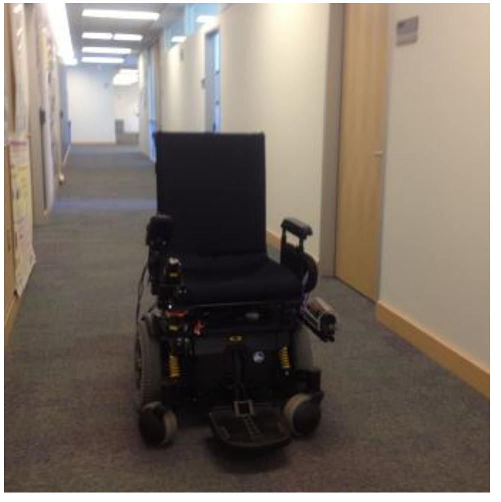
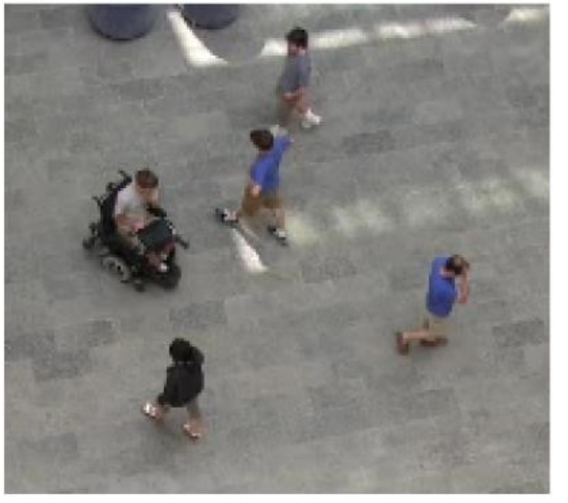
Robot Navigation with Model Predictive Equilibrium Point Control (MPEPC)

Jong Jin Park, Collin Johnson and Benjamin Kuipers
University of Michigan, USA

Robot Navigation Faces Dynamic and Uncertain Environments





- Tight rectilinear spaces require high precision motion control
- Pedestrians and inaccurate robot model introduce dynamics and uncertainty
- Need to accommodate user preferences, e.g. aggressiveness and comfort

Hierarchical Motion Planning
Is Needed in Dynamic and Uncertain Environments

Global Planner

Approximate, longer term navigation plan in the environment

↓

Local Planner

High fidelity local paths/trajectories in small scale space

Generate-and-test search for trajectories

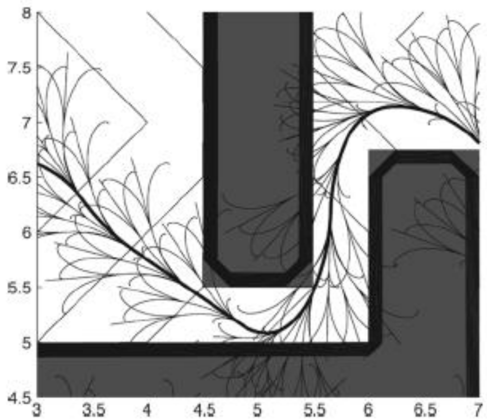
↓

Control

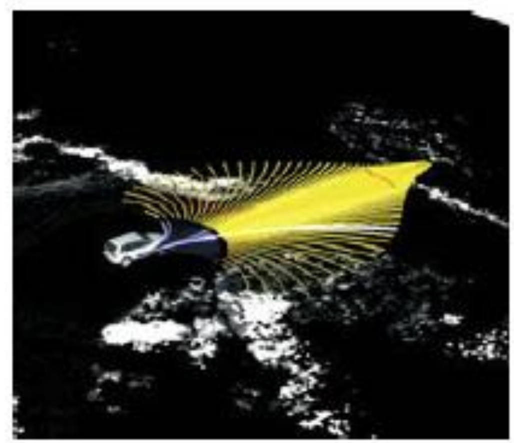
Low level controller for trajectory execution

The Space of Trajectories is Continuous and Infinite

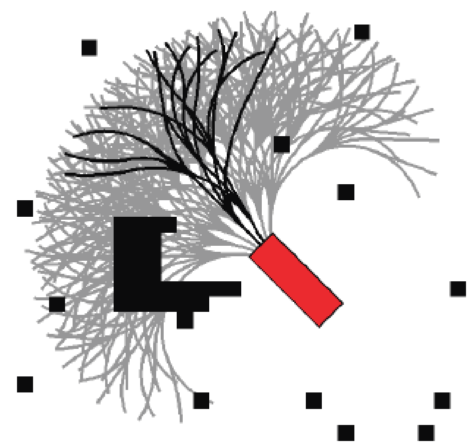
- Many current leading algorithms rely on a finite set of pre-determined candidate trajectories/paths.



[Ogren and Leonard 05]



[Hundelshausen et al. 08]



[Knepper and Mason 12]

- How to construct a good evaluation function is also an important question.
  - Determination of weights in multi-objective function, etc.

Our MPEPC Approach: Objectives

- Efficient search for candidate trajectories
- Efficient evaluation of candidate trajectories, considering robot and pedestrian motion uncertainties
- Easy and straightforward implementation
- Accommodation of user preferences

Our MPEPC Approach: Objectives

- **Efficient search for candidate trajectories**
- Efficient evaluation of candidate trajectories, considering robot and pedestrian motion uncertainties
- Easy and straightforward implementation
- Accommodation of user preferences

Our MPEPC approach to Hierarchical Motion Planning and Control

Global Planner

Approximate navigation plan in the static environment

Navigation Function (NF)

Local Trajectory Planner

High fidelity local trajectories in small scale space

Dynamic replanning with receding-horizon MPC

Control

Low level controller for trajectory execution

Pose-stabilizing feedback controller (EPC)

Pose-stabilizing Feedback Control

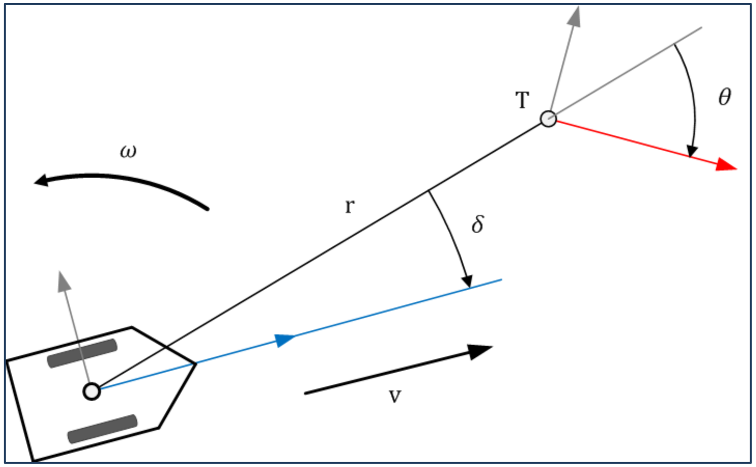
- We have developed a controller that allows the robot to reach an arbitrary target pose in a smooth curve.

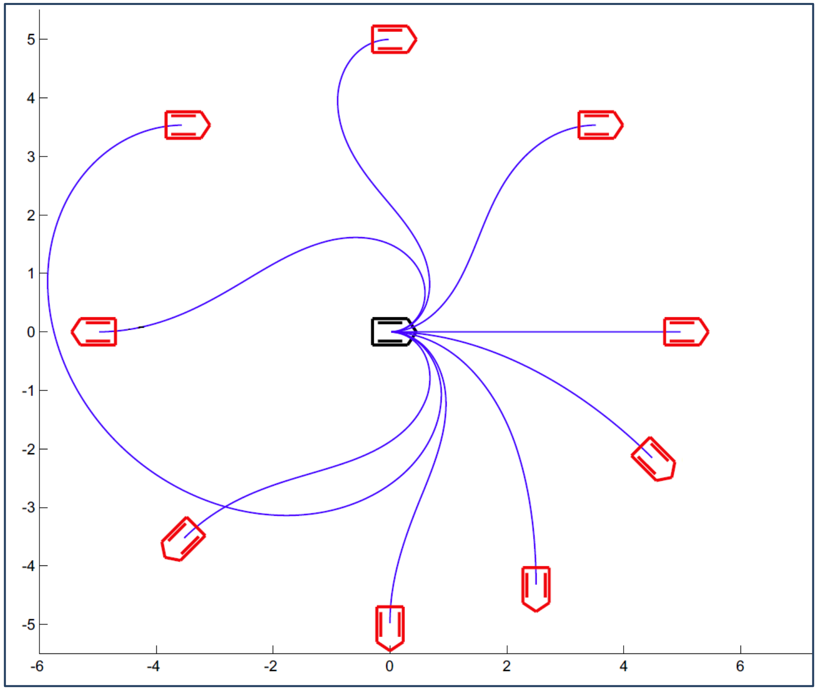
  [Park and Kuipers, *ICRA*-11]

  – While satisfying linear and angular velocity bounds, slowing down at high curvature points;
  – Without singularity at the target.
  – Target pose is exponentially stable.
- It allows us to compactly parameterize smooth and realizable robot trajectories in terms of the target pose and the gain value (4D).

Pose-stabilizing Feedback Control

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





- (r, θ, δ) describes the target T viewed from the vehicle in terms of the line of sight (LOS).
- At $r = 0$, LOS is aligned with T.

[Park and Kuipers, *ICRA*-11]

Pose-stabilizing Feedback Control

- Curvature-dependent choice of linear velocity

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

  – Guarantees bounded linear and angular velocities
- Slowdown rule near target pose

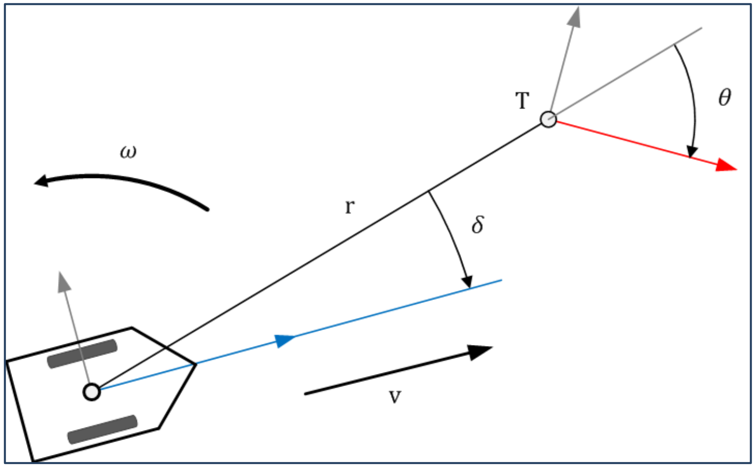
$$v = \min\left(\frac{v_{\max}}{r_{\text{thresh}}} r,\ v(\kappa)\right)$$

  – Removes singularity at $r \to 0$
  – Target pose is exponentially stable
  – $v_{\max}$ can be viewed as a gain value

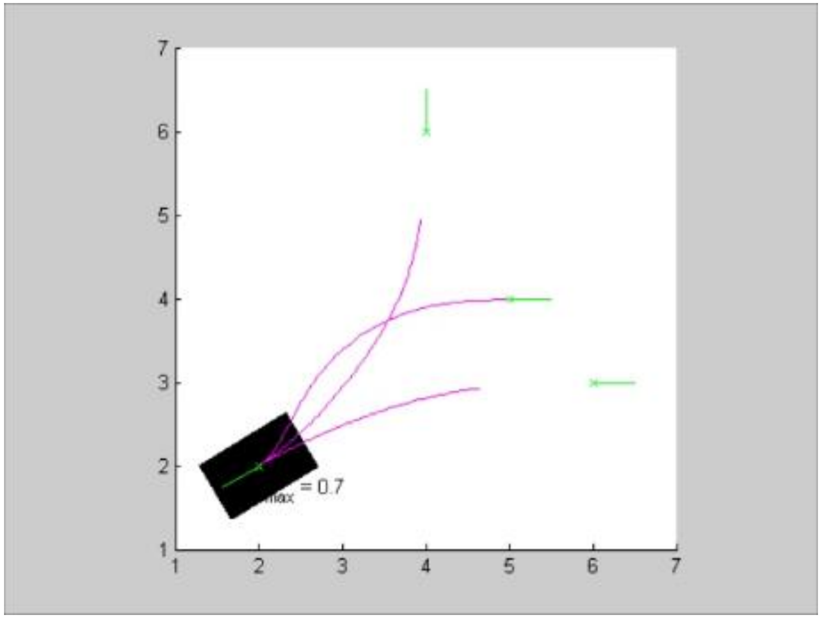
[Park and Kuipers, *ICRA*-11]

Combined Controller-Robot Model

- Closed-loop robot dynamic simulation with the controller target and gain, $z_* = (r, \theta, \delta, v_{\max})$
  - Non-holonomic, motor saturations, and P-controller for velocities (joystick)
  - z_* *parameterize the simulated responses of the robot system* under the feedback controller.

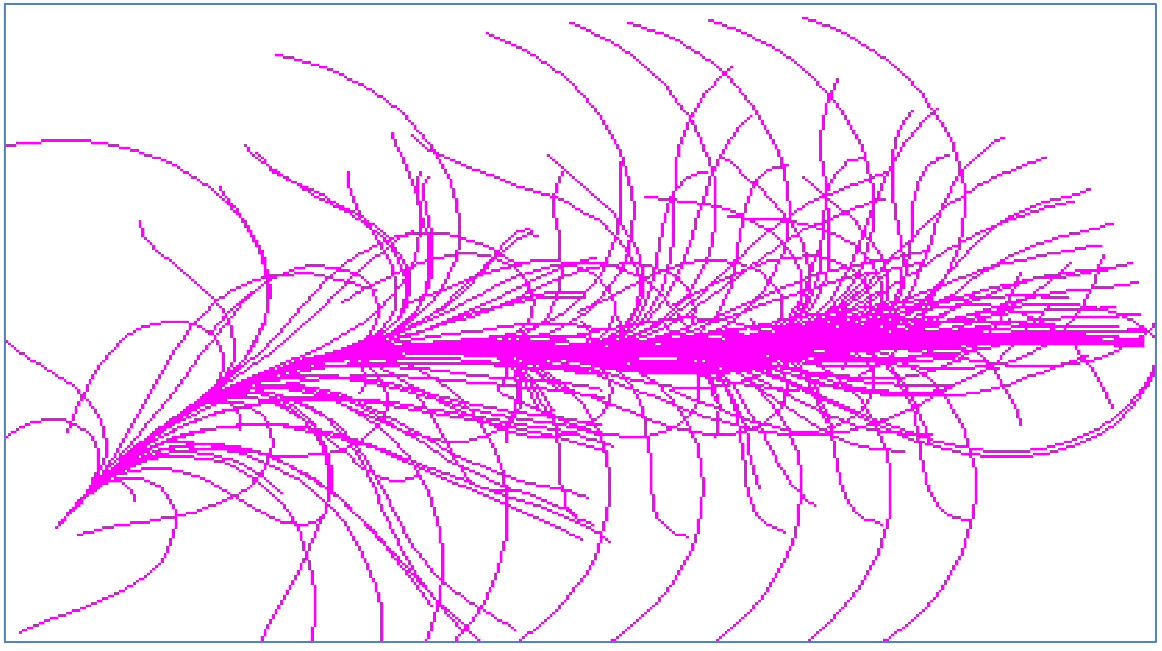


[Park and Kuipers, *ICRA*-11]



Defining Our Search Space:
Controller-based Trajectory Parameterization

- Our 4D parameterization $z_* = (r, \theta, \delta, v_{\max})$ defines a continuous space of closed-loop trajectories.
  - It identifies a useful subspace of the infinite and continuous space of possible trajectories that are smooth and realizable by construction.
- Compact parameterization allows efficient search.



Our MPEPC Approach: Objectives

- Efficient search for candidate trajectories
- **Efficient evaluation of candidate trajectories, considering robot and pedestrian motion uncertainties**
- Easy and straightforward implementation
- Accommodation of user preferences

Trajectory Evaluation

- Trajectories parameterized by z_*:

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

- Overall *expected* cost of a candidate trajectory, considering probability of collision

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

  - Negative progress over the static plan (Navigation Function, *NF*)
  - Penalty for probability of collision
  - Quadratic action cost (on velocities)

Incorporation of Motion Uncertainties Makes the Optimization Easier

- We construct probability weights as a function of robot and pedestrian motion uncertainties
  - We define simple approximations for:
    - Probability of collision and
    - Survivability of a trajectory segment.
  - Probability weights allow us to formulate the problem as unconstrained optimization over a smooth surface.

Discrete Approximation to Probability of Collision and Survivability

- For j-th sample along the trajectory, probability of collision to the i-th object in the map is approximated as:

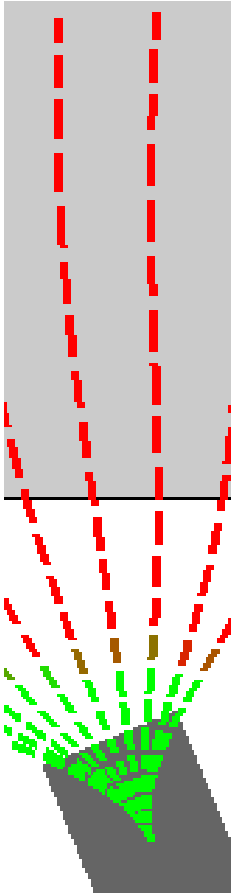
$$p_c^i(d_i(j), \sigma_i) = exp(-d_i(j)^2/\sigma_i^2)$$

  - $d_i(j)$ is the minimum distance from any part of the robot body to any part of the *i*-th object in the map at time *j*.
  - σ_i are uncertainty parameters.

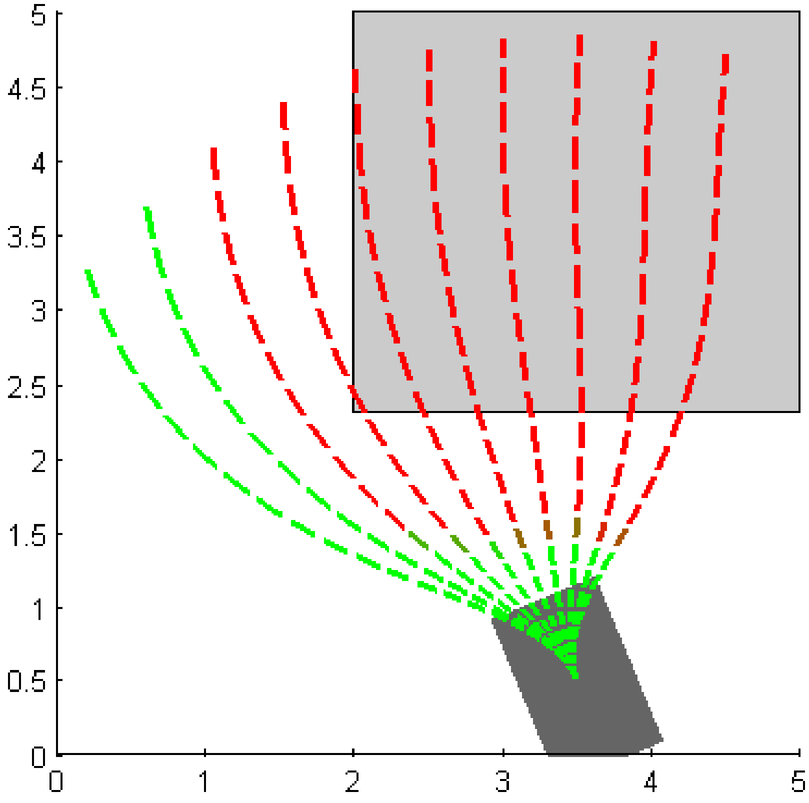
- Survivability of a trajectory segment is a probability that the trajectory segment will be collision free to any obstacles

$$p_s(j) \equiv \prod_{i=1}^{M} (1 - p_c^i(j))$$

  - $i \in [1...M], \quad j \in [1...N]$



Incorporating Probability Weights and Expected Values Creates a Smooth Optimization Surface



- Progress weighted by survivability

$$p_s(j) \cdot \Delta NF(j)$$

- Collision penalty weighted by probability of collision

$$\sum_{i=1}^{M} p_c^i(j) \cdot \phi_{\text{collision}}^i(j)$$

- Additive action cost to modify robot behavior

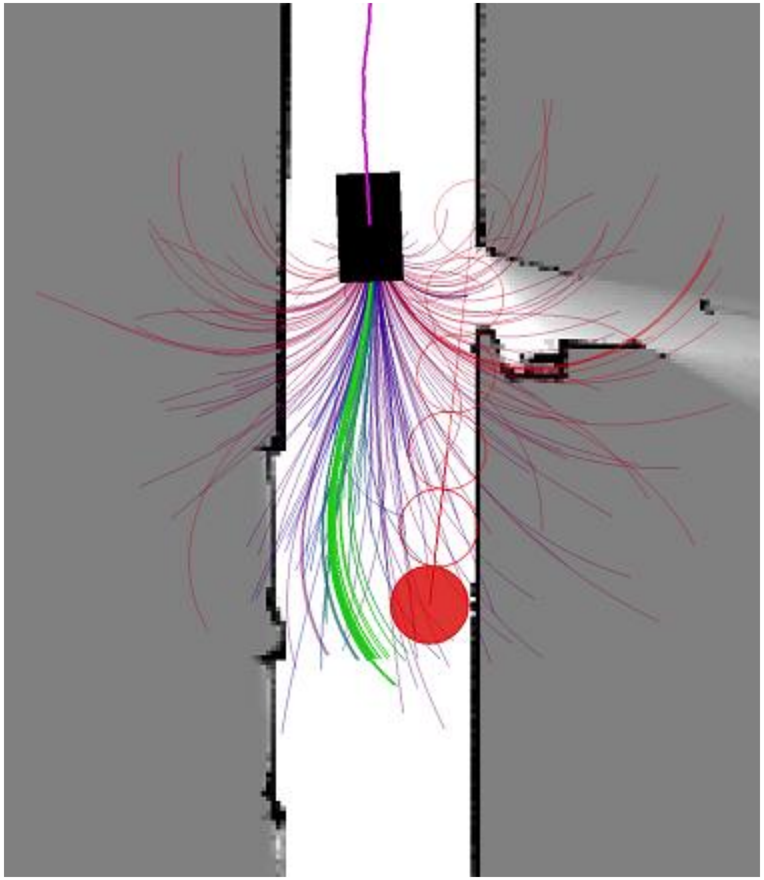
$$c_v v^2(j) + c_\omega \omega^2(j)$$

Expected Cost of a Trajectory Candidate

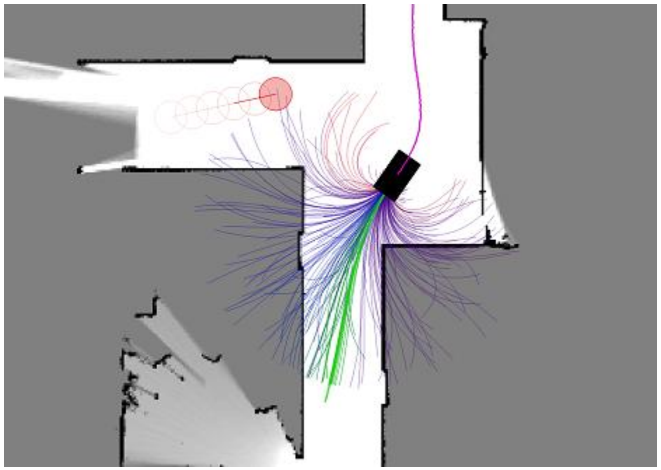
- The expected cost of a trajectory candidate is a probability-weighted time integral over [0*, T*]

- Probability weights create a smooth cost surface by setting physically meaningful soft boundaries around obstacles

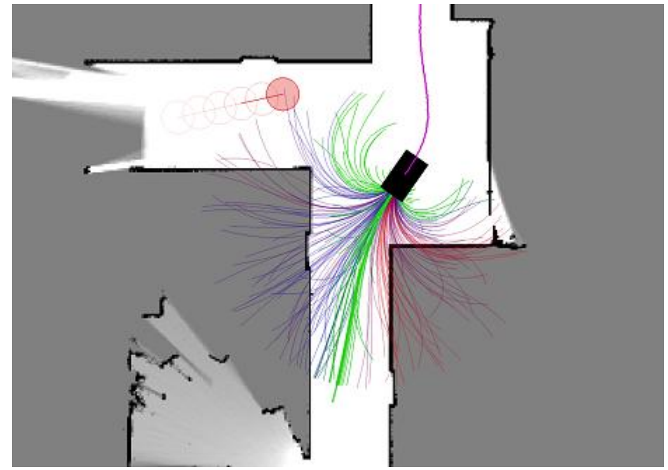
- Weights on action cost can be tuned to match user preferences



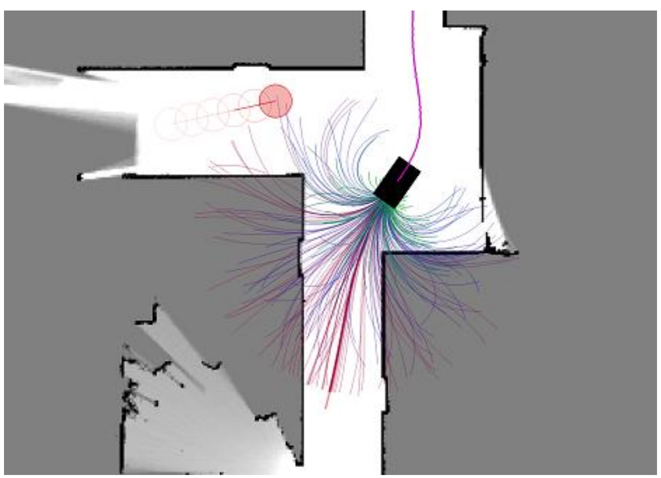
Expected Cost of a Trajectory Candidate



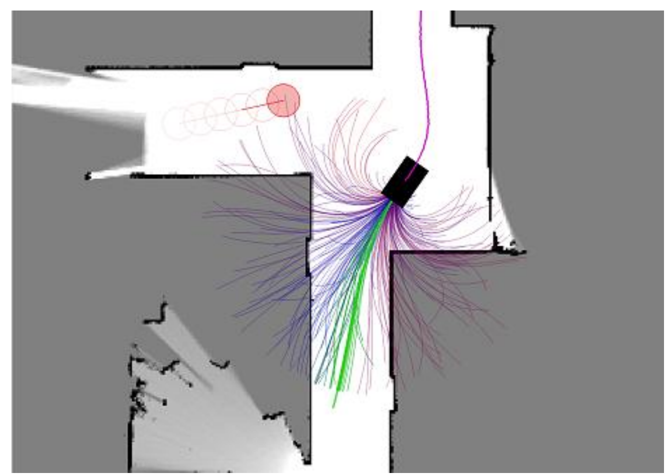
Progress



Collision



Action



Overall

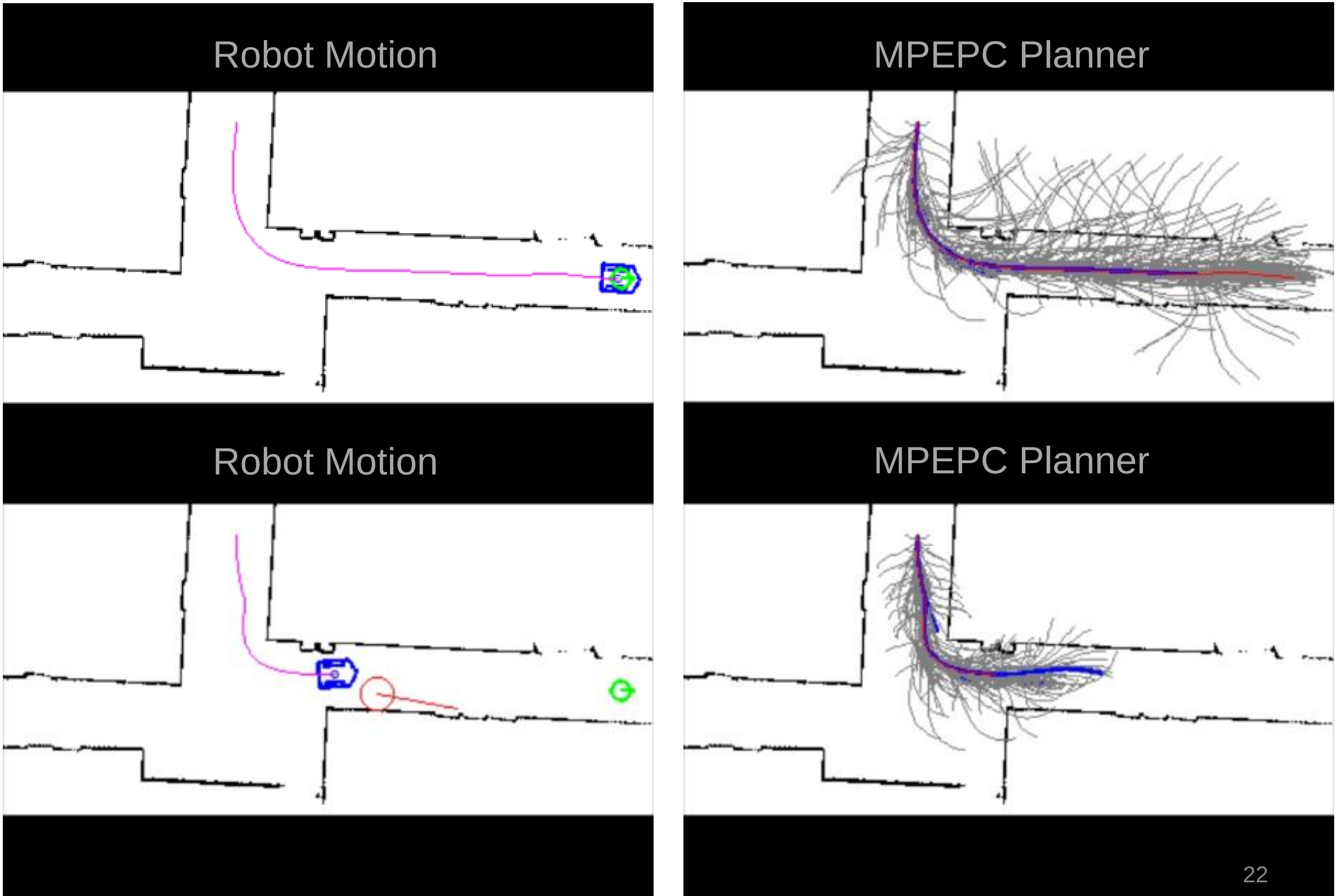
Our MPEPC Approach: Objectives

- Efficient generation of motion hypothesis and fine motion control
- Efficient evaluation of candidate trajectories, considering robot and pedestrian motion uncertainties
- **Implementation is easy and straightforward**
- **Action costs express user preferences**

Implementation is Straightforward

- Off-the-shelf optimization packages
  - Low-dimensional unconstrained optimization on continuous domain
  - No special post processing or optimization techniques
  - Real-time operation (C++)
- Two-phase optimization
  1. Coarse pre-sampling of the search space to find a good initial condition.
  2. Local gradient-based search from the best candidate from the pre-sampling phase.

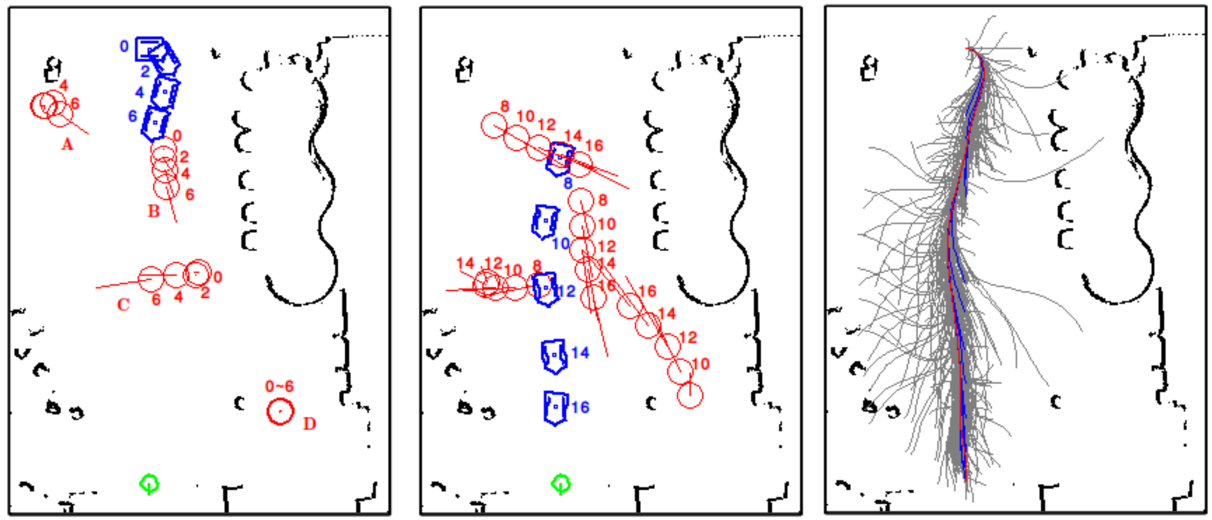
MPEPC in Action



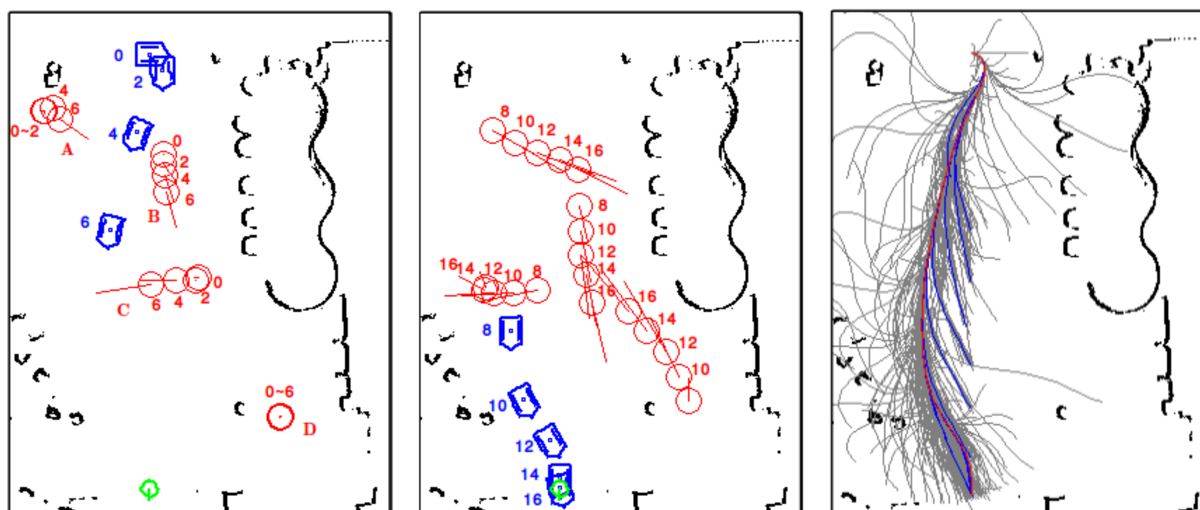
Different People Have Different Preferences

The proposed navigation algorithm handles multiple dynamic objects.
We can shape robot behavior by changing weights in action cost.

Moving slowly in a cluttered hall with multiple pedestrians (high weights on action cost)



Moving aggressively in a cluttered hall with multiple pedestrians (low weights on action cost)



Initial Tests on a Physical Platform

Navigation is a Constant Decision-Making Process

- The navigation problem can be factored by decomposing the task in the hierarchical architecture.
- The search for the optimal trajectory can be made easier by integrating planning and control.
- Motion uncertainties need to be considered explicitly.
- What do they teach in driving school?

Navigation is a Constant Decision-Making Process

- The navigation problem can be factored by decomposing the task in the hierarchical architecture.
- The search for the optimal trajectory can be made easier by integrating planning and control.
- Motion uncertainties need to be considered explicitly.
- Identify, predict, decide and execute.
  - Minimize the probability that you might get in trouble, while progressing along the road.

Conclusion

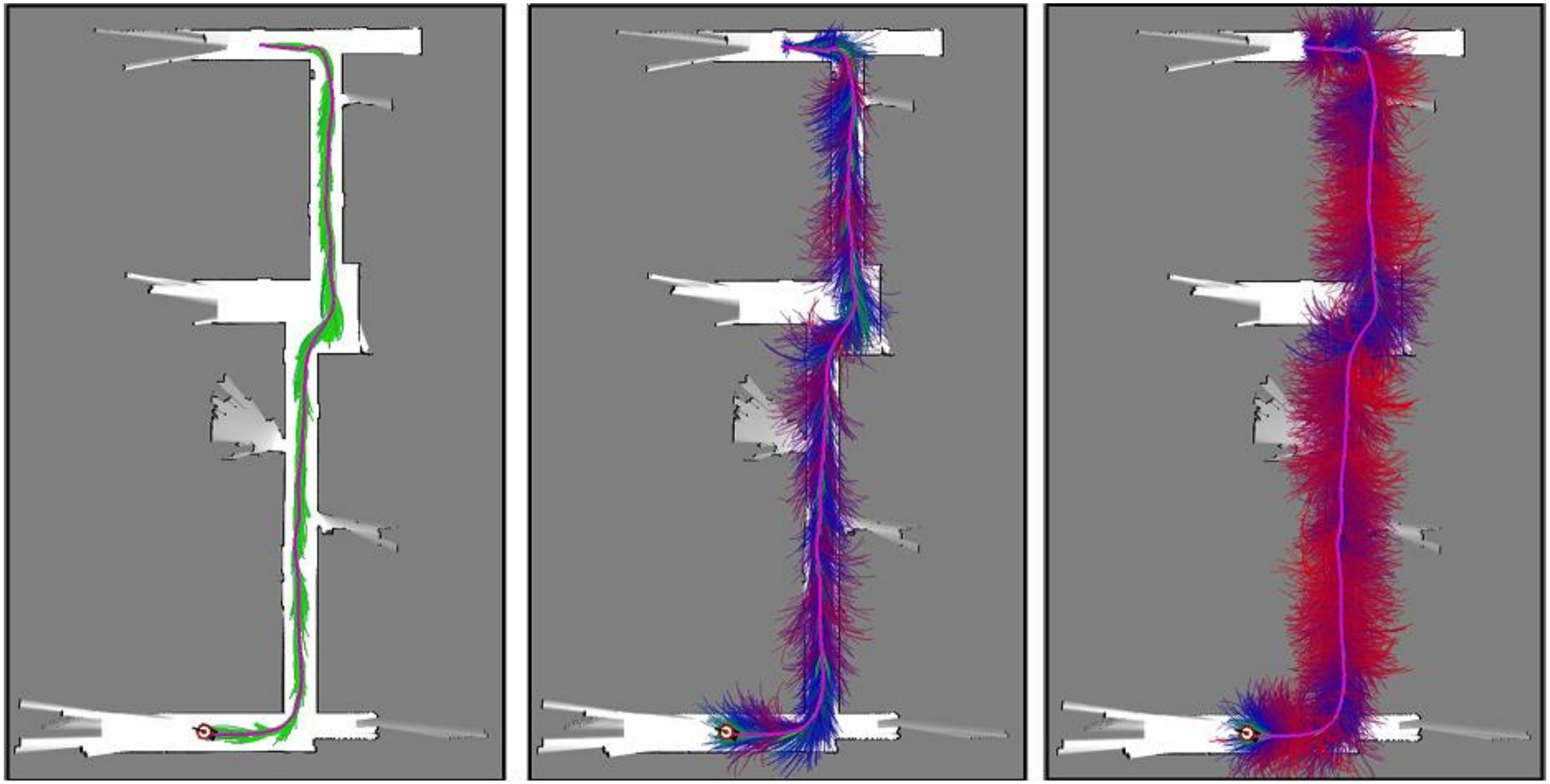
- We provide a compact representation of a space of smooth and realizable trajectories.

- We formulate local motion planning as an unconstrained optimization problem by computing expected values, using probability weights.

- The formulation allows straightforward low-dimensional optimization on a continuous domain.

- We have simple, easy to understand tunable parameters for qualitative robot behavior.

Thank You



References

[1] Jong Jin Park, Collin Johnson and Benjamin Kuipers, "Robot navigation with Model Predictive Equilibrium Point Control", *IROS-12*

[2] Jong Jin Park and Benjamin Kuipers, "A smooth control law for graceful motion of differential wheeled mobile robots in 2D environment", *ICRA-11*

[3] Knepper and Mason, "Path diversity is only part of a problem", *ICRA-09*

[4] Jong Jin Park and Benjamin Kuipers, "Graceful navigation via model predictive equilibrium point control (MPEPC) in dynamic and uncertain environments", *in preparation*.

[5] Ogren and Leonard, "A convergent dynamic window approach to obstacle avoidance", *IEEE Trans. Robot.,* 2005

[6] Hundelshausen, Himmelsbach, Hecker, Mueller and Wuensche, "Driving with Tentacles: Integral structures for sensing and motion", *J. Field. Robot.,* 2008

[7] Knepper and Mason, "Real-time informed path sampling for motion planning search", *IJRR*, 2012