Intelligent robot cleaner

Stanislav Slušný[1] Michal Zerola[2]

January 8, 2010

[1] Institute of Computer Science
[2] Nuclear Physics Institute

Table of Contents

Contents

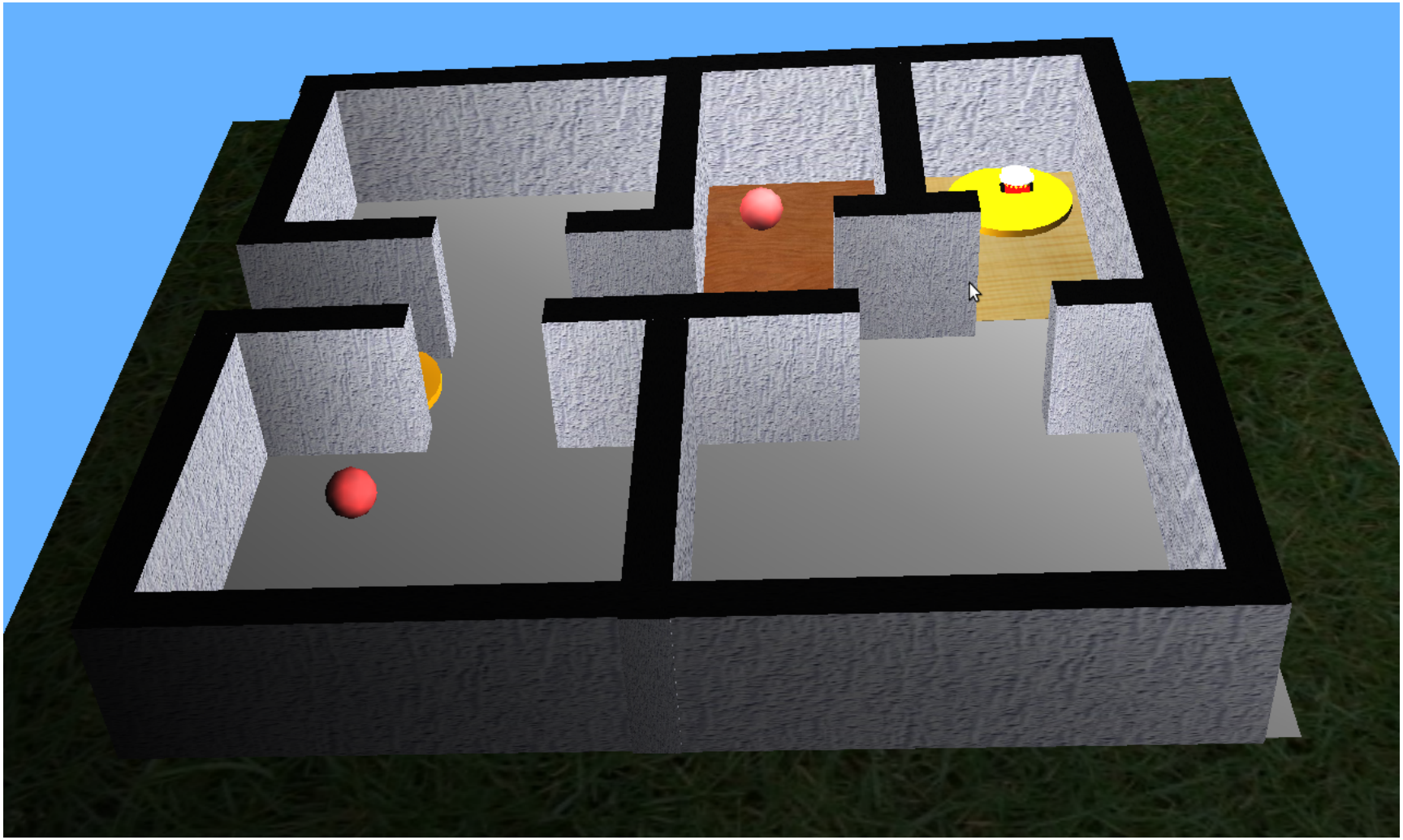
Contents

Contents

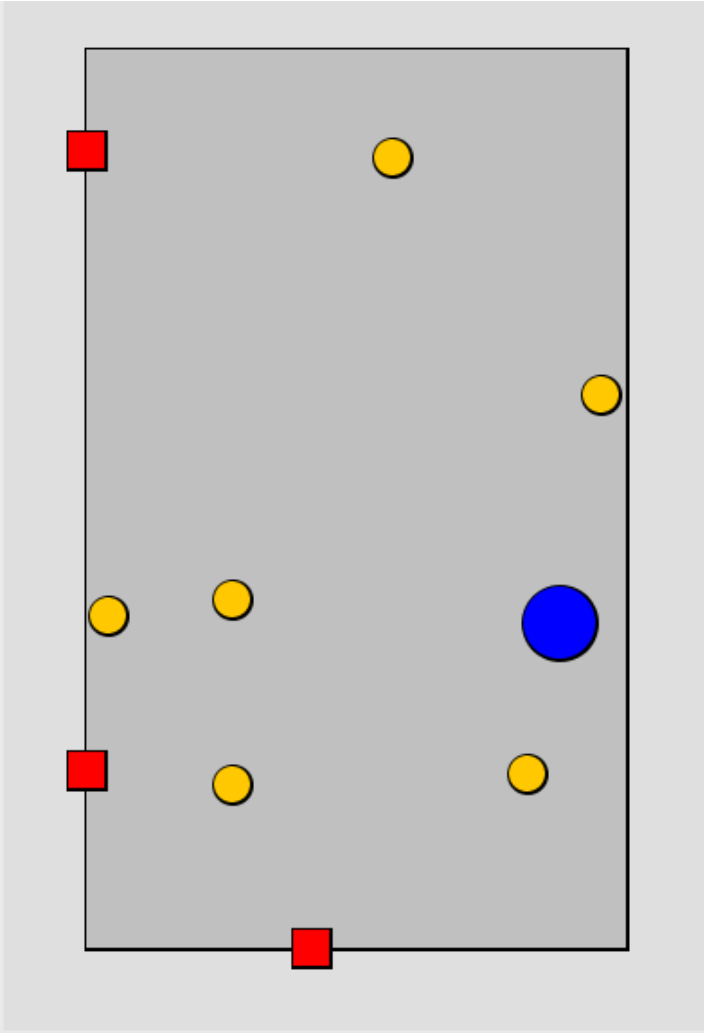
Contents

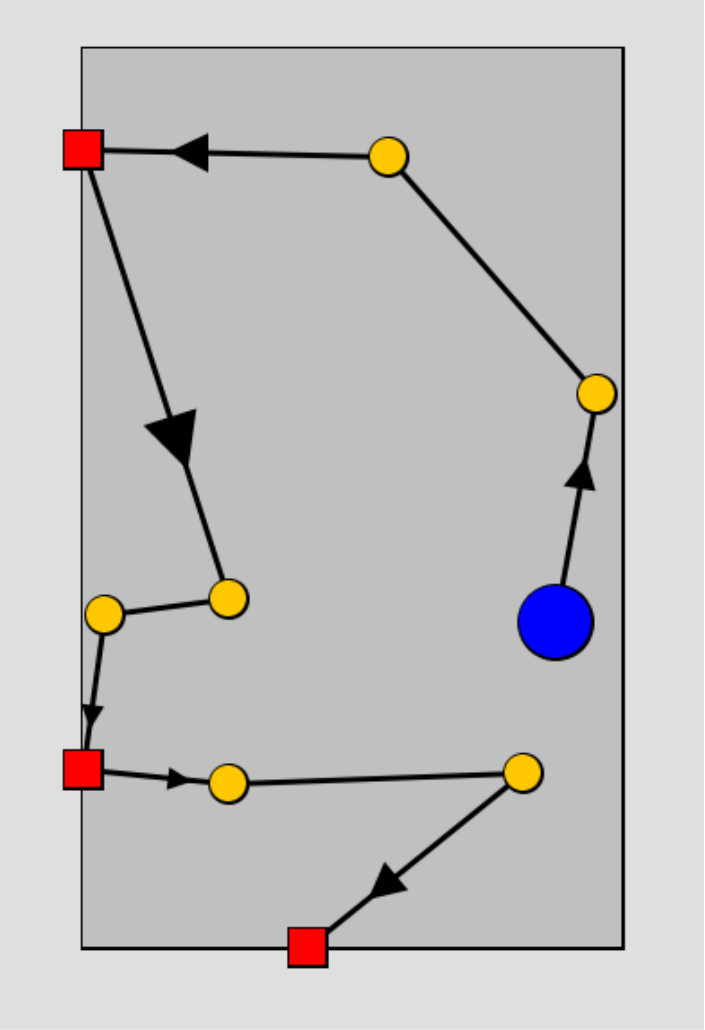
Contents

The problem

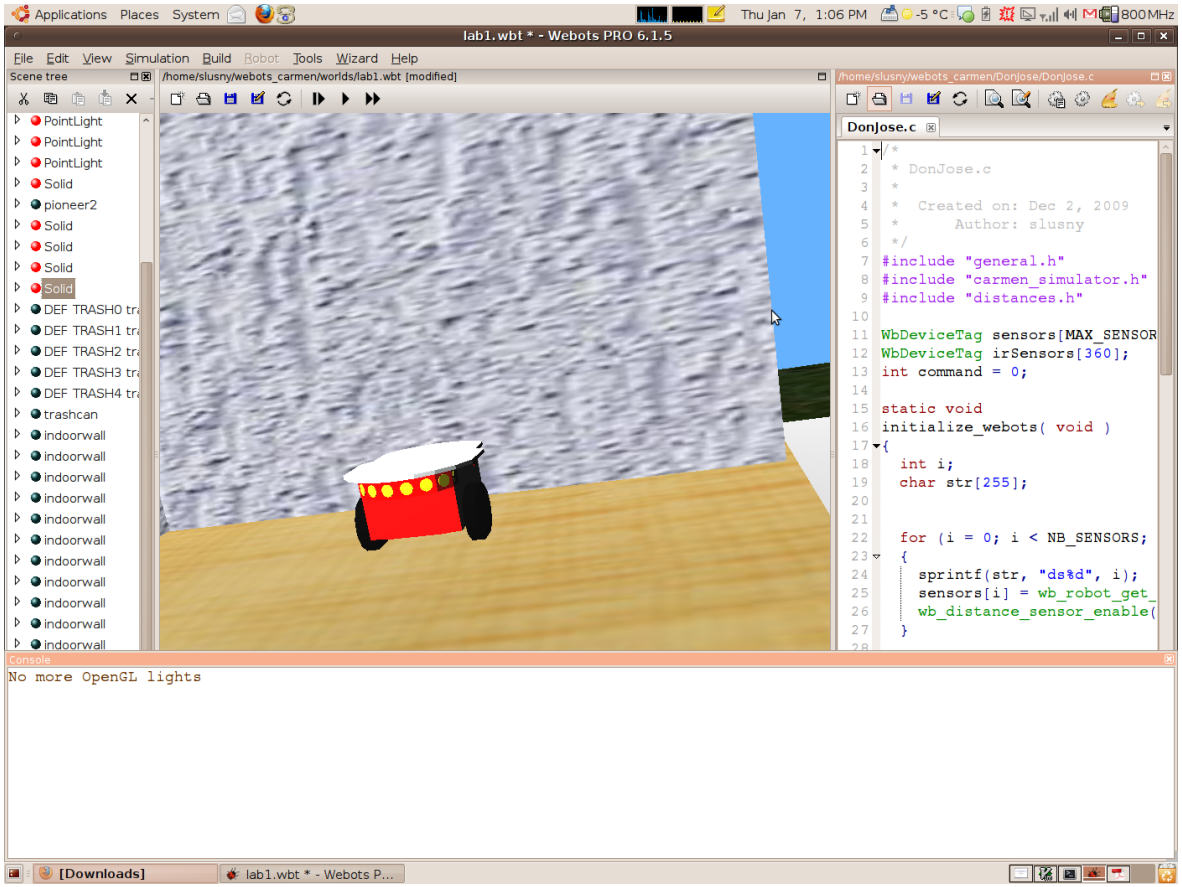


Plan - the solution

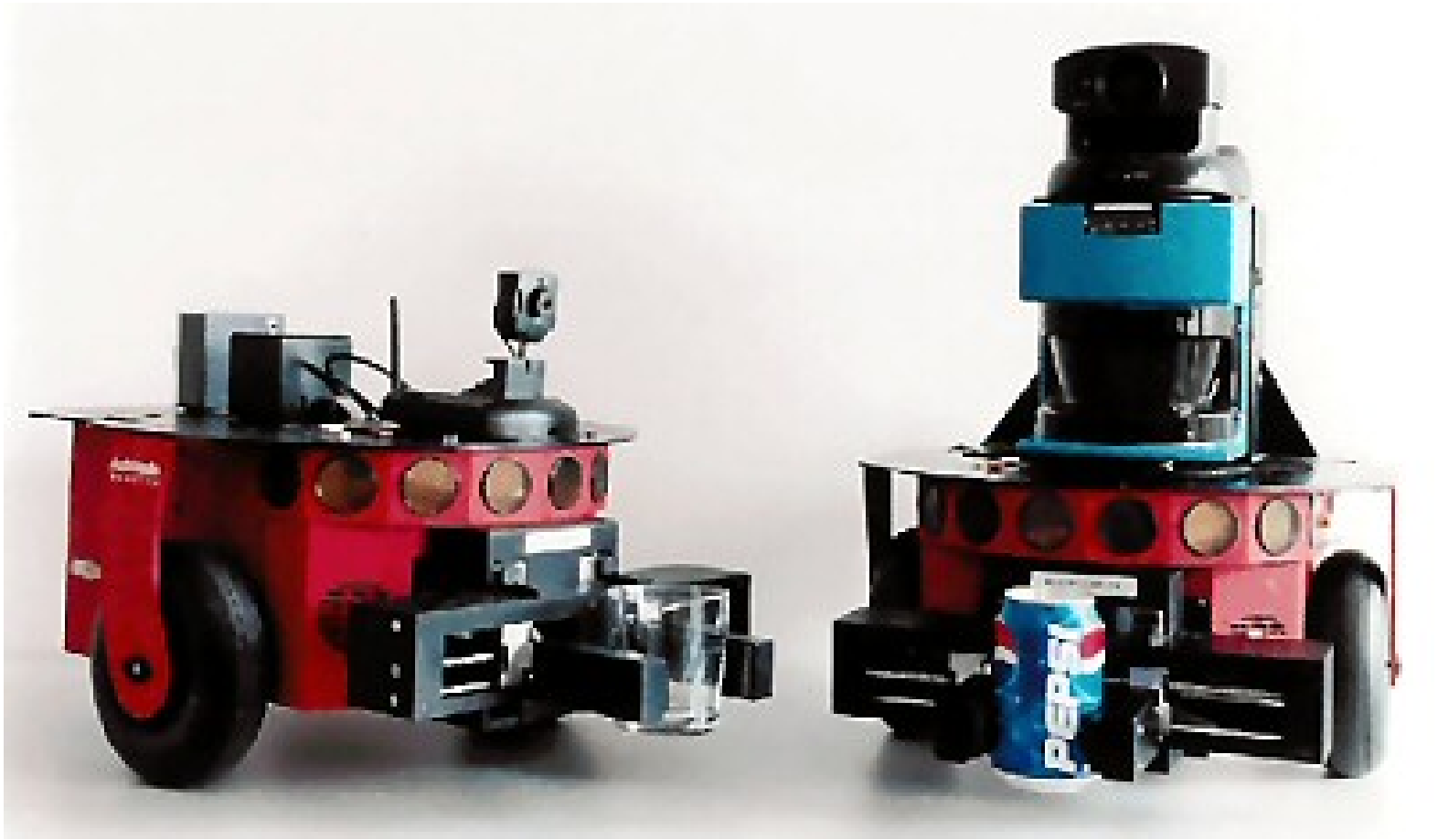




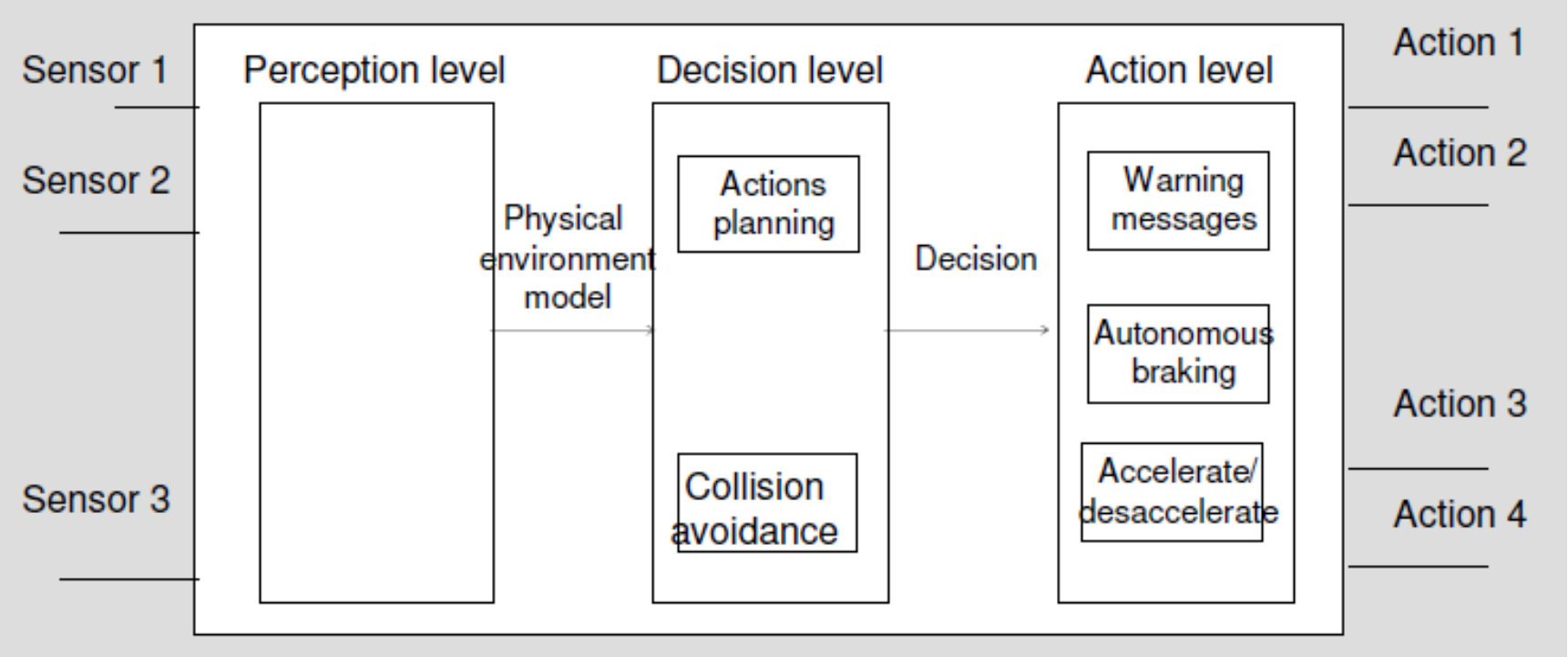
Webots - professional simulator



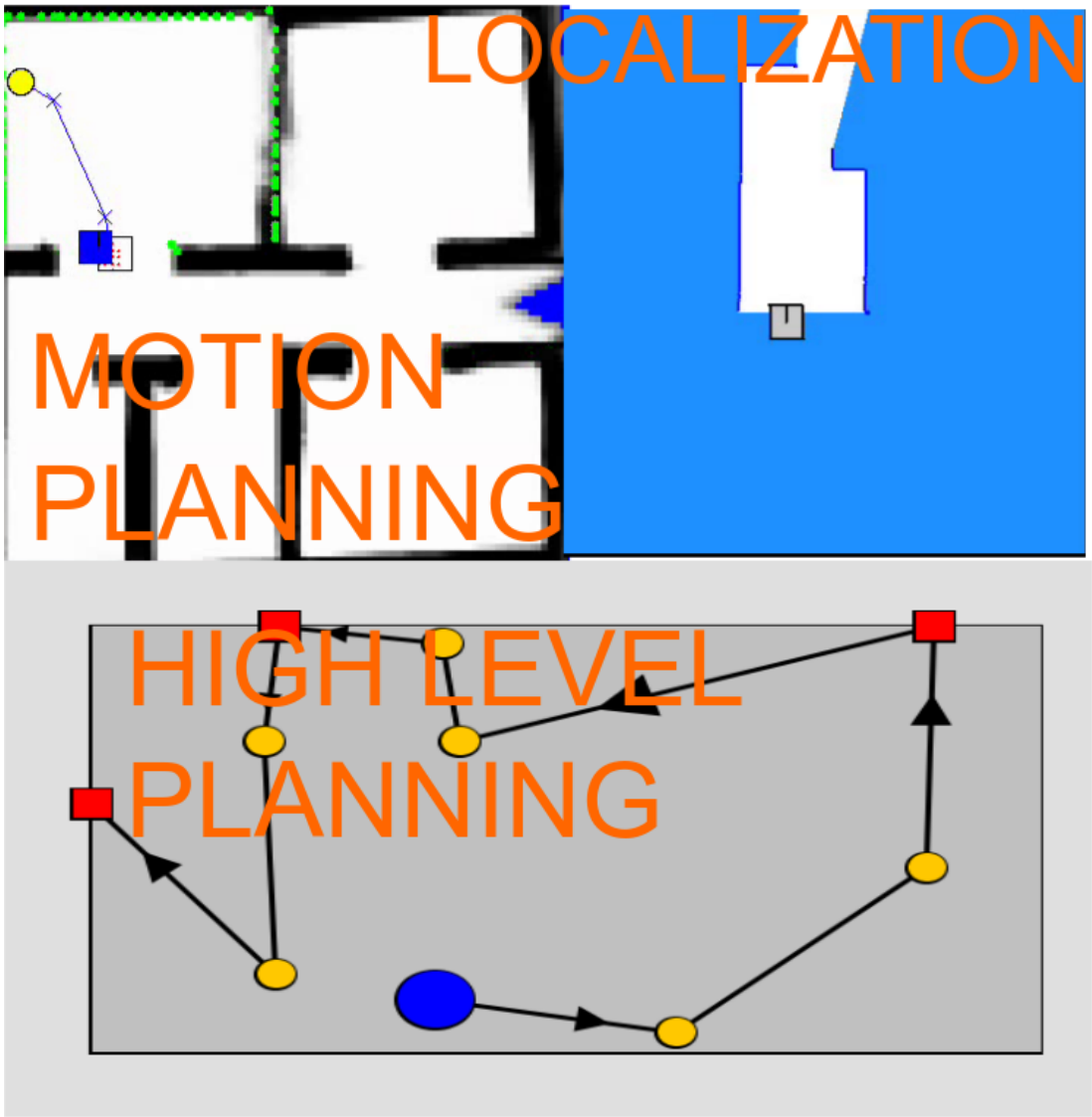
Pioneer-2



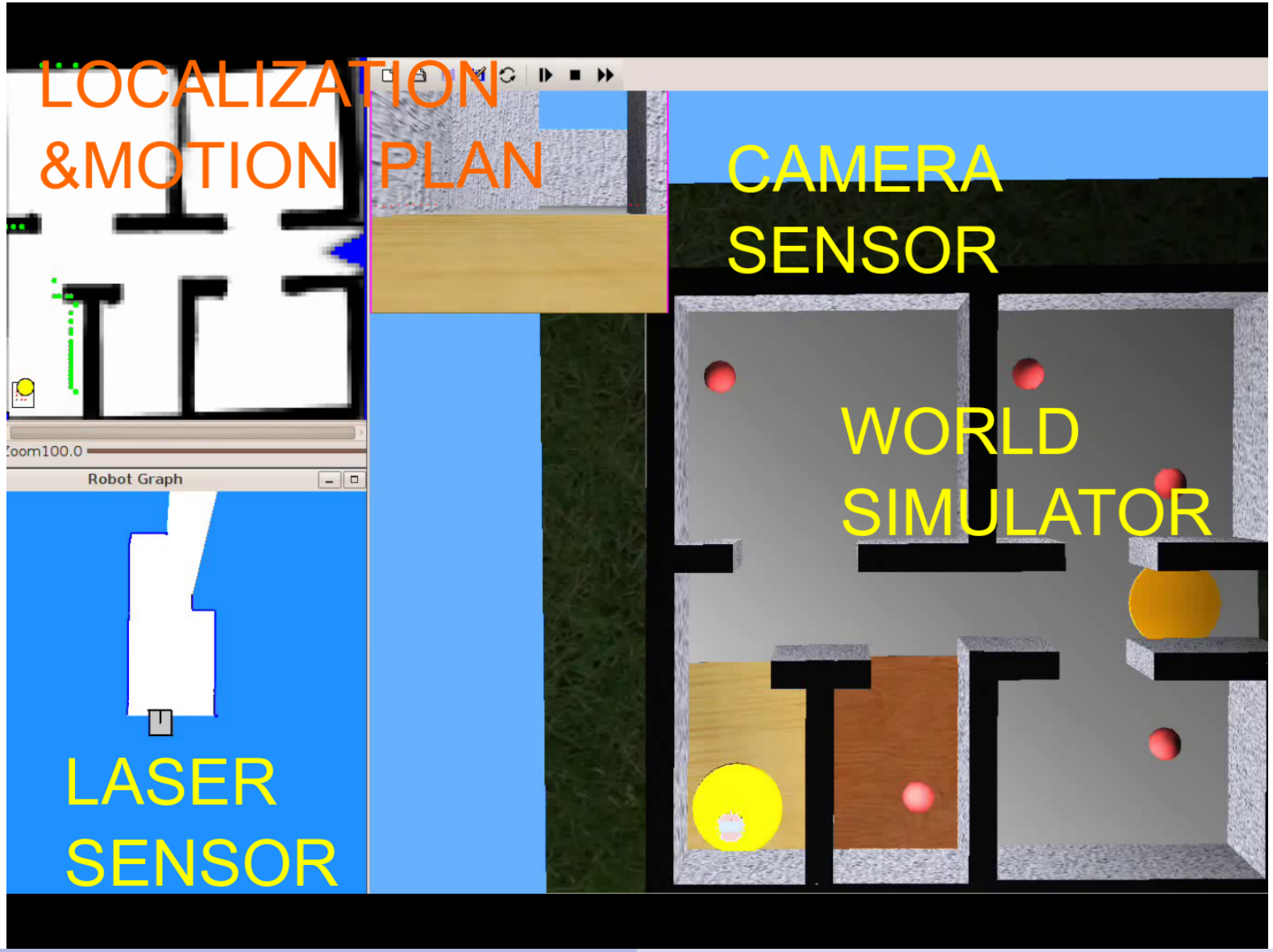
Robot planning



Included subproblems

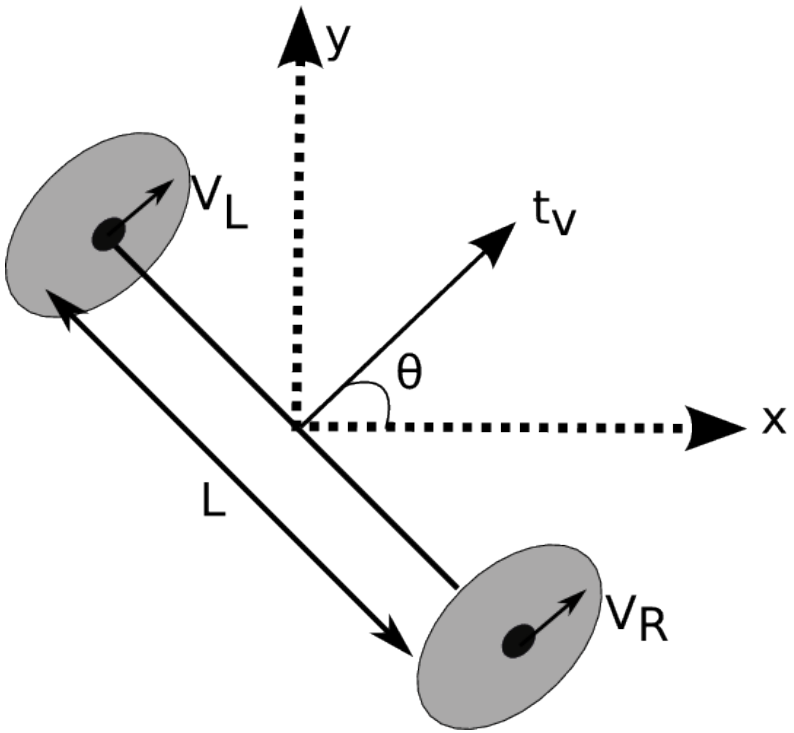


Demo


LOCALIZATION
&MOTION
PLAN
CAMERA
SENSOR
WORLD
SIMULATOR
Robot Graph
LASER
SENSOR

Dead reckoning

Differential drive:



$$\begin{pmatrix} x \\ y \\ \theta \end{pmatrix} = \begin{pmatrix} x_{OLD} \\ y_{OLD} \\ \theta_{OLD} \end{pmatrix} + \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta\theta \end{pmatrix} \quad (1)$$

$$\Delta\theta = \frac{\Delta s_R - \Delta s_L}{L} \quad (2)$$

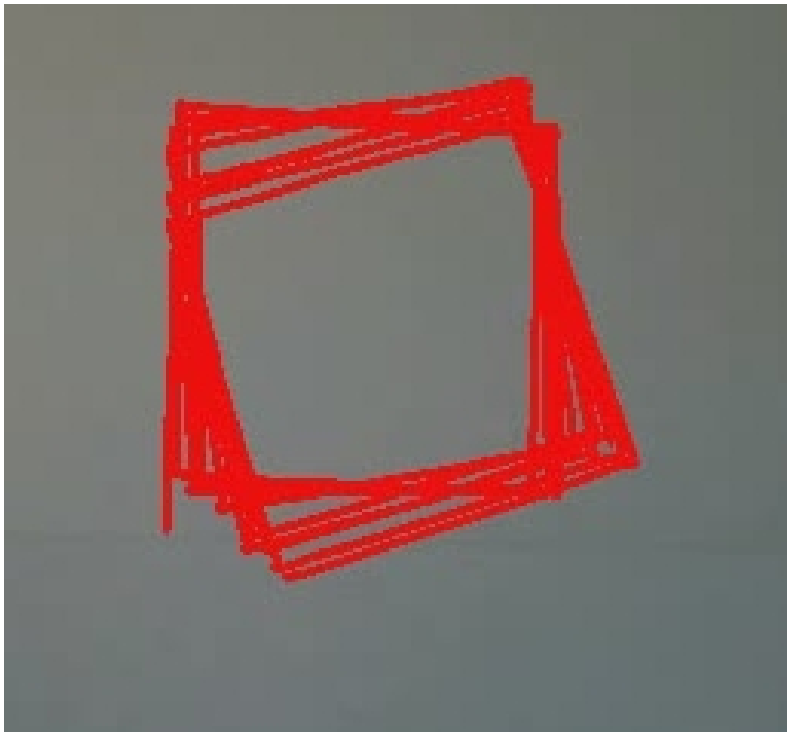
$$\Delta s = \frac{\Delta s_R + \Delta s_L}{2} \quad (3)$$

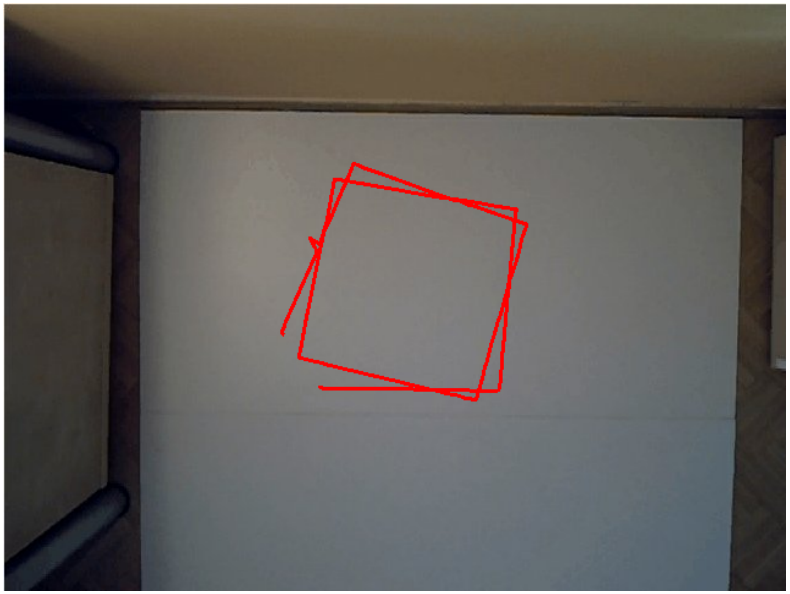
$$\Delta x = \Delta s.cos(\theta + \frac{\Delta\theta}{2}) \quad (4)$$

$$\Delta y = \Delta s.sin(\theta + \frac{\Delta\theta}{2}) \quad (5)$$

Reality

Make 10 squares of size 30 cm:





Monte-Carlo Localization

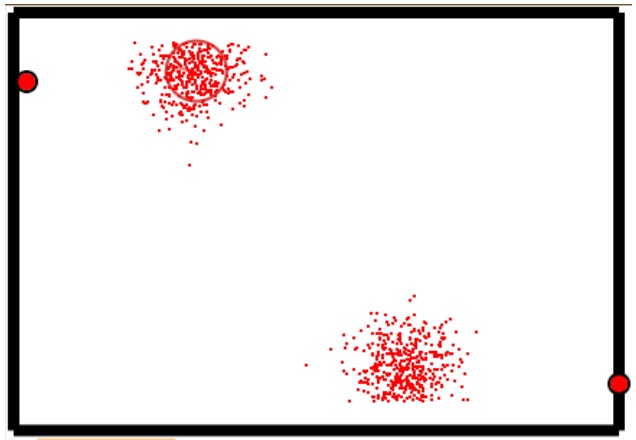
Takes command u_t and observation z_t, updates probability of robot location $p(x_{t-1})$

Input:

- motion model $p(x_t|x_{t-1}, u_t)$
- sensor model $p(z_t|x_{t-1})$

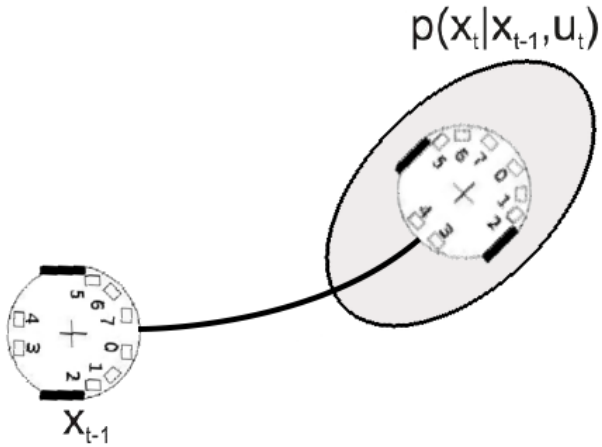
Output:

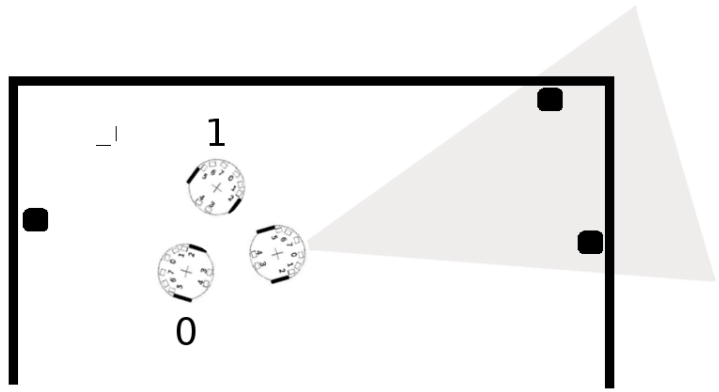
- $p_t(x)$



Monte-Carlo Localization

$p(x_t)$ - pblity, that robots is located at the position x_t in time t
u_t - control at time t
3 steps:





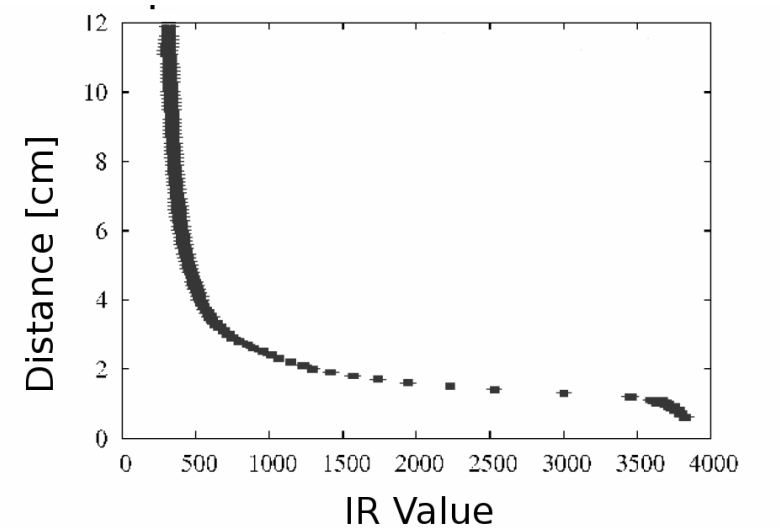
1. State prediction based on odometry

2. Correction step - Observation integration

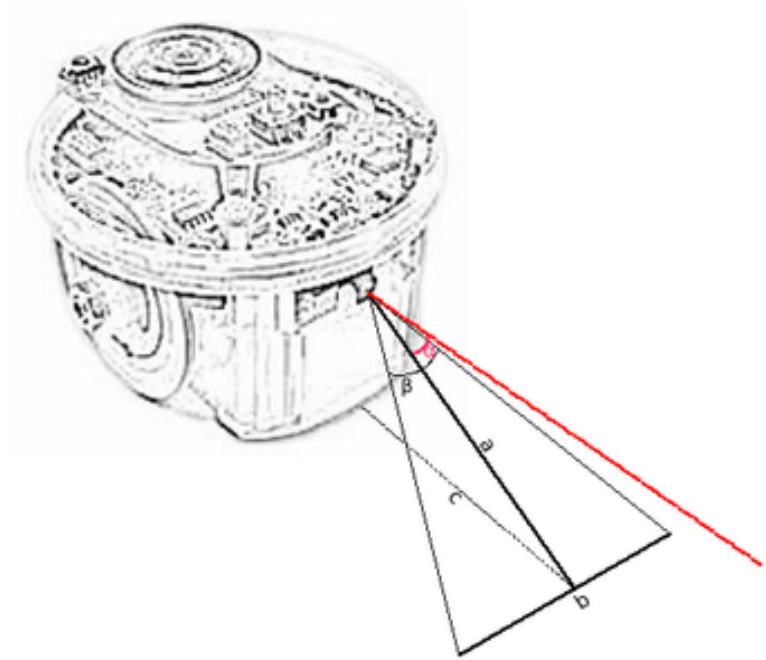
Third step is resampling.

2. Correction step - Observation integration

Distance sensors - bumpers only



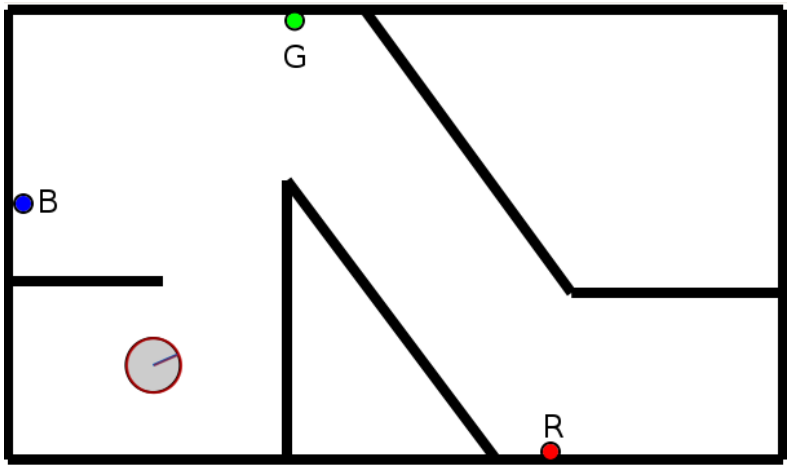
Camera - landmark detection



importance factor $w_t^{[m]}$: pbability of the measurement z_t under particle $x_t^{[m]}$, given by $w_t^{[m]} = p(z_t|x_t^{[m]})$.

Landmark detection and sensor fusion

Compute weight $w_t^{[m]}$ from detected contradictions:



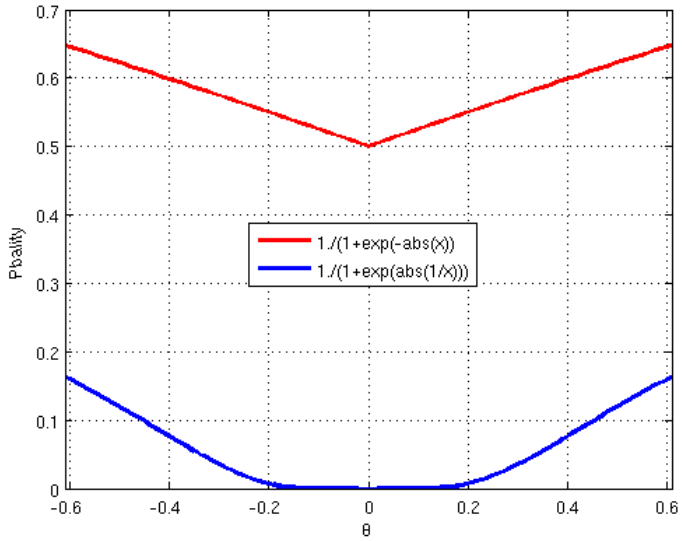
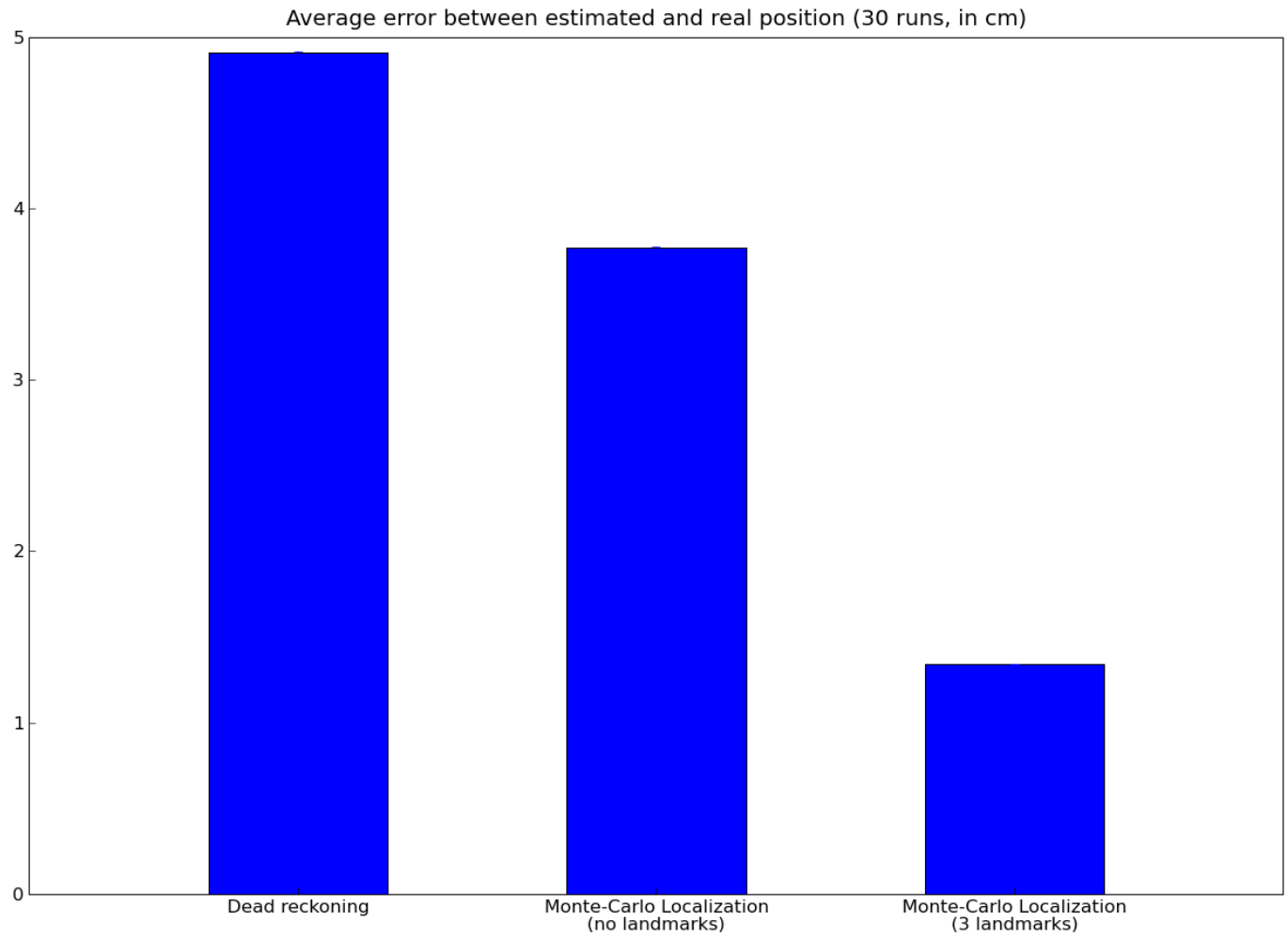


Figure: $p(z_t|x_t)$

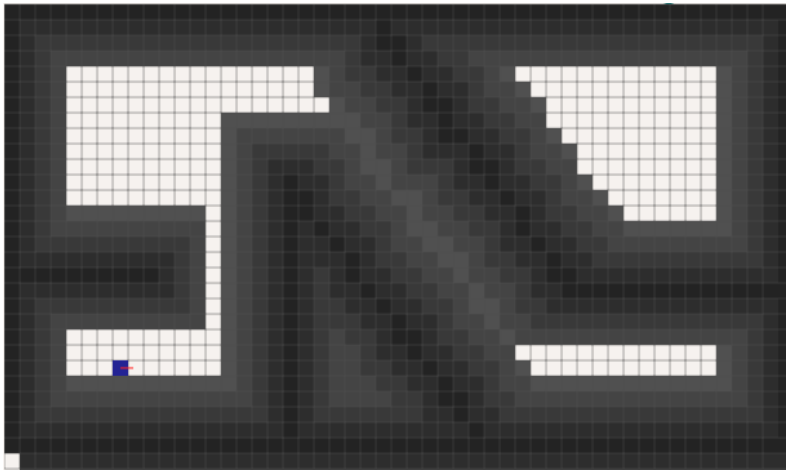
1. I should see the landmark (angle θ), but I cannot see it.
2. I should not see the landmark, but I see it (angle θ).
3. I should be near the wall but I am not
4. I am near the wall but I should not

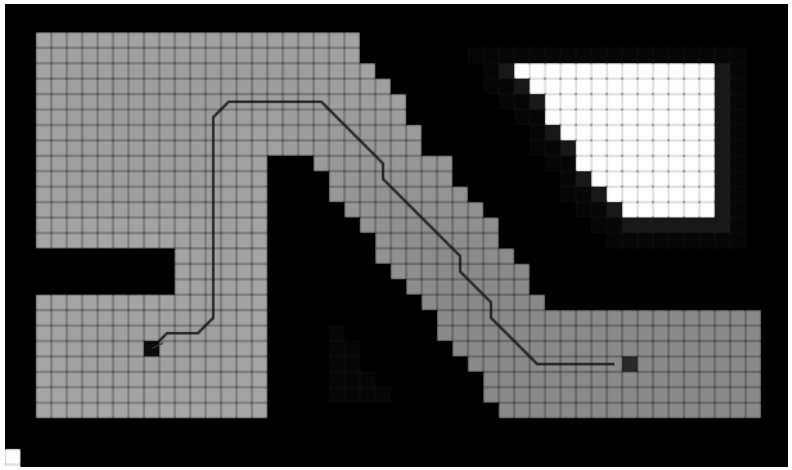
Localization - average errors



Motion planning

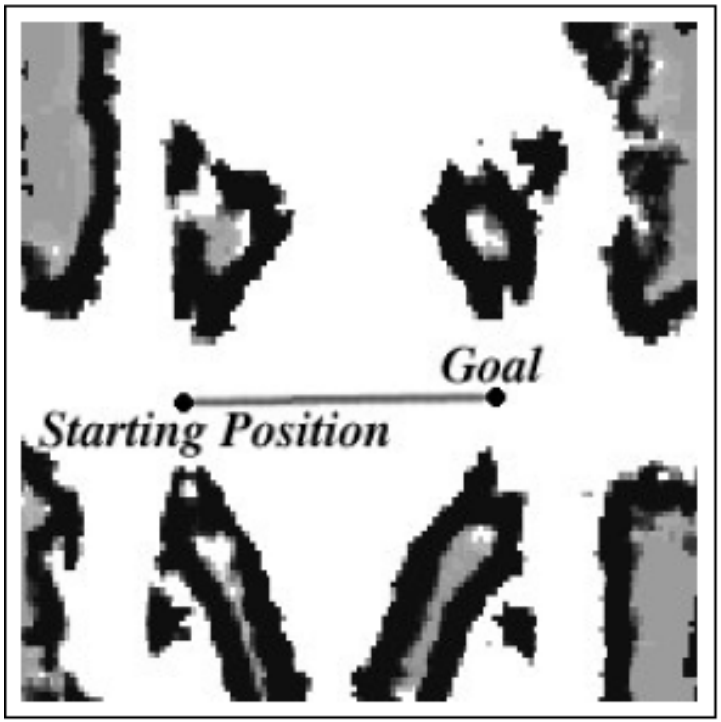
Planning with uncertainty:
Value iteration on occupancy grid (2x2 cm):

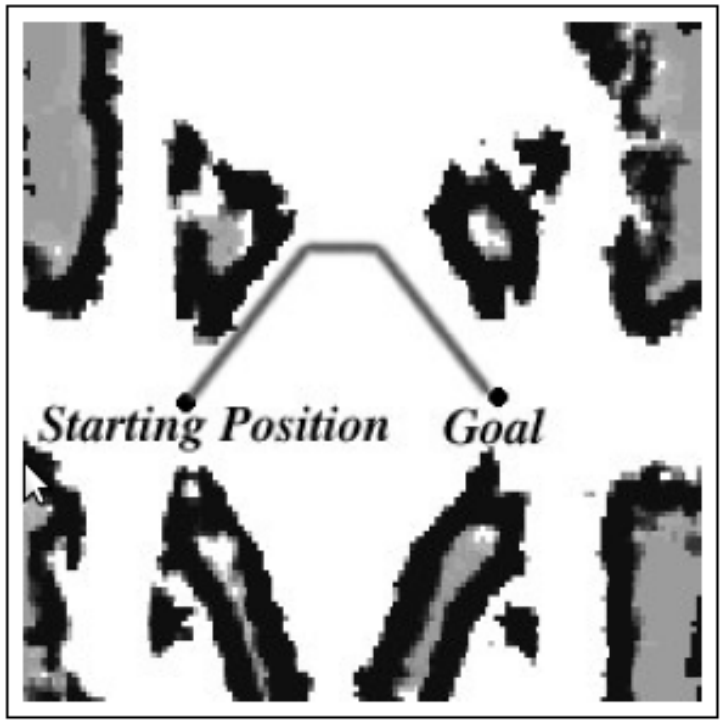




... plan your movements in order to arrive at goal with minimum uncertainty

Nick Roy: coastal navigation

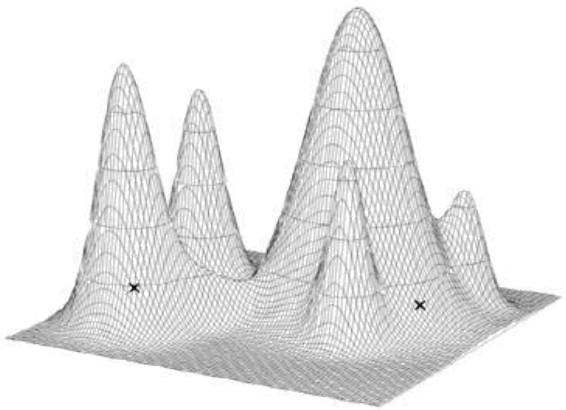




Local search - hill climber

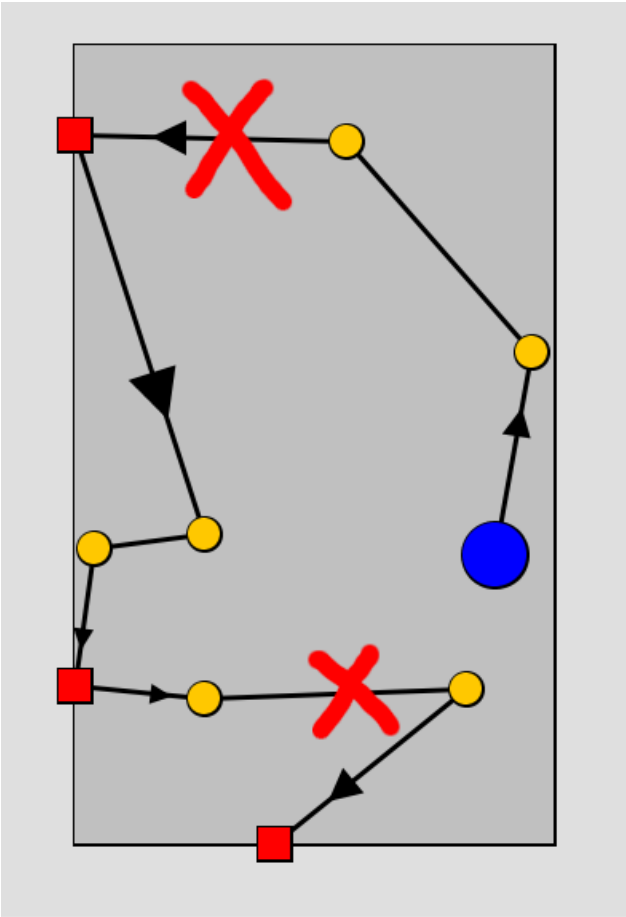
- Generate initial solution.
- Use neighborhood operator to generate new solution.
- If it is better, move there.
- Can fall into local optima
- Simulated annealing.

Beam search - hill climber



- Generate set of initial population
- Use neighborhood operator to generate new population
- Better solution substitutes its ancestor.
- More robust to local optima.

Neighborhood operator



Input: Previous plan.

Output: New (possibly) better plan.

Algorithm:

- Randomly choose predetermined number of edges, and delete them from the plan.
- Remaining edges are fixed (CP cannot touch them).
- Start CP to compute optimal plan from this partial plan.

Local search - pros and cons

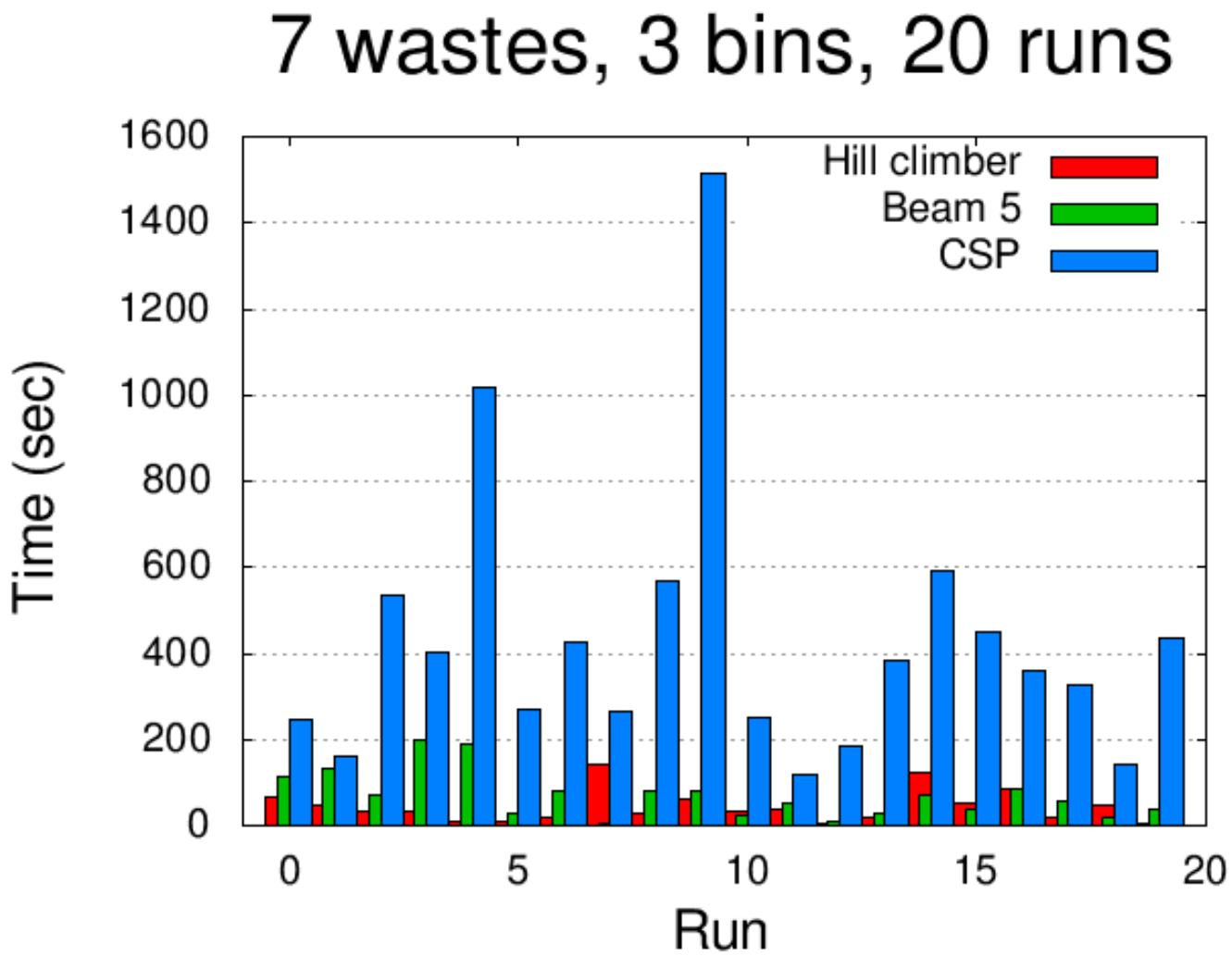
Local search methods

- Anytime planners
- Convergence to local optimum only
- Based on randomness

Global search methods (CP)

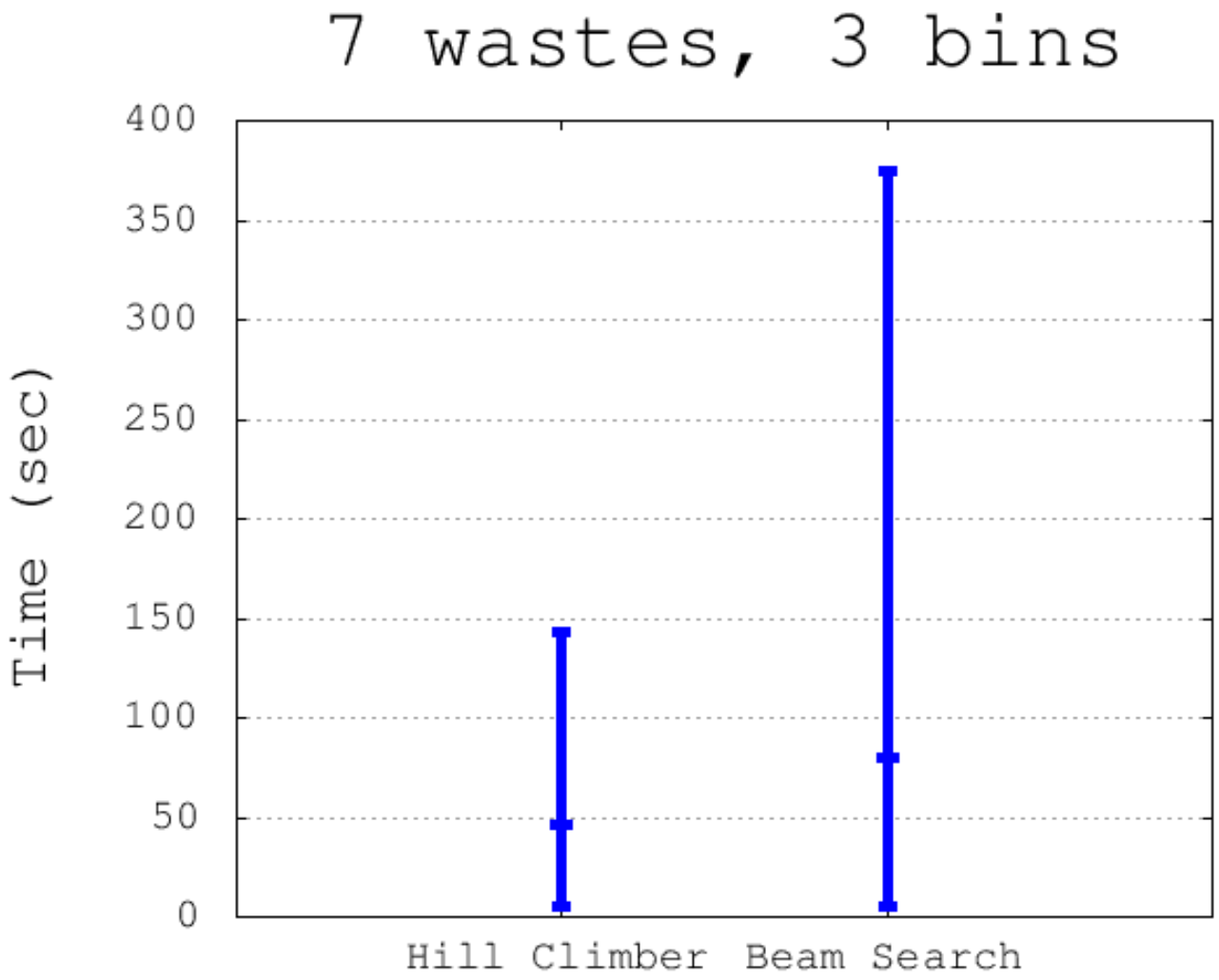
- Search for optimal solution
- Scalability
- Deterministic

Hill climbers vs Beam search vs CP (time)

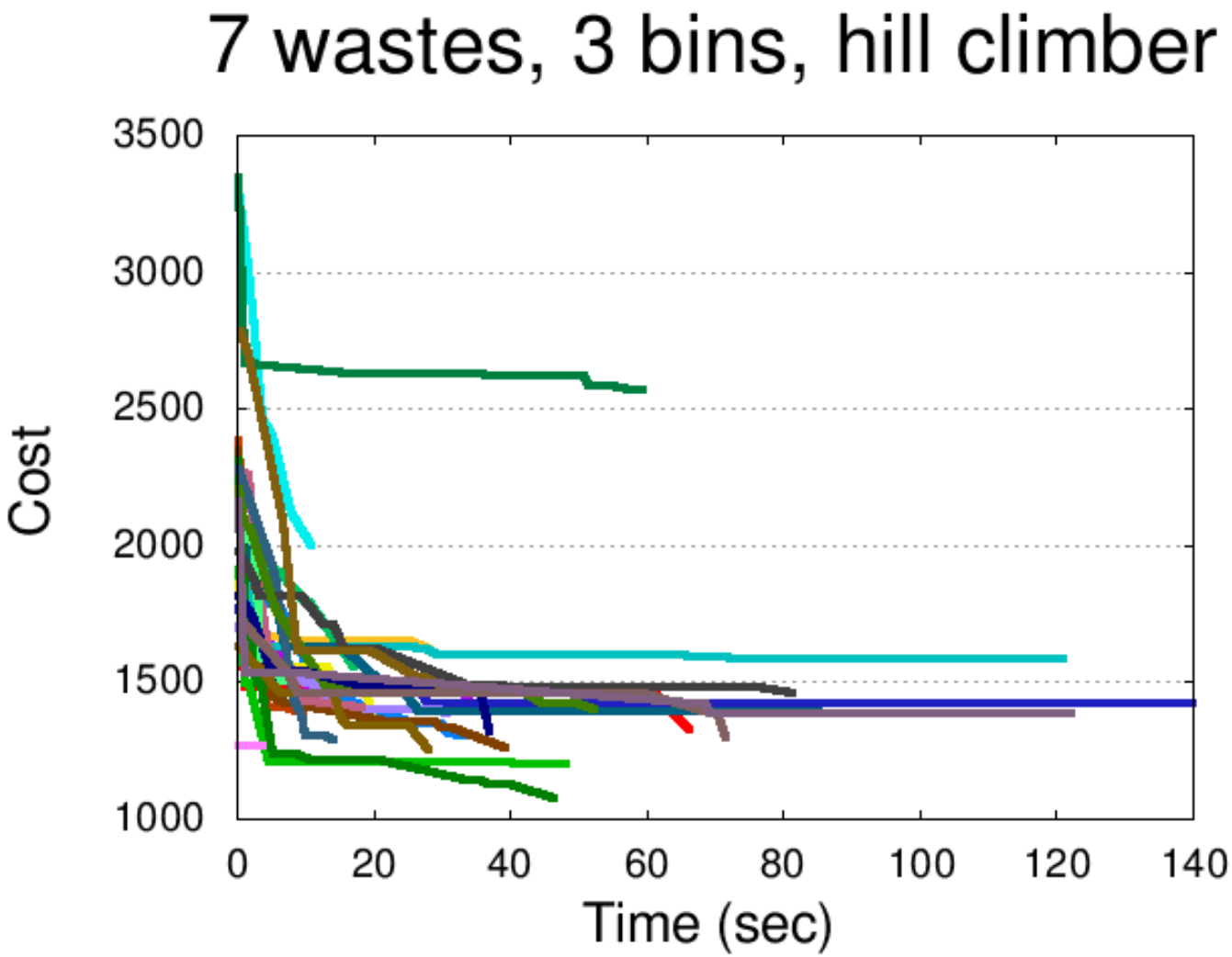


Hill climbers vs Beam search

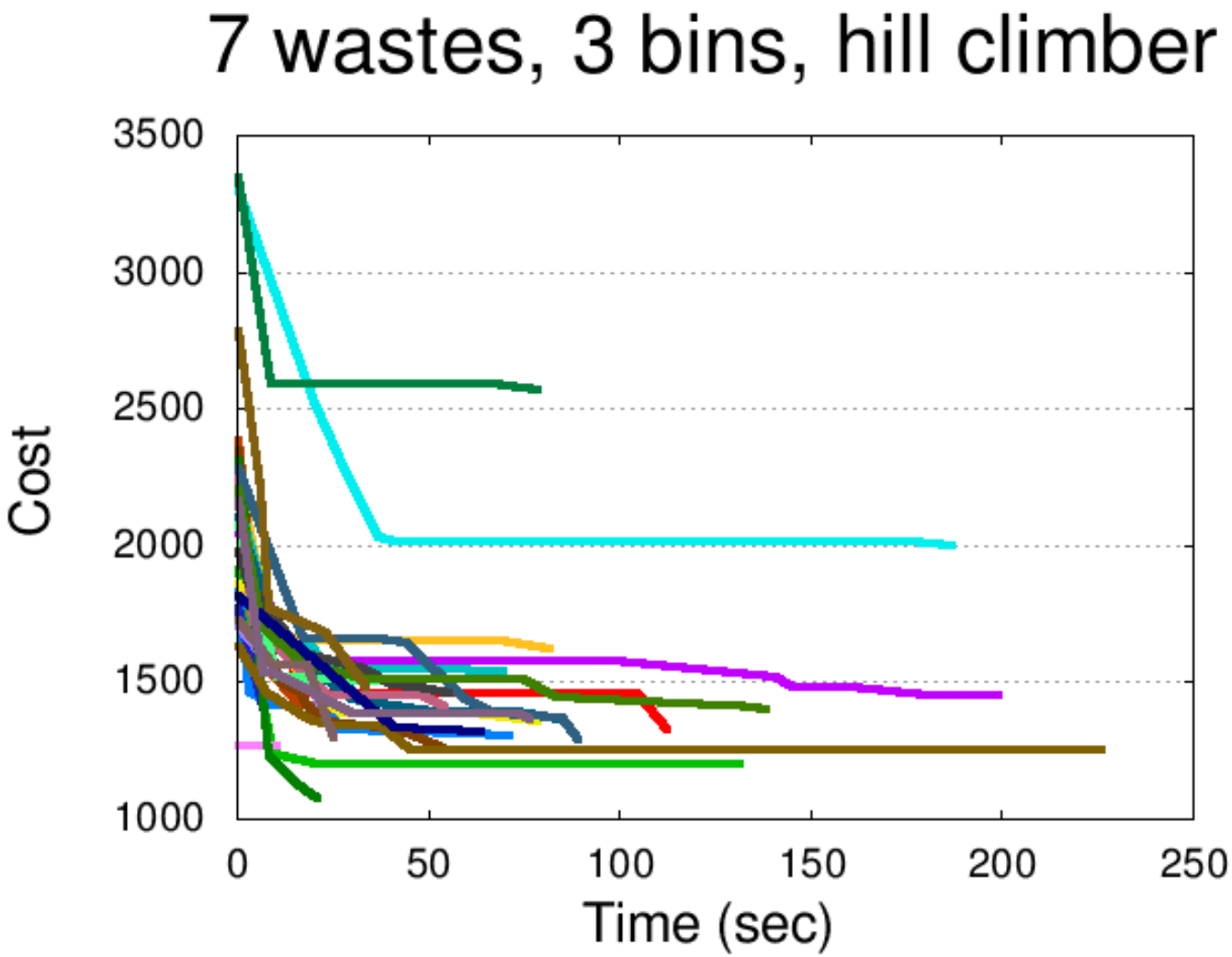
Hill climbers vs Beam search (average time)



Hill climbers - time to compute optimal solution



Beam search - time to compute optimal solution



Discussion

Beam search

- Always reached global optimum.
- Slower.
- More robust.
- Always outperformed CP.

Hill climbers

- 6/50 did not reach global optimum.
- Faster.
- If converged, outperformed CP.

Conclusions

- Global search is not scalable enough.
- Local search helped to solve bigger problems.
- Surprisingly good results with very simple operators.
- Anytime planners - suitable for robotics.

Future work

- More clever operators.
- Bigger problems.