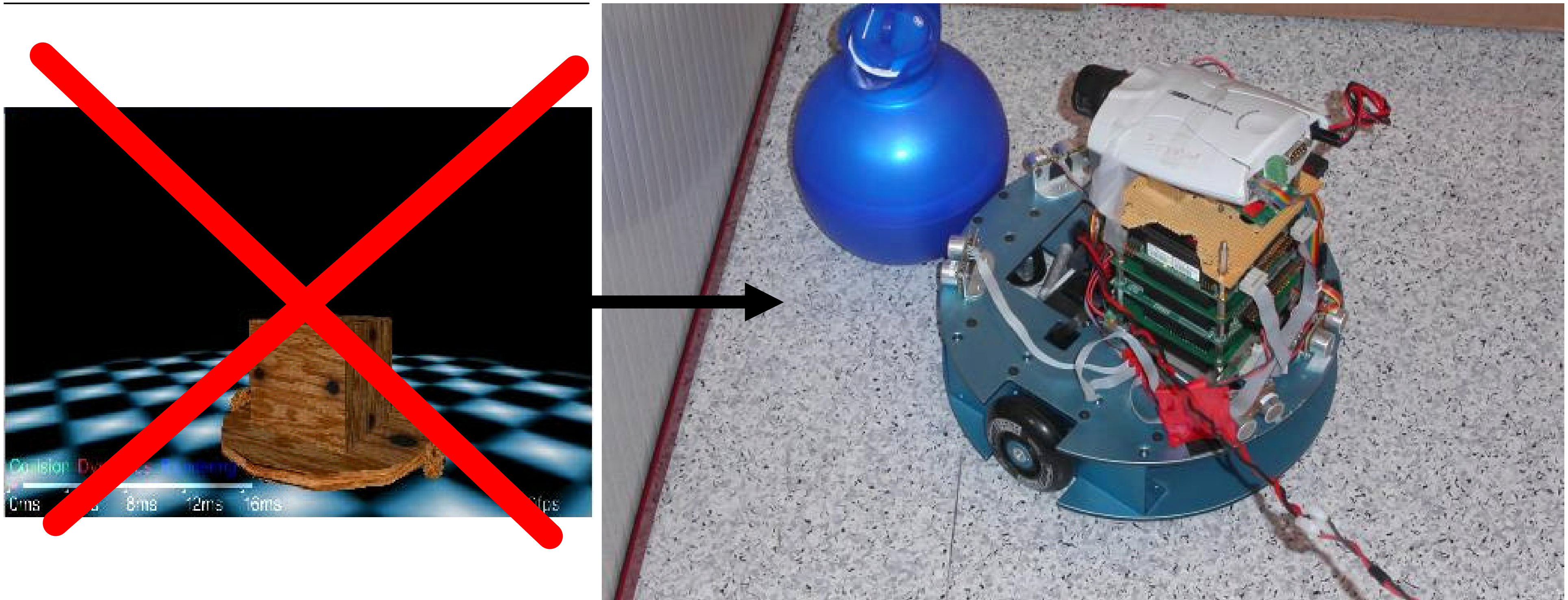
Metric State Space Reinforcement Learning for a Vision-Capable Mobile Robot

Viktor Zhumatiy[a], Faustino Gomez[a],
Marcus Hutter [a] and Jürgen Schmidhuber[a,b]
(a) *IDSIA, Switzerland*
(b)*TU Munich, Germany*

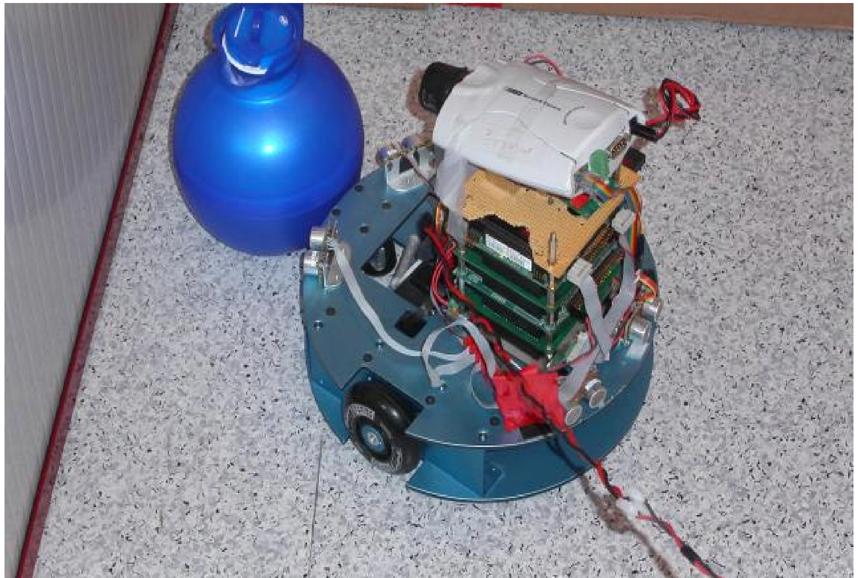
Learning environment: real robot, vision sensors

- Learning algorithm specifically targeted at real world mobile robots



Challenges of learning on vision-capable real robots

- Piecewise-continuous (PWC) control
- Partial observability (POMDP)
- High-dimensional sensors
- Costly exploration



Reinforcement learning

- RL: policy learning by autonomous environment exploration from reward signal
- Q-learning: estimation of discounted reward for each state-action pair

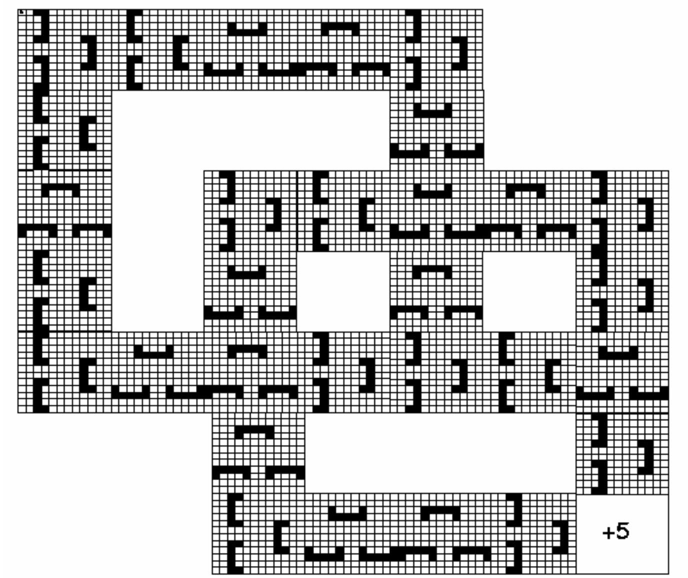
$$Q_{t+1}(s_t, a_t) = (1-\alpha)Q_t(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q_t(s_{t+1}, a)]$$

- Assumes that states s_t are fully observable at each moment
- In practice, only incomplete observations are available

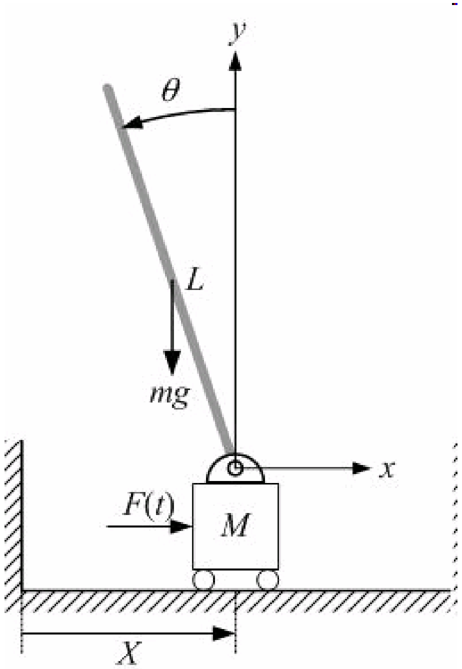
[redacted]

Discrete and continuous versus PWC

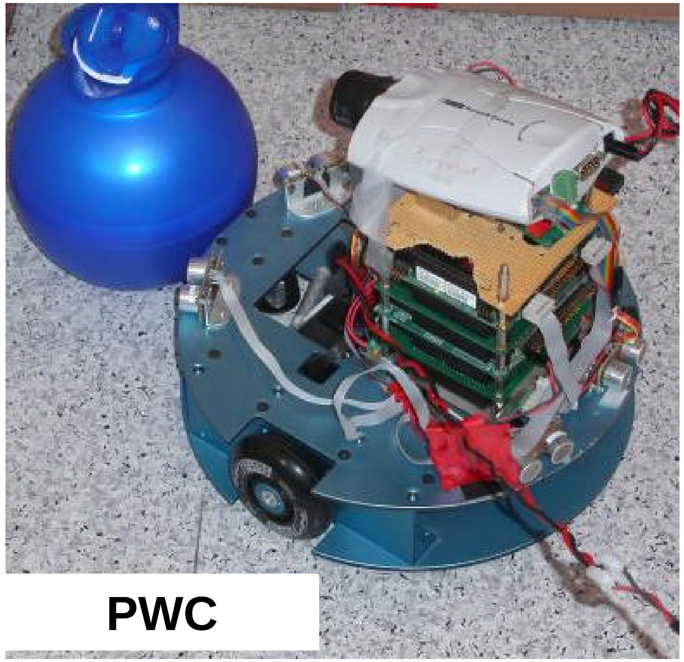
- Transitions and reinforcements on actual robots differ from well-studied continuous and discrete cases



Discrete



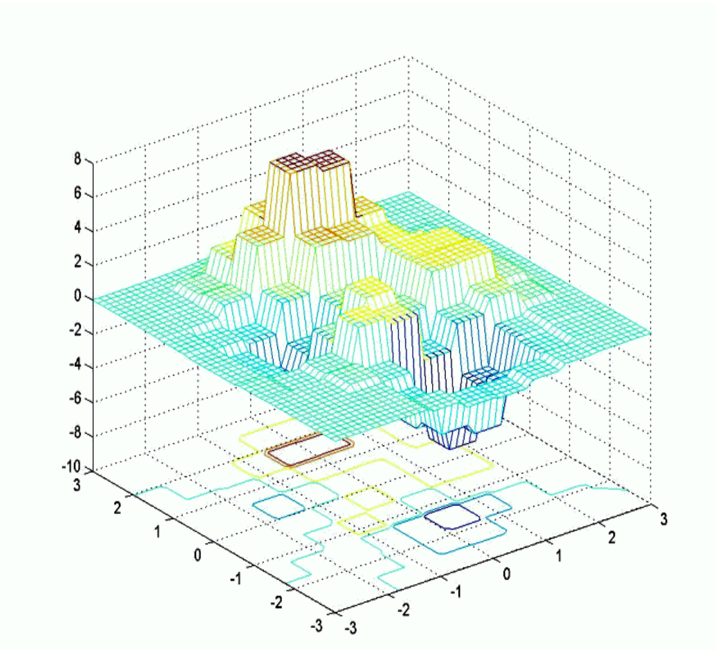
Continuous



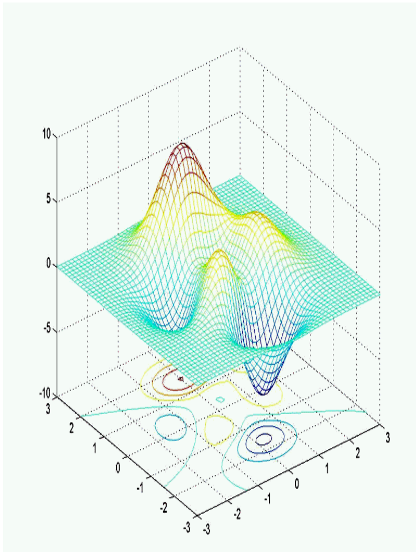
PWC

Continuous and discrete versus PWC

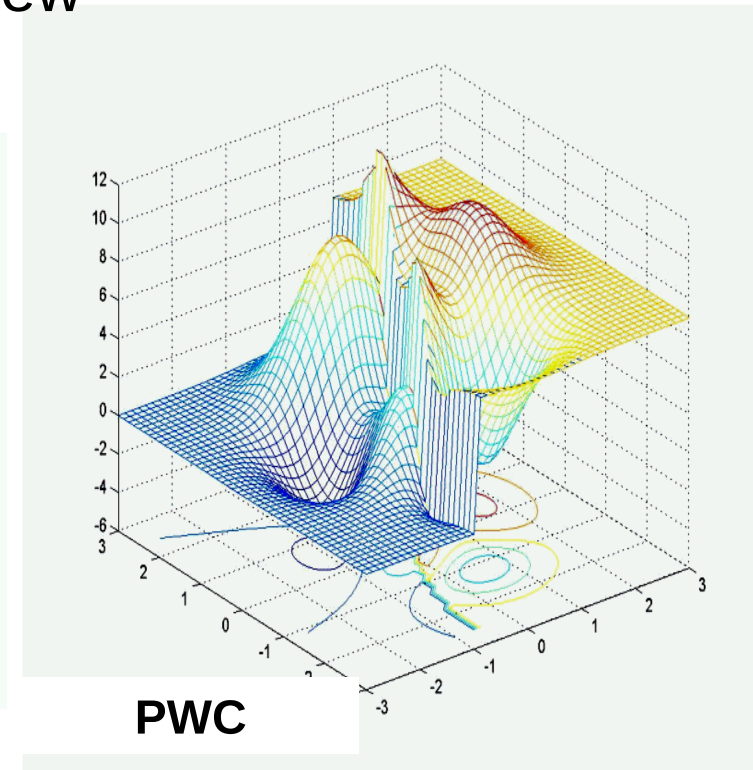
- PWC is characterized by continuous and differentiable structure broken by jumps that appear when, for example, an object is hidden from view



Discrete



Continuous



PWC

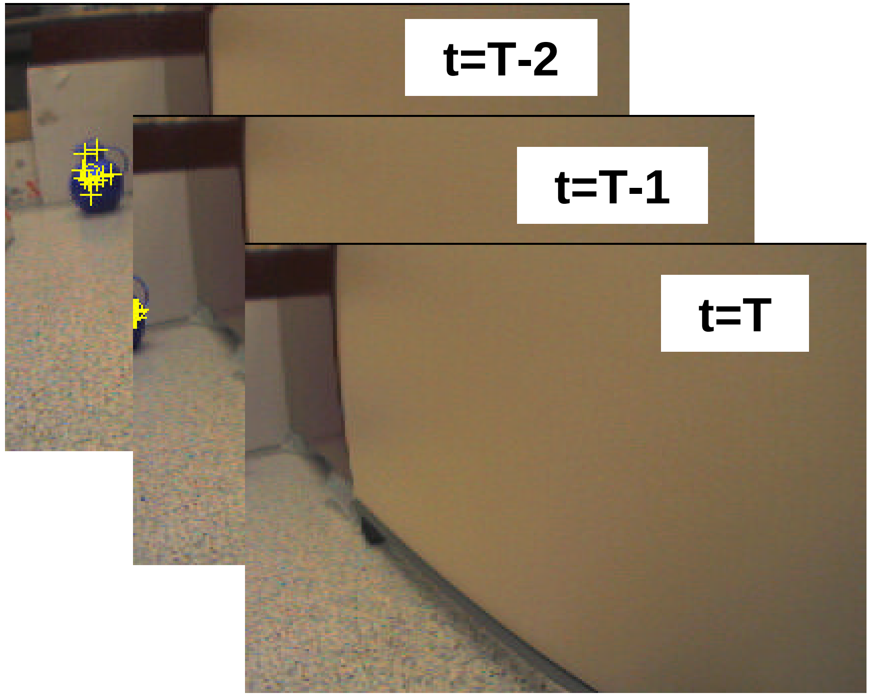
Candidate methods for PWC

- Discretizing state space with fixed /adaptive grid: artificial discontinuities
- Neural networks: do not model discontinuities
- CMAC & RBFs: knowledge of local scale required
- <u>Instance-based memory: OK</u>, but previously used with fixed scale

[redacted]

POMDP

- What if the goal is not seen?
- Solution: use chain of observations for control.

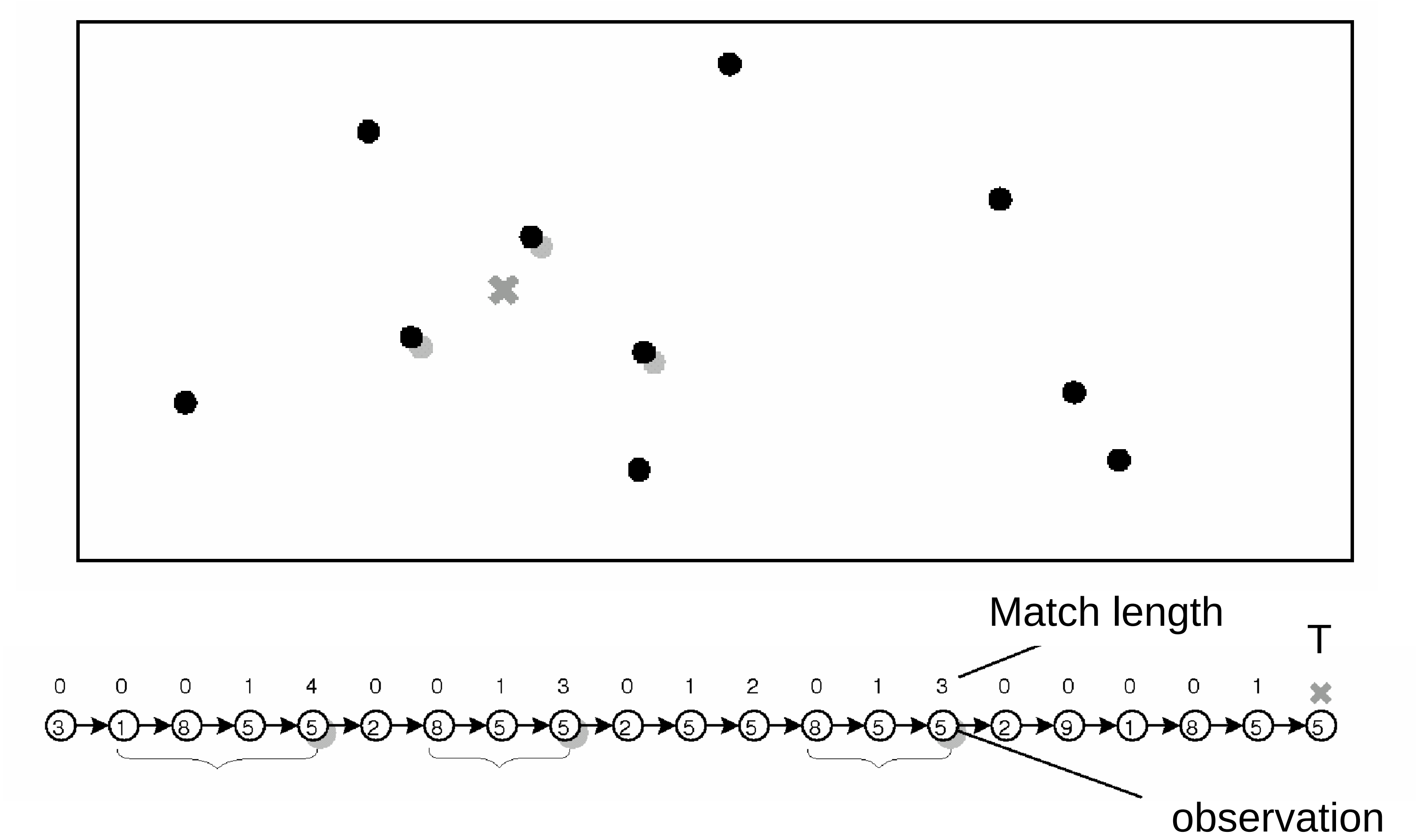


[redacted]

PC-NSM

- Nearest Sequence Memory (NSM) by McCallum: does everything we need, but discrete space and slow convergence
- Solution: modify to work in PWC + speed it up to use data more effectively = Piecewise-Continuous NSM (PC-NSM)

NSM Description slide



[redacted]

Change 1: for PWC

- Pseudometric in original McCallum:

  1/(1+<Number of matching observations>)

- Our metric:

$$\mu(h_t, h_{t'}) = \sum_{\tau=0}^{min(t,t')} \lambda^{\tau} ||o_{t-\tau} - o_{t'-\tau}||_2,$$

[redacted]

Change 2: endogenous updates

- McCallum: update only for t=T-1 (with traces)

$$q(h_t) \Leftarrow (1-\beta)q(h_t) + \beta(R_t + \gamma \max_a Q(h_{t+1}, a))$$

- PC-NSM update:

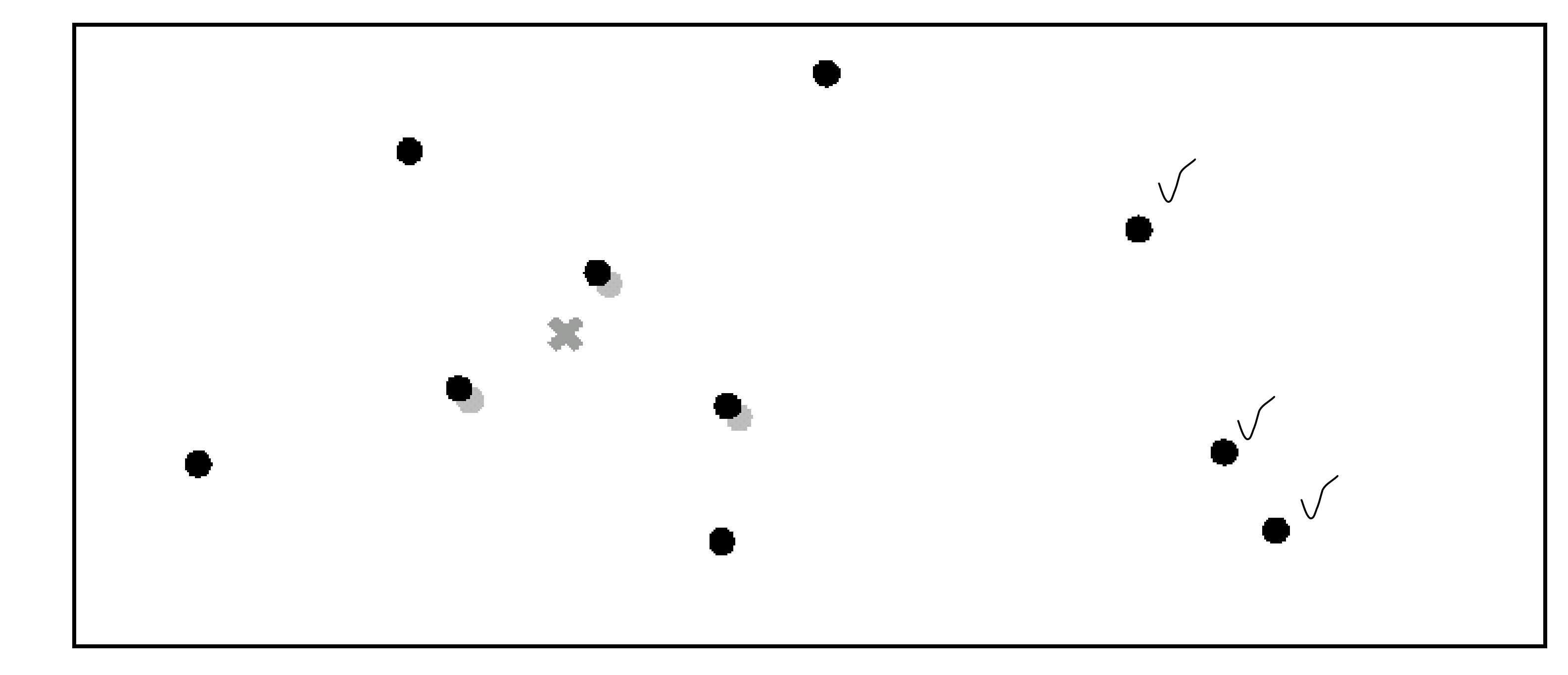
$$q(h_t) \Leftarrow (1-\beta)q(h_t) + \beta(R_t + \gamma \max_a Q(h_{t+1}, a))$$

- Updates through all history needed since neighbourhoods change with new experience

[redacted]

Change 3: directed exploration

- Make least explored action greedily



[redacted]

Demonstration on a mobile robot

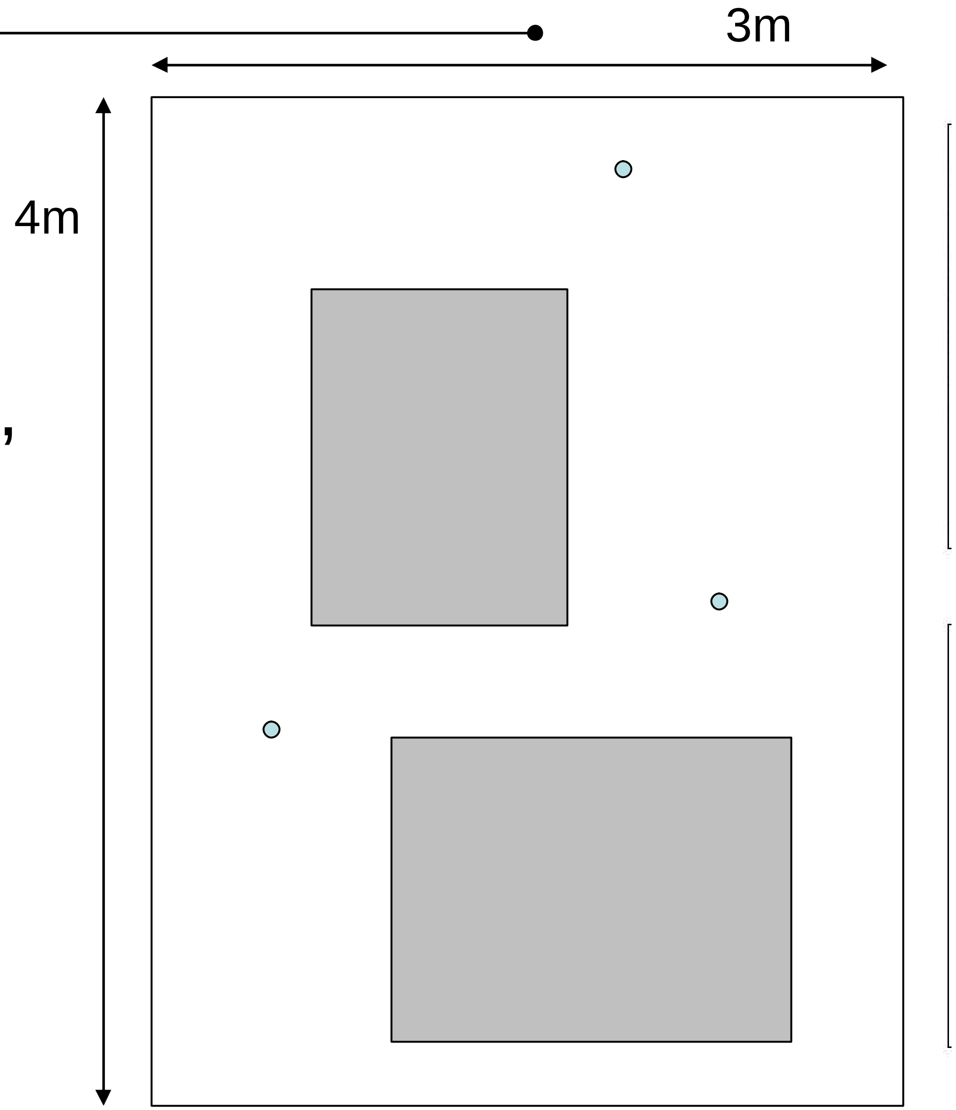
Robot setup

- Sensory input vector:

(x, y, isVisible, f, b)

- Goal: avoid wall collisions,

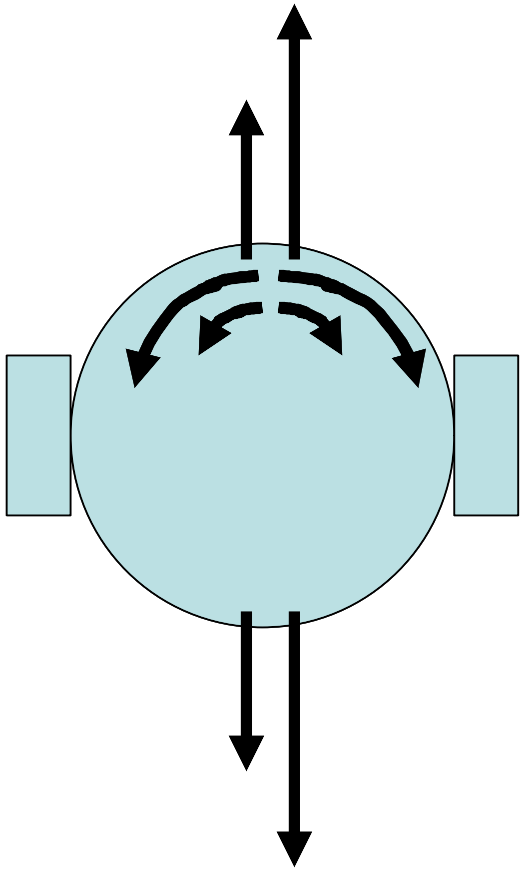
go to the blue teapot



[redacted]

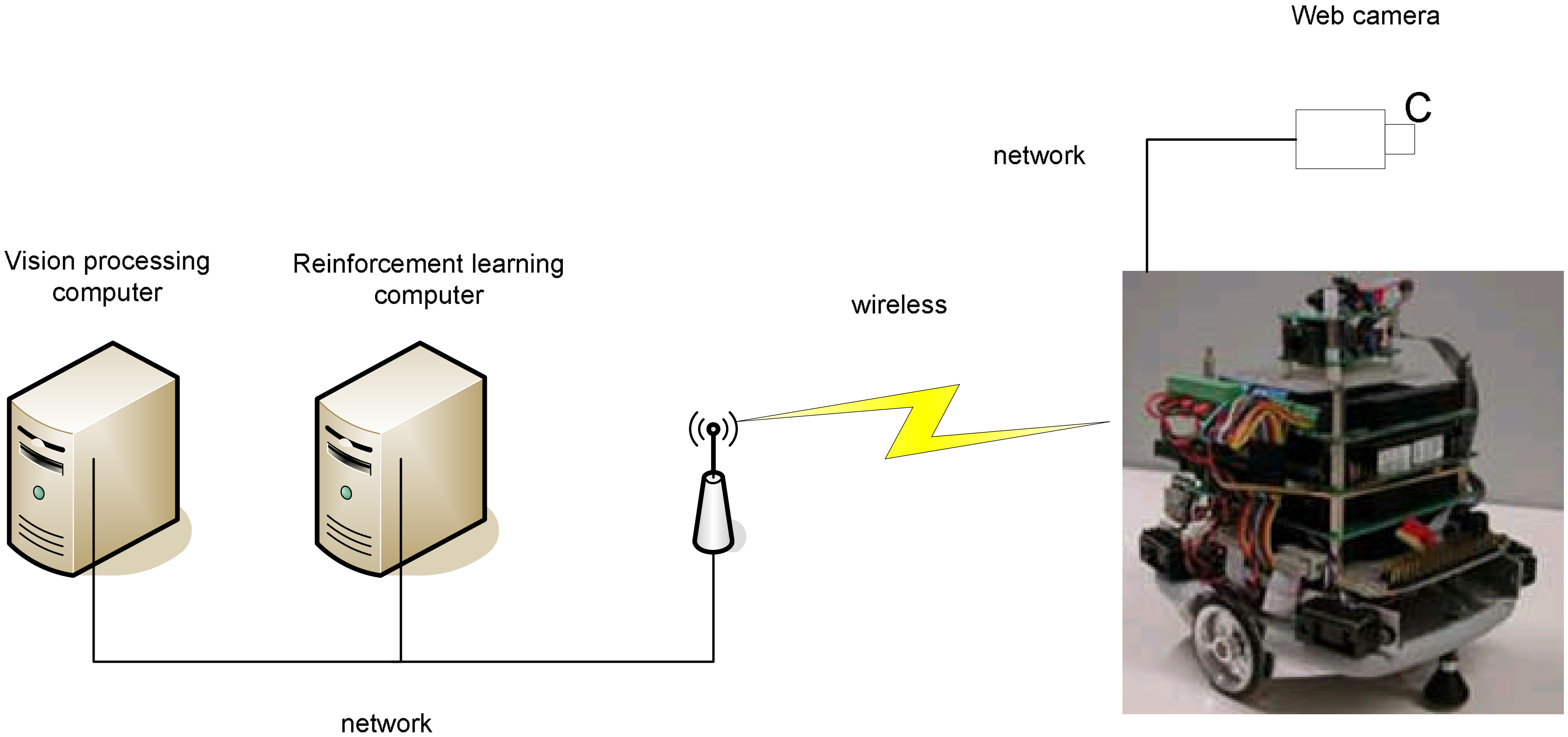
Actions

- Move forward / backward approx 5 cm / 15 cm
- Turn left / right 22.5^{o} / 45^{o}
- Exact values unimportant
- Stand still action
- Wait until robot stops before making the next action



[redacted]

Complete learning system



[redacted]

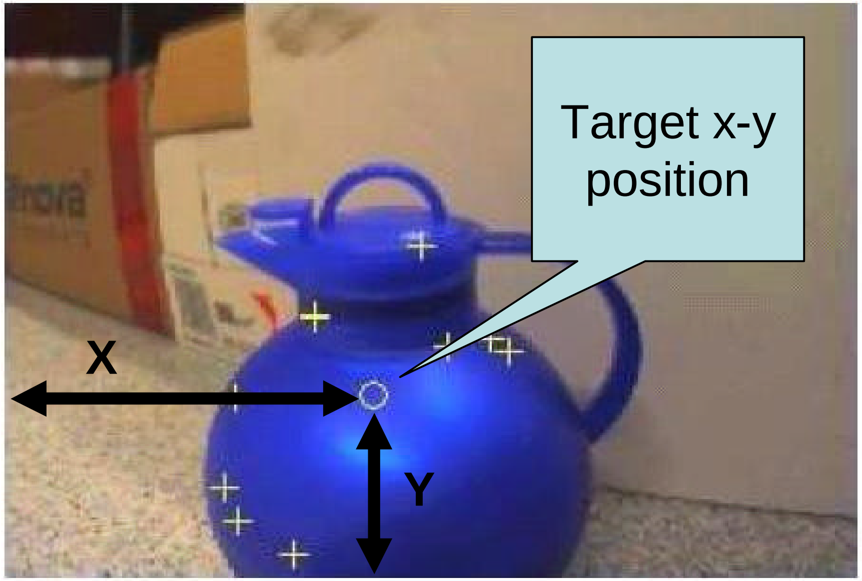
PC-NSM parameters

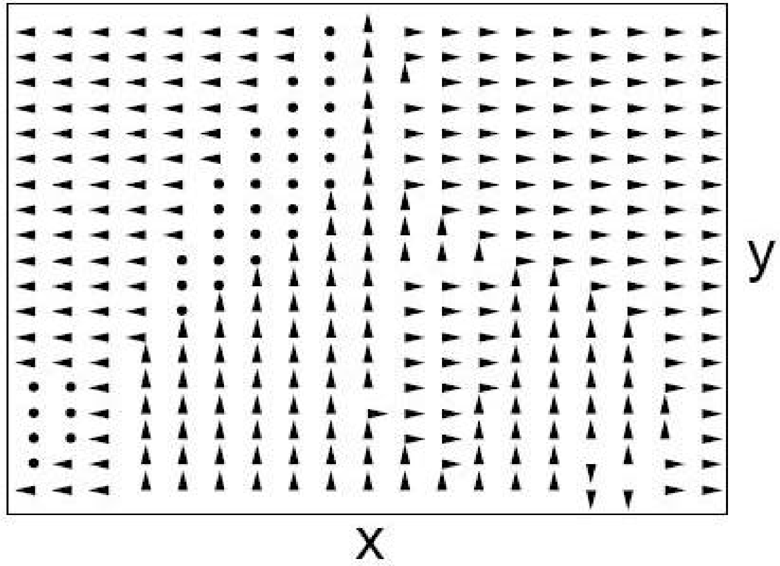
- epsilon-greedy policy with *epsilon* set to 0.3. (30% of the time the robot selects an exploratory action).
- The appropriate number of nearest neighbors, k, used to select actions, depends upon the noisiness of the environment. For the amount of noise in our sensors, we found that learning was fastest for k=3.

Reinforcement structure

$$R = \underbrace{-20/\max(0.01, \min(f, b))}_{R_{\text{obstacle}}} + \underbrace{p \cdot (500 - 50|x| - 250y + c_p)}_{R_{\text{target}}}$$

Results: learned policy





< turn left

> turn right

^ move forward

v move backward

o stand still

- Learned policy dimensionality reduction in the sensor space: varaible **x, y**; r, b walls are always far

Results: example trajectories

- A trajectory after learning
- White boxes mark the controller's confusion resulted from sound-reflecting wall joints

Contributions and limitations

- An algorithm capable of learning on real vision-controlled robots is developed
- The algorithm is able to use modern vision preprocessing algorithms thanks to reliance on metric

Limitations:

- Single metric may be too strict a limitation
- Exploration scheme is greedy

[redacted]

Future work

- Improved exploration
- Multimetric learning

[redacted]

Conclusion

- Requirements for real-world mobile robot learning defined
- An algorithm to satisfy these requirements is proposed
- Feasibility study on an actual robot is made

[redacted]