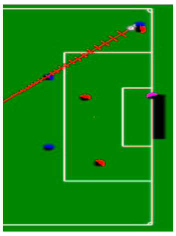
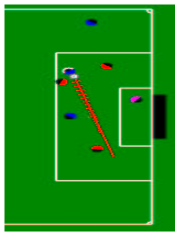
The Brainstormers -
Learning on the Tactical Level

GOAL: LEARN TO USE THE BASIC SKILLS BY SUCCESS OR FAILURE



Tactics: cooperative team play



Tactics: cooperative team play

Tactics

- *actions: basic skills with varying length (kick, intercept, go2pos, ...)*
- *need for cooperation ⇒ multi-agent reinforcement learning*
- *clearly defined goal: score!!!*

AI used in competition team

- *estimation of success probabilties (used in 1999; (Buck, Riedmiller, 2000))*
- *Depth-limited search for with-ball tactical decisions (since 2000)*
- *Neural network based cooperative attack positioning (since 2001)*
- *Neural network based complete cooperative attack behaviour (since 2002)*

Realized, but in research stage

- *Feature-Q-learning based attack (double-passes! (Ehrmann, Riedmiller)*

Research Issues

- *Reinforcement learning in cooperative multi-agent systems*
- *MAS-RL with features*
- *learning to communicate (Moore, Schneider, Riedmiller, 2001)*
- *incorporation of prior knowledge (constraints, special actions (homepositions), partially fixed policies (goalshot))*
- *theory: MAS-q algorithm for deterministic domains (Lauer, Riedmiller, 2000)*
- *theory: MAS-q algorithm for stochastic domains (Lauer, Riedmiller, 2001)*
- *theory: MAS-RL under varying conditions (e.g. exchange of information during learning): NAI, FAI, PSI, FSI, ...*

Learning 7 vs 8 Attack

- *individual decisions;*
- *actions: no-ball: 8 directions, default position*
  with-ball: pass, dribble, hold-ball
- *prior knowledge: intercept and goalshot are decided otherwise with priority*
- *cooperate (central) learning and distribution of value function*
- *⇒ theoretically justified: optimal cooperate policy can be found*
- *practical compromises: model based instead of Q-learning*
- *use of an approximate model*
- *raw coding of positions: $(7 + 8) \times 2 + 4 = 34$ inputs*

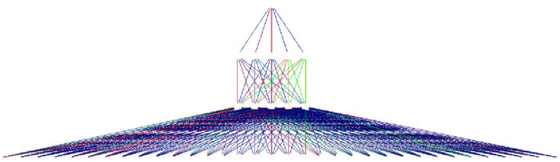
MAS learning algorithm

```
FOR all Situations s repeat n times{
   set_situation;
   play_and_record();
   /* until sequence terminates
}
evaluate&generate_training_patterns();
train_NN_by_Rprop();
distribute_NN();
REPEAT playing
```

Results - example: positioning

Policy type	Average Seq.Length	Success
random value function	*139.8*	*0.16 (0.10)*
attack 2000 (handcoded)	*129.4*	*0.18 (0.12)*
Neuro Positioning (2001)	*118.0*	*0.29 (0.18)*

improved by 1.7 over vice-world champion team 2000!



Neural Network for Positioning

Individual action selection

```
GenerateActionSet();
FOR all actions{
   AbstractModel(s,a,s');
   evaluate(s')
   determine_best_action();
}
```

GENERATEACTIONSET(): *depends on situation (w. ball, no. ball); estimation of success*

ABSTRACT MODEL(): *Rough estimation of successor state*

Adaptive Systems Group
Prof. Dr. Martin Riedmiller, Artur Merke
Lehrstuhl Informatik 1
University of Dortmund, Germany

sponsored by DFG under grant SPP 1125