Autonomous Person Pacing and Following with Model Predictive Equilibrium Point Control

Jong Jin Park and Benjamin Kuipers

Abstract— **The ability to follow or move alongside a person is a necessary skill for an autonomous mobile agent that works with human users. To accomplish the task, the robot must be able to track and follow the person it is accompanying while maneuvering through obstacles without collision. Also, the robot must be able to respect user preferences and exhibit behaviors that are intuitive and socially acceptable. That is, the robot is required to make complex decisions on-line, in environments that are almost always dynamic and uncertain in the presence of pedestrians.**

This paper discusses a versatile motion planning algorithm for *person pacing*, which refers to the capability to walk next to another person at user-preferred distance and orientation [1]. The algorithm is based on the Model Predictive Equilibrium Point Control (MPEPC) framework [2] which allows a robot to navigate gracefully in dynamic, uncertain, and structured environments.

We show that with a simple task description for person pacing, an agent with the MPEPC navigation algorithm can make intelligent decisions on-line, maximizing the expected progress toward achieving the task while minimizing the action cost and the probability of collision. We present navigation examples generated from real data traces, where a wheelchair robot exhibits very reasonable behaviors across a wide range of situations.

I. INTRODUCTION

To be genuinely useful to human users, an autonomous agent must be able to carry out high-level navigational instructions on its own (e.g. "go to my office desk," or "follow the guide.") However, autonomous navigation is a difficult challenge since the environment is almost always dynamic and uncertain, e.g. structured indoor environments with multiple pedestrians. The robot needs to perceive the environment, identify and track dynamic objects, constantly make intelligent decisions to avoid collision with static and dynamic hazards and make progress while conforming to user preferences.

In this paper, we introduce a navigation algorithm for *person pacing*, which refers to the capability to walk next to another person at desired distance and orientation [1]. Person pacing is an important skill for a service robot, especially for passenger carrying applications such as an autonomous wheelchair where a passenger commonly wants to accompany a person side by side. People walking in groups have a strong tendency to walk abreast, although exact formation varies depending on characteristics of group members [3]. It is also widely accepted that people have preferred inter-personal distances (so-called personal zones [4]) which depend on the relationship between them. Likewise, several studies [5] [6] [7] [8] have found that people have varying preferences for distance and approach angles when interacting with a robot, which depends on individual traits, robot characteristics as well as type of the interaction.

Existing literature on this topic tends to focus on *person following*, where the objective is simplified to maintain relative distance without considering orientation. Note that when following a person from behind, the robot is implicitly given the free space in the wake of the leading person, thus obstacle avoidance may be easier. For person following, there are published methods based on P- or PID-control and its variants [9] [10] [11] [12] [13], simple distance and distance-and-velocity based control laws [14] [15], a finite state machine [16], and on-line probabilistic roadmap [17] [18], and a variant of pure pursuit [19]. For person pacing, there is a classical method based on velocity obstacle [20]. However, smoothness and comfort of robot motion, and adaptation to user-specified clearance and orientations are largely neglected in the existing literature.

Our goal is to develop a versatile motion planning algorithm that is able to generate behaviors that are appropriate across a wide range of situations, while respecting the user-specified distance and orientation preferences. The algorithm must consider the robot and the person's current configuration, observed location and velocities of pedestrians, and static structures in the environment, while moving as smoothly and comfortably as possible.

The Model Predictive Equilibrium Point Control (MPEPC) [2] is an on-line local trajectory planning and control algorithm for robot navigation in dynamic and uncertain environments [2] [21]. We show that with a simple task description for person pacing, an agent with the MPEPC algorithm can make intelligent decisions on-line, maximizing the expected progress toward achieving the task while minimizing the action cost and the probability of collision. We present navigation examples generated from real data traces, where a wheelchair robot exhibits very reasonable behaviors moving in tandem with another person in an open hall and in a tight corridor with multiple pedestrians.

We describe the MPEPC framework, controller design and trajectory parameterization in Section II, and present a novel task description for person pacing in Section III, and our

This work has taken place in the Intelligent Robotics Lab in the Computer Science and Engineering Division of the University of Michigan. Research of the Intelligent Robotics lab is supported in part by grants from the National Science Foundation (CPS-0931474, IIS-1111494, and IIS-1252987), and from the TEMA-Toyota Technical Center.

J. Park is with the Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI 48109, USA jongjinp@umich.edu

B. Kuipers is with Faculty of Computer Science and Engineering Division, Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109, USA kuipers@umich.edu

low-overhead pedestrian tracking with lidar in Section IV. Results are shown in Section V.

II. MODEL PREDICTIVE EQUILIBRIUM POINT CONTROL

The MPEPC framework for navigation [2] is an on-line local trajectory planning and control algorithm that works well in dynamic and uncertain environments. The MPEPC-navigator is a receding-horizon model predictive planner, where the search space for the planner is defined by a pose-stabilizing feedback controller which was shown to work well with a physical robot [22]. As will be explained in this section, we use target pose and overall gain of the controller to compactly parameterize a continuous space of realizable closed-loop trajectories for the robot.

The MPEPC planner constantly replans as new information becomes available. At each planning cycle, the planner searches for the best trajectory that optimizes the expected value of robot behavior within a fixed time horizon for a given task.

A. MPEPC and Probabilistic Trajectory Evaluation

MPEPC formulates the problem of local navigation as a continuous, low dimensional and unconstrained optimization problem which is easy to solve. During the optimization process, we evaluate each trajectory candidate based on its *expected* utility for a given task, smoothly discrediting trajectories that may lead to collision based on estimated probability of collision along the trajectory.

Formally, we write

$$\underset{z_*=(x_*,\zeta)}{\text{minimize}} \quad J(q,x_*,\zeta,T) \tag{1}$$

$$\text{subject to} \quad \dot{q} = f_{\pi_\zeta}(q,\pi_\zeta(q,x_*)) \tag{2}$$

$$q(0) = q_I \tag{3}$$

where q is the robot state, x_* is the target pose for the controller, ζ is the controller gain, and T is the time horizon. Equation (1) is the multi-objective cost function for the expected cost (or negative utility) of a given trajectory that needs to be minimized, (2) is the dynamics of the robot under the stabilizing control policy which determines control input $u = \pi_\zeta(q,x_*)$, and (3) is the initial state of the robot at the beginning of the planning cycle. MPEPC optimizes over the space of trajectories parameterized by the controller target and gain, $z_* = (x_*,\zeta)$.

B. Controller Design, Robot Model and Trajectory Parameterization

We use the controller-based trajectory parameterization to efficiently narrow down the search space for the planner to the space of trajectories that are smooth, and are realizable by the controller.

The pose-stabilizing feedback controller that we use [2] is based on the kinematic control law (4) developed in [22] to provide fast, intuitive, smooth and comfortable motion. We have shown that linear velocity v and angular velocity

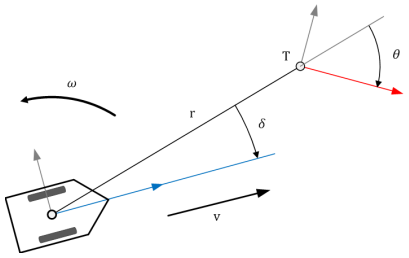

Fig. 1. Egocentric polar coordinate system with respect to a vehicle. From an observer situated on the vehicle and fixating at a target pose T, r is the radial distance to the target, θ is the orientation of T with respect to the line of sight from the observer to the target, and δ is the orientation of the vehicle heading with respect to the line of sight. At $r = 0$, the line of sight is aligned with the target. Here, both θ and δ have negative values.

ω satisfying the control law (4) will globally drive the robot toward any target pose by singular perturbation [23]. We have

$$\omega = -\frac{v}{r}\left[k_2(\delta - \arctan(-k_1\theta)) + (1 + \frac{k_1}{1+(k_1\theta)^2})\sin\delta\right] \tag{4}$$

where k_1 and k_2 are shape parameters[1] and the egocentric polar coordinates $(r,\theta,\delta)^T$ describe the relation between a robot pose and a target pose for the controller (Fig. 1).

The path curvature κ resulting from the control law is

$$\kappa = -\frac{1}{r}\left[k_2(\delta - \arctan(-k_1\theta)) + (1 + \frac{k_1}{1+(k_1\theta)^2})\sin\delta\right] \tag{5}$$

with a relation $\omega = \kappa v$ which holds for planar motion.

One of the key features of the control law is that its convergence property does not depend on the choice of positive linear velocity v. We choose a curvature-dependent linear velocity to make motion comfortable:

$$v(\kappa) = v(r,\theta,\delta) = \frac{v_{\max}}{1+\beta|\kappa(r,\theta,\delta)|^\lambda} \tag{6}$$

where the parameters β and λ determine how the vehicle slows down at a high curvature point, and $v_{\max}$ is a user-imposed maximum linear velocity, with $v \to v_{\max}$ as $\kappa \to 0$. It can be shown that all velocities, acceleration and jerks are bounded under (4) and (6) so that the motion of the robot is smooth and comfortable. Here, the parameters are fixed to $k_1 = 1.5$, $k_2 = 3$, $\beta = 0.4$ and $\lambda = 2$.

By adding a slowdown rule near the target pose with

$$v = \min(\frac{v_{\max}}{r_{\text{thresh}}}r,\, v(\kappa)) \tag{7}$$

with some user-selected distance threshold r_{thresh}, the target T becomes a non-linear attractor that the vehicle will exponentially converge to.[2] The maximum linear velocity parameter $v_{\max}$ can be understood as a gain value which determines the strength of the non-linear attractor. For our experiments, we use $r_{\text{thresh}} = 1.2\,\text{m}$.

[1] k_1 determines how fast the orientation error is reduced relative to the distance error. E.g. $k_1 = 0$ reduces the controller to pure way-point following. k_2 is a gain value.

[2] With (7), $v = v(\kappa)$ when $r > r_{\text{thresh}}$ since $v(\kappa) \le v_{\max}$. Also, $v \sim r$ near $r = 0$ canceling $1/r$ in (4) and rendering the system exponentially stable. Proofs shown in [22].

Fig. 2. **Left**: The wheelchair robot ($0.76 \times 1.2\,\mathrm{m}$). The wheelchair is differentially driven, such that linear and angular velocity are controlled independently. The robot is underactuated such that the linear velocity is always aligned with the orientation of the robot (i.e. the robot cannot move sideways). The motor becomes saturated at linear and angular accelerations of $0.4\,\mathrm{m/s^2}$ and $1.0\,\mathrm{rad/s^2}$. The robot model for forward prediction fully reflects these constraints. **Right**: Randomly selected 300 samples from the continuous space of smooth and realizable *closed-loop* trajectories, parametrized by the four dimensional vector $z_* = (r, \theta, \delta, v_{\max})^T$ over a time horizon $[0, T]$. The trajectories shown are generated from the dynamically simulated vehicle (with non-holonomic constraints and actuator saturations) under the stabilizing kinematic controller.

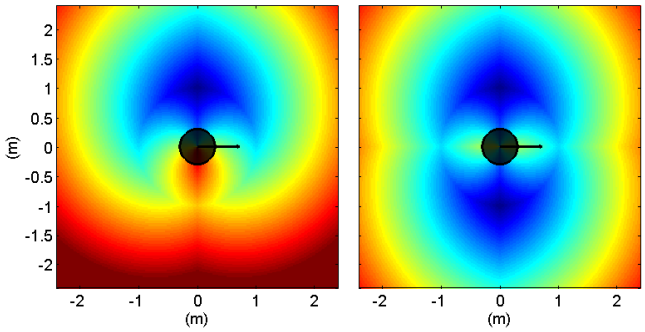

Fig. 3. (Best viewed in color) Example cost surfaces for person pacing which encode the user specified desired distance and orientation. The person to follow is depicted as a circle in the middle, with arrow indicating the person's heading. Red-to-blue color represents high-to-low cost. The cost surface is attached to the estimated pose of the person. See text for details. **Left**: A cost surface with the user specified desired orientation of $\pi/2\,\mathrm{rad}$ (on the left side of the person) and the desired distance of $1\,\mathrm{m}$. **Right**: The desired distance is the same at $1\,\mathrm{m}$, but with the preferred orientation of $\pm\pi/2\,\mathrm{rad}$ (on either side of the person).

Now, we have a 4-dimensional vector

$$\begin{aligned} z_* &= (\mathrm{T}, v_{\max})^T \\ &= (r, \theta, \delta, v_{\max})^T \end{aligned} \tag{8}$$

which completely describes any target of the vehicle system in the configuration-time space, and the trajectory of the vehicle converging to that target. Thus z_* also parametrizes *a space of smooth trajectories generated by the feedback control*, (4) - (7). The configuration-time space for the robot is 4-dimensional, so our parameterization is as compact as possible.

Although the control law is kinematic, the model for the MPEPC is constructed to closely match the non-linear dynamics of the physical wheelchair robot (Fig. 2, Left). For forward estimation of the robot trajectory, we simulate the response of the combined controller-robot system via the dynamic model which includes internal controller gains, torque saturations and input deadzones. The resulting estimated future trajectory of the robot (Fig. 2, Right) is evaluated for the expected progress, the cost of action and the cost of collision as shown in the next section.

III. TASK DESCRIPTION FOR PERSON PACING AND TRAJECTORY EVALUATION

A. Person Pacing

We want our robot to be able to accompany a person at user-defined distance and orientation. Furthermore, we want to be able to accommodate cases when there are more than one desired orientation, e.g. when walking on either side of the target person is equally desirable.

Let $\rho(t)$ be the estimated radial distance between the robot and the target person at time t, and $\eta(t) \in (-\pi, \pi]$ denote the estimated relative orientation of the robot as measured from the heading of the target person at time t. We introduce a straightforward cost metric $F(\cdot)$ for person pacing (which is in a unit of distance),

$$F(t) = |\rho(t) - \rho_d| + c_1 \cdot \rho_d \cdot |g(\eta(t) - \eta_d)| \tag{9}$$

where ρ_d and η_d denote the user-specified desired distance and orientation in the person-frame,[3] c_1 is the weight for the orientation error, and $g(\cdot)$ is a function that wraps any angle to interval $(-\pi, \pi]$.

If there is no specific preference for the orientation (or when the heading of the person to follow is indeterminate), the problem reduces to person following and $F(t) = |\rho(t) - \rho_d|$. If there are more than two desired orientations, we replace the orientation error $|g(\eta(t) - \eta_d)|$ with $\min(|g(\eta(t) - \eta_{d1}|, |g(\eta(t) - \eta_{d2})|, \ldots)$. Example cost surfaces are shown in Fig. 3.

B. Trajectory Evaluation

In our setting, the problem of person pacing for the planner is to find the most desirable trajectory parameterized by the 4-dimensional vector z_* at each planning cycle, so that the robot can make the most progress for a given task while minimizing discomfort and the chance of collision.

Formally, let

$$q_{z_*} : [0, T] \to C \tag{10}$$

be a trajectory of the robot parametrized by z_* within a finite time horizon T, where $C \simeq \mathrm{R}^2 \times \mathrm{S}^1$ is the configuration space of the robot. The optimization problem for the predictive planner is to select z_* which minimizes the following sum of *expected* costs over the trajectory $q_{z_*}([0, T])$:

$$\begin{aligned} J(x, z_*, T) &= E[\phi(q_{z_*})] \\ &= E[\phi_{\text{progress}} + \phi_{\text{action}} + \phi_{\text{collision}}] \end{aligned} \tag{11}$$

where ϕ_{progress} is the negative progress, ϕ_{action} is the cost of action, and $\phi_{\text{collision}}$ is the cost of collision.

[3]The desired orientation is defined with respect to the heading of the target person, and is not a function of robot orientation in the reference frame.

1) Probability of Collision: Suppose a robot is expected to travel along a trajectory q_{z*} at time t_j. We model the probability p_c^i of collision of the robot with the i-th object in the environment over a small time segment $[t_j, t_j + \Delta t]$, including the target person the robot is following, as a bell-shaped function:

$$p_c^i(d_i(j), \sigma_i) = exp(-d_i(j)^2/\sigma_i^2) \tag{12}$$

where j is a shorthand notation for the j-th sample pose of the robot at time t_j along the trajectory with $t_N = T$, $d_i(j)$ is the minimum distance from any part of the robot body to any part of the i-th object at time t_j which is computed numerically, and σ_i is an uncertainty parameter. We treat static structures as a single object in the environment.

From (12), we define the *survivability* $p_s(j)$, which represents the probability that the robot motion over the small time segment along the trajectory will be collision free, as

$$p_s(j) \equiv \prod_i^m (1 - p_c^i(d_i(j), \sigma_i)) \tag{13}$$

where m is the number of obstacles including the static structure. In our experiments, we use empirically chosen values for the uncertainty parameters: $\sigma = 0.1$ for the static structure, which represents robot position/simulation uncertainties; and $\sigma = 0.2$ for dynamic objects, which represents combined uncertainties of the robot and dynamic objects.

2) Expected Cost of a Trajectory: The negative progress $\phi_{\text{progress}}(j)$ over $q_{z*}([t_j, t_j + \Delta t])$ can be written as

$$\phi_{\text{progress}}(j) = F(t_j + \Delta t) - F(t_j) \tag{14}$$

where the relevant values of $\rho(t_j)$ and $\eta(t_j)$ is computed from the estimated pose of the robot $q_{z*}(t_j)$ and the estimated pose of the target person at time t_j from the tracker result.

With (13) and (14), we can write the *expected negative progress* $E[\phi_{\text{progress}}]$ as an integral of the negative progress weighted by the survivabiliy,

$$E[\phi_{\text{progress}}] \equiv \sum_{j=1}^{N} p_s(j) \cdot \phi_{\text{progress}}(j) \tag{15}$$

To guarantee the robot will never voluntarily select a trajectory leading to collision, after any sample with $p_c^i(k) = 1$, we set $p_c^i(j) = 1$ (thus $p_s(j) = 0$) for all $j \geq k$.

Similarly, with (12), we can write the *expected cost of collision* as

$$E[\phi_{\text{collision}}] \equiv \sum_{j=1}^{N} \sum_{i}^{m} p_c^i(j) \cdot \phi_{\text{collision}}^i(j) \tag{16}$$

where $\phi_{\text{collision}}^i(j)$ is the agent's idea for the cost of collision with the i-th object at the j-th sample. In this paper, we treat all collisions to be equally bad and use $\phi_{\text{collision}}^i(j) = 0.1$ for all i, j.

For the action cost, we use a usual quadratic cost on velocities:

$$\phi_{\text{action}} = \sum_{j=1}^{N} (c_2 v^2(j) + c_3 \omega^2(j)) \Delta t \tag{17}$$

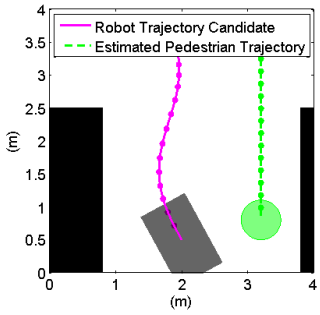

Fig. 4. Robot evaluating a candidate future trajectory. The expected negative progress of a candidate robot trajectory (red) is evaluated by integrating the negative progress (14) over the moving cost surface (Fig. 3) attached to the estimated trajectory of the target person (green), weighted by the estimated survivability (13) along the candidate robot trajectory. The total expected cost also considers the expected cost of collision (16) and the quadratic cost of action (17). The cost function is generated with user-specified desired distance and orientation (9). See text for details.

where c_3 and c_4 are weights for the linear and angular velocity costs, respectively. For the *expected cost of action*, we assume $E[\phi_{\text{action}}] = \phi_{\text{action}}$.

The overall *expected* cost of a trajectory is the sum of (15), (16), and (17), as given in (11). The probability weights p_c^i and p_s (12) - (13) can be understood as mixing parameters, which properly incorporates the collision constraint into the cost definition. The use of probability weights renders the collision term completely dominant when the robot is near an object, while letting the robot focus on progress when the chance of collision is low. With (11), the optimization problem (1) is now in continuous, low-dimensional and unconstrained form.

For our experiments, the weights are fixed to $c_1 = 0.75$, $c_2 = 0.2$, and $c_3 = 0.1$.

C. Implementation

We have used off-the-shelf optimization packages[4] for implementation. For our experiments, the receding horizon was set at $T = 5\,\text{s}$, the plan was updated at 2 Hz, and the underlying feedback controller ran at 20 Hz.

In practice, we found the performance of the optimizer (for both speed and convergence to the global minimum) depends greatly on the choice of the initial conditions. Thus the optimization process is implemented in two phases. First, we coarsely sample the search space[5] to find a good initial condition. The best candidate from the pre-sampling phase is passed to a gradient-based local optimizer as an initial condition for the final output.

With our problem formulation the computational cost for the numerical optimization is small, achieving real-time performance. With our C++ implementation on a 2.66-GHz laptop, a typical evaluation of a trajectory takes less than a millisecond. In our experiments, on average ~ 62 trajectories were evaluated per optimization (see Section V).

[4] *fmincon* in MATLAB and *NLopt* [24] in C++.

[5] For speed, we use a dozen pre-selected sample trajectories (including the optimum from the previous time step and typical turns) in MATLAB. We sample by Controlled Random Search with local mutation [24] in C++.

IV. REAL-TIME PEDESTRIAN TRACKING

Here we briefly describe our implementation of the pedestrian tracker, although the main focus of the paper is on versatile motion planning. Our tracker is a low-overhead solution at minimal computational cost using an on-board laser sensor. The tracker can detect and track multiple pedestrians in the sensor's field of view and can handle occlusions for a short period of time.

From the static map of the environment and a current laser scan (which is typically very noisy, e.g. Fig. 5), we first identify portions of the laser scans that are not explained by the static map. In our current implementation, laser rays that fall in cells which were previously marked as free space are classified as dynamic [25].

We cluster rays in the dynamic scan by identifying connected components, and the resulting clusters are validated for a pedestrian model based on size and shape (similar to identification of leg candidates in [26] [27] [28]). The validated clusters are tracked via Kalman Filters for position and linear velocity. Future trajectory of detected pedestrians are estimated using constant velocity model.

To estimate the heading of the person, we assume that the heading of a person is aligned with the estimated linear velocity if the speed is larger than a threshold ($> 0.1\,\mathrm{m/s}$). Otherwise the heading of the person is indeterminate. We handle temporary occlusions by keeping track of an object for a short period of time even when it is not observed. An example result is shown in Fig. 6.

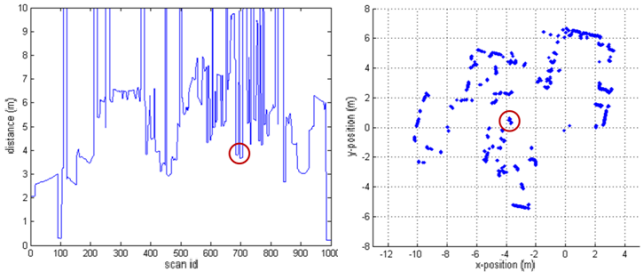

Fig. 5. **Left**: A (noisy) scan from a laser range finder in an environment with a single pedestrian, with the legs of a pedestrian in a red circle. **Right**: Scan values projected onto map coordinates.

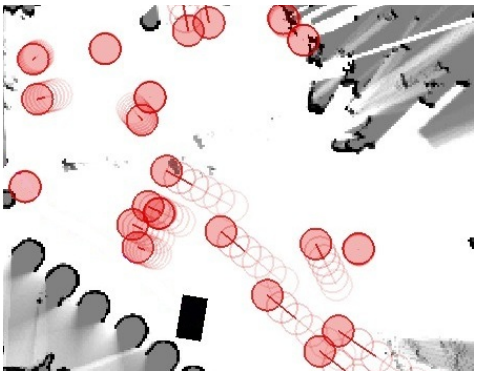

Fig. 6. Tracker result in a crowded hall from the wheelchair robot. Robot pose is depicted as a black rectangle. Red circles and protruding lines denote the current position and estimated velocities of the pedestrians being tracked. Estimated future positions (up to 5 seconds) are shown by empty circles with constant velocity model.

V. RESULTS

In this section, we show three example scenarios that are typical in real life situations. The wheelchair robot navigates in challenging environments with the proposed algorithm, under varying user preferences in the distance ρ_d and the orientation η_d. The maps and the pedestrian trajectories in the experiments are collected from the physical wheelchair robot, and the robot behaviors are simulated using the real data traces.

In Fig. 7, the robot starts at some distance away from the target pedestrian and quickly sets itself up on the left side of the person, with the desired distance and orientation of $\rho_d = 1.5\,\mathrm{m}$ and $\eta_d = \pi/2$. The mission is relatively easy, and the robot is able to follow the person smoothly near the desired distance and orientation (Fig. 8, Right). The full set of trajectories explored over the navigation run is shown on Fig. 7, Right. Overall, the robot moves smoothly, except a brief slowdown near 14 s (Fig. 8, Left) to avoid collision with a pedestrian coming across.

Fig. 9 shows the trajectory of the robot in the same environment, but it is now instructed to pace the person on the right, with the desired distance and orientation of $\rho_d = 1.5\,\mathrm{m}$ and $\eta_d = -\pi/2$. The mission is more difficult as the robot has less space to maneuver, but the robot can successfully speed up, slow down, change its heading to avoid a pedestrian and return to its desired position. The full set of trajectories that were explored over the navigation run is shown on Fig. 7, Right. As can be seen in this example, the robot with the proposed navigation algorithm is able to temporarily move away from the user-specified distance and orientation preferences to avoid collision with other obstacles (Fig. 10, Right).

Fig. 11 show the most challenging example, where the robot ($0.76 \times 1.2\,\mathrm{m}$) has to move in a tight corridor (2 m wide) with multiple pedestrians. In this example no specific preference in orientation is given, and the problem is reduced to person following with the desired distance of $\rho_d = 1.8\,\mathrm{m}$. The example shown is very difficult as the robot needs to move in confined space and there is a pedestrian quickly moving in from behind. The robot is able to successfully follow the person, speeding up, switching sides, slowing down and veering as needed. See Fig. 11 - 13 for details.

Over the three example runs shown, 10240 trajectory candidates were evaluated in 166 planning cycles, averaging 61.7 trajectory evaluations per optimization, achieving real-time performance. The robot exhibits very reasonable behaviors in realistic scenarios using noisy data.

VI. DISCUSSION

Person pacing is a task that can greatly benefit from an excellent on-line trajectory planning algorithm. For a robot moving alongside a person, both the goal (target person) and the obstacles (pedestrians) are mobile, and it is difficult to accurately predict motion of a pedestrian for more than a few seconds; the environment is highly dynamic and uncertain.

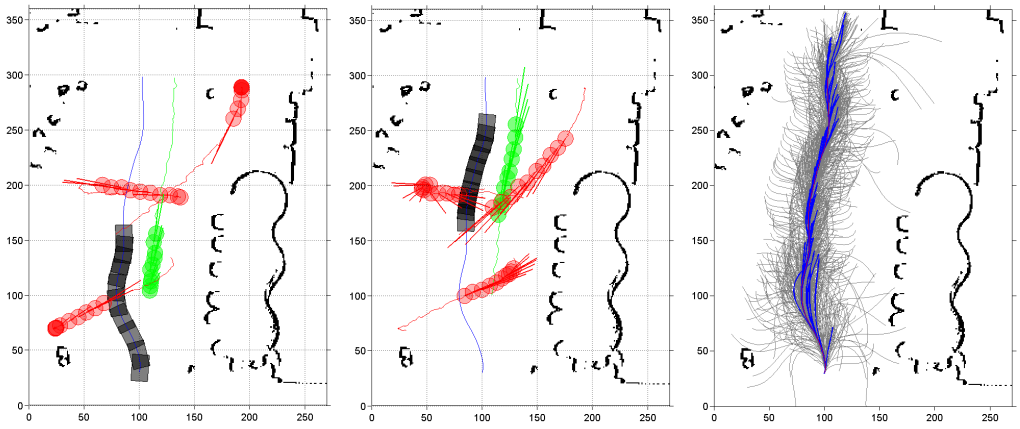

Fig. 7. (Best viewed in color) In an open hall (13.5×18 m) with multiple pedestrians (circles), the robot (rectangle) estimates the trajectories of dynamic objects over the planning horizon (protruding lines) and paces the target person (green) on the left without collision. This is a relatively easy mission. **Left and Center**: The trajectories overlaid with snapshots of the robot (rectangle) and the pedestrians (circles) at 1 s intervals. The axis are in grid coordinates (5cm per grid). The robot starts from the bottom pf the map, trying to pace the person in the middle on the left side. **Right**: Trajectories sampled by the planner (gray), time-optimal plans selected at each planning cycle (blue) and the actual path taken (red).

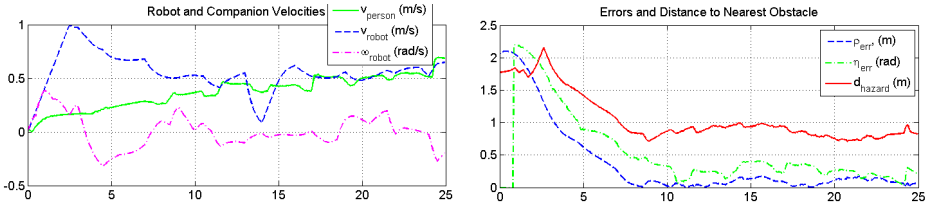

Fig. 8. **Left**: Raw velocity estimates of the target person and the robot. The robot reaches the desired distance and orientation near 8 s, and paces the target person at a similar speed. The robot briefly slows down near 14 s to avoid collision with a pedestrian coming across its path. **Right**: Distance and orientation error for the person pacing, and distance to nearest obstacle over the trajectory. As the mission is relatively easy, the error stays low for most of the trajectory.

Motion planning in dynamic and uncertain environments is a challenging problem which, in our opinion, requires probabilistic trajectory planning. For example, path planning alone (e.g. RRT [29]) is not sufficient, since a path is defined in the configuration space and does not include the important timing information that is required to efficiently follow or avoid dynamic objects. Potential-field methods can be simple, but they have problems such as being trapped in local minima, and are generally not suitable in highly dynamic environments [16].

The MPEPC framework provides a tool for on-line local trajectory planning in dynamic and uncertain environments, by constant replanning with information feedback[6] and by efficient optimization with compact parameterization of the trajectory space. The overall algorithm is efficient since the search space is small and does not require post-processing to compute the control inputs. Planning and control is truly integrated in our framework, as the planner only searches in the space of closed-loop trajectories that are smooth and realizable by construction.

The decision-theoretic trajectory evaluation, with the estimated probability of collision, allows easy integration of user-preference into the objective function. With the probability weights, any trajectory that may lead to collision is smoothly discredited based on the estimated probability of collision along the trajectory, regardless of the form of the progress metric. This gives us great flexibility in terms of cost design. As can be seen from the examples, the algorithm works well with the straightforward progress metric (9) which is a function of user preferences.

One limitation of the current implementation of the algorithm is the constant velocity model for pedestrian behaviors. Pedestrians can take a sharp turn around a corner, and can actively avoid other agents, including the wheelchair robot. As the model does not capture such agent intentions, the resulting robot motion may appear too submissive which may not be suitable for a wheelchair robot.[7] We believe that incorporation of intention estimation will significantly improve the pedestrian behavior model, although it is out of scope of this paper.

The proposed planner is able to generate very reasonable behaviors by making locally optimal decisions with the information available, i.e. the noisy tracker outputs and the conservative model which predicts that other agents will not make way for the robot. We plan to improve the pedestrian model for the implementation of the algorithm on the physical robot.

[6]We have found the fast information feedback is crucial. The quality of robot motion dropped significantly when the planning rate was slowed to 1 Hz. Data not shown.

[7]We thank the reviewers for making this point.

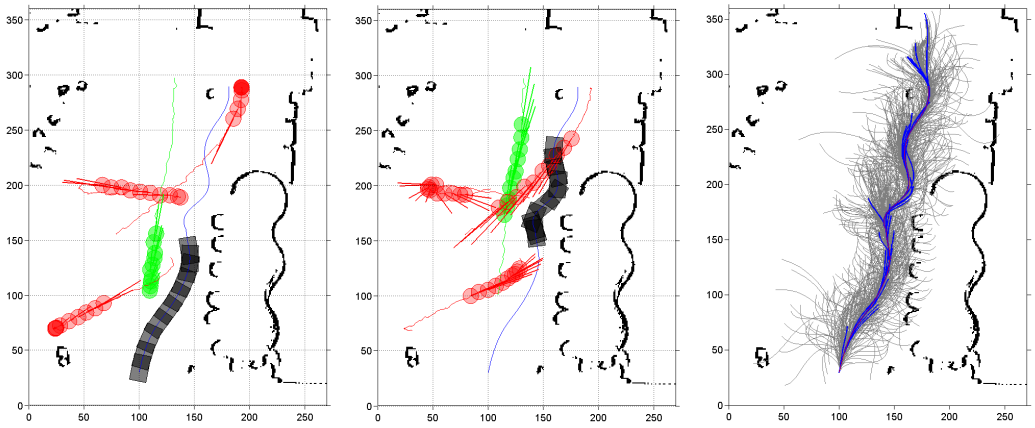

Fig. 9. (Best viewed in color) In an open hall (13.5×18 m) with multiple pedestrians (circles), the robot (rectangle) estimates the trajectories of dynamic objects over the planning horizon (protruding lines) and paces the target person (green) on the right without collision. The robot has to maneuver in a tighter space and avoid an oncoming pedestrian. The robot successfully speeds up, slows down, changes its heading to avoid the pedestrian and returns to its desired position. **Left and Center**: The trajectories overlaid with snapshots of the robot (rectangle) and the pedestrians (circles) at 1 s intervals. The axis are in grid coordinates (5cm per grid). The robot starts from the bottom, trying to pace the person in the middle on the right. **Right**: Trajectories sampled by the planner (gray), time-optimal plans selected at each planning cycle (blue) and the actual path taken (red).

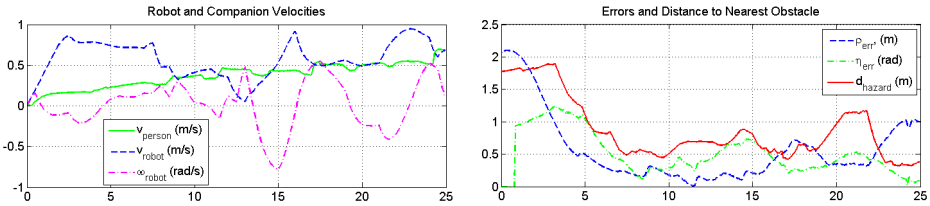

Fig. 10. **Left**: Raw velocity estimates of the target person and the robot. Overall, the robot moves smoothly to pace the target person, stopping (13 s), speeding up, and turning as needed to avoid collision. **Right**: Distance and orientation error for the person pacing, and distance to nearest obstacle over the trajectory. Compared to Fig. 8, this figure shows that the robot is able to move away from the user-specified distance and orientation to avoid collision.

VII. CONCLUSIONS

In this paper, we have introduced a novel motion planning algorithm for person pacing, which is one of the most common and important navigation tasks when people interact with other people, and with a service robot. We have described the requirements for the movement of a robot that has to accompany a human, and cast the task as a low-dimensional, continous, and unconstrained optimization problem for which we have developed a task-specific cost function. We have demonstrated that the proposed algorithm allows the robot to accompany a person across a wide range of situations, while respecting user-specified distance and orientation preferences.

To accomplish the task, the algorithm considers the robot and the person's current configuration, observed location and velocities of pedestrians, and static structures in the environment, making decisions on-line so that the robot can move along with a person at desired distance and orientation smoothly and comfortably while maneuvering through obstacles. The performance of the algorithm depends on the integrated planning and control architecture of the MPEPC framework, and the decision-theoretic trajectory evaluations based on the estimated probability of collision.

The proposed algorithm is important as it addresses the difficult problem of navigating in an uncertain and dynamic environment safely and comfortably while avoiding hazards and respecting user preferences, which is a necessary task for autonomous passenger vehicles.

ACKNOWLEDGMENT

The authors would like to thank Kate Tsui, Collin Johnson, Grace Tsai and Paul Foster.

REFERENCES

[1] D. Miller, "Moving in tandem: automated person pacing for wheelchair users," in *Developing Assistive Technology for People with Disabilities*, ser. AAAI-96 Fall Symposium Series, Nov 1996, pp. 86–88.

[2] J. Park, C. Johnson, and B. Kuipers, "Robot navigation with Model Predictive Equilibrium Point Control," in *2012 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Oct 2012, pp. 4945–4925.

[3] M. Costa, "Interpersonal distances in group walking," *Journal of Nonverbal Behavior*, vol. 34, no. 1, pp. 15–26, Mar 2010.

[4] E. Hall, *The hidden dimension*. NY: Doubleday, 1966.

[5] M. Walters, K. Dautenhahn, R. te Boekhorst, K. L. Koay, C. Kaouri, S. Woods, C. Nehaniv, D. Lee, and I. Werry, "The influence of subjects' personality traits on personal spatial zones in a human-robot interaction experiment," in *IEEE Intl. Workshop on Robot and Human Interactive Communication (ROMAN)*, Aug 2005, pp. 347–352.

[6] R. Gockley and M. Mataric, "Encouraging physical therapy compliance with a hands-off mobile robot," in *Proceedings of Human-Robot Interaction, Salt Lake City, Utah*, Mar 2006, pp. 150–155.

[7] H. Hutenrauch, K. Eklundh, A. Green, and E. Topp, "Investigating spatial relationship in human-robot interaction," in *2006 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Oct 2006, pp. 5052–5059.

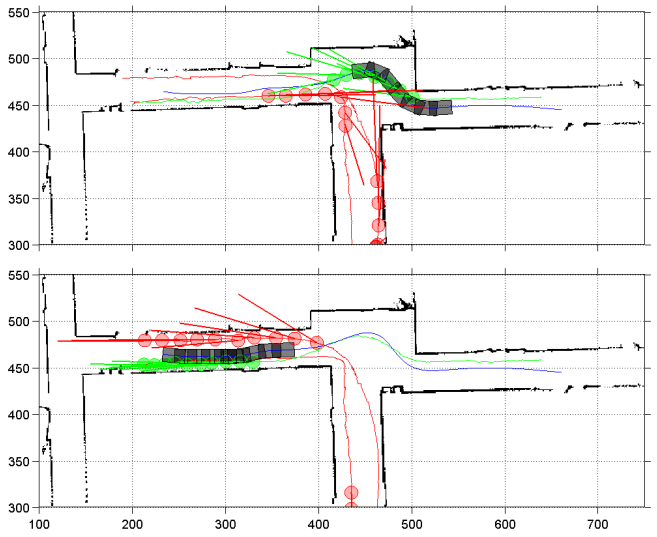

Fig. 11. (Best viewed in color) In a tight corridor environment (32.5 × 12.5 m, 2 m in width) with multiple pedestrians (circles), the robot (rectangle) estimates the trajectories of dynamic objects over the planning horizon (protruding lines) and follows the target person (green), without orientation preference. The axis are in grid coordinates (5cm per grid). **Top**: The robot follows the person slightly on the left to keep distance from walls, and moves away from estimated trajectory of an oncoming pedestrian (red lines). **Bottom**: After the robot moves into a hallway, the target person walks along the left wall and the robot switches to the right side of the person to keep distance from walls. When the robot detects a quickly approaching pedestrian from behind, it slows down and veers slightly left to avoid collision with the pedestrian and the person the robot is following.

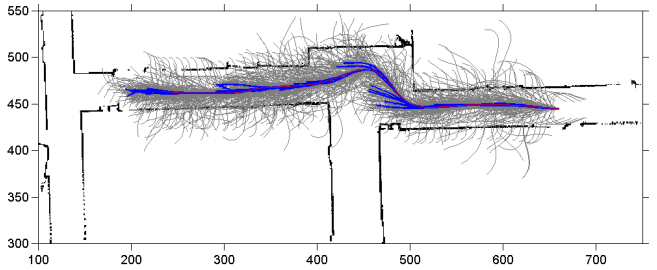

Fig. 12. Densely sampled candidate trajectories in the neighborhood of the robot over the navigation run in Fig. 11 (gray), time-optimal plans selected at each planning cycle (blue) and the actual path taken (red).

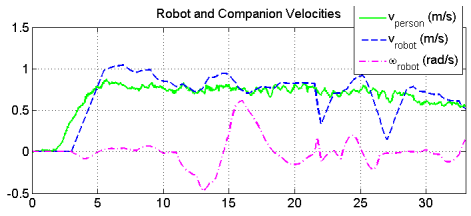

Fig. 13. Raw velocity estimates of the leading pedestrian and the robot. The robot generally moves at the same speed as the target person. It slows down and veers to side as needed when it detects another person approaching from behind (22 - 29 s).

[8] M. Walters, K. Koay, S. Woods, D. Syrdal, and K. Dautenhahn, "Robot to human approaches: Preliminary results on comfortable distances and preferences," in *Proc. AAAI Spring Symp. Multidisciplinary Collaborat. Socially Assistive Robot*, 2007.

[9] E. Topp and H. Christensen, "Tracking for following and passing persons," in *2005 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Aug 2005, pp. 2321–2327.

[10] R. Gockley, J. Forlizzi, and R. Simmons, "Natural person-following behavior for social robots," in *Human-Robot Interaction (HRI), 2007 2nd ACM/IEEE International Conference on*, Mar 2007, pp. 17–24.

[11] X. Ma, C. Hu, X. Dai, and K. Qian, "Sensor integration for person tracking and following with mobile robot," in *2008 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Sep 2008, pp. 3254–3259.

[12] T. Germa, F. Lerasle, N. Ouadah, V. Cadenat, and M. Devy, "Vision and rfid-based person tracking in crowds from a mobile robot," in *2009 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Oct 2009, pp. 5591–5596.

[13] C.-H. Hu, X.-D. Ma, and X.-Z. Dai, "Reliable person following approach for mobile robot in indoor environment," in *2009 Intl. Conf. on Machine Learning and Cybernetics*, vol. 3, Jul 2009, pp. 1815–1821.

[14] J. Satake and J. Miura, "Robust stereo-based person detection and tracking for a person following robot," in *2009 ICRA Workshop on Person Detection and Tracking*, May 2009.

[15] H. Takemura, N. Zentaro, and H. Mizoguchi, "Development of vision based person following module for mobile robots in/out door environment," in *2009 IEEE Intl. Conf. on Robotics and Biomimetics (ROBIO)*, Dec 2009, pp. 1675–1680.

[16] C.-P. Lam, C.-T. Chou, K.-H. Chiang, and L.-C. Fu, "Human-centered robot navigation –towards a harmoniously human-robot coexisting environment," *IEEE Transactions on Robotics*, vol. 27, no. 1, pp. 99–112, Feb 2011.

[17] S. Frintrop, A. Knigs, F. Hoeller, and D. Schulz, "A component-based approach to visual person tracking from a mobile platform," *Intl. J. of Social Robotics*, vol. 2, pp. 53–62, 2010.

[18] F. Hoeller, D. Schulz, M. Moors, and F. Schneider, "Accompanying persons with a mobile robot using motion prediction and probabilistic roadmaps," in *2007 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Nov 2007, pp. 1260–1265.

[19] S. Hemachandra, T. Kollar, N. Roy, and S. Teller, "Following and interpreting narrated guided tours," in *2011 IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2011, pp. 2574–2579.

[20] E. Prassler, D. Bank, and B. Kluge, "Motion coordination between a human and a mobile robot," in *2002 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, vol. 2, 2002, pp. 1228–1233.

[21] J. Park, C. Johnson, and B. Kuipers, "Robot navigation with MPEPC in dynamic and uncertain environments: From theory to practice," in *IROS 2012 Workshop on Progress, Challenges and Future Perspectives in Navigation and Manipulation Assistance for Robotic Wheelchairs*, Oct 2012.

[22] J. Park and B. Kuipers, "A smooth control law for graceful motion of differential wheeled mobile robots in 2D environment," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, May 2011, pp. 4896–4902.

[23] H. K. Khalil, *Nonlinear Systems (3rd Edition)*, 3rd ed. New York, NY, U.S.: Prentice Hall, 2002.

[24] S. G. Johnson. The nlopt nonlinear-optimization package. [Online]. Available: http://ab-initio.mit.edu/nlopt

[25] J. Modayil and B. Kuipers, "The initial development of object knowledge by a learning robot," *Robot. Auton. Syst.*, vol. 56, no. 11, pp. 879–890, Nov 2008.

[26] K. Nakamura, H. Zhao, R. Shibasaki, K. Sakamoto, T. Ohga, and N. Suzukawa, "Tracking pedestrians using multiple single-row laser range scanners and its reliability evaluation," in *Syst. Comp. Jpn*, vol. 37, no. 7.

[27] T. Horiuchi, S. Thompson, S. Kagami, and Y. Ehara, "Pedestrian tracking from a mobile robot using a laser range finder," in *IEEE Intl. Conf. on Systems, Man and Cybernetics*, Oct 2007, pp. 931–936.

[28] C. T. Chou, J.-Y. Li, M.-F. Chang, and L. C. Fu, "Multi-robot cooperation based human tracking system using laser range finder," in *2011 IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2011, pp. 532–537.

[29] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006, available at http://planning.cs.uiuc.edu/.