Architecture

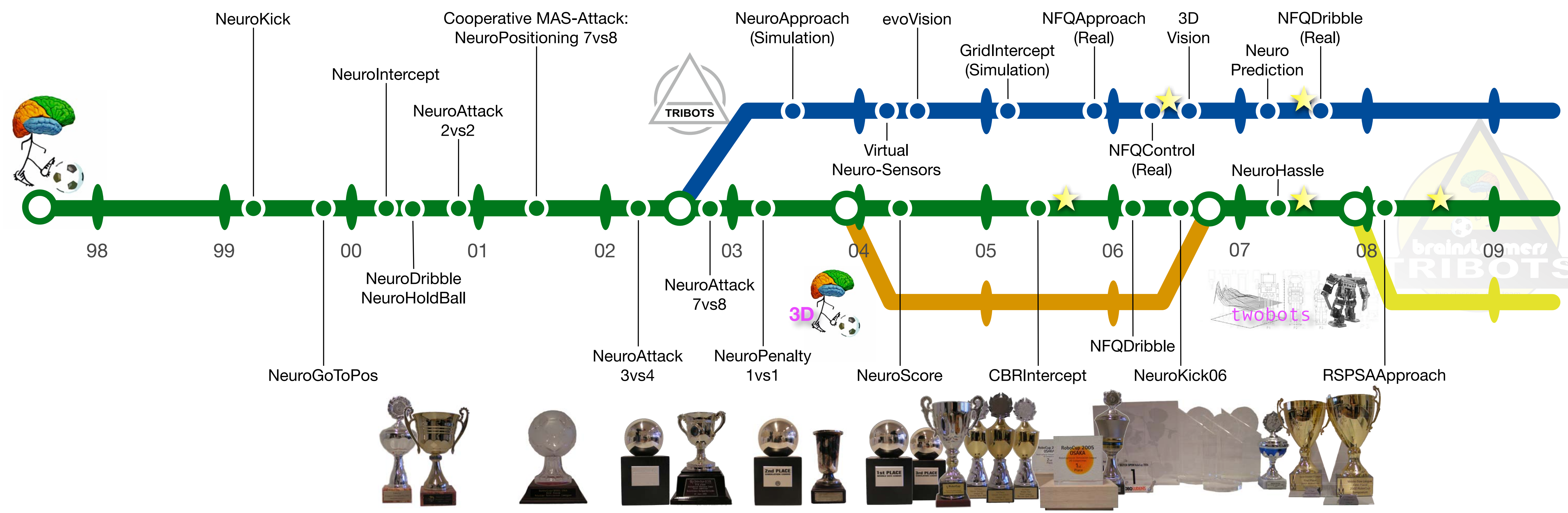
Sascha Lange, Christian Müller, Stefan Welker

Brainstormers Tribots

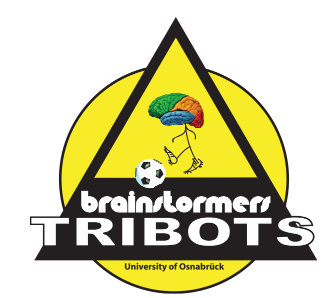
Neuroinformatics Group
University of Osnabrück
Albrechstr. 28
49069 Osnabrück

Tel: +49 541 969-2390
Fax: +49 541 969-2246

http://www.tribots.uos.de
Email: tribots@informatik.uni-osnabrueck.de



NeuroKick
NeuroIntercept
NeuroAttack 2vs2
Cooperative MAS-Attack: NeuroPositioning 7vs8
TRIBOTS
NeuroApproach (Simulation)
evoVision
GridIntercept (Simulation)
NFQApproach (Real)
3D Vision
Neuro Prediction
NFQDribble (Real)
Virtual Neuro-Sensors
NFQControl (Real)
NeuroHassle
98
99
00
01
02
03
04
05
06
07
08
09
NeuroDribble NeuroHoldBall
NeuroAttack 7vs8
3D
twobots
NeuroGoToPos
NeuroAttack 3vs4
NeuroPenalty 1vs1
NeuroScore
CBRIntercept
NFQDribble
NeuroKick06
RSPSAApproach
brainstormers TRIBOTS

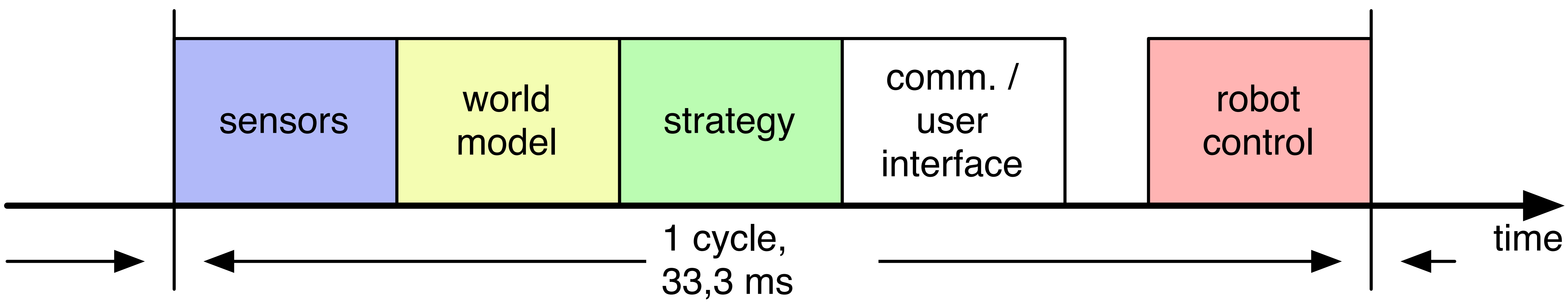


Relevant Design Goals

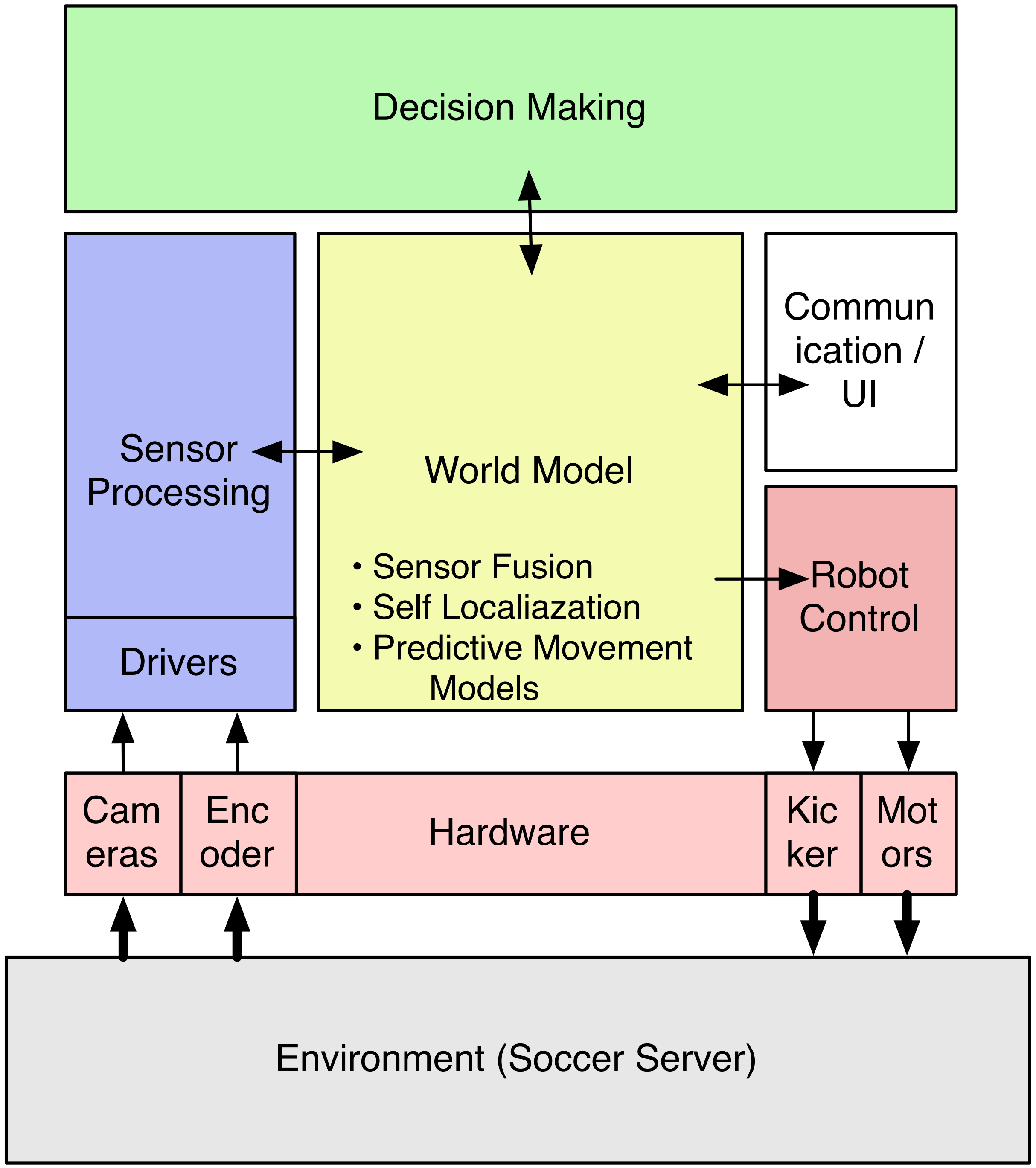
- Research goal: Value-function-based Reinforcement Learning (RL) on real systems
- Markov Decision Process (MDP)
  - discrete time steps
  - state representation with markov property
- As little restrictions as possible on behavior
  - reactive, subsumtion-like behaviors
  - planning

Control Loop

- Control Loop like in Closed Loop Control
  - discrete time steps
  - sequential processing of the modules
  - start of cycle synchronized to camera

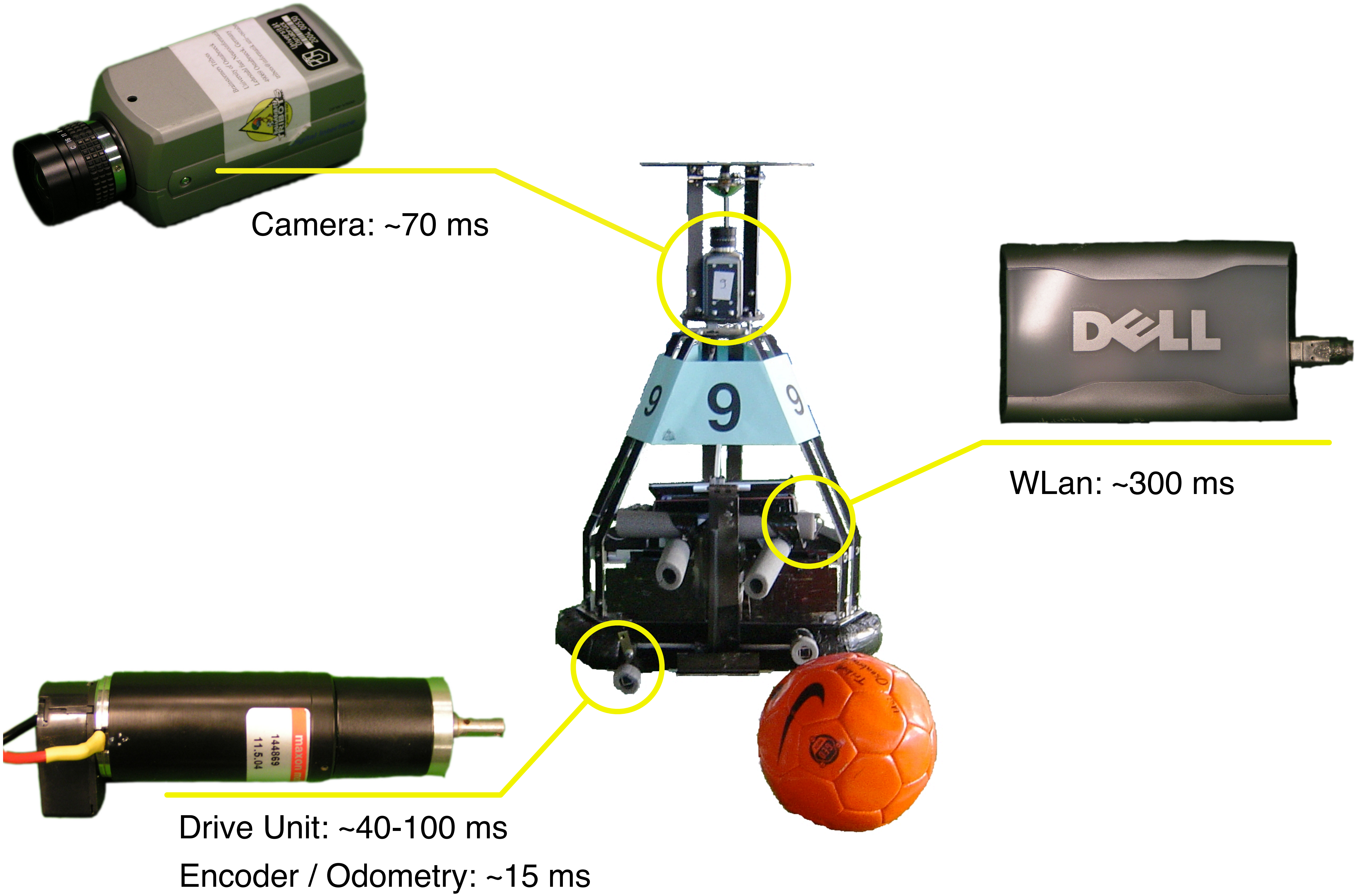


General Architecture



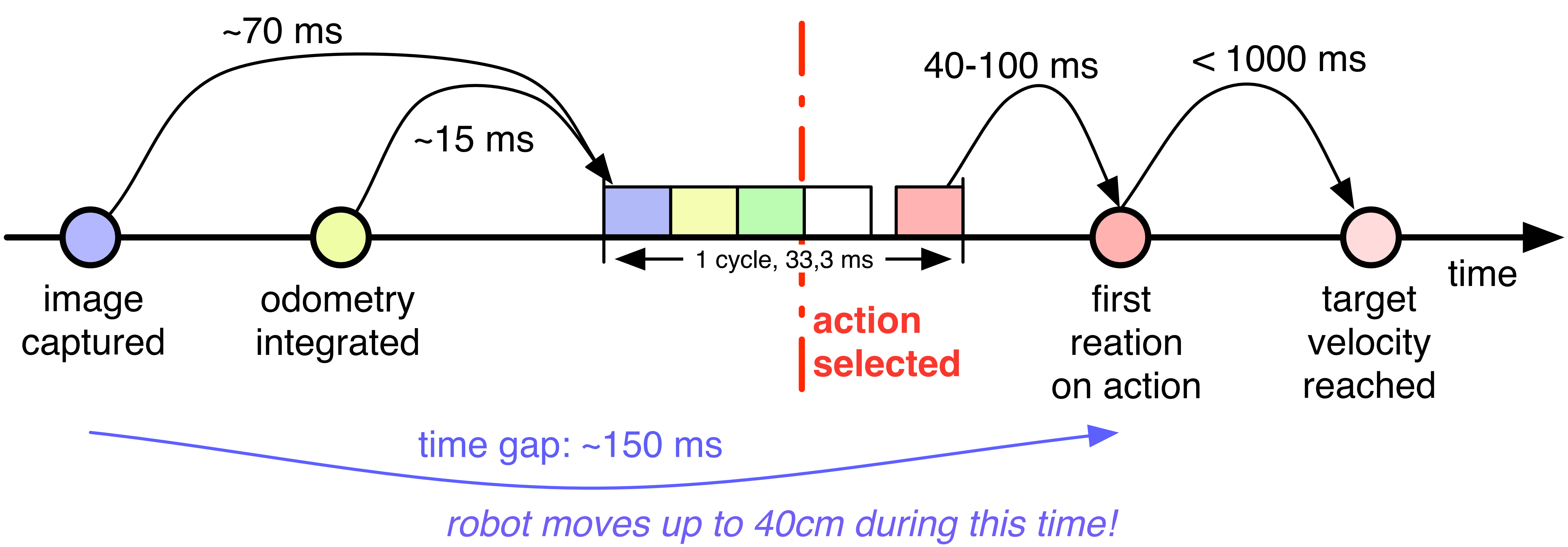
- the world model (WM) is the central module of the architecture, the single point of storage for all information
- all modules access the world model to get and store information
- no module has access to another module (besides the WM)
- all modules have abstract interfaces, implementations are exchangable at runtime through abstract factory-pattern

Delays


Camera: ~70 ms
WLan: ~300 ms
Drive Unit: ~40-100 ms
Encoder / Odometry: ~15 ms

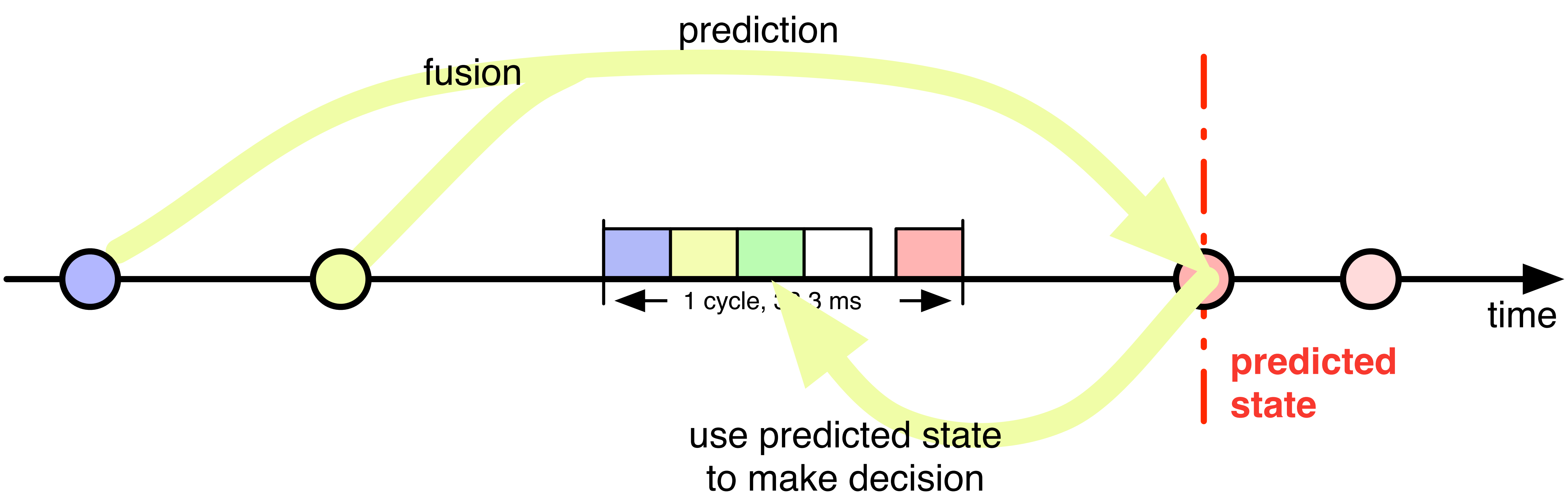
Delays in relation to one cycle

- information is more than 70ms old, when decision is made
- decision won't have any measurable physical effects for up to100 ms



Prediction

- use predictive models to bridge the time gap
- the extrapolated state resembles markov property as closely as possible
- all decisions are made on this extrapolated state information
- thus, decision making is done „delay free“ (ideally)



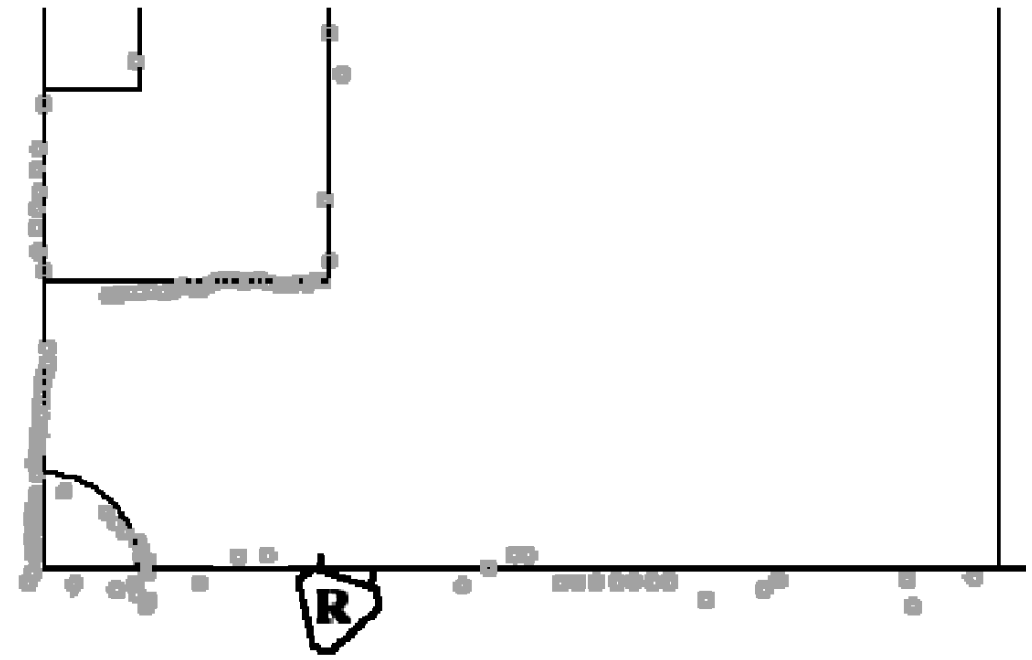
Predictive World-Model

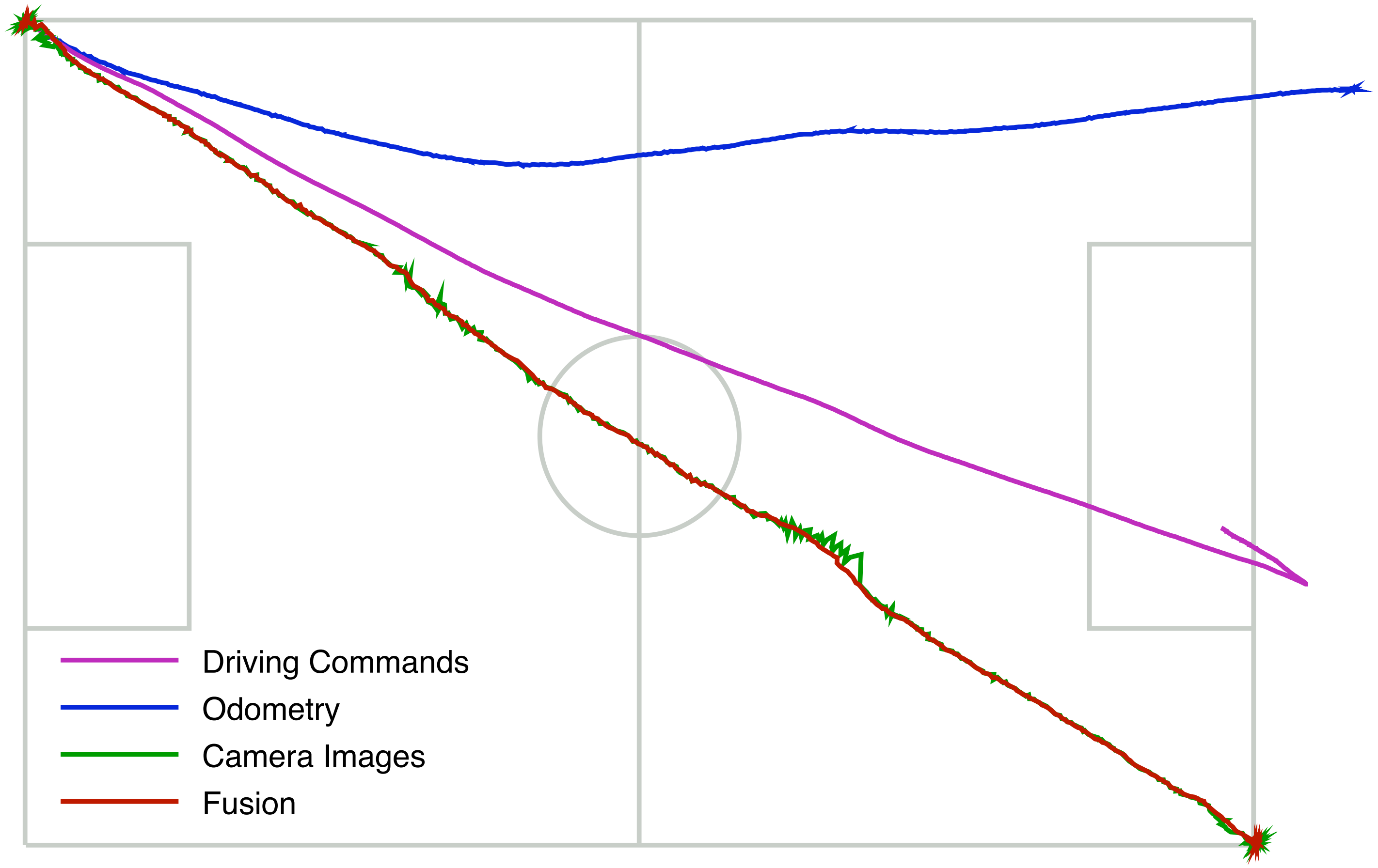
- all prediction is encapsulated in a „predictive world-model“
- all information and all querries are timestamped
- WM is able to „continously“ extrapolate the state into the future
- Sensor Fusion & Models used in the World-Model
  - Self-localization (minimizing an error using gradient descent)
  - Ball-model (robust regression / multiple hypothesis checking)
  - Self-model (robust regression / MLP)
  - Teammate / Opponent-Model (shared WM, robust regression)

Self-Localization

- Gradient descent with Rprop:

$$\underset{\vec{p},\phi}{minimize} \quad E := \sum_{i=1}^{n} err\left(d\left(\vec{p} + \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \vec{s_i}\right)\right)$$





Ego-Motion

$$\phi_{t+\tau} = \phi_t + \omega_t \cdot \tau \quad mod\ 2\pi$$

$$\vec{p}_{t+\tau} = \begin{cases} \vec{p}_t + \vec{u} \cdot \tau & \text{if } \omega_t = 0 \\ \vec{p}_t + \frac{1}{\omega_t} R_{\phi_t} \begin{pmatrix} \sin(\omega_t\tau) & \cos(\omega_t\tau)-1 \\ 1-\cos(\omega_t\tau) & \sin(\omega_t\tau) \end{pmatrix} R_{-\phi_t}\vec{u}_t & \text{if } \omega_t \neq 0 \end{cases}$$

$$\omega_{t+\tau} = \omega_t$$

$$\vec{u}_{t+\tau} = R_{\omega_t\tau}\vec{u}_t$$

estimating the velocity from observations $(p_{i,x}, p_{i,y}, \phi_i, t_i)$:

$$\omega = \frac{n\sum_{i=1}^n(\phi_i t_i) - \sum_{i=1}^n t_i \sum_{i=1}^n \phi_i}{n\sum_{i=1}^n t_i^2 + (\sum_{i=1}^n t_i)^2}$$

$$\begin{pmatrix} n & 0 & \sum s_i & \sum c_i \\ 0 & n & -\sum c_i & \sum s_i \\ \sum s_i & -\sum c_i & \sum(s_i^2+c_i^2) & 0 \\ \sum c_i & \sum s_i & 0 & \sum(s_i^2+c_i^2) \end{pmatrix} \begin{pmatrix} x_0 \\ y_0 \\ u_x \\ u_y \end{pmatrix} = \begin{pmatrix} \sum p_{i,x} \\ \sum p_{i,y} \\ \sum(s_i p_{i,y} - c_i p_{i,y}) \\ \sum(c_i p_{i,x} + s_i p_{i,y}) \end{pmatrix}$$

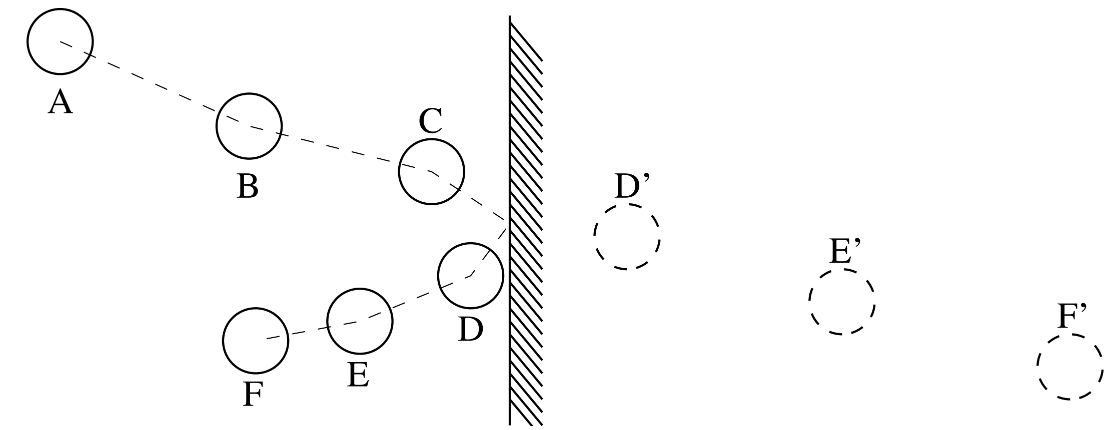
with $s_i = \frac{\sin(\omega t_i)}{\omega}$ and $c_i = \frac{\cos(\omega t_i)-1}{\omega}$

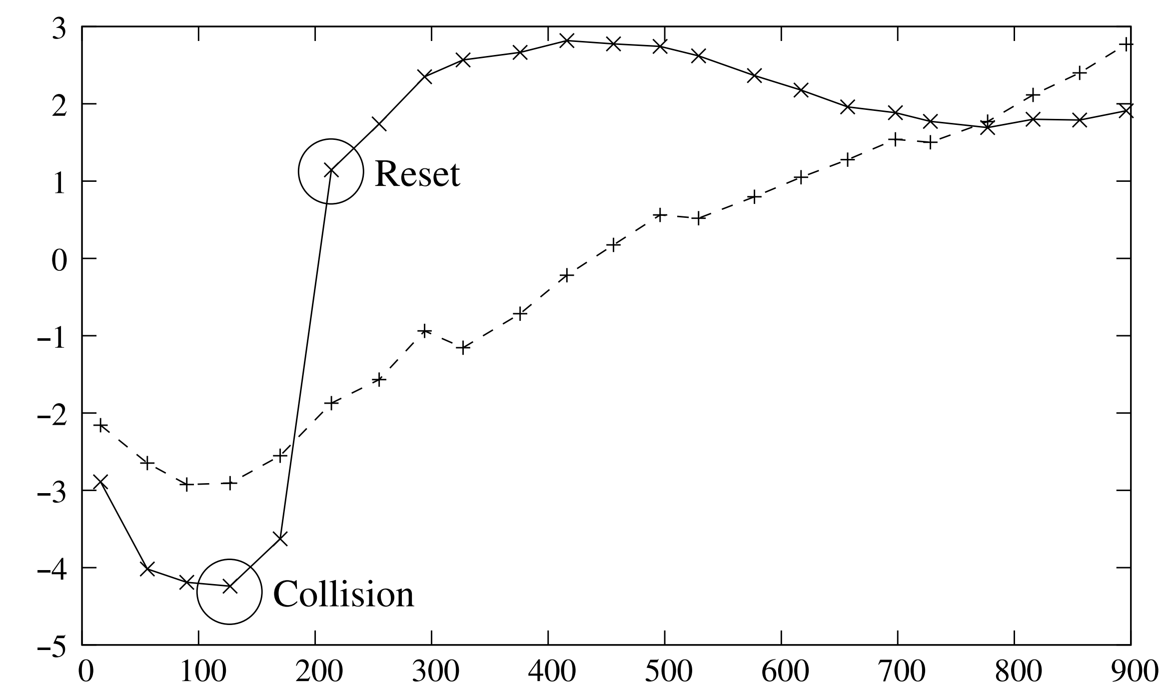
Solving this equation with respect to u_x, u_y yields the velocity estimate.

Alternative: Time series prediction, MLP trained to the transition model.
(similar setup as in „Predicting away the delay“, FU-Fighters)

Ball Motion

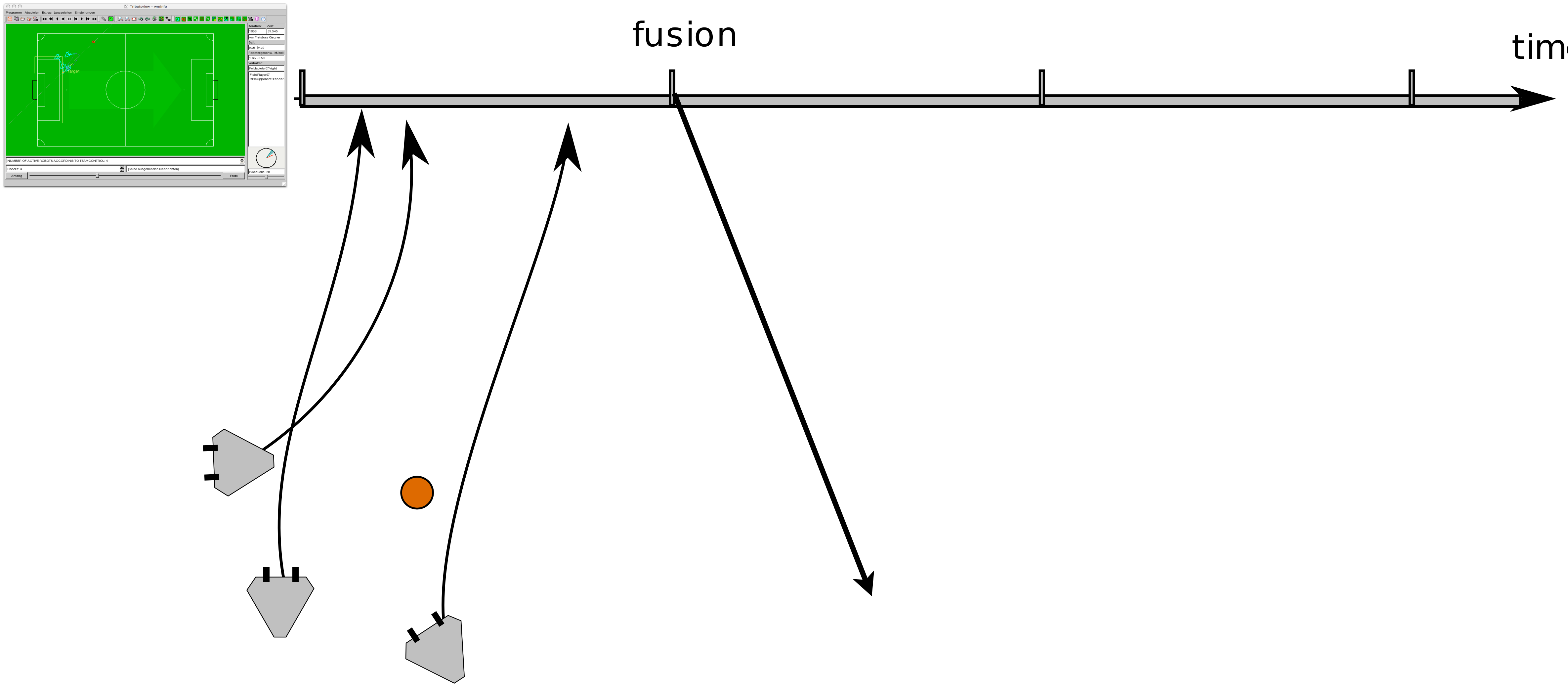
- sequence of linear motion models
- ridge regression to estimate parameters
- collision detection
- detection of the ball being chipped
    - switch to 3D-motion model, if 2nd camera is available





$$\underset{\boldsymbol{p}_0,\boldsymbol{v}}{minimize}\ \frac{1}{2}\sum_{i=1}^{n}||\boldsymbol{p}_0+\boldsymbol{v}(t_i-t_0)-\boldsymbol{p}_i||^2+\frac{\lambda}{2}||\boldsymbol{v}||^2$$

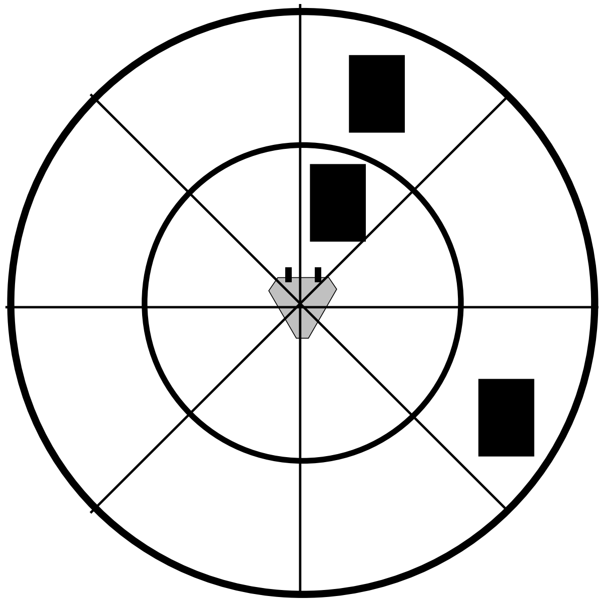
Shared WM



- client-server architecture
- full communication cycle up to 300ms

Shared WM

- information shared with the individual robots
  - position of the ball
  - for each of the teammates
    - position, velocity
    - occupancy grid
  - whiteboard (plain text, unstructured, unrestricted)



Other robots' motion

- we can not discriminate obstacles visually
- tracking by establishing correpsondences between frames minimizing some criterion on the movement of the indistinguishable objects
- same model as for the ego-motion
- communicated positions of teammates are matched to cloeset obstacles (iff non-ambigious) and used for identification only

history of object positions

new measurement

standard criterion from feature point tracking for matching indistinguishable objects

$$\delta\left(\mathbf{p}_{t-2,i}, \mathbf{p}_{t-1,j}, \mathbf{p}_{t,k}\right) = \tag{2.22}$$
$$w_1\left(1 - \frac{(\mathbf{p}_{t-2,i} - \mathbf{p}_{t-1,j}) \cdot (\mathbf{p}_{t-1,j} - \mathbf{p}_{t,k})}{||\mathbf{p}_{t-2,i} - \mathbf{p}_{t-1,j}|| \cdot ||\mathbf{p}_{t-1,j} - \mathbf{p}_{t,k}||}\right)$$
$$+w_2\left(1 - \frac{2\left[||\mathbf{p}_{t-2,i} - \mathbf{p}_{t-1,j}|| \cdot ||\mathbf{p}_{t-1,j} - \mathbf{p}_{t,k}||\right]^{\frac{1}{2}}}{||\mathbf{p}_{t-2,i} - \mathbf{p}_{t-1,j}|| \cdot ||\mathbf{p}_{t-1,j} - \mathbf{p}_{t,k}||}\right)$$

Tools

- Tribotsview
- 2 Simulators (team behavior / physically more adequate simulation of drive unit)
- Teamcontrol
- ColorTool / Camera Calibration
- Tacticsboard - modifiable tactics
- iPhone Control

- If you‘re interested in one tool, please ask...

Conclusion

- Sequential processing in control loop
- All information stored in World-Model
- No direct communication between other modules besides World-Model
- Prediction to „remove“ delay
- Markov Decision Process