



Master in Artificial Intelligence (UPC-URV-UB)

Vision-based Navigation and Reinforcement Learning Path Finding for Social Robots

Xavier Pérez Sala

Dr. Sergio Escalera (UB) and Dr. Cecilio Angulo

Motivation

- Path finding:
 - Map
 - To optimize the route
- Navigation:
 - Follow the route

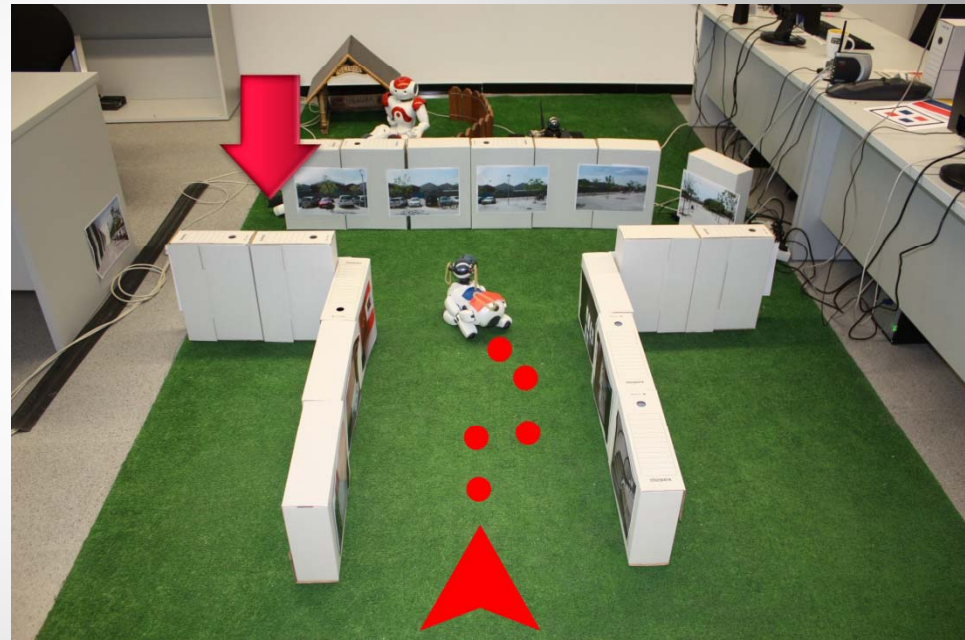


Motivation

- Avoid the necessity to know the world's map

- Robot finds the best solution by itself

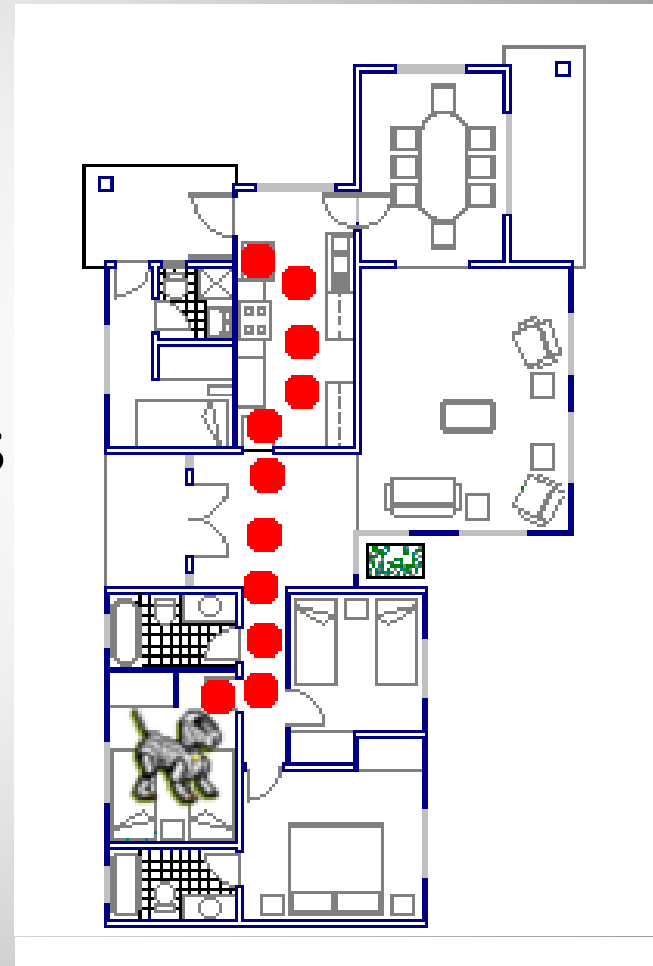
- Robot really follows the route



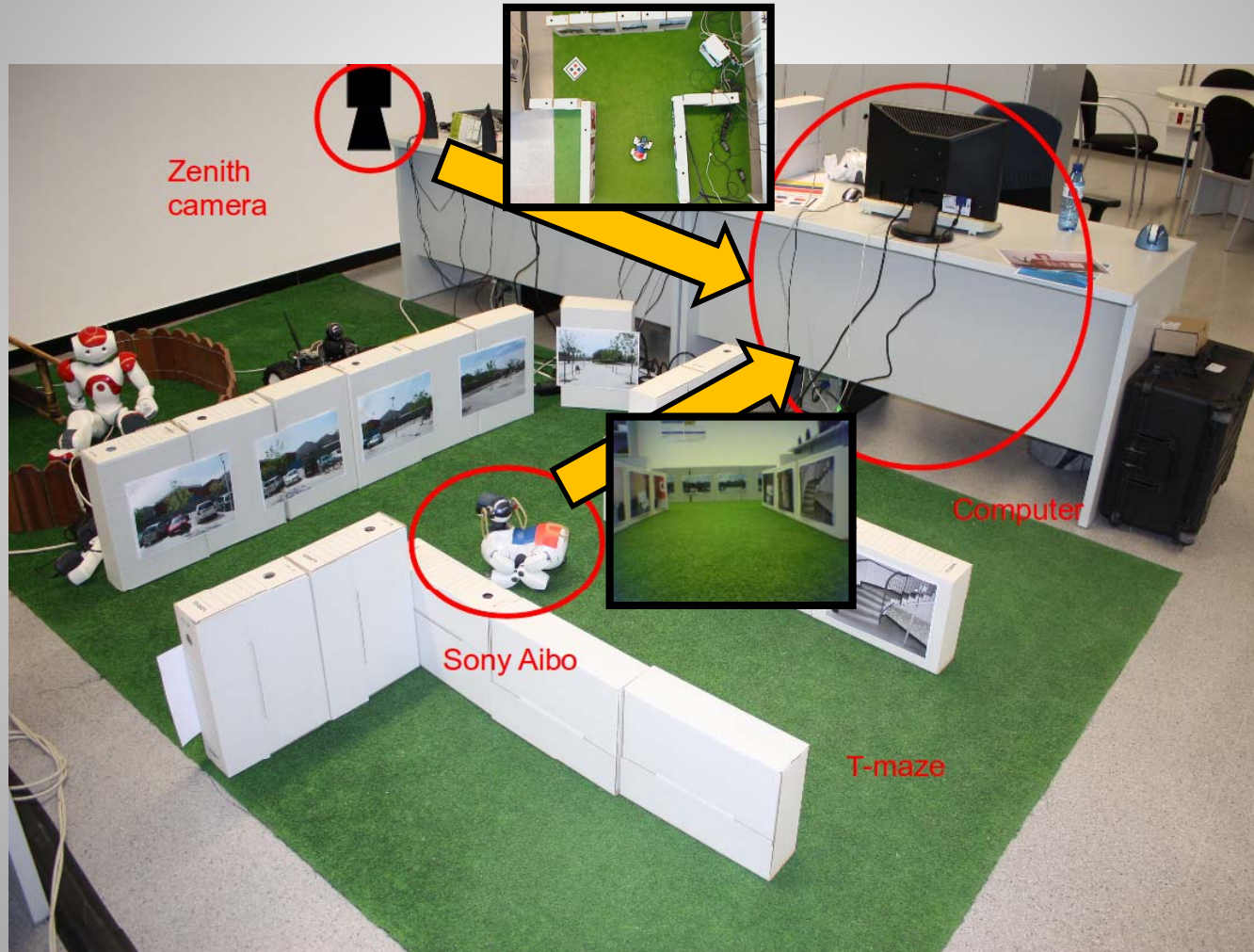
- Robot only needs information from its sensors

Motivation

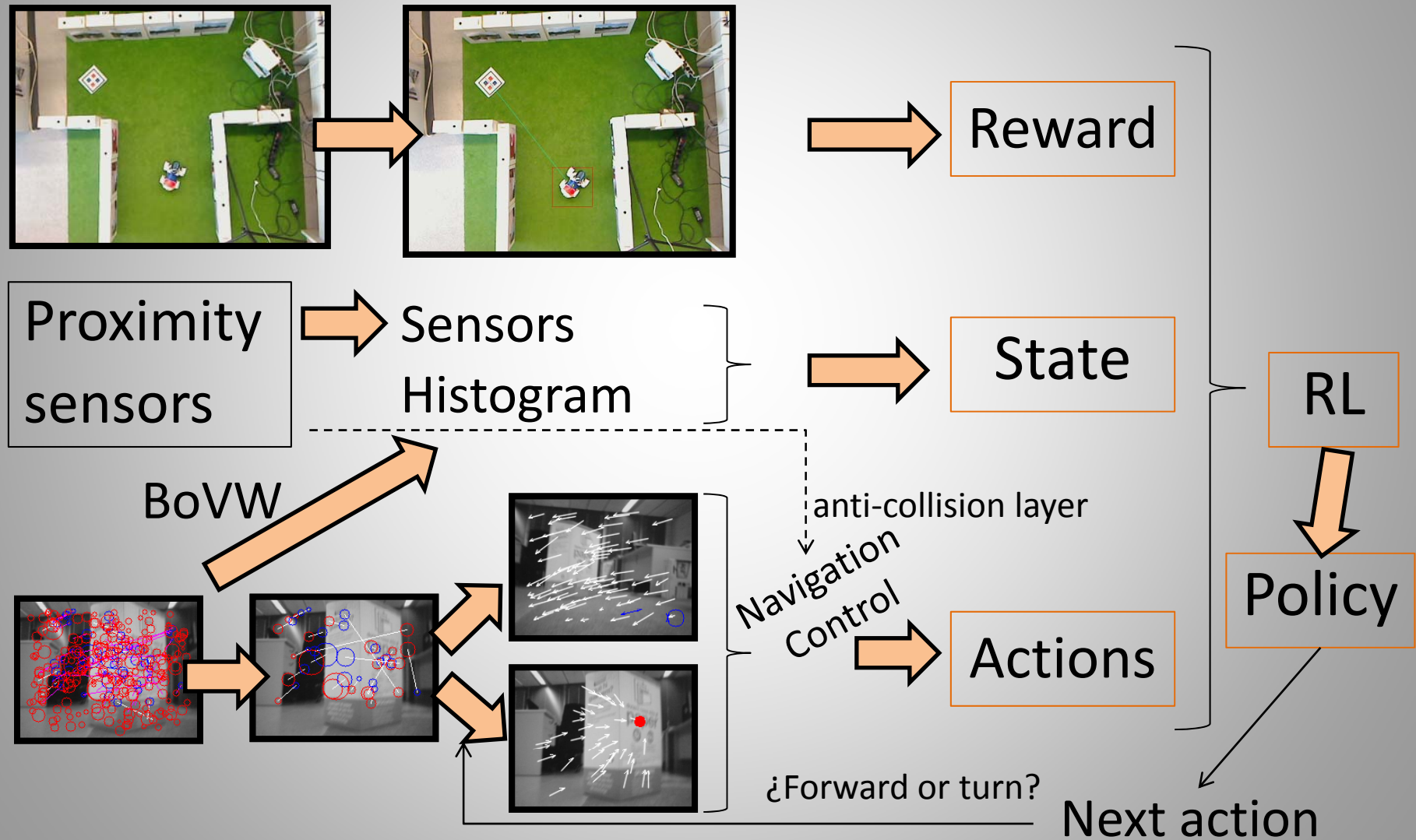
- Unknown maze
 - Unknown house
- Leaving the maze
 - Route between rooms
- Social Robotics



Overview



Overview



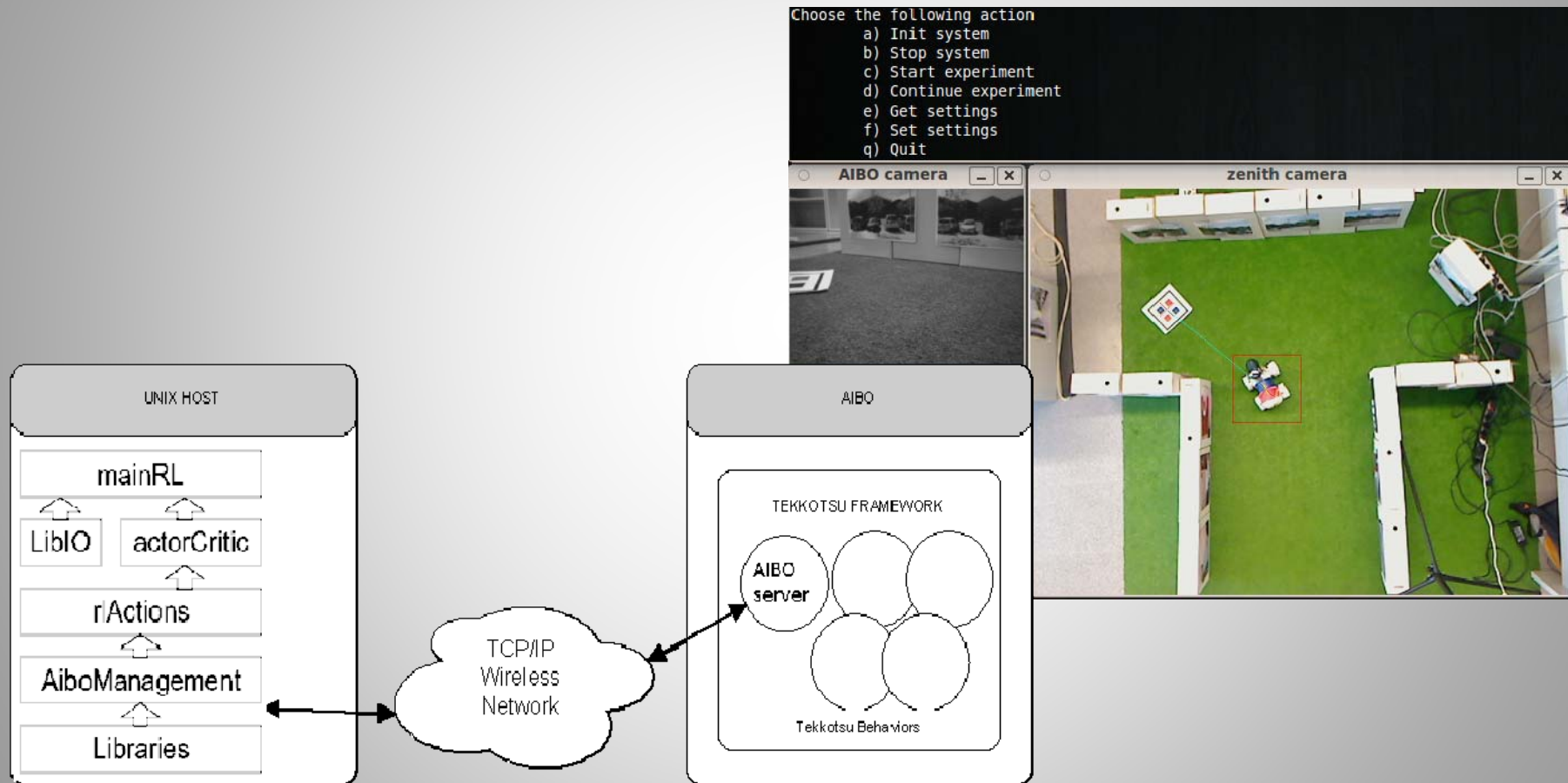
Goals

1. Working environment:
 - a) Robot framework
 - b) Computer and communication platforms
 - c) Exchange protocols
2. To implement Reinforcement Learning algorithms
3. Leave the T-maze in the real world
4. To track the robot and to compute the reward
5. Reliable state representation
6. Vision Based Navigation:
 - a) Anti-collision reactive layer
 - b) Reliable actions: controlled forward and controlled turn

Working Environment

1. Working environment:
 - a) Robot framework
 - b) Computer and communication platforms
 - c) Exchange protocols
2. To implement Reinforcement Learning algorithms
3. Leave the T-maze in the real world
4. To track the robot and to compute the reward
5. Reliable state representation
6. Vision Based Navigation:
 - a) Anti-collision reactive layer
 - b) Reliable actions: controlled forward and controlled turn

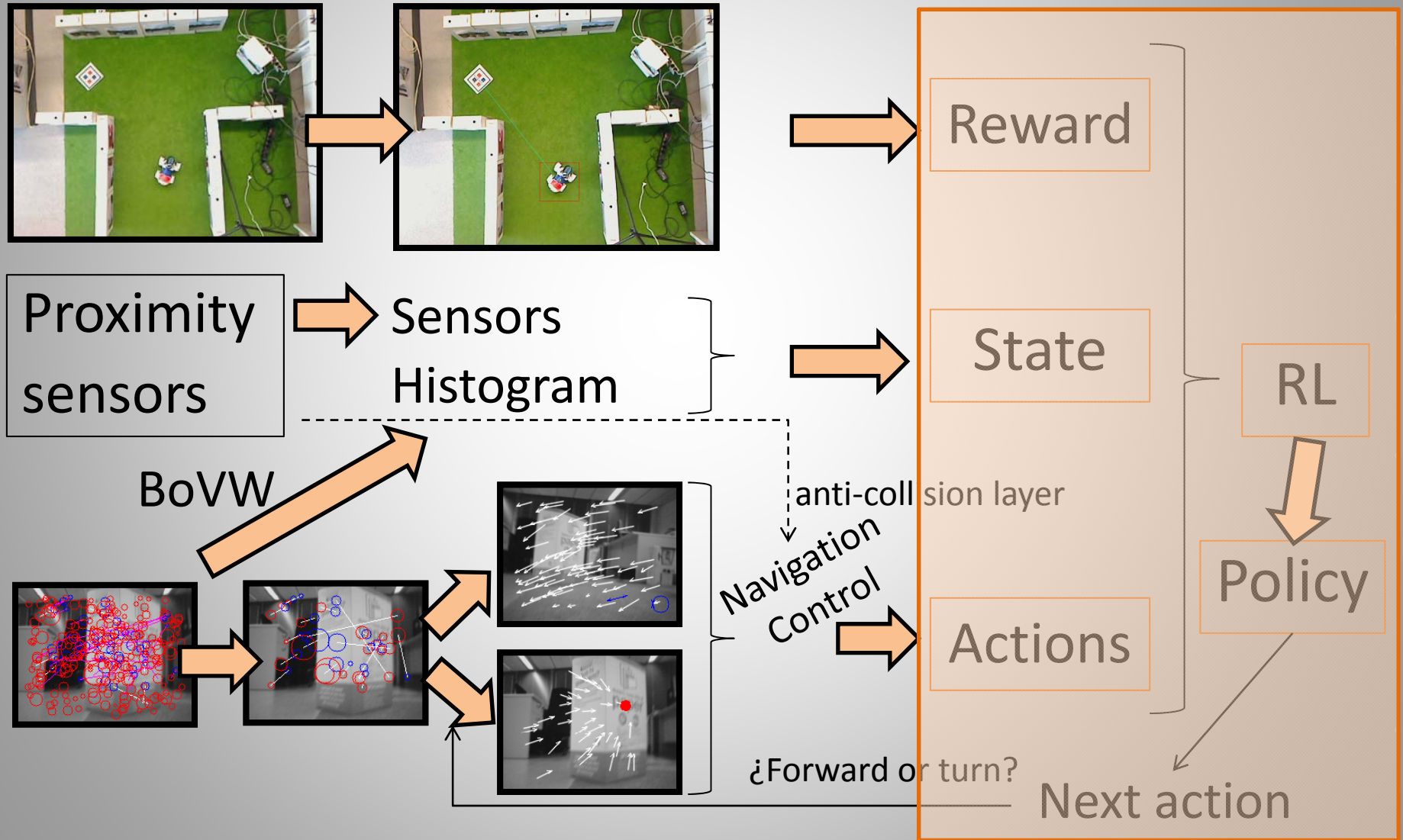
Working Environment



Reinforcement Learning

1. Working environment:
 - a) Robot framework
 - b) Computer and communication platforms
 - c) Exchange protocols
2. To implement Reinforcement Learning algorithms
3. Leave the T-maze in the real world
4. To track the robot and to compute the reward
5. Reliable state representation
6. Vision Based Navigation:
 - a) Anti-collision reactive layer
 - b) Reliable actions: controlled forward and controlled turn

Reinforcement Learning



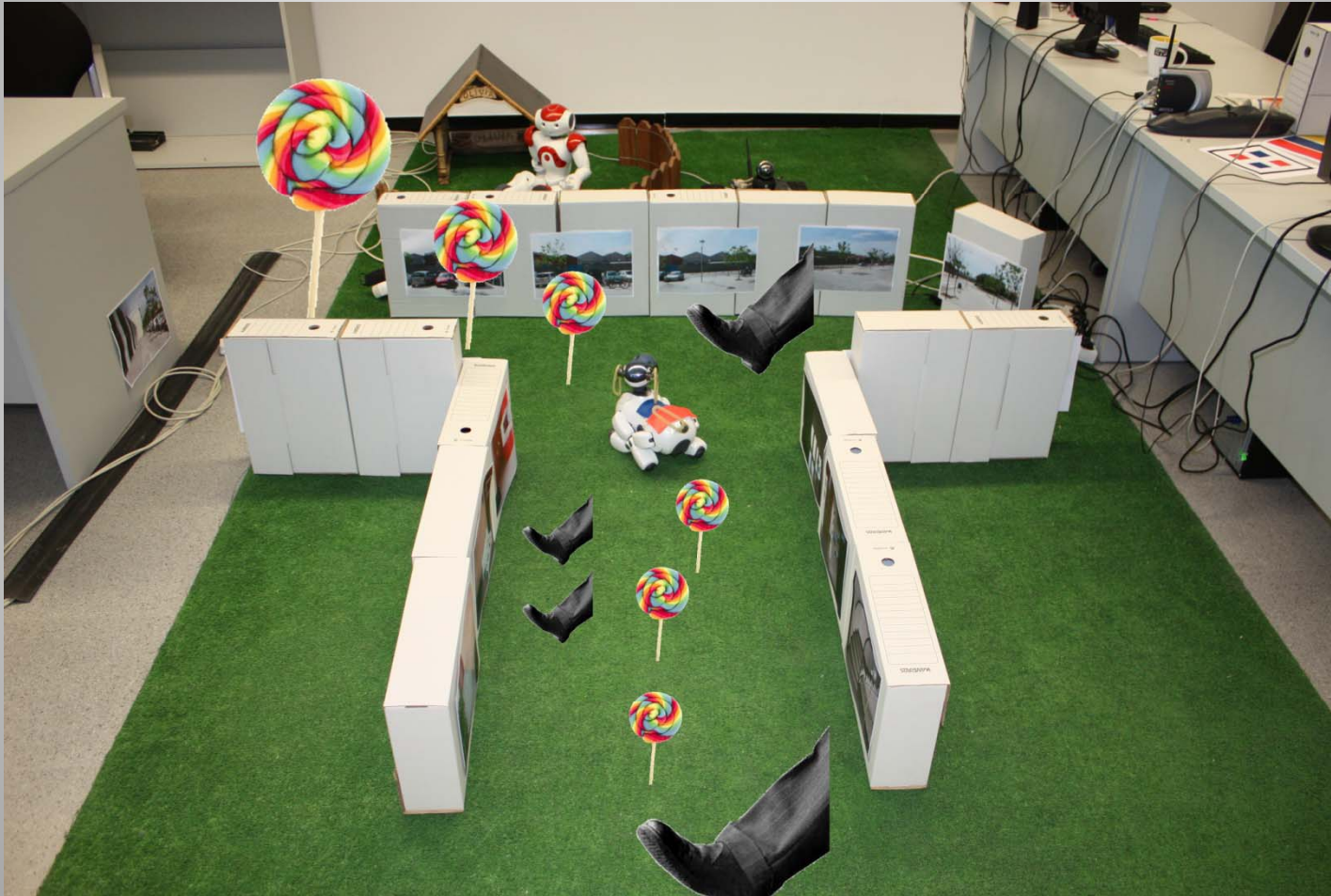
Reinforcement Learning

- Set of world states X
- Set of actions U
- Set of scalar *rewards*



- At each $X_t \rightarrow u(x_t) \rightarrow$ receives x_{t+1} , reward $_{t+1}$
- Interactions with the world \rightarrow policy π

Reinforcement Learning



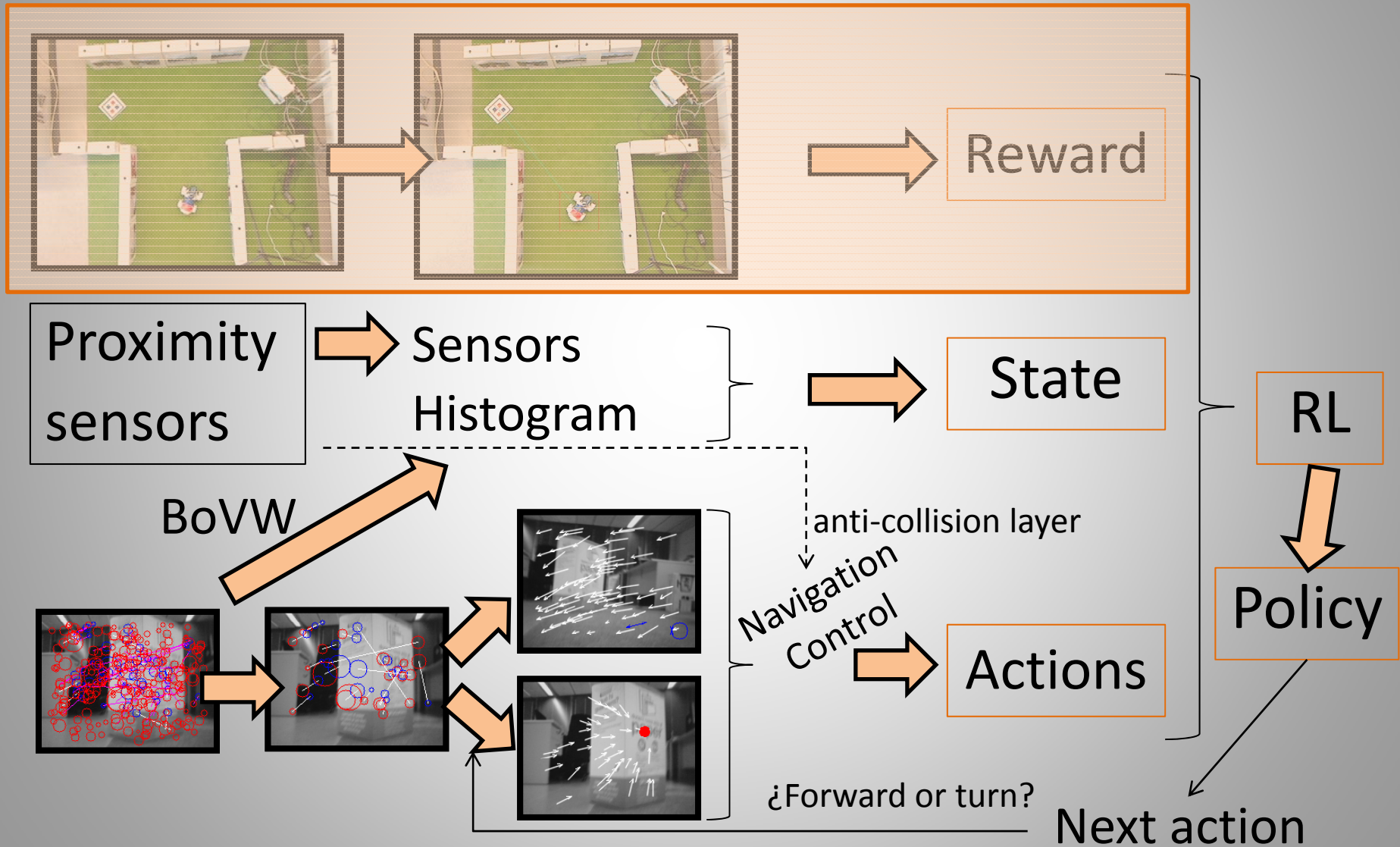
Reinforcement Learning

- Our approach:
 - $X \in \mathbb{R}^n$, where $n = \text{dictionary size} + \text{sensors used} = 53$
 - $U \in [\text{FORWARD}, \text{BACKWARD}, \text{LEFT}, \text{RIGHT}]$
 - Reward $\in \mathbb{R}$
- High state space dimensionality
 - Too much to grid the state space
 - Policy Gradient \rightarrow Natural Actor Critic

Reward

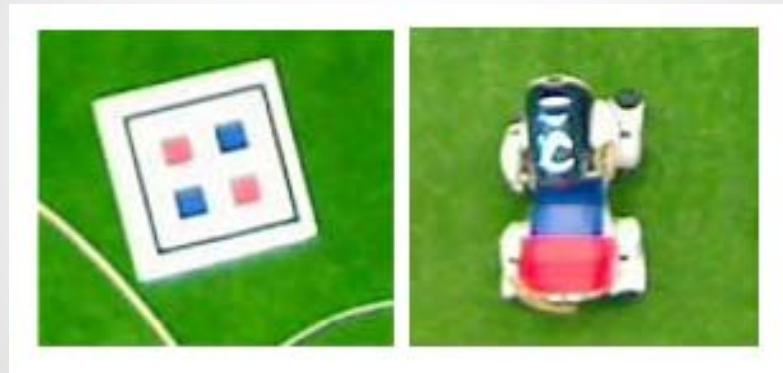
1. Working environment:
 - a) Robot framework
 - b) Computer and communication platforms
 - c) Exchange protocols
2. To implement Reinforcement Learning algorithms
3. Leave the T-maze in the real world
- 4. To track the robot and to compute the reward**
5. Reliable state representation
6. Vision Based Navigation:
 - a) Anti-collision reactive layer
 - b) Reliable actions: controlled forward and controlled turn

Reward



Reward

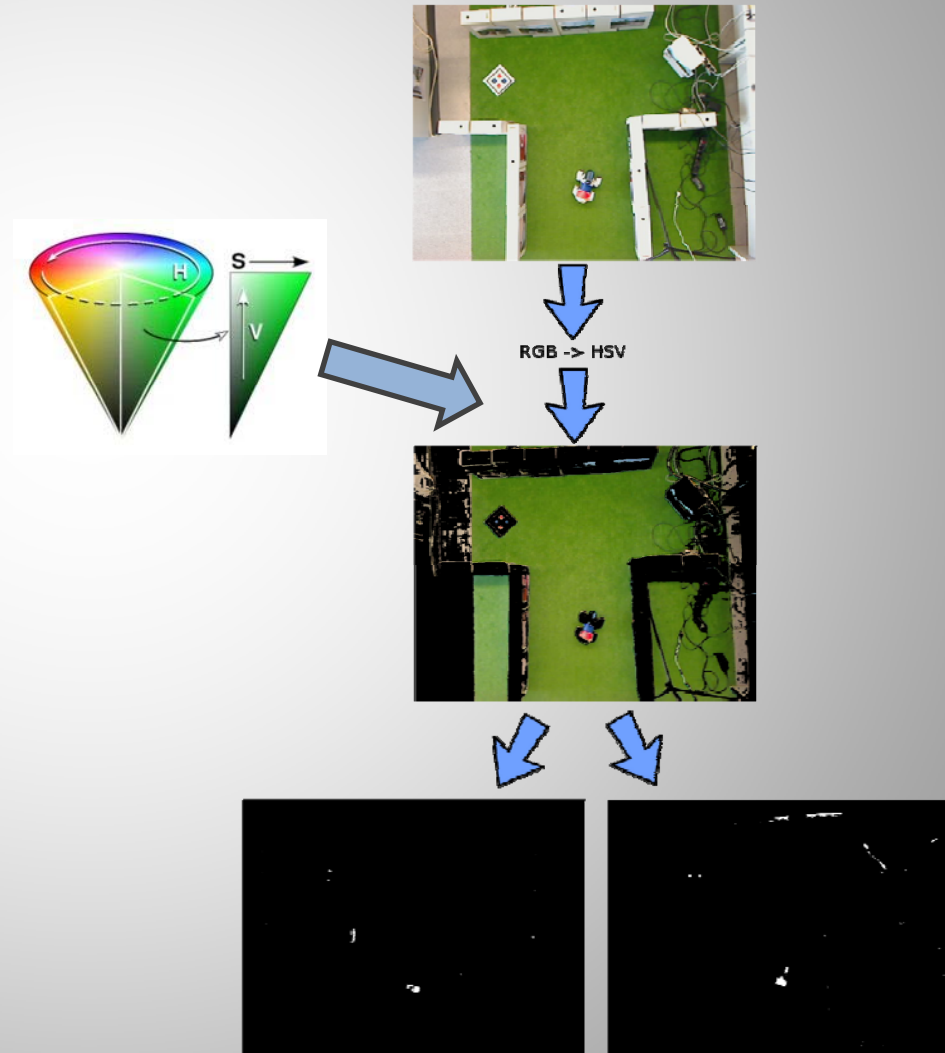
- Distance and relative orientation goal – robot
- Landmarks only here



- Industrial environment
- Filter by color

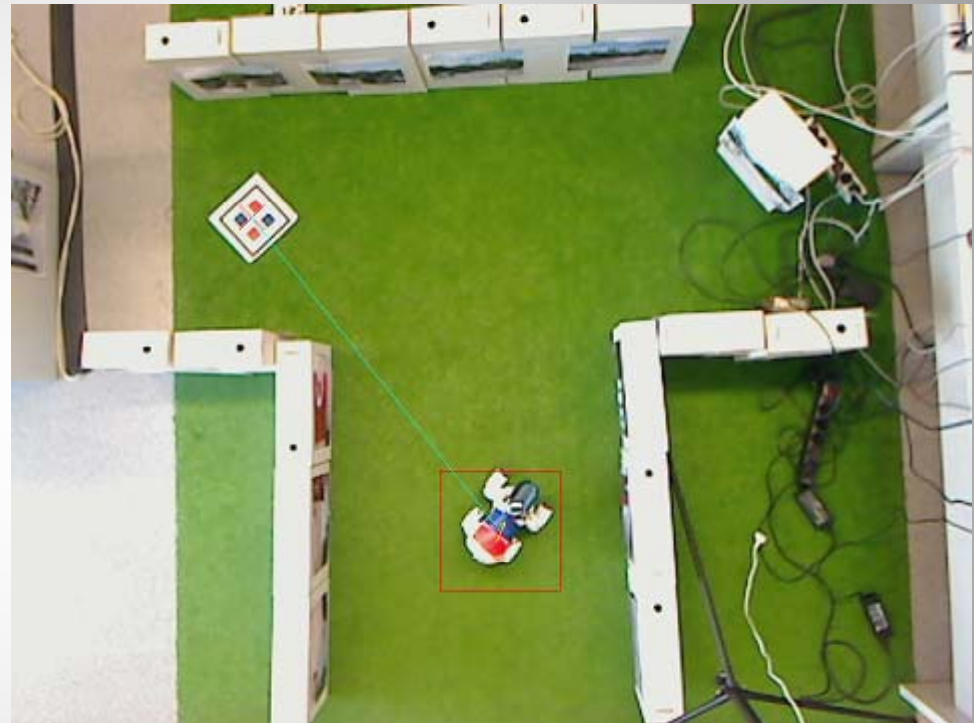
Reward

1. RGB \rightarrow HSV
2. Important pixels
3. Filter by Hue
4. Label images
5. Filter by area
6. Filter by distance
7. Filter by square



Reward

- Resulting image:
- Anti-errors layer
 - ROI
 - Uncertainly



First execution:

202.365ms

Other executions:

54.108ms

Reward

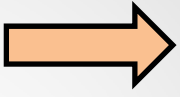
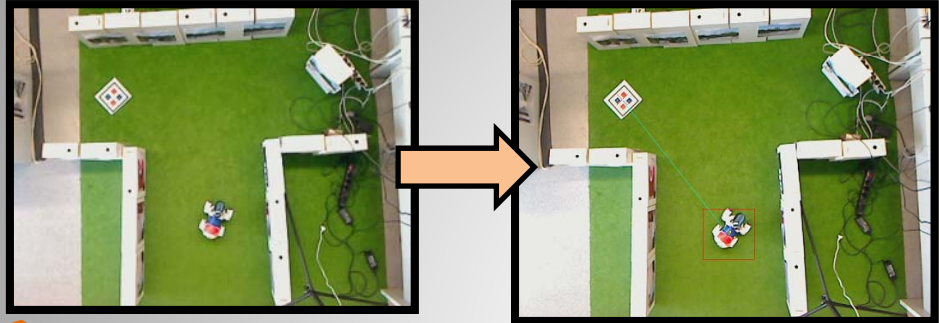
Results



State

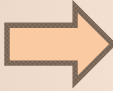
1. Working environment:
 - a) Robot framework
 - b) Computer and communication platforms
 - c) Exchange protocols
2. To implement Reinforcement Learning algorithms
3. Leave the T-maze in the real world
4. To track the robot and to compute the reward
5. **Reliable state representation**
6. Vision Based Navigation:
 - a) Anti-collision reactive layer
 - b) Reliable actions: controlled forward and controlled turn

State

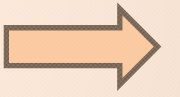


Reward

Proximity sensors

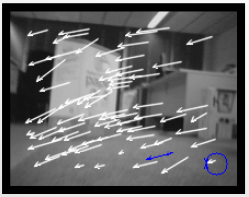


Sensors Histogram



State

BoVW



Navigation Control



Actions

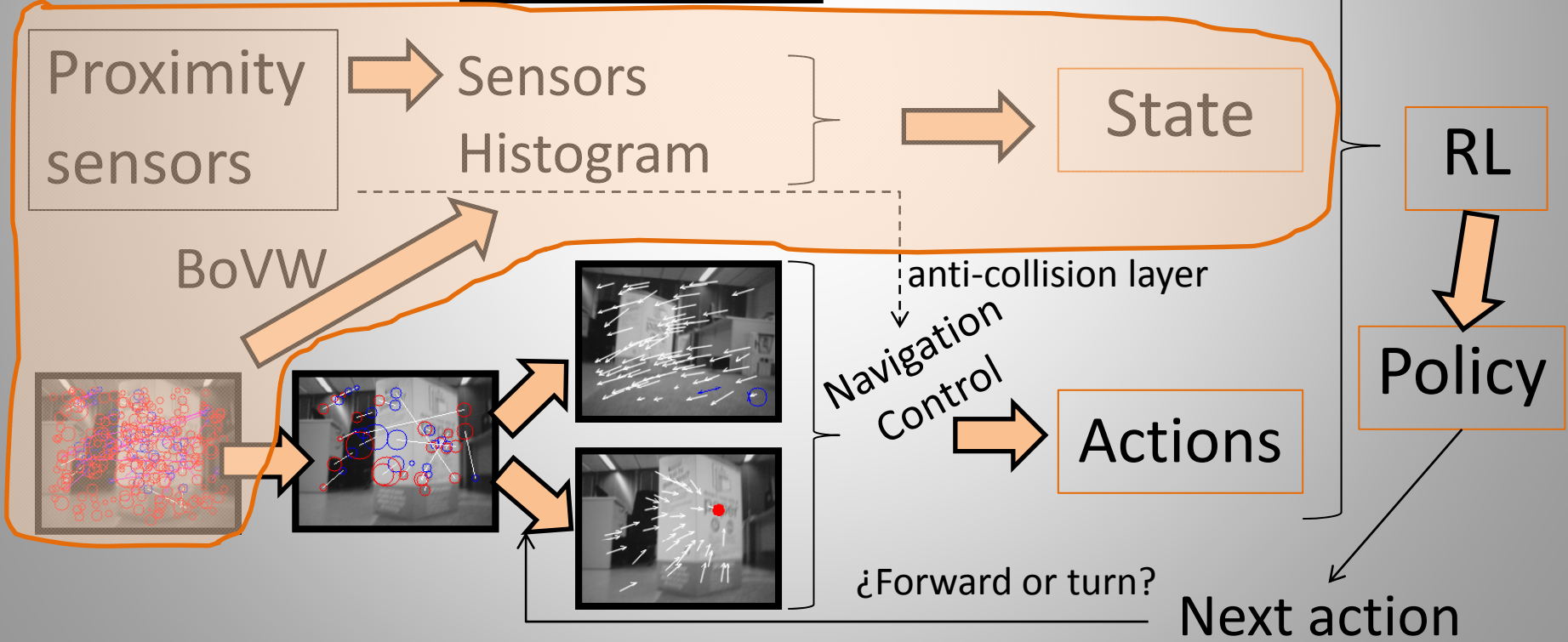
¿Forward or turn?

Next action

RL



Policy



State

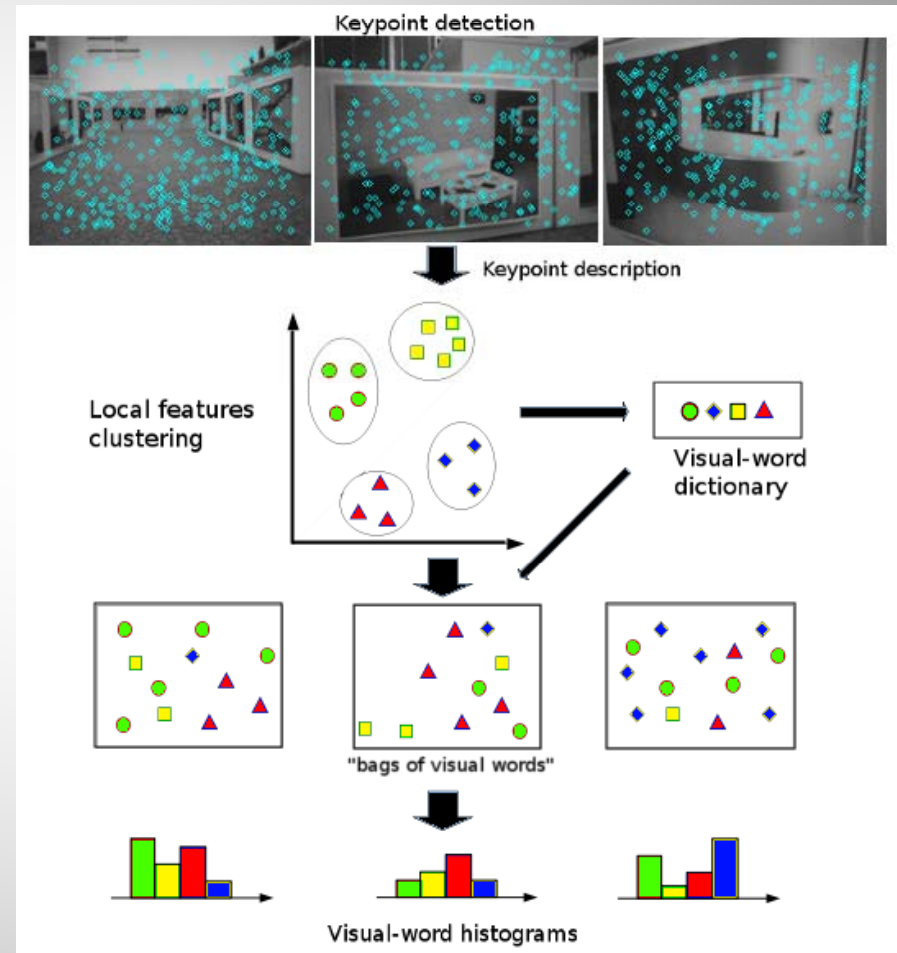
(x, y, ψ) position

vs.

Robot view





















“Bag of Visual Words”

- It comes from BoW
- SURF features
- Dictionary is needed
- Image \rightarrow Histogram



State

Results (50 Words)

Query	Output 1	Output 2	Output 3	Output 4
				
	0.925	0.896	0.880	0.871
				
	0.868	0.860	0.844	0.840
				
	0.851	0.846	0.843	0.838
				
	0.885	0.859	0.847	0.837

State



$x = [50 \text{ BoVW histogram}, 3 \text{ sensors}]$

Actions

1. Working environment:
 - a) Robot framework
 - b) Computer and communication platforms
 - c) Exchange protocols
2. To implement Reinforcement Learning algorithms
3. Leave the T-maze in the real world
4. To track the robot and to compute the reward
5. Reliable state representation
6. Vision Based Navigation:
 - a) Anti-collision reactive layer
 - b) Reliable actions: controlled forward and controlled turn

Actions

- Turn 90° left
- Turn 90° right
- Go forward (2 seconds)
- Go backward (2 seconds)

We can trust on them?

Actions

Open loop vs. closed loop

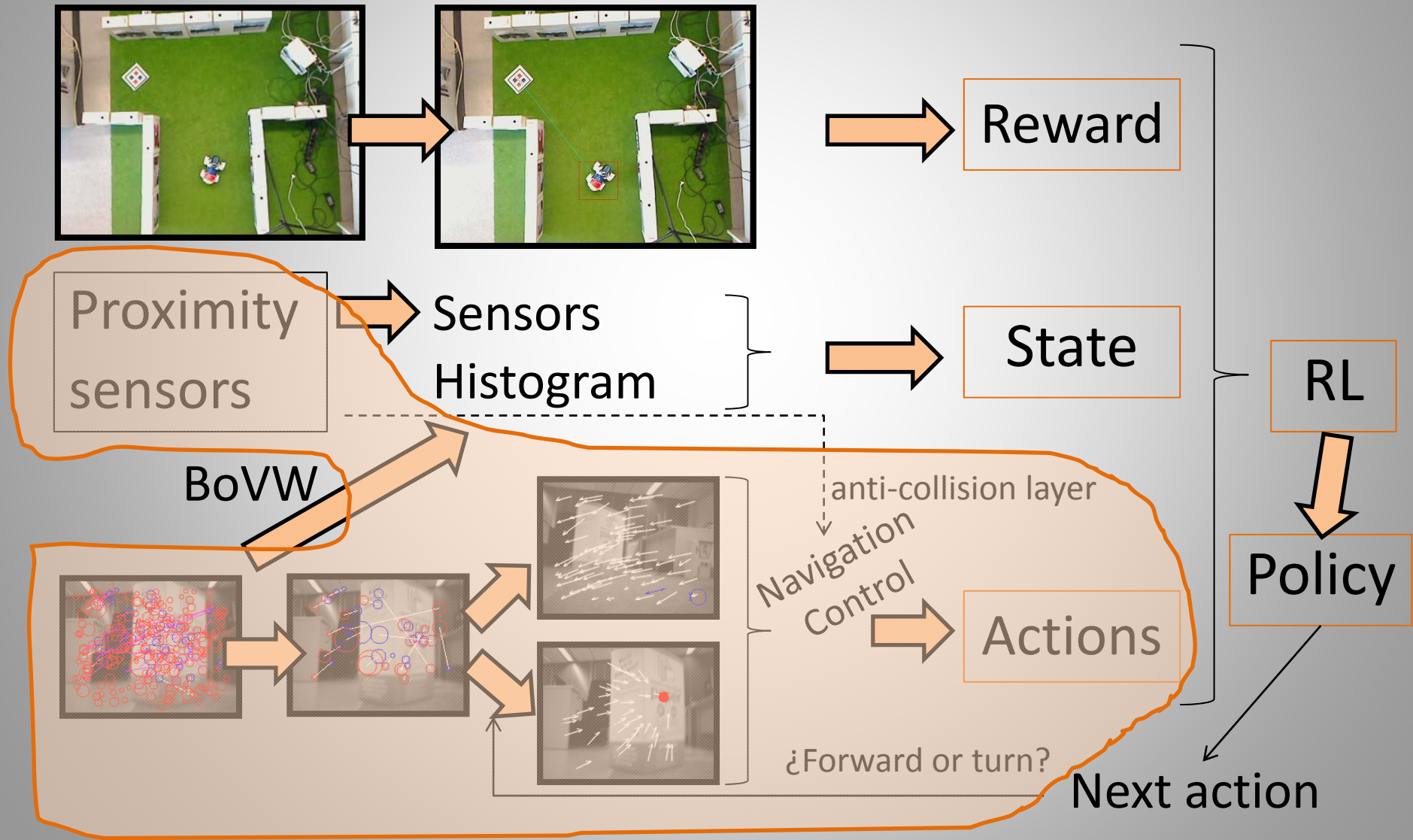


Actions

! CLOSED LOOP IS NEEDED !

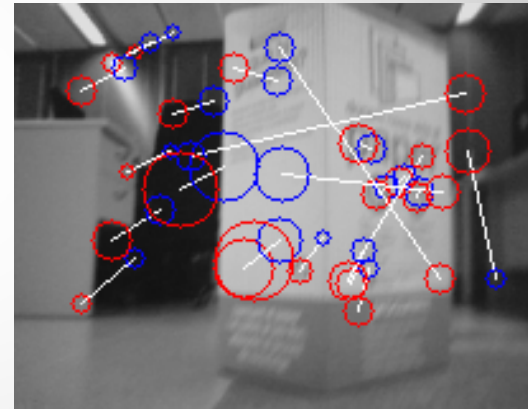
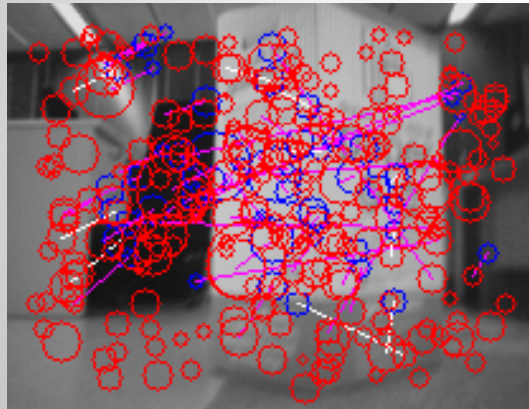
(Vision Based Navigation)

Actions



Vision Based Navigation

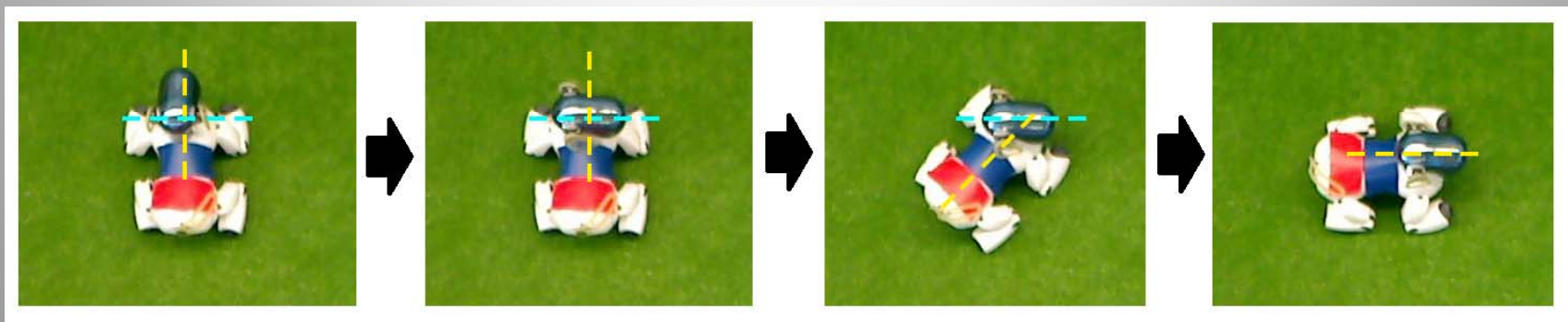
- Extract motion information from consecutive images



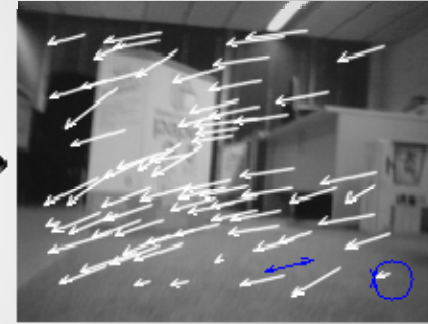
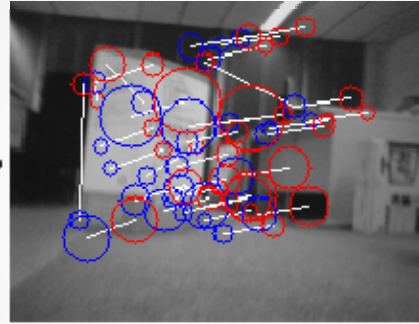
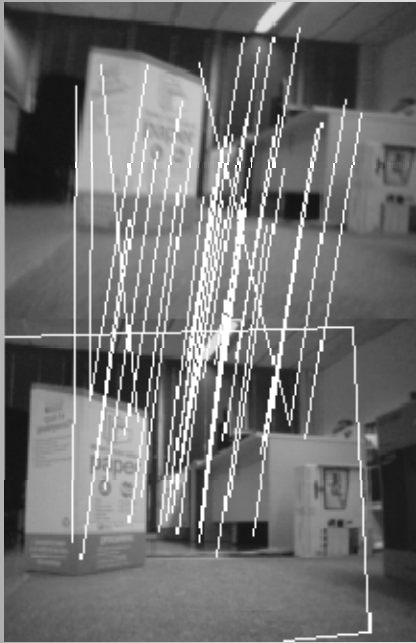
- Simple odometry \rightarrow smart correspondences
- Anti-collision layer

Turn control

1. To turn the head in an specific angle
2. Start turning the while robot keeps its head still
3. Turn is completed when head and body are aligned

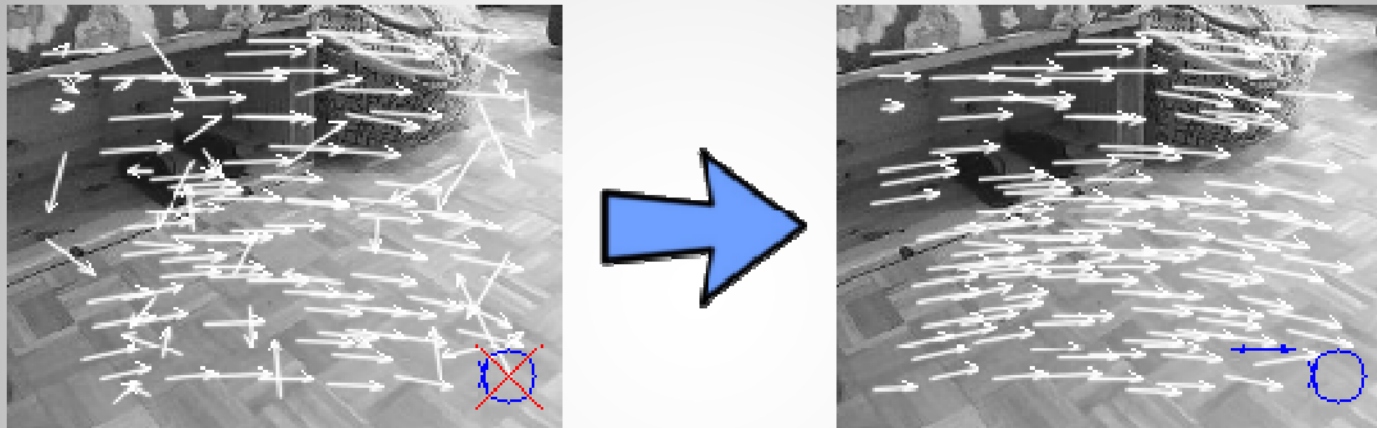


Turn control



error = steering angle = mean motion vector

Turn control



Correspondences \rightarrow most common angle \rightarrow
oriented correspondences \rightarrow mean angle

Turn control

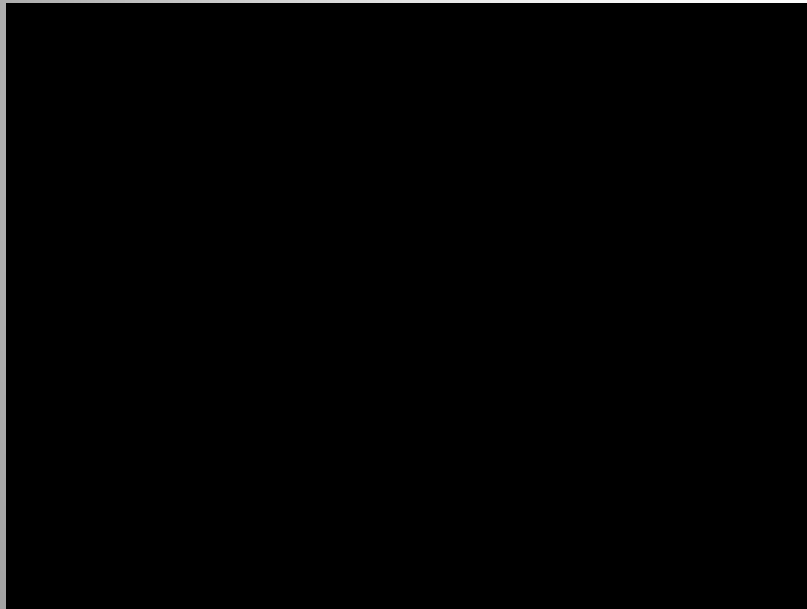
Results (robot view)



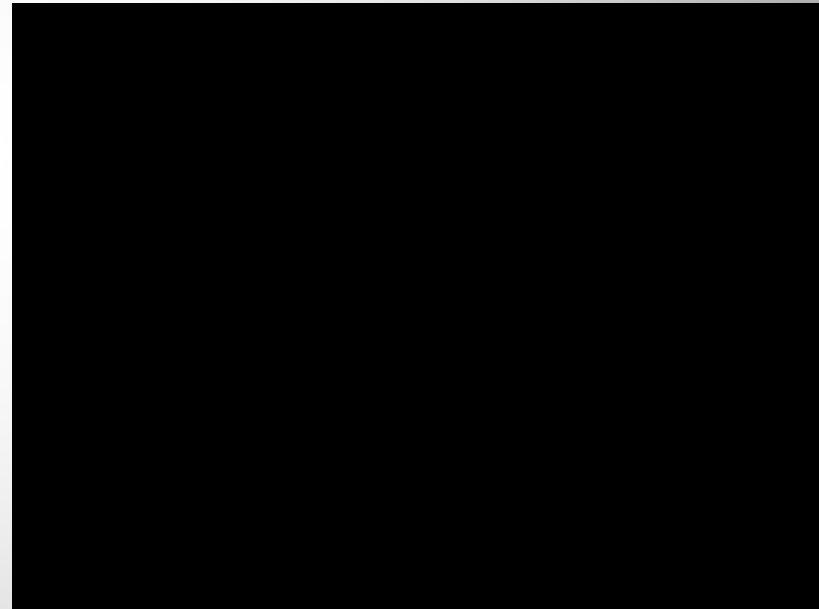
Turn control

Results

Open loop

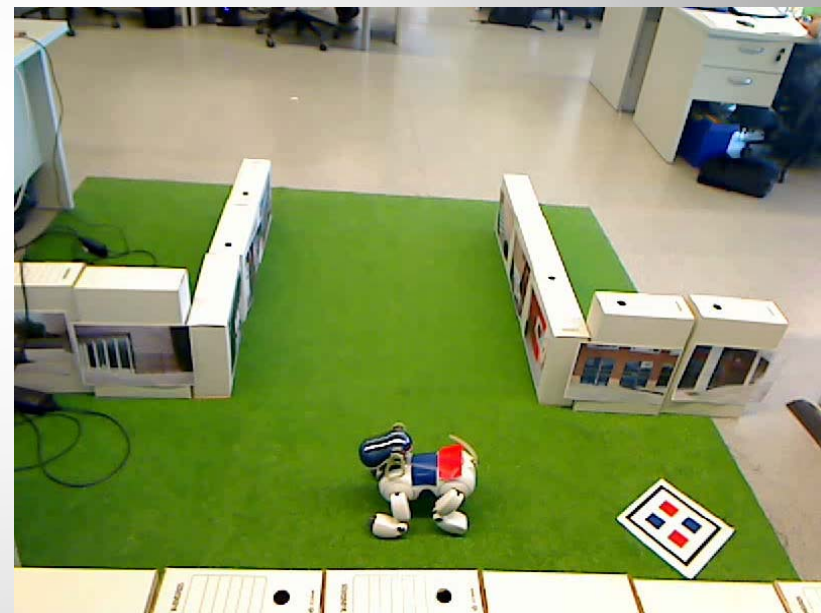


Closed loop

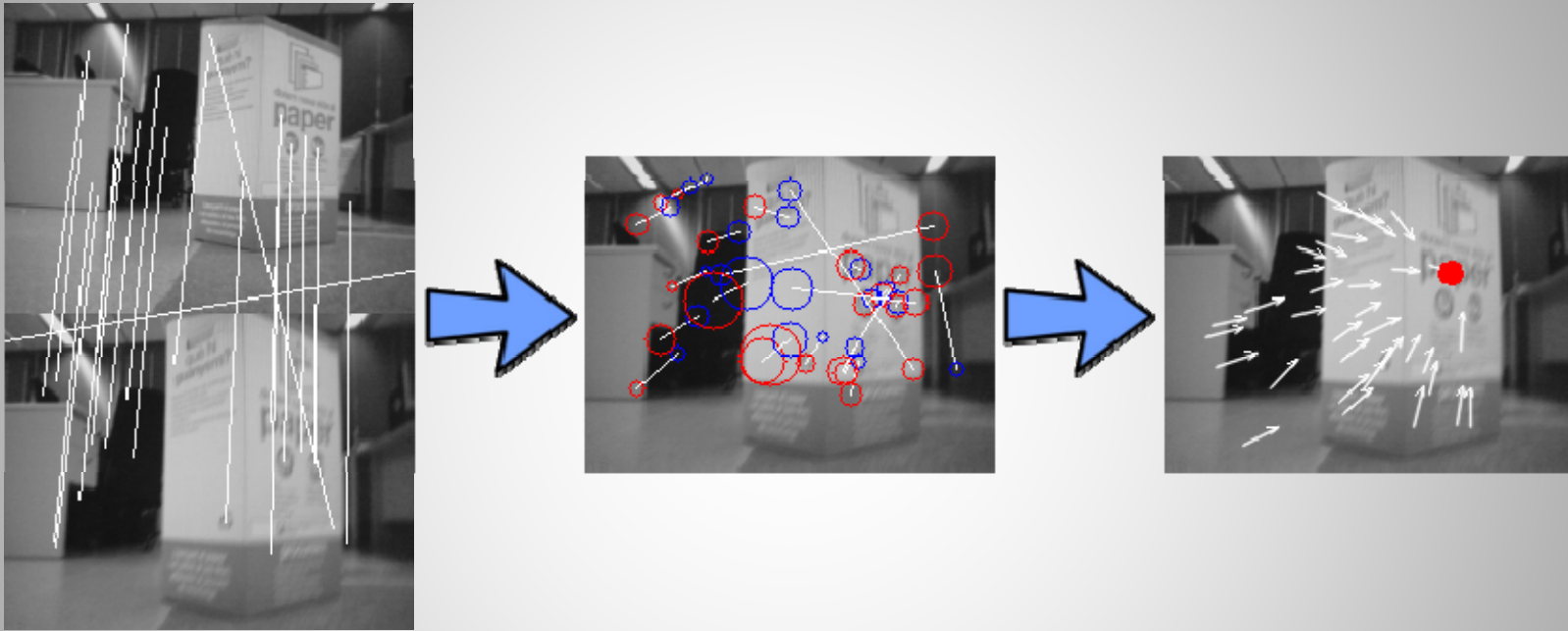


Turn control

Results (left turn)



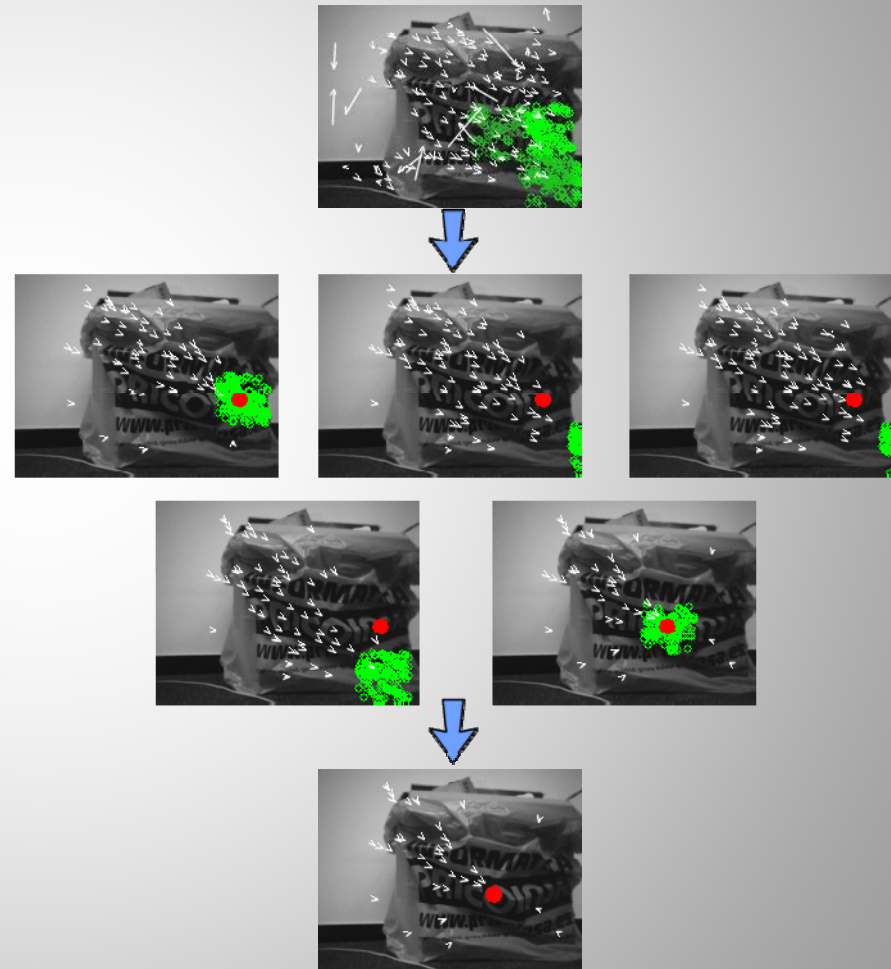
Forward control



error = center – average last vanishing points

Forward control

1. Correspondences
2. Motion vectors
3. Vanishing point (VP)
4. Iterate over various hypotheses
5. Choose the best VP
6. Average last VP's



Forward control

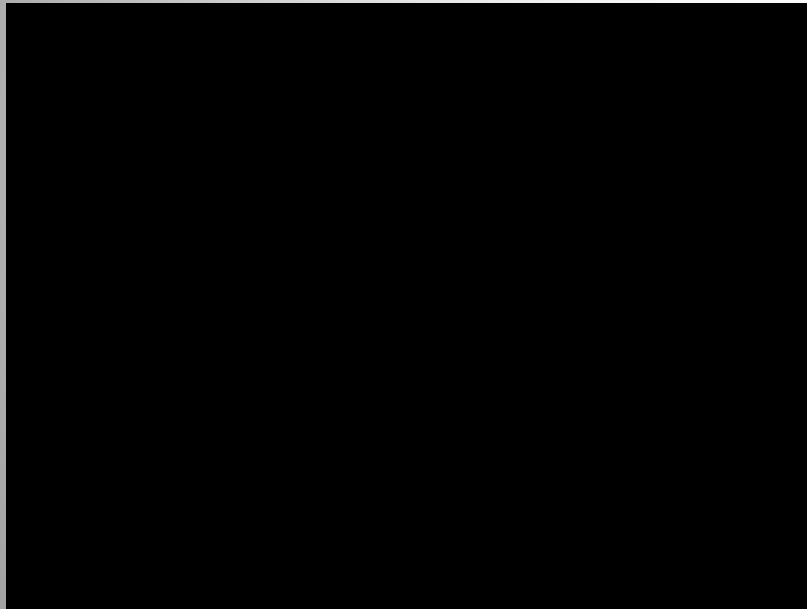
Results (robot view)



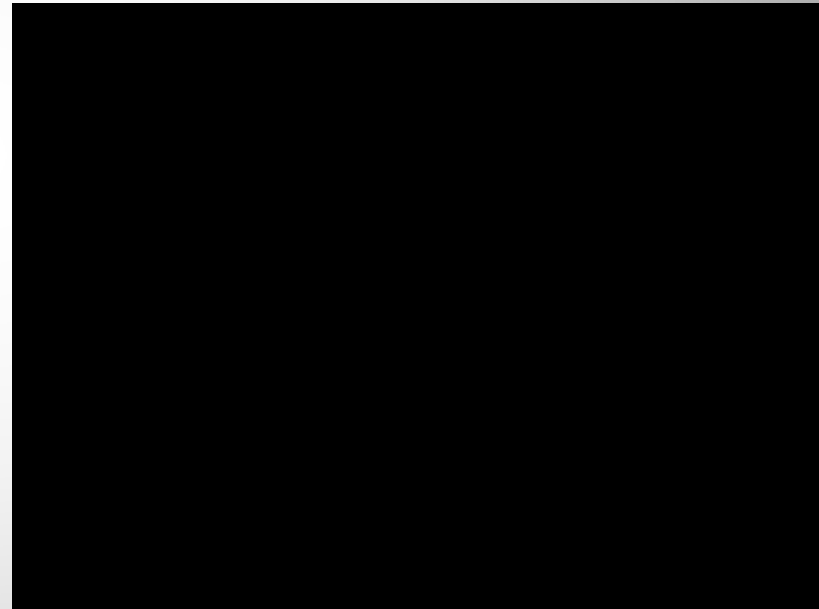
Forward control

Results

Open loop

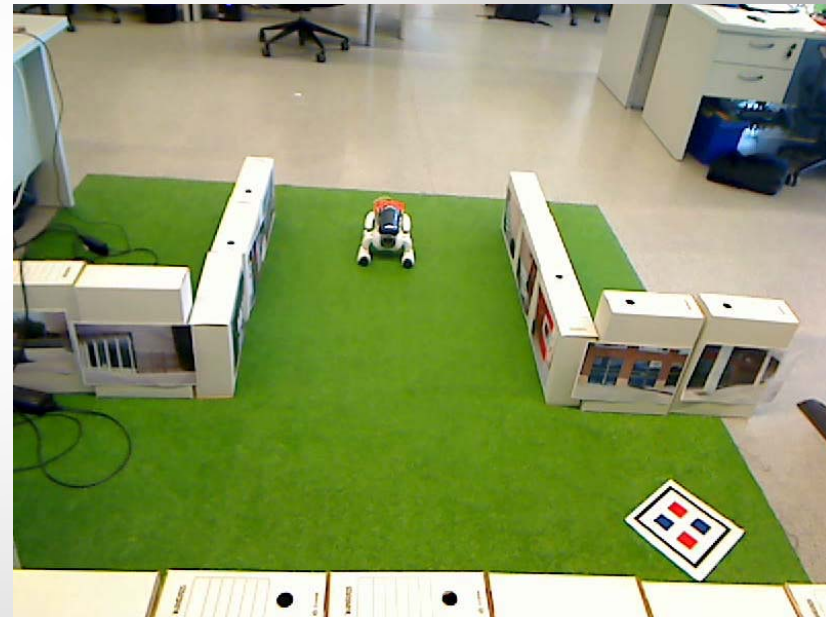


Closed loop



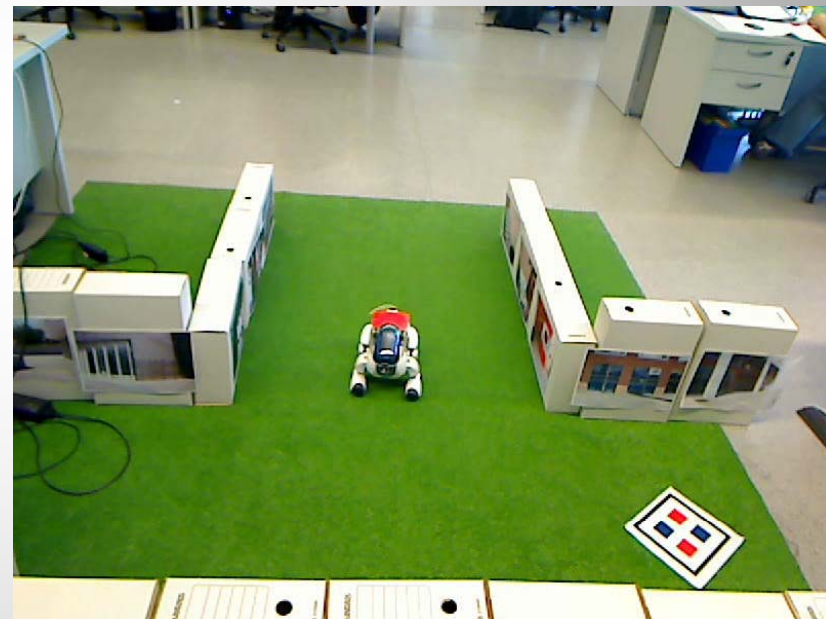
Forward control

Results



Reinforcement Learning

Results



Conclusions

- ✓ Working environment:
 - ✓ Robot framework
 - ✓ Computer and communication platforms
 - ✓ Exchange protocols
- ✓ To implement Reinforcement Learning algorithms
- ✗ Leave the maze in the real world
- ✓ To track the robot and to compute the reward
- ✓ Reliable state representation
- ✓ Vision Based Navigation :
 - ✓ Anti-collision reactive layer.
 - ✓ Reliable actions: controlled forward and controlled turn

Future work

- Improve Reinforcement Learning part, looking for a better learning parameters.
- Strong research on colors spaces (CIE-LAB).
- Improve error signal for forward control:
 - Averaged vanishing point → Joints information
- Test our Control Navigation in a wheeled robot

Thanks for your attention

Doubts & questions