for such methods:

**Algorithm 3** EXTEND$(s_{parent}, q_r^g, \{q_{p_i}^{goal}\}, \boldsymbol{\beta})$

**while** $t(s_{parent}) < t_{horizon}$ & $s_r(s_{parent}) \not\subseteq q_r^g$ **do**
  $u_r^{new} = \boldsymbol{f}_r(s_{parent}, q_r^g)/m_r$
  $s_r^{new} = s_r(s_{parent}) + \int_{\Delta t} \dot{s}_r$
  **for** $i = 1, \ldots, N$ **do**
    **if** $s_{p_i}(s_{parent}) \not\subseteq q_{p_i}^{goal}$ **then**
      $u_{p_i}^{new} = \boldsymbol{f}(q_{p_i}^{goal}, s_{parent}, s_r^{new})/m_i$
      $s_{p_i}^{new} = s_{p_i}(s_{parent}) + \int_{\Delta t} \dot{s}_{p_i}$
    **end if**
  **end for**
  $s_{parent} = [s_r^{new}, s_{p_1}^{new}, \ldots, s_{p_N}^{new}]$
  $S^{new} \leftarrow S^{new} \cup s_{parent}$
  $U^{new} \leftarrow U^{new} \cup [u_r^{new}, u_{p_1}^{new}, \ldots, u_{p_N}^{new}]$
**end while**
**return** $[S^{new}, U^{new}]$

the set of trajectories that can be obtained is highly dependent on the environment configuration, and thus, we may be biasing the search space into only a subset of it.

Since we need a set of paths that covers as much of the solution space as possible, the sampling of $q_r^g$ expands the paths in many different directions, and additionally, we propose to overcome this problem by introducing randomness into the steering function, the ESFM. We aim to introduce some randomness in the generation of robot trajectories by attributing random behaviors to the robot when trying to reach a destination $q_r^g$. The resultant force

$$\boldsymbol{f}_r = \beta_1 \ \boldsymbol{f}_r^{goal} + \beta_2 \ \boldsymbol{f}_{people}^{int} + \beta_3 \ \boldsymbol{f}_{obstacles}^{int} \tag{11}$$

determines the generated trajectory, being $\boldsymbol{\beta} = [\beta_1, \beta_2, \beta_3]$ a set of values associated to each sampled trajectory. Sampling $\boldsymbol{\beta}$ will enrich the generation of paths to be evaluated by the planner.

### 5.4 Cost functions

As stated before, we aim to obtain a navigation algorithm that considers different cost functions. In this subsection these costs functions are defined. The cost to reach a goal $J_d$, is defined as

$$J_d(S, s^{goal}) = \sum_{t=t_{ini}}^{t_{end}} \|\boldsymbol{x}_r(t) - \boldsymbol{x}^{goal}\|^2, \tag{12}$$

where we obtain the accumulated square distance value from the initial state at time $t_{ini}$ in the set of states $S$, to the final state at time $t_{end}$.

The cost orientation $J_{or}$ expresses the difference between the desired orientation and the current orientation

$$J_{or}(S, s^{goal}) = \sum_{t=t_{ini}}^{t_{end}} \|\theta_r(t) - \theta^{goal}\|^2, \tag{13}$$

representing the accumulated distance to the desired goal orientation $\theta^{goal}$.

We additionally measure the cost associated to the robot control $J_r$ in the following way

$$J_r(U) = \sum_{t=t_{ini}}^{t_{end}} \|u_r(t)\|^2, \tag{14}$$

that sums the robot inputs $u_r$ throughout the calculated trajectory.

Similarly, we define the cost function for the pedestrians $J_p$ as

$$J_p(U) = \sum_{t=t_{ini}}^{t_{end}} \sum_{i=1}^{N} \|u_{p_i}(t)\|^2, \tag{15}$$

where the inputs $u_{p_i}$ due to the robot influence to other pedestrians are summed over time for each person while walking towards their goals.

We also take into account the cost produced by nearby obstacles to the robot $J_o$ as

$$J_o(U) = \sum_{t=t_{ini}}^{t_{end}} \sum_{i=1}^{O} \|u_{o_i}(t)\|^2, \tag{16}$$

where again we consider the perturbation occasioned to the robot due to nearby obstacles $u_{o_i}$, since we want to avoid collisions. It will be explained below how to combine these different metrics in order to obtain the best path.

### 5.5 Cost-to-go

We have defined different cost functions, and thus a multi-objective problem, that can also be used to evaluate the cost-to-go from a state to another state.

Euclidean distance is a good metric in a geometrical planning problem, nevertheless, it is not so efficient in kinodynamic planning schemes and cost-to-go functions work better, as we will demonstrate later.

Many works [39–41] calculate the cost-to-go in the absence of obstacles. In our case, we consider dynamical obstacles and include these interactions, thanks to the ESFM. In Fig. 7 is drawn the process to calculate the nearest vertex in the search tree. We extend a virtual path towards the goal $q_{new}$ from each of the vertices, and calculate the accumulated cost to reach it plus the previous cost.

We are highly concerned to provide a good representation of the solution space if we want to obtain a good "best trajectory" solution. We will discuss in Sec. 7 the coverage of the solution space explored using a cost-to-go function to find the nearest vertex and compare it to an Euclidean approach. It will be demonstrated later the performance of our contributions, either in simulations as well as in real experiments.

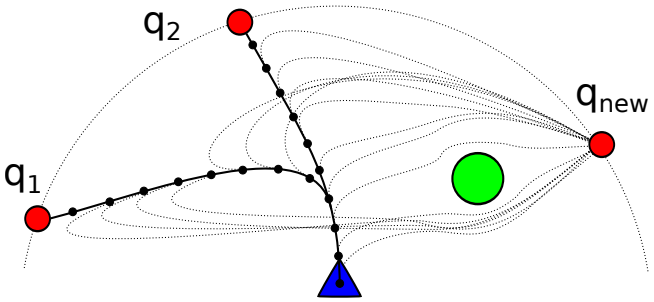


**Fig. 7** Scheme of the cost-to-go function to reach $q_{new}$ from each of the vertices in the tree.

## 6 Multi-objective optimization

Multi-objective functions are built after a scalarization of different variables, expressed in a function like $J = c_1 \cdot J_1 + c_2 \cdot J_2 + \ldots + c_I \cdot J_I$, if the $c_i$ parameters are known, which is not always the case. Each of the $J_i$ may be expressed in different units, and they are projected into the real space $J : \mathbb{R}^I \rightarrow \mathbb{R}$. In our previous work [20] we followed an scalarization approach for unknown $c_i$. It turns out that even learning a set of weight parameters, they are only valid in a limited number of scenarios meeting similar conditions. In this section we will describe a procedure to maintain the same set of parameters for all our testing environments.

There are multiple objectives to be minimized expressed as a vector function consisting of $I$ components

$$\boldsymbol{J}(S, s_{goal}, U) = [J_d, J_{or}, J_r, J_p, J_o], \tag{17}$$

in our case $I = 5$, and we should use different and independent criteria instead of a single-objective composed of different variables. In our problem, these functions are described in (12)-(16).

To this end, inspired by a multi-objective optimization technique, we solve the problem of finding a set of optimal solutions. The weighted-sum method is a popular and simple approach, although it also presents limitations. We propose to avoid the scaling effect by normalizing the objective functions according to:

$$\bar{J}_i(X) = \text{erf}\,(\frac{x - \mu_x}{\sigma_x}). \tag{18}$$

The variables $\mu_x, \sigma_x$ are estimated after the tree $\mathcal{T}$ is built. The multi-objective cost function becomes a single-objective by applying a three step calculation: first, the individual costs for each criteria to be considered $\boldsymbol{J} : \mathcal{S} \times \mathcal{U} \rightarrow \mathbb{R}^I$. Second, a normalization to $(-1, 1)$ for each of the costs (18), and finally, a projection via a weighted sum $J : \mathbb{R}^I \rightarrow \mathbb{R}$ as follows

$$J(S, s_{goal}, U) = \sum_i w_i \bar{J}_i(S, s_{goal}, U). \tag{19}$$

At the end, we have obtained a cost $J \in \mathbb{R}$, but after a normalization and a projection. We will demonstrate in Sec. 7.1 that the $\boldsymbol{w} = [w_d, w_{or}, w_r, w_p, w_o]$ parameters are valid for a large number of different scenarios, which is one of our initial motivations.

### 6.1 Identifying the non-dominated set

Our algorithm is inspired in a multi-objective approach in order to calculate the best trajectory among the paths calculated during the planning step. Prior to provide a solution, we seek to obtain the set of *non-dominated* solutions depending on the different cost functions $J_i$ defined in Sec. 5.4.

In general, a solution $x_1$ is said to dominate other solution $x_2$ if both of the following conditions are true:

1. $J_i(x_1) \leq J_i(x_2), \quad \forall i \in I$
2. $J_i(x_1) < J_i(x_2)$ for at least one $i \in I$

When both conditions are asserted, then $x_1$ dominates $x_2$ and we formulate it mathematically $x_1 \preceq x_2$. If we obtain the set of *non-dominated* solutions, then any pair of solutions do not dominate each other. Ideally, if we could calculate all the solutions in the solution space, then the *non-dominated* set would be *Pareto-optimal set*. No solution of this set is by itself better than other, we have only discarded those solutions that are dominated.

Obtaining a set of *non-dominated* solutions is of great importance [36,37], since the AKP algorithm uses this information to chose a better solution. The selection of the *non-dominated* is done before any scalarization or normalization and this fact is useful specially when dealing with non-convex sets of solutions since it allows us to discard local-optimal solutions in front of global ones, before reprojecting into $J \in \mathbb{R}$.

## 7 Simulations

Robot experimentation is a delicate matter specially when there are people involved. For this reason, we believe that it is mandatory to validate any robotic system

in a simulated environment before any real interaction with people takes place.

The simulation framework used is a custom project built around ROS. Human motion is simulated using the work presented in [10], where we generate people tracks, that is, a sequence of positions over time with constant *id*, and model them as random variables. The robotic platform is simulated according to a unicycle model that is controlled by the AKP calculations. The simulation scenario as well as the AKP code can be found *soon* at the project web-page `http://www.iri.upc.edu/groups/lrobots/akp`

We have carried out all the simulations in a Intel Core2 Quad CPU Q9650 @ 3.00GHz and memory 3.8 GB, at an average rate of 5Hz. The hardware used for simulations is similar to the PC on-board the real robot. The simulated scenarios are as follows: the robot receives a query to a goal, in different scenarios that are built combining a different number of obstacles and pedestrians walking in the area, as can be seen in the supplemental material at the web-page.

In this section we will discuss, by using these different simulated scenarios, some of the most relevant topics regarding the AKP, such as the initial learning of the weighting parameters $\boldsymbol{w}$, the importance of a correct cost-to-go function compared to an Euclidean metric, and finally the multi-objective optimization performance with respect to a reactive approach.

### 7.1 Parameter learning

In order to correctly characterize the effect of the $\boldsymbol{w} = [w_d, w_{or}, w_r, w_p, w_o]$ parameters, we have used a Monte Carlo approach to sample the weights of the normalized costs, carrying out more than 20k simulations. We have used many different scenarios which consist of a variable number of people and obstacles, and we have calculated the costs associated to the sampled $\boldsymbol{w}$ parameters for each configuration.

The results of the simulations have been averaged in order to address the fact that the scenario is dynamic and the outcome for the same set of parameters can be different depending on the initial conditions, those are, the position of the simulated pedestrians and their corresponding destinations.

In Fig. 8 is depicted the costs and average cost for the distance and obstacle objectives, depending on the same parameter $w_{obs}$. As it can be seen, there is not a clear value for the weight cost that can minimize simultaneously the obstacle's and people's costs. In addition, there is a high variance in the results, since we are testing a highly dynamic environment.

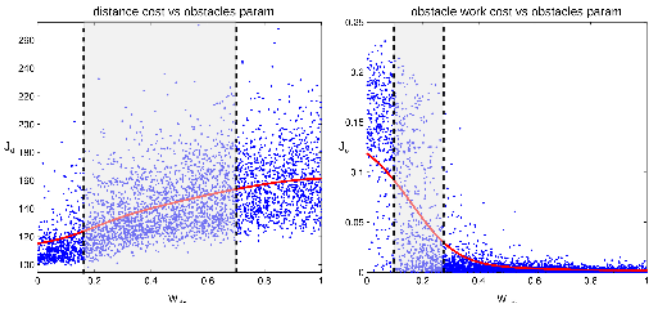


**Fig. 8** On the *left* distance cost and on the *right* obstacle cost w.r.t. the $w_{obs}$ weight parameter. Blue dots correspond to single experiments and the red line is the calculated expectation using a Gaussian likelihood. In both graphics appear the interval of the *acceptable* region, intersection around $w_{obs} = 0.2$.

For these reasons, we have used an heuristic method to select which are the most convenient values of $\boldsymbol{w}$. Each of the costs considered in this work is drawn as a function of a weight parameter, in the case of Fig. 8, the parameter is $w_{obs}$. We describe an *acceptable* region starting at 20% of the maximum cost value to the 80%. All the average costs are monotonic functions. The intersection, if possible, of the different regions leads to an approximate value for the parameter, after some adjusting by try and error.

The parameters used throughout the remaining of the work are $w_{distance} = 0.7$, $w_{orientation} = 0.4$ , $w_{robot} = 0.5$, $w_{ppl} = 0.3$ and $w_{obs} = 0.2$.

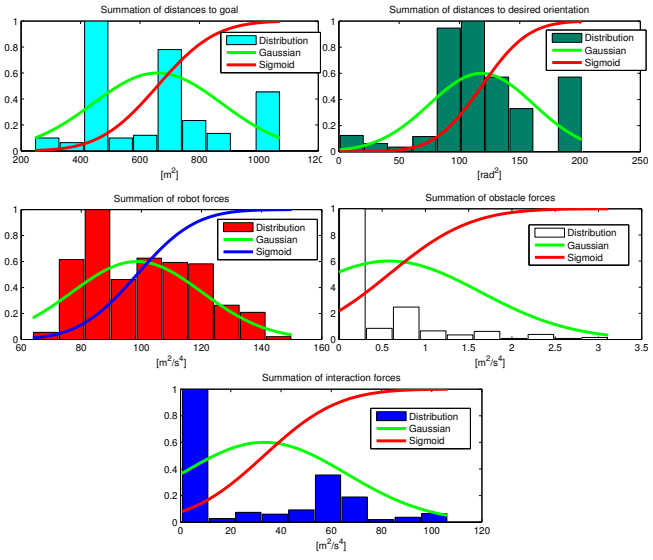


**Fig. 9** Distributions of the cost parameters $J_i$. Adaptively normalizes the different cost values into $(-1, 1)$.

In Fig. 9 we can observe the distribution of the cost parameters, prior to normalization, and the corresponding normalization function. The magnitudes of the variables are too different, and any attempt to scale them

and sum (former approach) represents a real challenge, if not an ill-posed problem.

We have observed an interesting feature after comparing the graphical results for the cost functions in different scenarios: the expectation curves obtained are very similar for the same cost in different scenarios, *e.g.* one person, two people and two obstacles, etc. In other words, they are not dependent on the number of interacting obstacles or people, but we can chose the same set of parameters $\boldsymbol{w}$ since the navigation outcome is relatively invariant to the scenario configuration, and thus, we can make use of the same weigh values in all situations for future experiments and simulations.

### 7.2 Coverage of the solution space

In this subsection we measure the coverage of the workspace comparing the em cost-to-go method with an Euclidean distance to select the closest vertex for FIND_NEAREST_VERTEX(). In Fig. 10-*left* is depicted two of the scenarios chosen, among many, to illustrate the problem. On the top, there is an obstacle free scenario and on the bottom an scenario with some obstacles. The tree generated by an Euclidean function, as formulated in [20], can be seen in Fig. 10-*right* and the tree generated by the cost-to-go metric and randomness in the ESFM is depicted in Fig. 10-*center*.

The area in red corresponds to an average of visited regions and the blue area corresponds to non visited. It is not possible to generate paths that can cover all the solution space since there is a strong time restriction $t_{horizon} = 5s$. We observe that there is no difference in the obstacle free scenario. Both approaches visit a similar area. However, in the scenario with some obstacles, there is a great difference: the Euclidean approach presents more difficulties to scape the obstacle area. The cost-to-go and the randomness to the resultant force make possible to explore a wider region of the solution space.

These qualitative measures need to be quantified. To this end, we have run another experiment, that measures the area visited in the obstacle scenario, while growing the number of vertices in the AKP.

The results are depicted in Fig. 11. Clearly, the cost-to-go and randomness approach outperforms the Euclidean distance, at least under this configuration. The computation overhead generated by using the cost-to-go metric to find the nearest vertex compensates when considering that to obtain the same area covered by the Euclidean approach, we require multiple times the number of vertices used.

### 7.3 Performance

We have conducted a large number of simulations in different environments as described above: with multiple obstacles, multiple people, very near obstacles while navigating with people, etc. We want to provide a challenging testing for diverse environments.

To illustrate this idea, we have performed more than $5k$ experiments per method, comparing our approach, the AKP using the cost-to-go distance, with the AKP using Euclidean metrics and a third approach: a pure reactive planning [19]. We have set the number of vertices to $K = 800$ for both AKP approaches. The planner was able to provide a path at a rate not lower than 5Hz.

All the objective cost functions are plotted in Fig. 12, averaged for the different scenarios analyzed, which are presented jointly. Both of our approaches clearly outperform in all objectives the reactive approach, which has demonstrated to behave badly in dynamic environments, since we often observed a *go-stop-go* strategy.

Both AKP behave similarly in simple scenarios, however the AKP with Euclidean metric presents a more "straight behavior" towards its goal, specially in complicated environments (lots of obstacles). Under this circumstances is where the AKP with cost-to-go really shines. The AKP is able to explore more solutions and to provide more candidates that result in a better choice for the best trajectory. Observing these results we can conclude that our algorithm performs successfully in many situations with the same set of parameters $\boldsymbol{w}$.

## 8 Experiments

In the second part of the testing, the real experimentation, we have evaluated the AKP in real scenarios with volunteers, as we believe that social robot navigation should be largely tested in real situations, since all the simulations have limitations. The two environments used for the experimentation are a controlled environment consisting of a limited number of obstacles and people, and a real environment were no instructions are given to pedestrians. The experiment environments are intended to be semi-crowded. Idle people are not considered as part of the dynamic obstacles.

The Tibi&Dabo robots are based on a two-wheeled Segway RMP200 platform. To perceive the environment they are equipped with two Hokuyo UTM-30LX 2D laser range sensors used to detect obstacles and people, giving scans over a local horizontal plane at 40cm above the ground, facing forward and backward. We use range information to navigate as well as to generate people detections needed for the AKP.

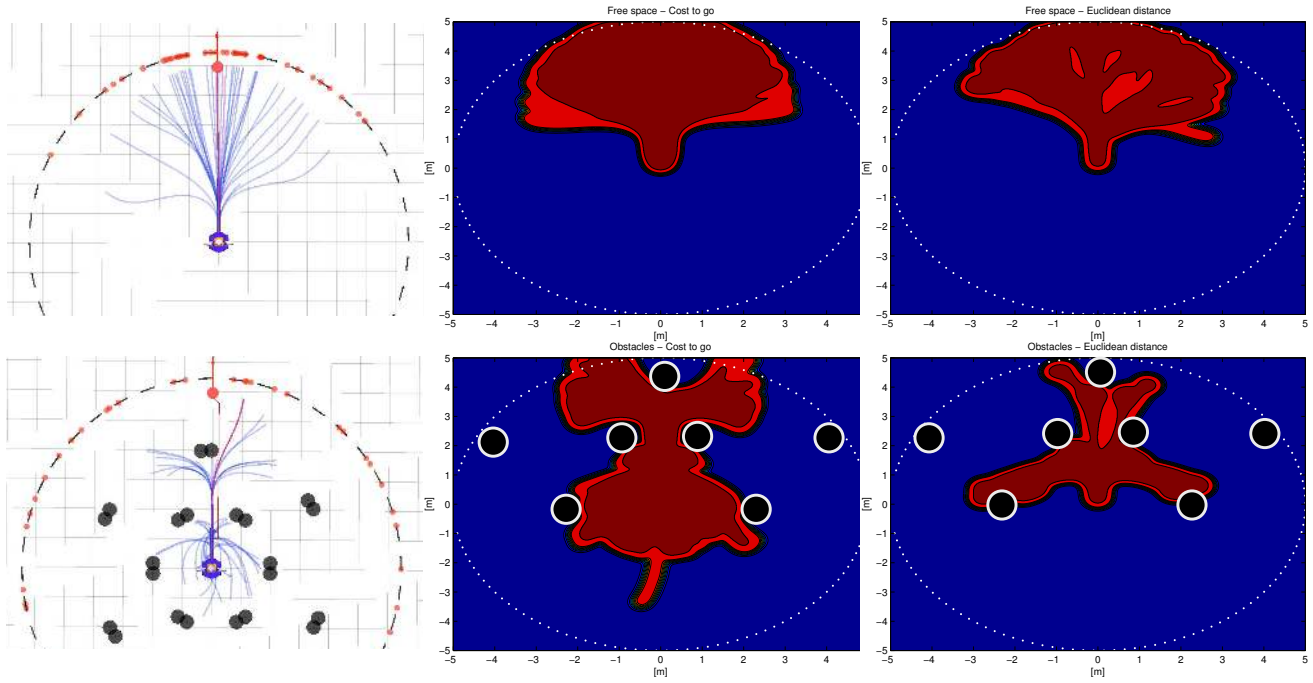


**Fig. 10** Coverage of the workspace according to the cost-to-go and randomness in the ESFM, and the Euclidean metric for the same number of vertices and obstacles. In red, typically visited areas averaged over a large number of iterations, and in blue, typically non visited regions. The gradient between red and blue represents rarely explored regions by the AKP.

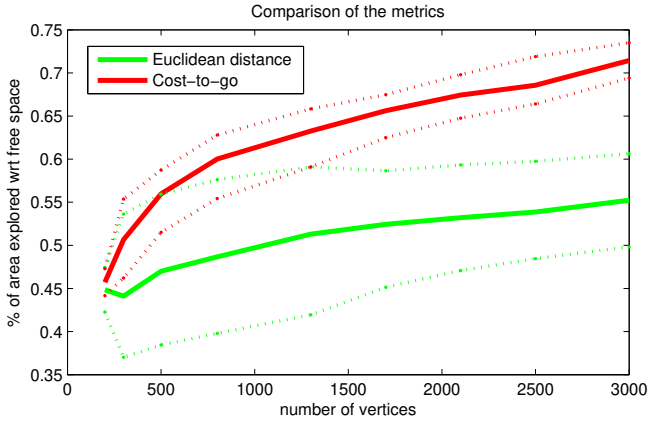


**Fig. 11** Coverage of the workspace with respect to the number of vertices used in a obstacle scenario. Results normalized to the coverage of a free obstacle scenario, being 1 a perfect coverage. Average and ± the standard deviation are plotted for bith methods.

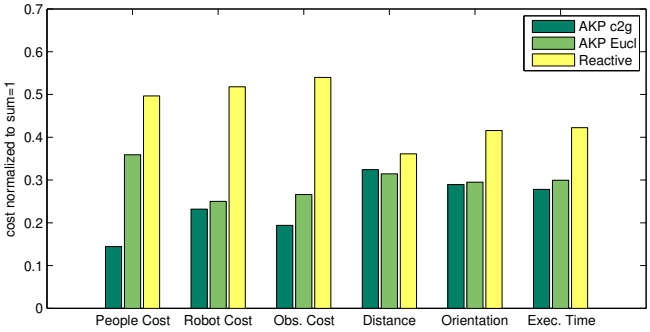


**Fig. 12** Results for the different cost functions $J_i$ for the AKP using cost-to-go function, compared with the AKP with Euclidean metrics, and a pure reactive approach. All results have been normalized to sum 1 for visualization reasons.

Using one of these platforms, we have conducted 52 experiments in a controlled environment, the *Facultat de Matemàtiques* (FME), which consists of an open space surrounded by some walls. The volunteers were told to walk towards a destinations in the scene, while simultaneously the robot performed a *go-to* query that entailed an interaction with the pedestrian. The idea behind these scenarios is to provide an isolated and unconditioned interaction between a person and the robot, and thus, analyze the corresponding results which are guaranteed to minimize other interactions or noise. We have tested three different scenarios: on the Fig. 13-*Left* is depicted the first scenario, where the volunteer, in green, is told to walk towards a destination in a straight line, while the robot approaches the person from behind, just in time to intersect with the volunteer path. A side-frontal interaction, see Fig. 13-*Center*, and a completely frontal interaction in Fig. 13-*Right* are the remaining experiments.

In order to compare the AKP method with other navigation methods, we have carried out the same setting of experiments for two additional methods. The first one is the *Reactive* approach [19] that we pre-

sented in the simulations section. This method makes use of the social force (ESFM) but is not able to plan in advance, which will be demonstrated later to be a great disadvantage in challenging scene configurations. The second one is the Dynamic Window Approach [22] implementation in the ROS navigation stack. We have configured it to consider as local obstacles a region centered at the robot of $10m \times 10m$, to be equivalent to the area considered by the AKP which is a circle of radius $6m$.

In Fig. 13 are depicted the results of the experiments. We have chosen to show two variables: the accumulated trajectory cost of people due to the interaction with the robot, that is, the direct impact of the robot towards its environment. The second one is the average time for the robot to reach the goal. Since all methods are compared under the same conditions, and that entails the same distance to goal, with this measure we can compare objectively the different approaches in terms of efficiency of execution. Both of the results appearing in Fig. 13 are normalized to sum the unity in all the bar diagrams in order to facilitate the visualization.

In the first scenario, the side-rear interaction, the results considering the people's costs show a better performance for the AKP approach than for the rest of the approaches. Considering time of execution, the AKP presents an slightly better performance. In the second scenario, the side-frontal interaction, the DWA is able to avoid people successfully since it can re-plan on time and their results obtained are on par with the AKP and far better than the reactive approach, that fails to avoid interaction with people. The time of execution, is better for the AKP, since it can avoid collision more efficiently than the other two approaches. In the third scenario we observe a clear winner, the AKP. When considering the people's cost, the AKP is able to reduce its impact towards the scene and execute its task successfully. The average time of execution is not considerably better for the AKP than the other approaches, but the fact is that our approach is really avoiding a possible collision while the other two approaches fail to re-plan on time and follow a too straight trajectory.

We can observe that the results are dependent on the configuration, the more complex the configuration is, the better relative results are obtained by our approach. We already demonstrated this statement in simulations, however it had to be verified in a real scenario.

We highly recommend to watch the videos at the project web-page for a more detailed idea of the outcome of the experiments, since movement is difficult to be captured in images or words.

In order to further validate our approach, we have tested it in a more complex scenario, where the robot receives *go-to* queries in a urban environment and more interactions take place. The testing is done under uncontrolled conditions, that is, no instructions are given to people in the scene while the robot tries to navigate. Two environments are tested, the FME where we carried out the comparison explained before, and the *Barcelona Robot Lab.* (BRL), which is a urban area in the university campus, when many students were there. The experiment's main purpose is to demonstrate the success of navigation queries in a real and uncontrolled environment. To this end, we performed and recorded the experience on a video. As stated before, simple configurations with few interactions do not represent a challenge for most navigation algorithms, however the scene can increase in complexity and more challenging configurations may appear. Is in these situations when the AKP approach might help us to avoid possible collisions or abnormal robot behaviors, and deal with these situations more successfully while minimizing its impact towards the pedestrians in the scene.

A sequence of one navigation experiment is depicted in Fig. 14, where we observe the interaction of several people with the robot at different instants of time and how the robot is able to avoid them successfully.

For more details, check the multimedia material at `http://www.iri.upc.edu/groups/lrobots/akp/`.

## 9 Conclusions

The main motivation of the present work, as stated in the introduction, was to minimize the robot impact towards all those nearby pedestrians, while navigating and executing tasks. Throughout this paper we have proposed an approach that is able to consider multiple objectives for robot navigation and demonstrated to be effective in real situations. In addition, the AKP approach has proven to foresee people's trajectories and their corresponding reactions to each of the planned actions, and act accordingly.

The solution trajectories, calculated in an RRT fashion, take into account kinodynamic and nonholonomic restrictions which are mandatory considerations for a realistic navigation in such highly time-variant scenarios like urban environments.

We have presented a seamlessly integrated prediction algorithm that builds predictions simultaneously with plans. We have demonstrated that a joint approach can be achieved without increasing much the system complexity, and we have obtained multiple benefits by integrating prediction and planning.

The cost for being anticipative is a more intensive processing since we must propagate the state of moving

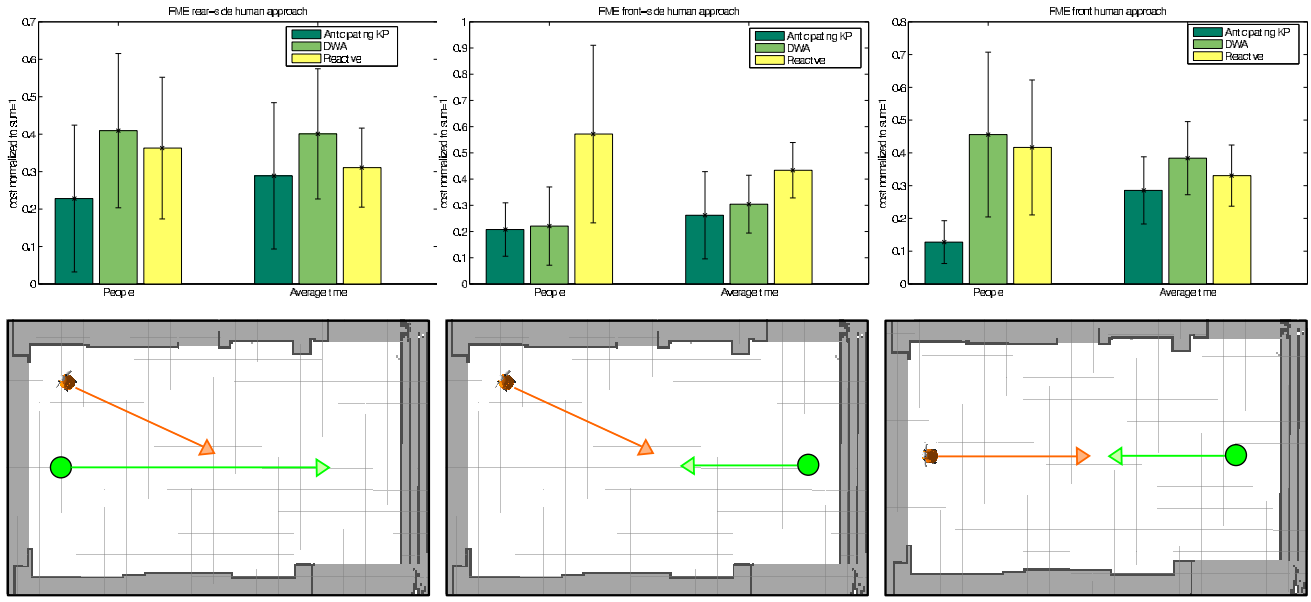


**Fig. 13** Results for different scenarios in the FME environment. On the top row is depicted the bar diagram for the people's costs and the average time of execution. Their corresponding standard deviation is drawn on each bar of the diagram. On the bottom row appears an scheme of each of the experiments.

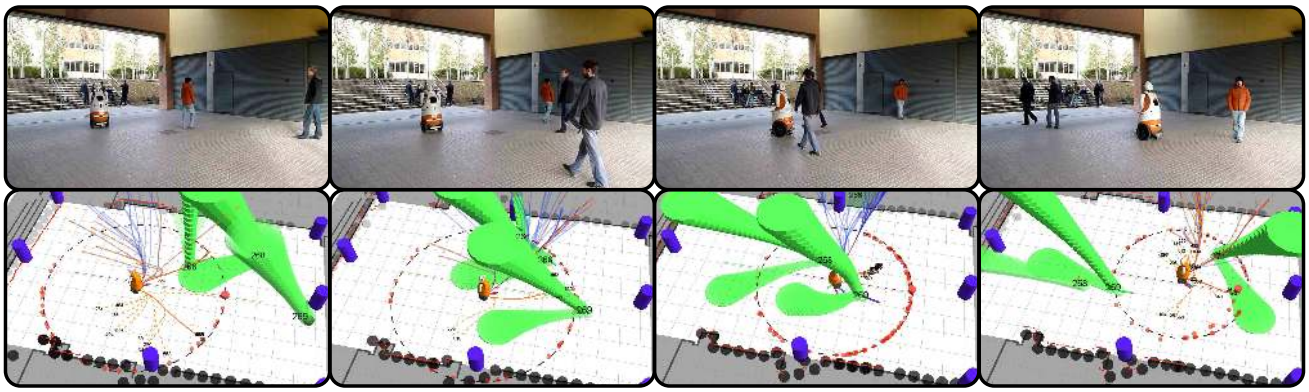

**Fig. 14** Sequence of a navigation experiment interacting with multiple pedestrians.

pedestrians accordingly to the robot propagation. The overhead generated by this feature has been demonstrated to be justified, since challenging and complex scenarios are more successfully solved by the AKP than other state of the art navigation methods.

The multi-objective approach studied in this paper has shed light into the construction of cost functions. If there are few parameters and costs involved, a direct scalarization of the costs may be fine, however we have shown that it might not hold when considering several costs. One contribution is that we have obtained a practically invariant set of navigation parameters that can be used in many different scenarios.

The theoretical approach presented is well supported by plenty of experiments and simulations. There are in addition plenty of multimedia material and *soon* all the code available at the web-page which better illustrates the results that have been discussed in this work.

## References

1. D. Helbing and P. Molnár, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, no. 5, p. 4282, 1995.
2. M. Moussaïd, D. Helbing, and G. Theraulaz, "How simple rules determine pedestrian behavior and crowd disasters," *Proceedings of the National Academy of Sciences*, vol. 108, no. 17, pp. 6884–6888, 2011.
3. F. Zanlungo, T. Ikeda, and T. Kanda, "Social force model with explicit collision prediction," *EPL (Europhysics Letters)*, vol. 93, no. 6, 2011.
4. G. Arechavaleta, J.-P. Laumond, H. Hicheur, and A. Berthoz, "An optimality principle governing human walking," *IEEE Transactions on Robotics*, vol. 24, no. 1, pp. 5–14, 2008.

5. J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1928–1935, IEEE, 2008.
6. D. Vasquez, "Novel planning-based algorithms for human motion prediction," in *IEEE Conference on Robotics and Automation*, 2016.
7. A. Foka and P. Trahanias, "Probabilistic autonomous robot navigation in dynamic environments with human motion prediction," *International Journal of Social Robotics*, vol. 2, no. 1, pp. 79–94, 2010.
8. G. Ferrer and A. Sanfeliu, "Bayesian human motion intentionality prediction in urban environments," *Pattern Recognition Letters*, vol. 44, pp. 134–140, 2014.
9. G. Ferrer and A. Sanfeliu, "Behavior estimation for a complete framework of human motion prediction in crowded environments," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 5940–5945, 2014.
10. G. Ferrer, A. Garrell, F. Herrero, and A. Sanfeliu, "Robot social-aware navigation framework to accompany people walking side-by-side," *Autonomous Robots*, pp. 1–19, 2016.
11. T. Gindele, S. Brechtel, J. Schroder, and R. Dillmann, "Bayesian occupancy grid filter for dynamic environments using prior map knowledge," in *Intelligent Vehicles Symposium*, pp. 669–676, IEEE, 2009.
12. M. Bahram, A. Wolf, M. Aeberhard, and D. Wollherr, "A prediction-based reactive driving strategy for highly automated driving function on freeways," in *Intelligent Vehicles Symposium*, pp. 400–406, IEEE, 2014.
13. O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.
14. Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *Proceedings IEEE International Conference on Robotics and Automation*, pp. 1398–1404, IEEE, 1991.
15. J. Barraquand and J.-C. Latombe, "Robot motion planning: A distributed representation approach," *The International Journal of Robotics Research*, vol. 10, no. 6, pp. 628–649, 1991.
16. E. A. Sisbot, L. F. Marin-Urias, R. Alami, and T. Simeon, "A human aware mobile robot motion planner," *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 874–883, 2007.
17. M. Svenstrup, T. Bak, and H. J. Andersen, "Trajectory planning for robots in dynamic human environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4293–4298, 2010.
18. C. Fulgenzi, A. Spalanzani, and C. Laugier, "Probabilistic motion planning among moving obstacles following typical motion patterns," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4027–4033, IEEE, 2009.
19. G. Ferrer, A. Garrell, and A. Sanfeliu, "Social-aware robot navigation in urban environments," in *European Conference on Mobile Robotics, ECMR.*, pp. 331–336, 2013.
20. G. Ferrer and A. Sanfeliu, "Proactive kinodynamic planning using the extended social force model and human motion prediction in urban environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1730–1735, 2014.
21. G. Ferrer and A. Sanfeliu, "Multi-objective cost-to-go functions on robot navigation in dynamic environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3824–3829, 2015.
22. D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.
23. R. Simmons, "The curvature-velocity method for local obstacle avoidance," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 4, pp. 3375–3382, 1996.
24. O. Brock and O. Khatib, "High-speed navigation using the global dynamic window approach," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 1, pp. 341–346, 1999.
25. S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.
26. D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock, "Randomized kinodynamic motion planning with moving obstacles," *The International Journal of Robotics Research*, vol. 21, no. 3, pp. 233–255, 2002.
27. C. Stachniss and W. Burgard, "An integrated approach to goal-directed obstacle avoidance under dynamic constraints for dynamic environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 508–513, 2002.
28. 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, pp. 31–48, 2005.
29. B. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. Bagnell, M. Hebert, A. Dey, and S. Srinivasa, "Planning-based prediction for pedestrians," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3931–3936, 2009.
30. M. Luber, L. Spinello, J. Silva, and K. O. Arras, "Socially-aware robot navigation: A learning approach," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 902–907, 2012.
31. H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard, "Socially compliant mobile robot navigation via inverse reinforcement learning," *The International Journal of Robotics Research*, 2016.
32. E. Kim, S. Choi, and S. Oh, "Structured low-rank matrix approximation in gaussian process regression for autonomous robot navigation," in *Proceedings IEEE International Conference on Robotics and Automation*, pp. 69–74, IEEE, 2015.
33. Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 285–292, IEEE, 2017.
34. P. Trautman, J. Ma, R. M. Murray, and A. Krause, "Robot navigation in dense human crowds: the case for cooperation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2145–2152, 2013.
35. M. Kuderer, C. Sprunk, H. Kretzschmar, and W. Burgard, "Online generation of homotopically distinct navigation paths," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 6462–6467, 2014.
36. R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and multidisciplinary optimization*, vol. 26, no. 6, pp. 369–395, 2004.
37. K. Deb, "Multi-objective optimization," in *Search methodologies*, pp. 403–449, Springer, 2014.

38. D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *Journal of Artificial Intelligence Research*, 2013.
39. E. Glassman and R. Tedrake, "A quadratic regulator-based heuristic for rapidly exploring state space," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 5021–5028, IEEE, 2010.
40. A. Perez, R. Platt, G. Konidaris, L. Kaelbling, and T. Lozano-Perez, "Lqr-rrt*: Optimal sampling-based motion planning with automatically derived extension heuristics," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2537–2542, IEEE, 2012.
41. B. Kunz and M. Stilman, "Probabilistically complete kinodynamic planning for robot manipulators with acceleration limits," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3713–3719, 2014.