Cognitive Robotics

Introduction

Hans-Dieter Burkhard
June 2014

Organizational Issues

Tuesday 17.6. - Wednesday 25.6.2014

Lectures and Labs: Prof. Hans-Dieter Burkhard

Labs with Framework RoboNewbie

Slides will be provided after lectures on
http://www.naoteamhumboldt.de/projects/RoboNewbie_Plovdiv2014

Programming Exercises

are based on the RoboNewbie Framework

developed by Monika Domańska

Required general resources

(download and install from net)

1. WindowsXP or newer
2. Java Development Kit 7
3. NetBeans (v. 7.1 or later, JavaSE or JavaEE)
4. Java 3D

Programming Exercises

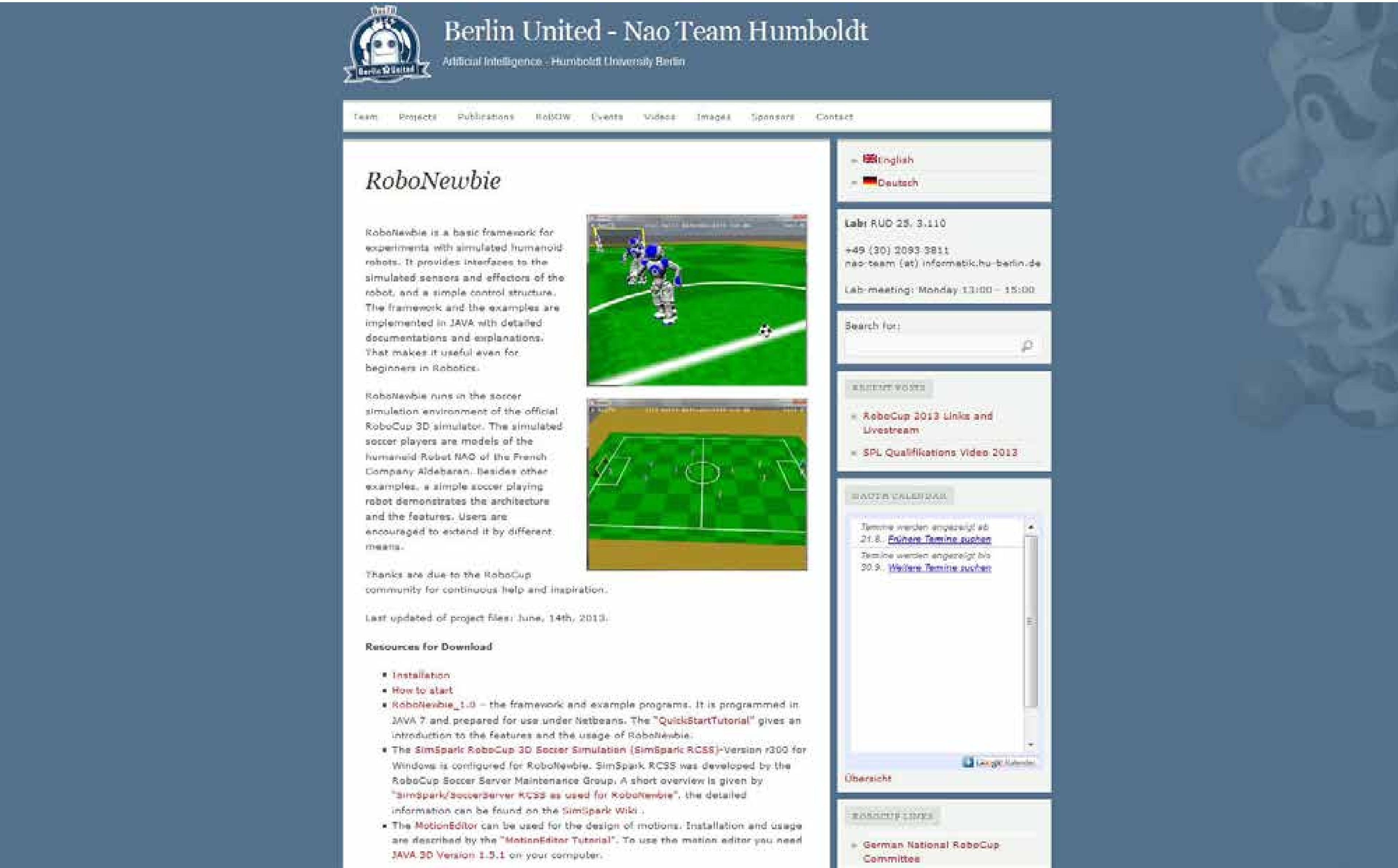
Required special resources, download from

http://www.naoteamhumboldt.de/projects/robonewbie

1. RoboNewbie
2. MotionEditor
3. SimSpark RoboCup 3D Soccer Simulation (SimSpark RCSS)

Additional materials for installation on that page.

Programs and related instructions are available on http://www.naoteamhumboldt.de/projects/robonewbie/



Outline Introduction

Introduction

Simple Example

RoboCup

RoboCup: 3D-Simulation League

Locomotion

Acting in SimSpark/RoboCup

Keyframe Motions

Perception

Perceptors in SimSpark

All topics will be explained later in more details.

Decisions Performed by Machines

Examples:

- Chess
- Search Engines
- Computer Aided Design
- Language Translation
- Industrial Robotics
- Photography
- Driver assistance systems
- Space discovery

Not all of them are "intelligent"

Assistance for humans
Guidance of humans
Autonomous machines

DARPA Grand Challenges

Pictures by DARPA
and Telepolis (H.A. Marsiske)

1. Competition (desert):	2004
2. Competition (simple desert):	2005
Urban Challenge :	2007
Robotics Challenge:	2012-14

1. DARPA Grand Challenge 2004

2. Grand Challenge 2005

DARPA Urban Challenge 2007

Recent Developments

Google using street view

DARPA Robotics Challenge 2012-14

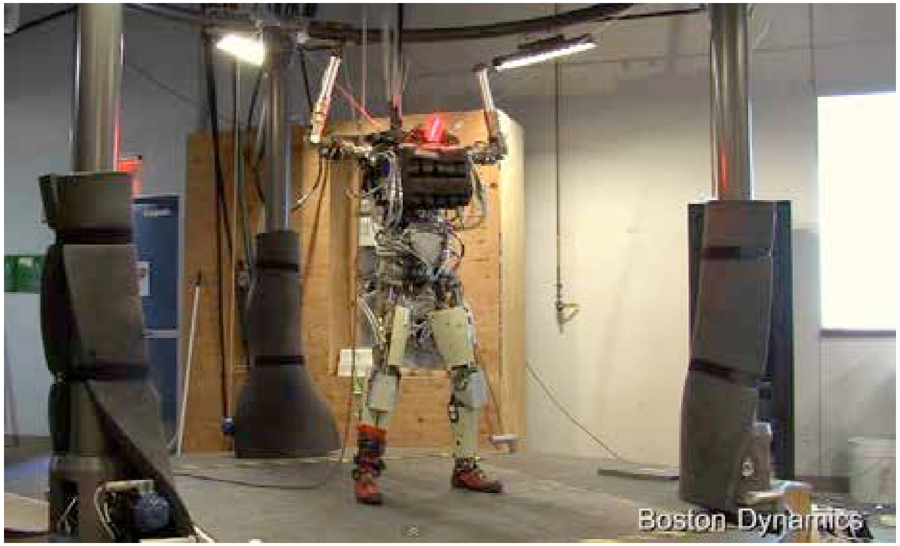
Robots in desaster response scenario

The robot has to

1. use an unmodified vehicle to drive to disaster area
2. traverse through divested area
3. remove debris blocking an entry
4. open a door and enter a building
5. climb a ladder and traverse industrial walkway
6. break through wall using appropriate tools
7. locate and close a valve near a leaking pipeline
8. replace a defect component

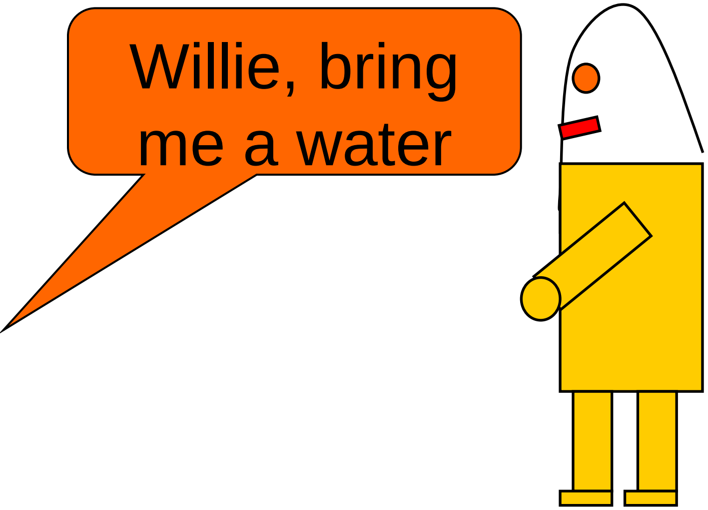
DARPA Robotics Challenge

- Semi-autonomy
- Control by non-expert operators
- Acting in normal environment after a catastrophe
- Usage of standard tools
- Extern power supply allowed as far as conform with tasks



A robot platform like PETMAN from Boston Dynamics was provided for selected participants.

Example: Service Robots



Alternatives:

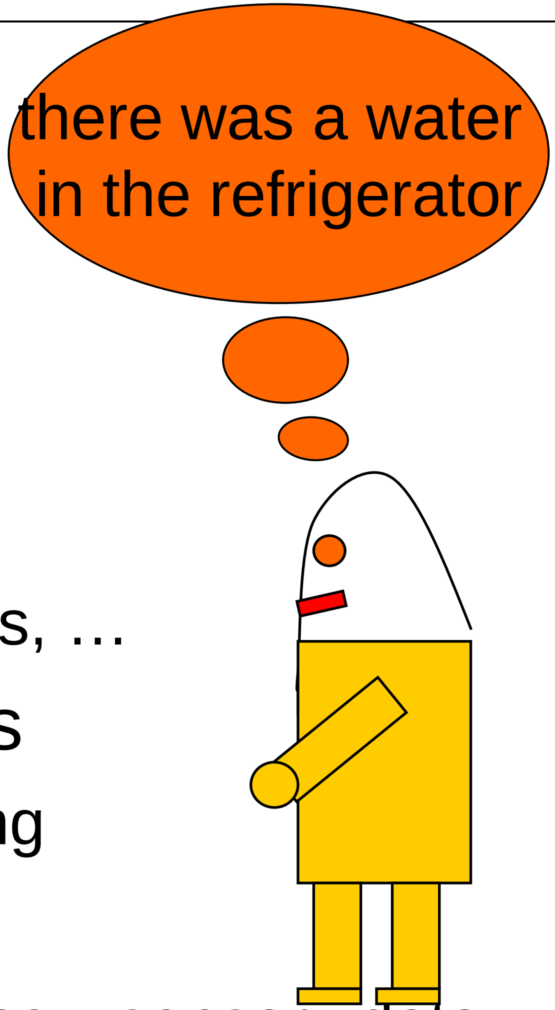
- from the refrigerator
- from the cellar
- from the neighbor
- from the shop
- from the internet
- …

Which alternative to choose?

What else is needed (glass, …)?

Robot Needs Knowledge about the world

World model:
Part of state in the program



Facts about the world
- maps, positions of objects, descriptions, …

Methods for processing sensory inputs
- language processing, image processing

Methods for integrating sensory data
- new world model from old model and new sensory data

World Model

Problems:

Environment is only partially observable

Observations are insecure and noisy

Scene interpretation with Bayesian methods, e.g.

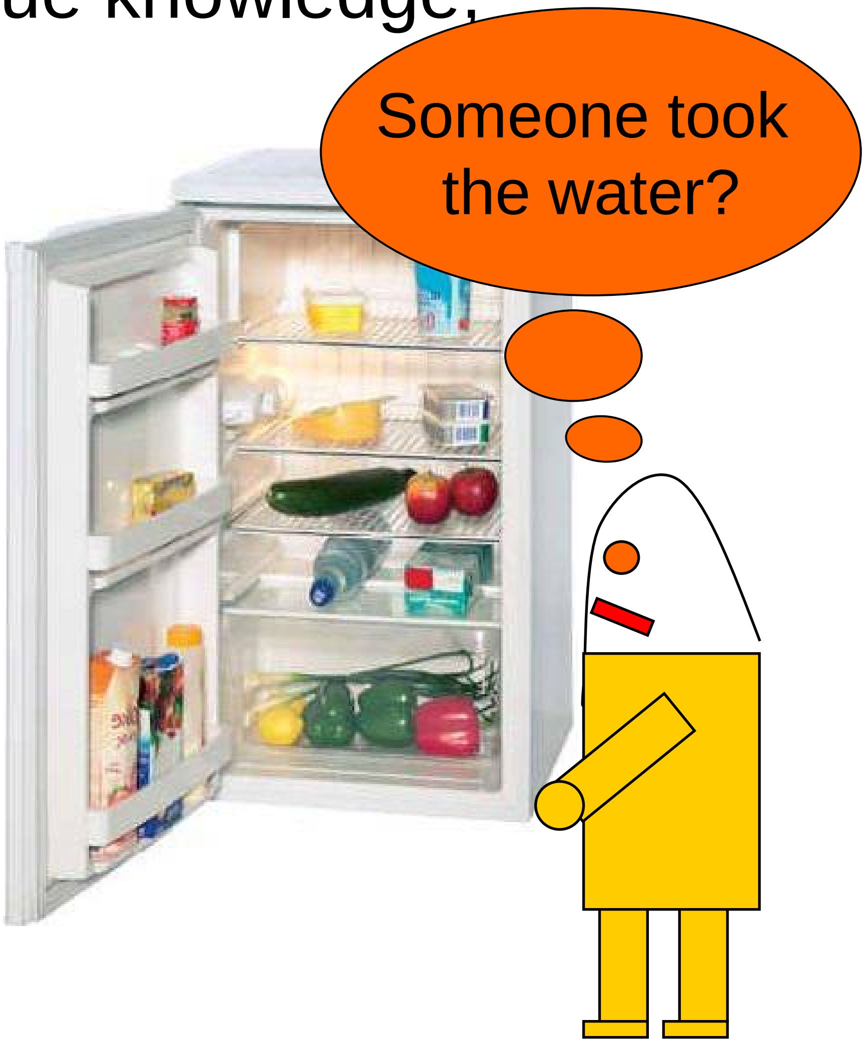
Probability to be at location s given an observation z:

$$P(s|z) = P(z|s) \cdot P(s) / P(z)$$

World Model

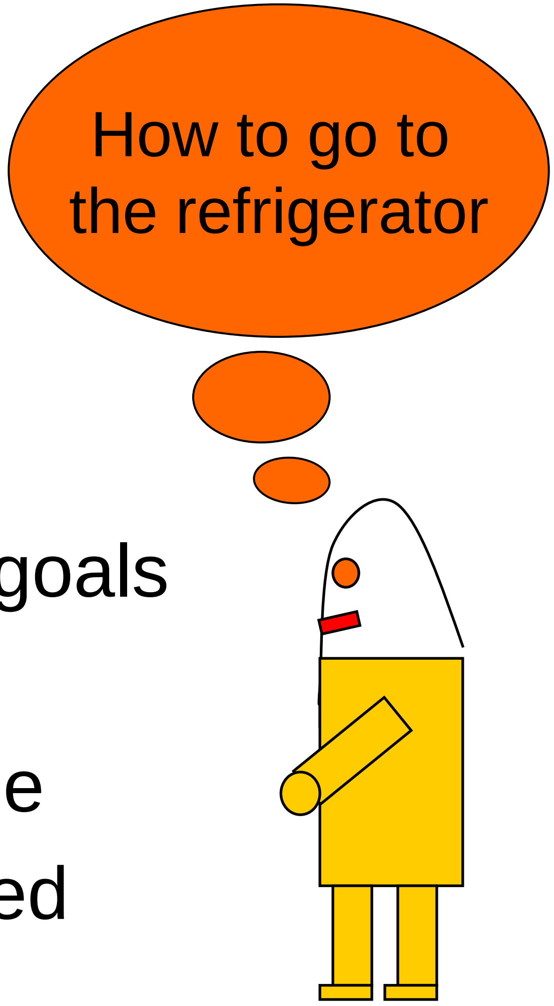
World model need **not** be true knowledge,

only **belief** of the agent.



Commitments

Commitments:
Part of state in the program



Tasks/Goals: Desired world states

Plans: Sequence of actions to reach goals

Rationality: Agents should only pursue goals/plans that can be achieved

Commitments

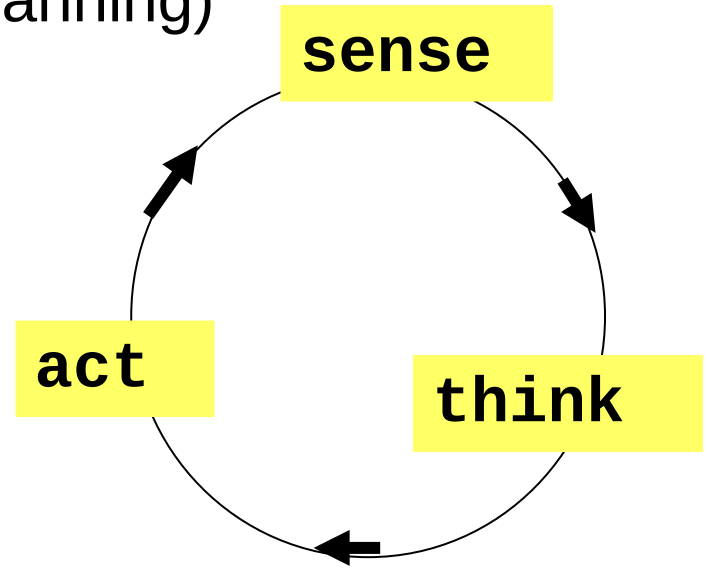
Plans may fail.

Need methods for revision.

Putting Together: Sense-think-act Cycle

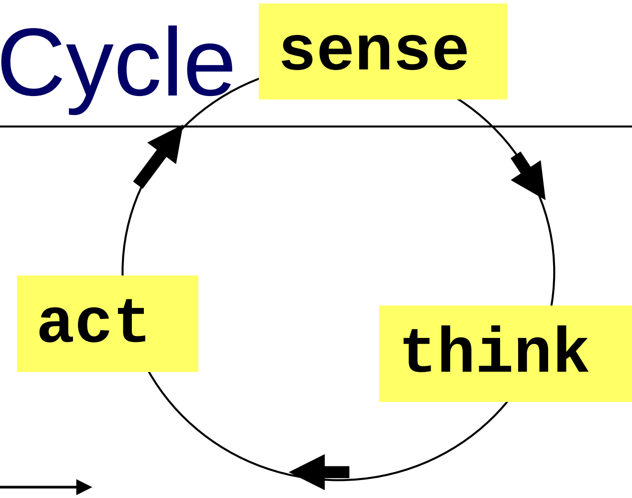
Ordering of intern processing of the agent

1. Sense („input“) + perception (interpretation, world model)
2. Think (“processing”: evaluation, planning)
3. Act („output“)

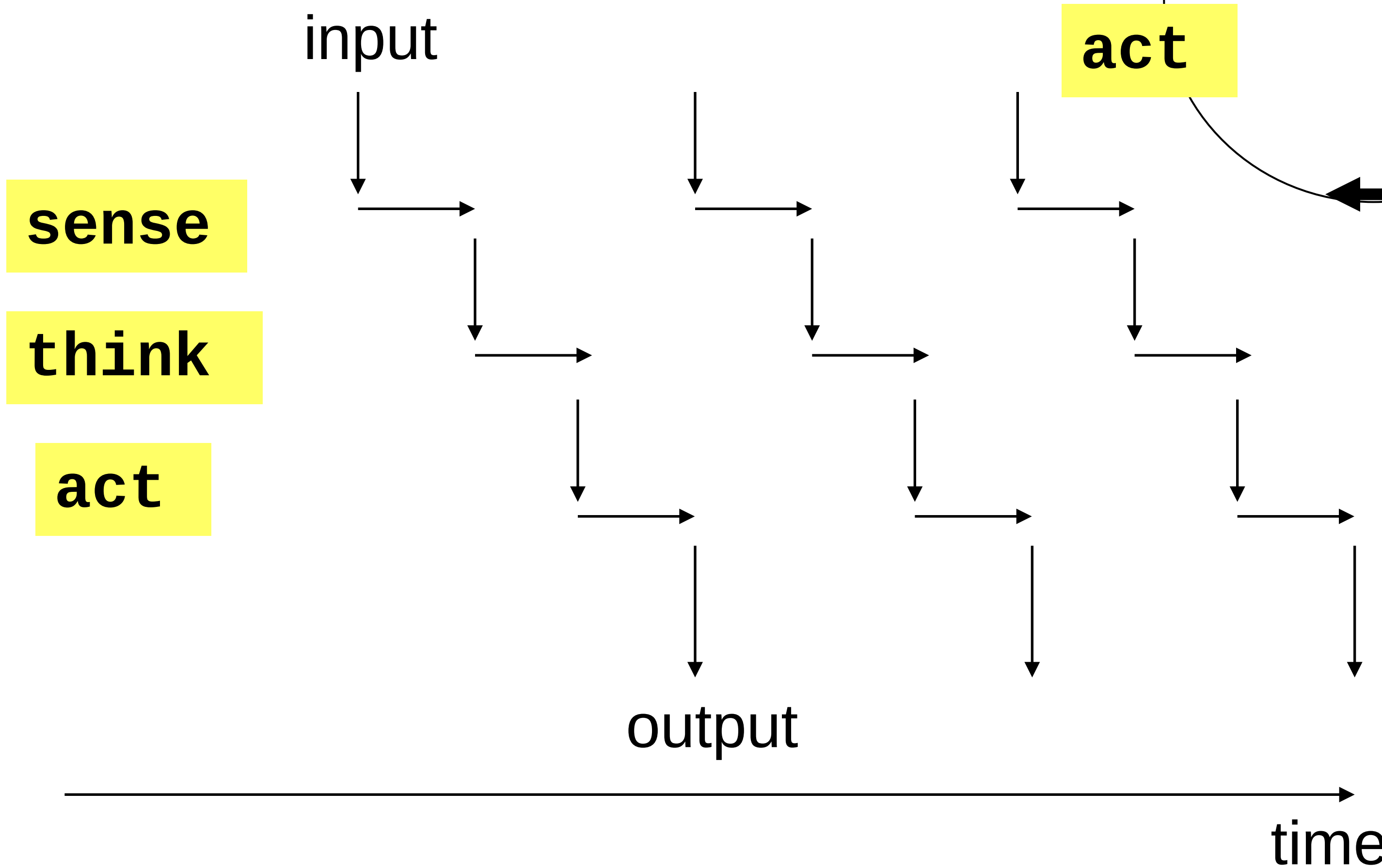


States in the program
- World model
- Commitments

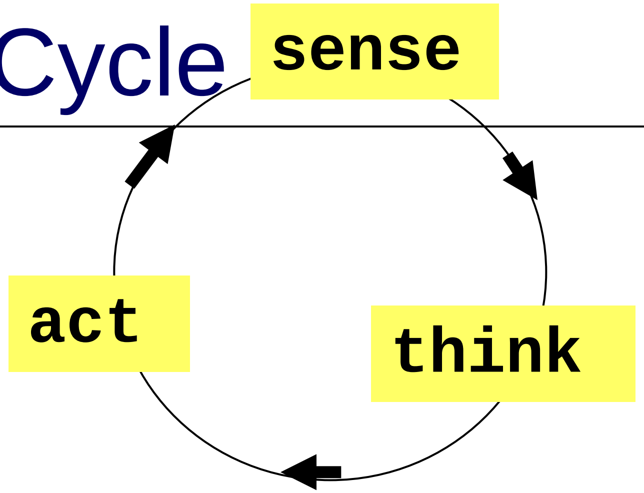
Sense-think-act Cycle



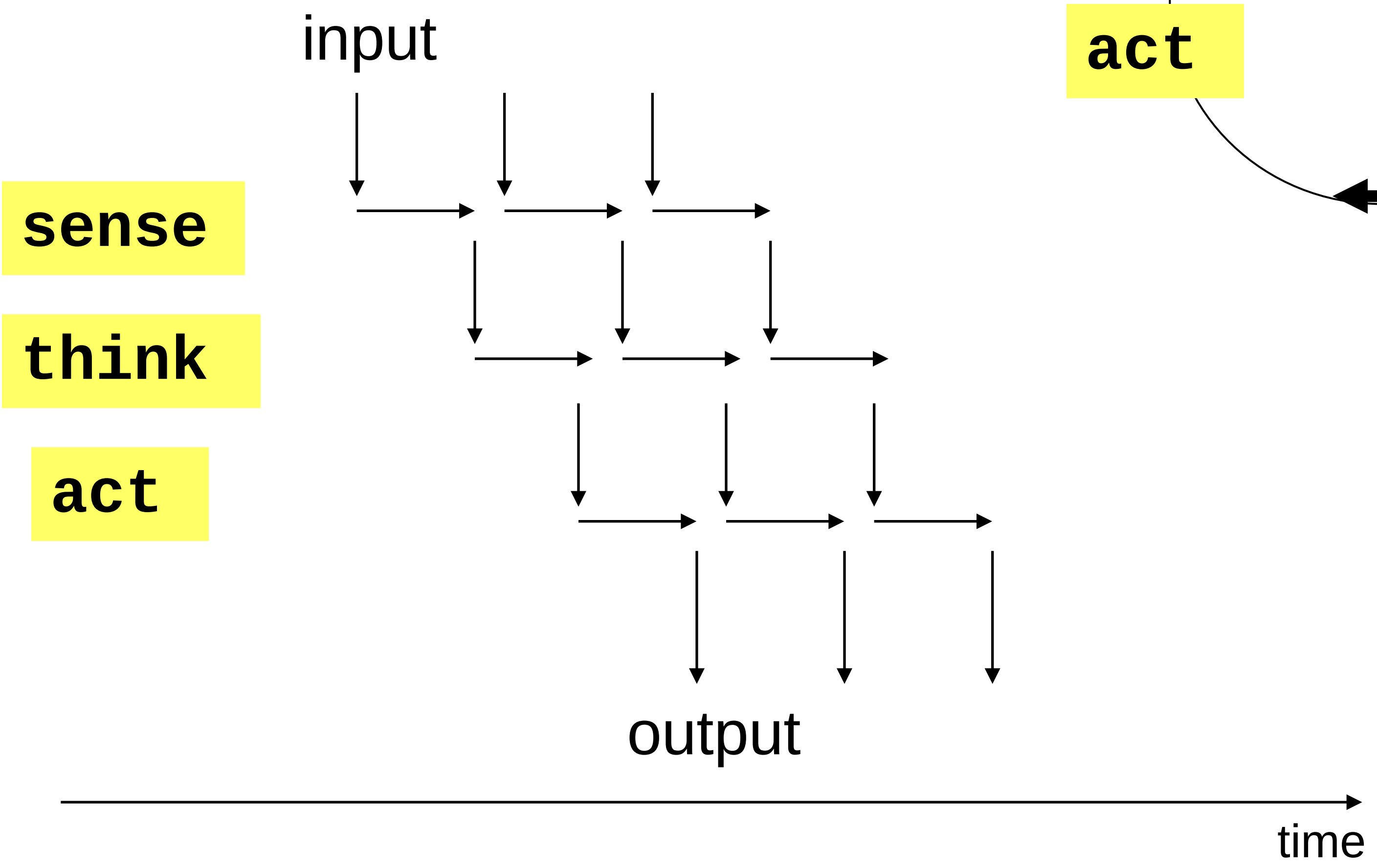
Synchronisation (sequential)



Sense-think-act Cycle

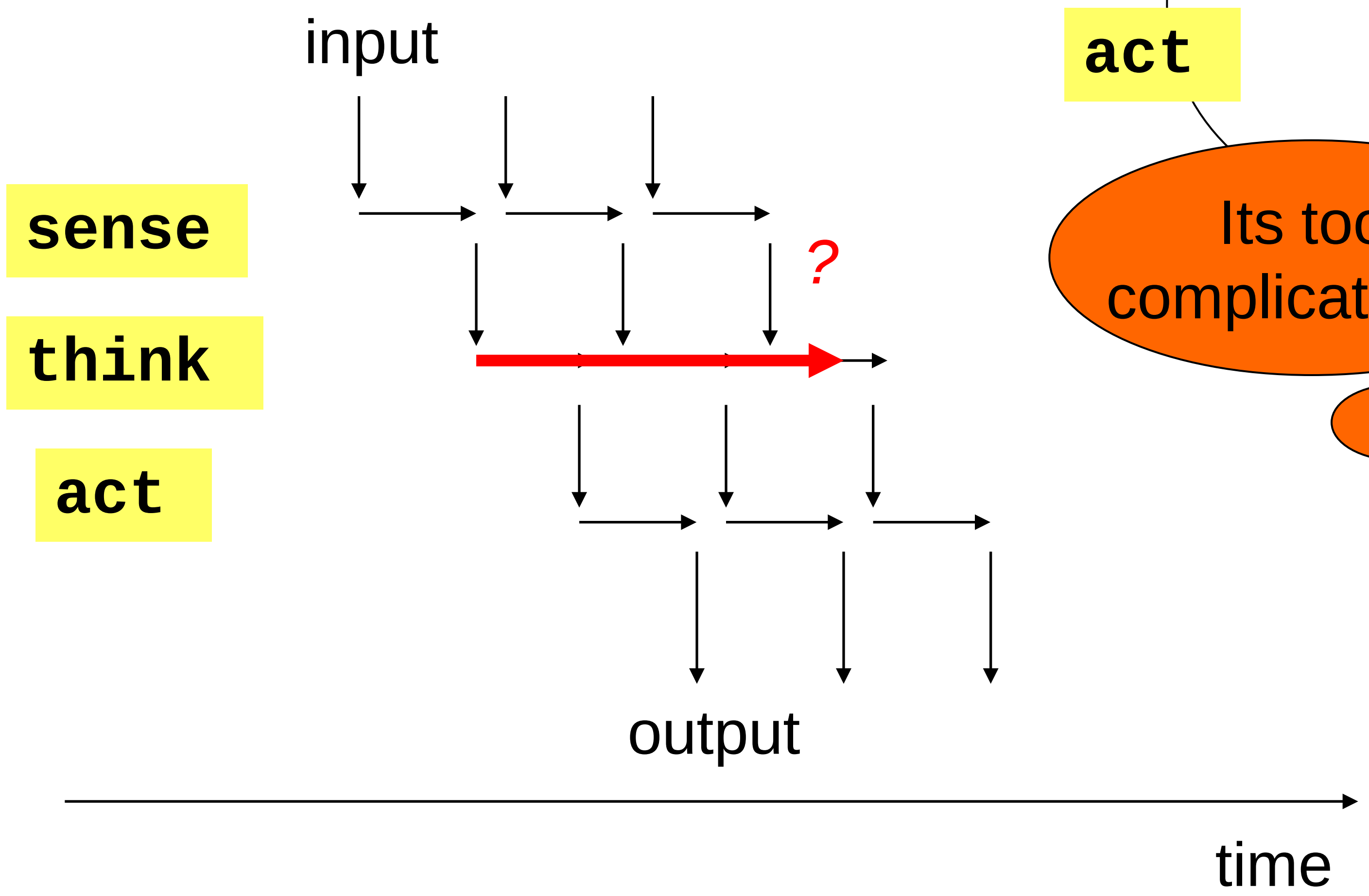


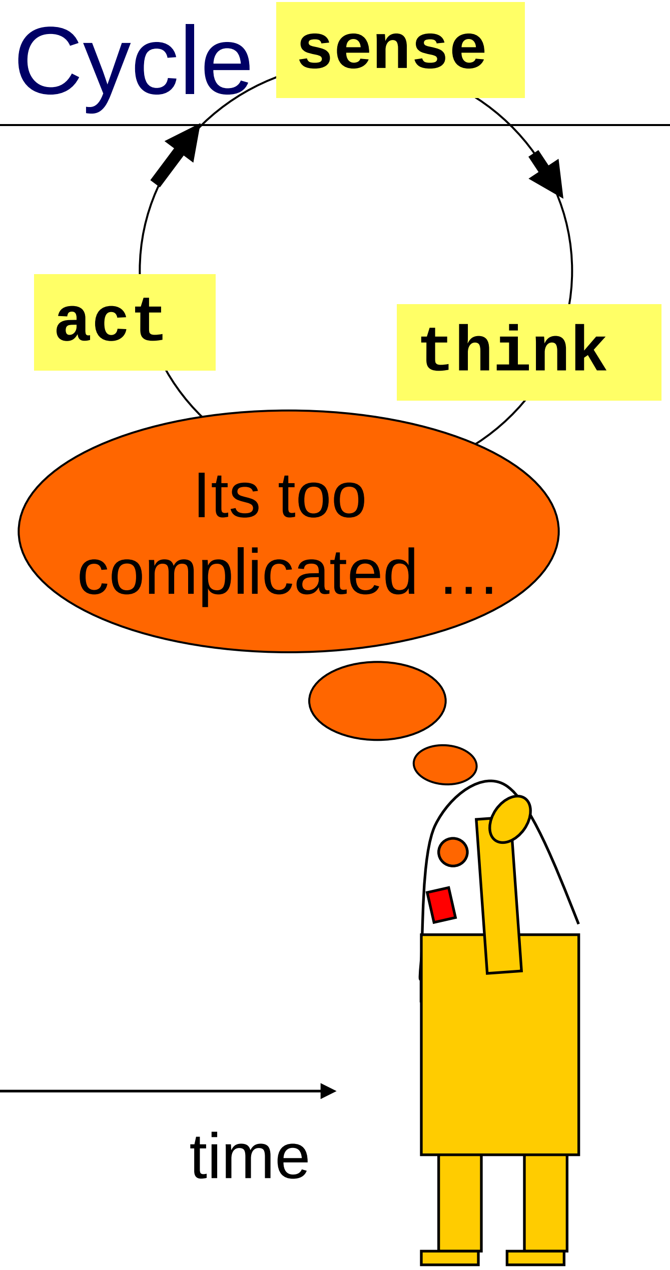
Synchronisation (concurrent)



Sense-think-act Cycle

Synchronisation problems





"Autonomous Agents"

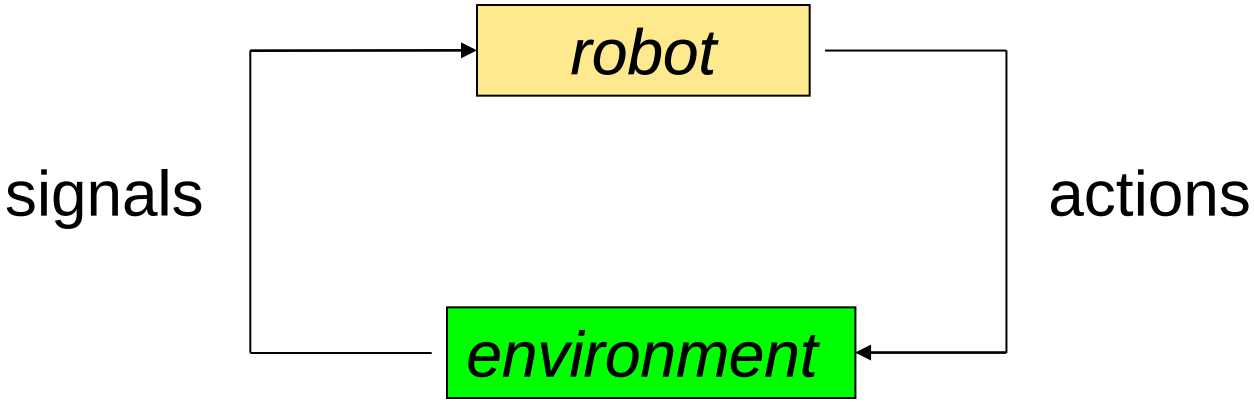
act in a certain environment on behalf of its user

… a long running program, where the work can be meaningfully described as autonomous completion of orders or goals while interacting with the environment.

Further attributes may be:

Intelligent, social, reactive, proactive, mobile, …
adaptive, learning ,…
goal-oriented etc. (modeling human-like attitudes) …

Acting in the environment



Software agents:

Clearly defined virtual environment

Robots:

Real environment with incomplete and unreliable information

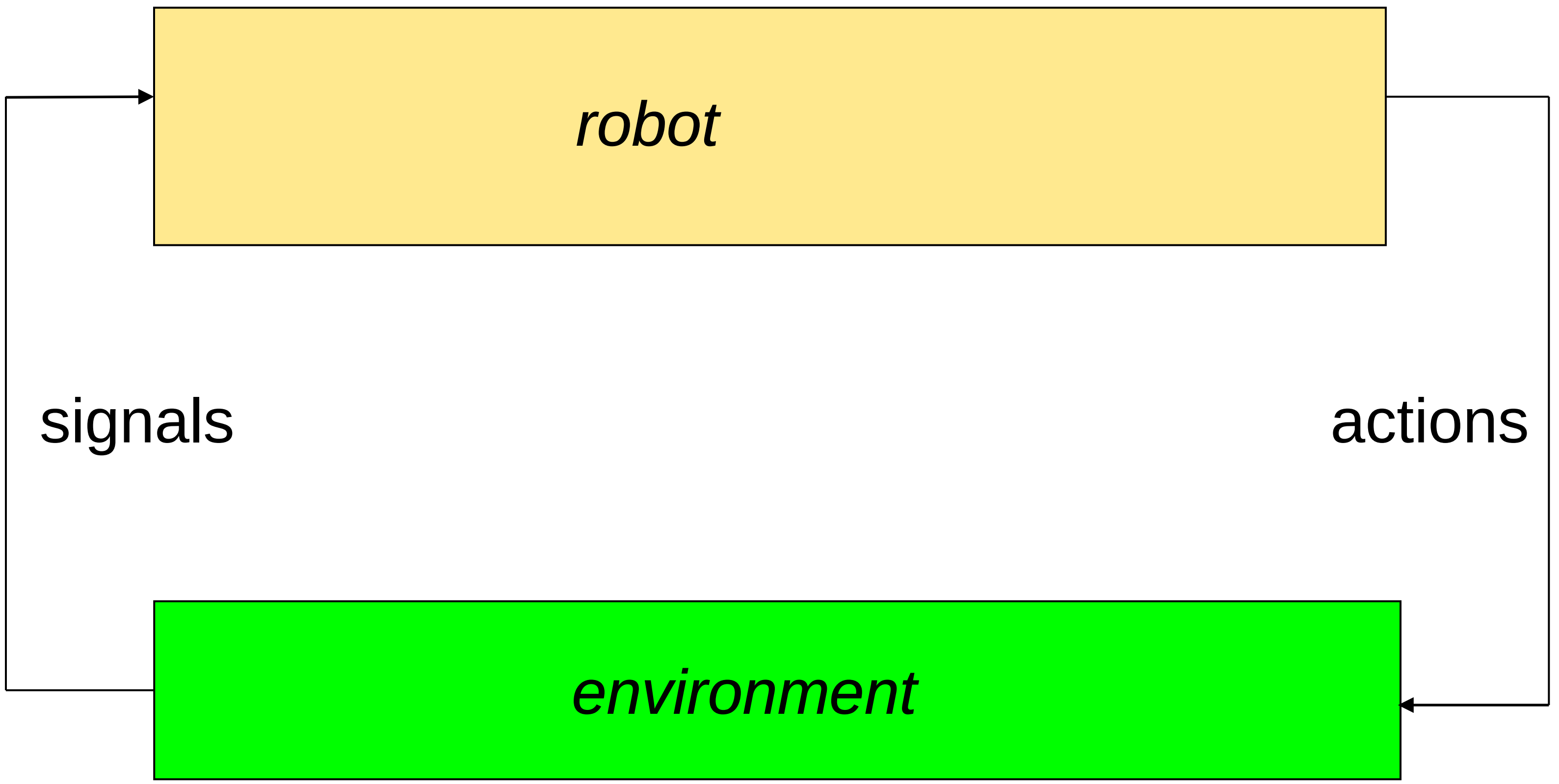
Chess program vs. Soccer robots

1997: Deep Blue wins against human chess champion Kasparov

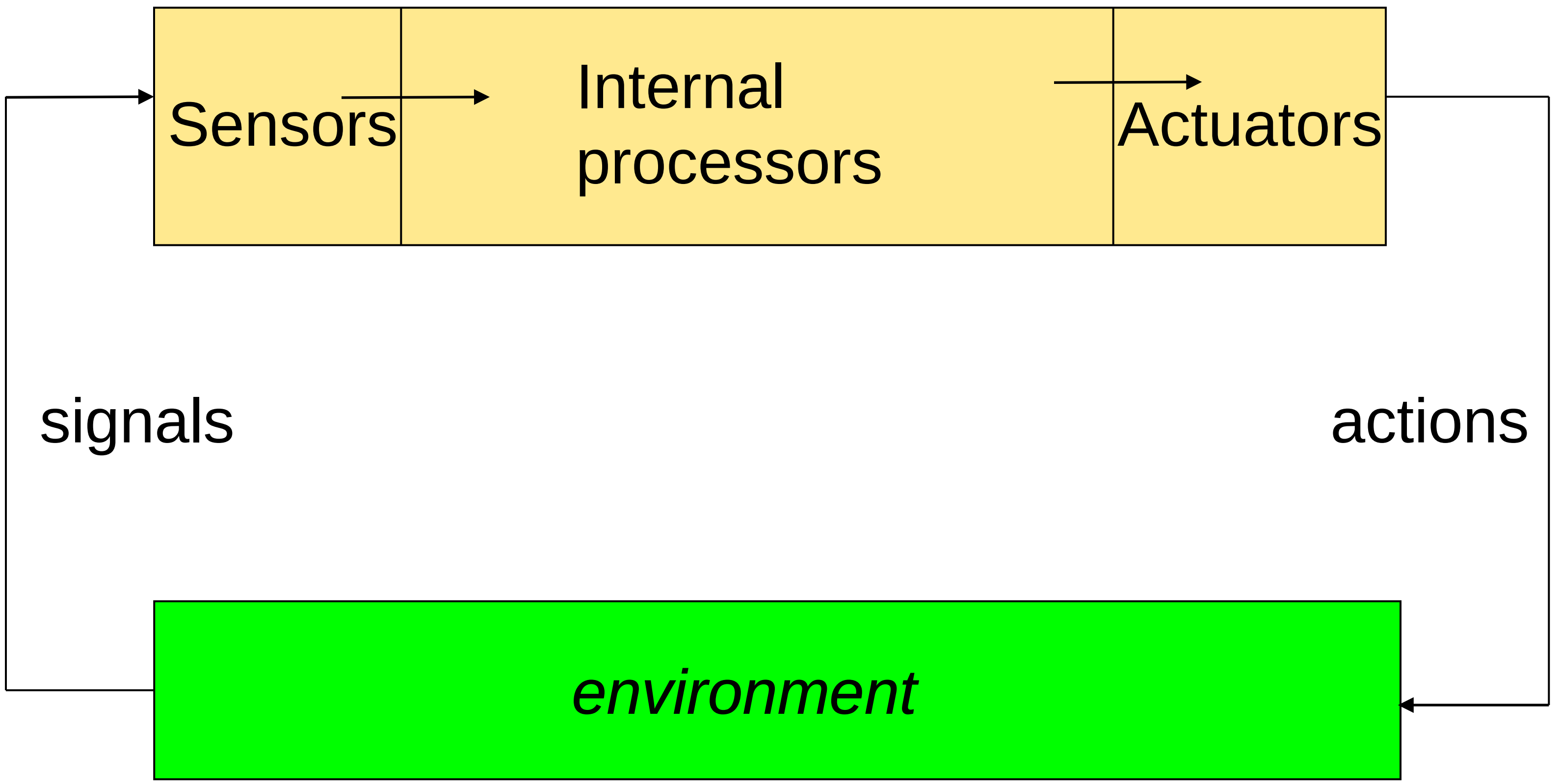
Chess:
- Static
- 3 Minutes per move
- Single action
- Single player
- Information:
  - reliable
  - complete

Soccer:
- Dynamic
- Milliseconds
- Sequences of actions
- Team
- Information:
  - unreliable
  - incomplete

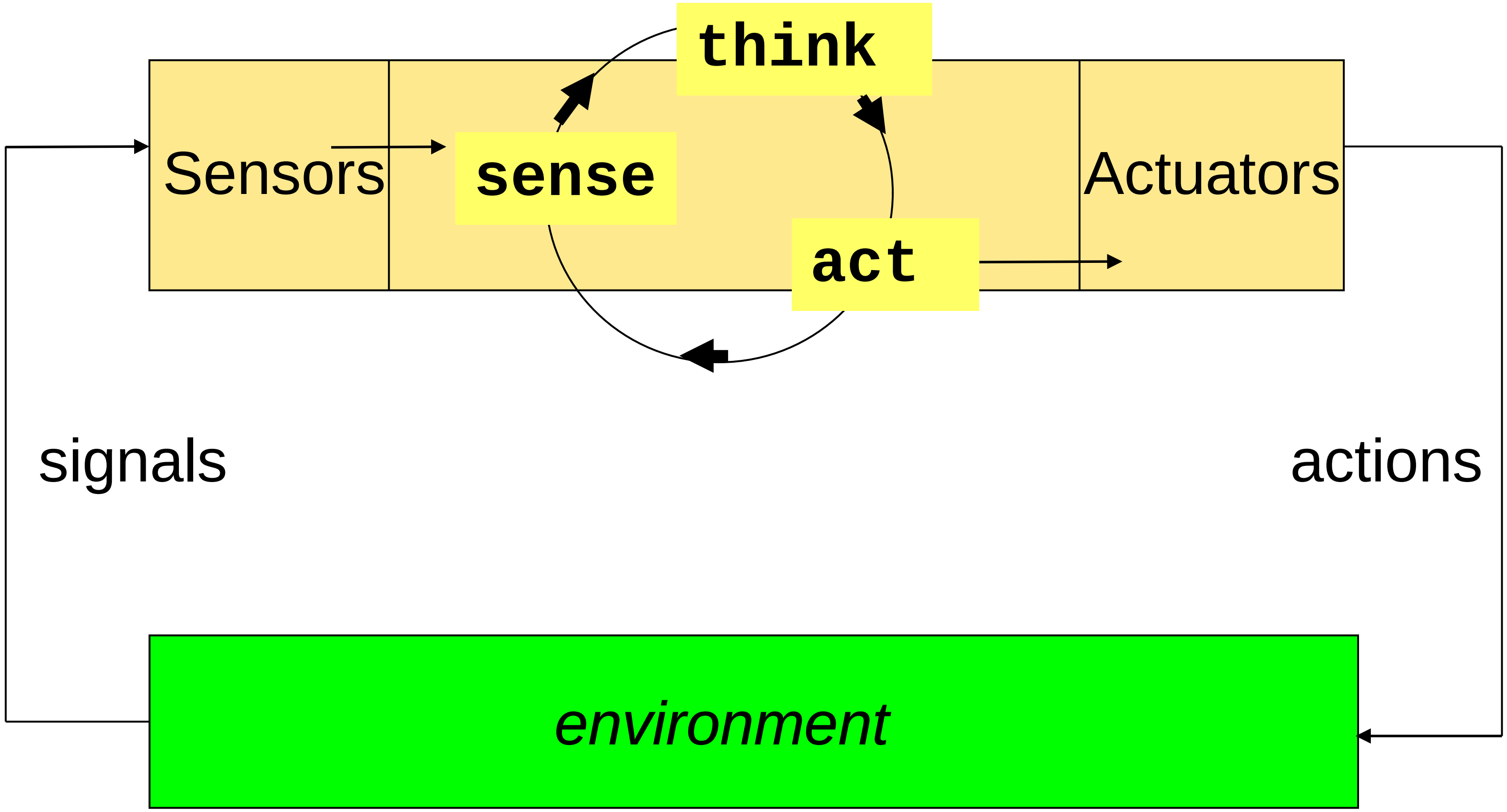
Acting in the environment



Inside the robot

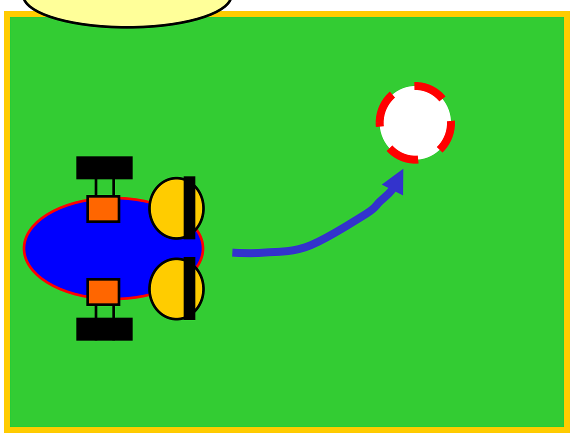


Inside the robot: Sense-think-act cycle



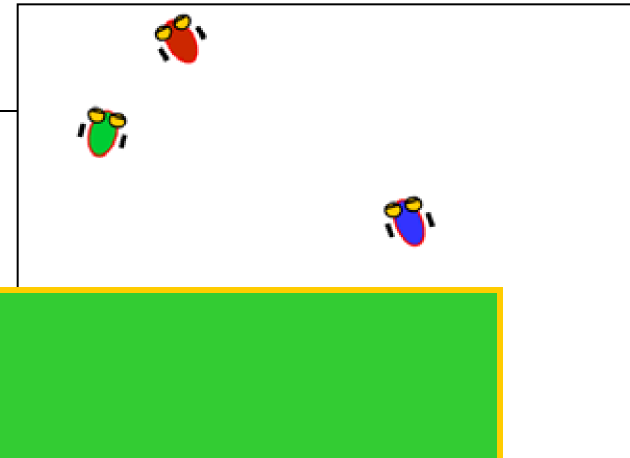
„Conscious“ Acting

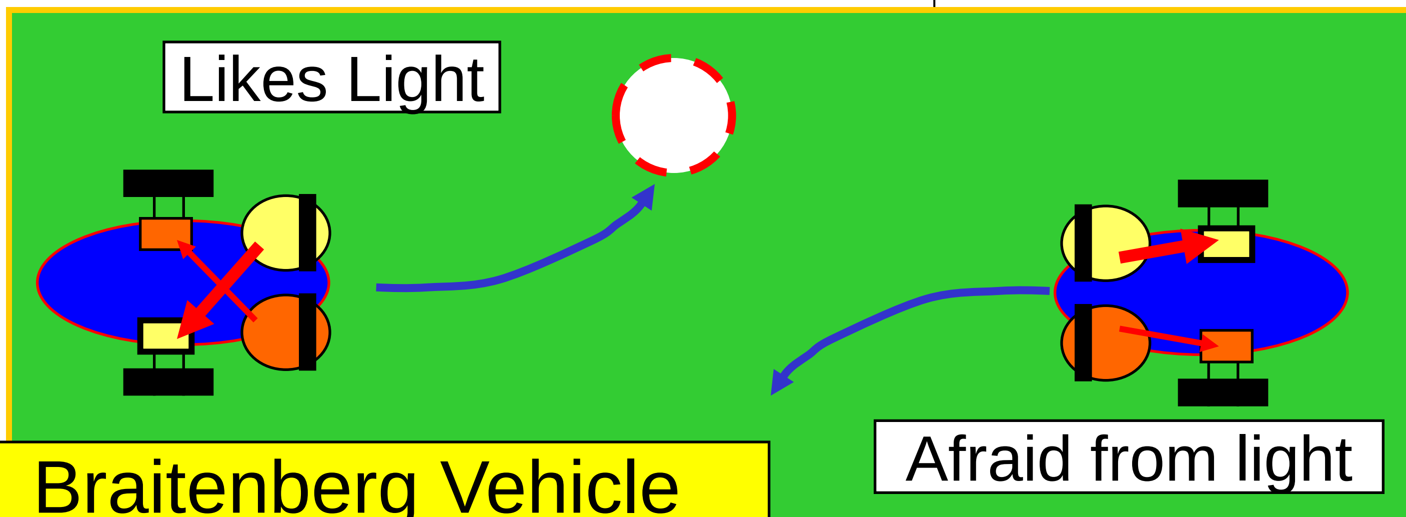
I see the light left in front.
I like the light.
I should go to the left.
I have to turn and walk.
My right wheel should move faster than the left one.
etc.



Alternatively:
Stimulus Response

Stimulus Response



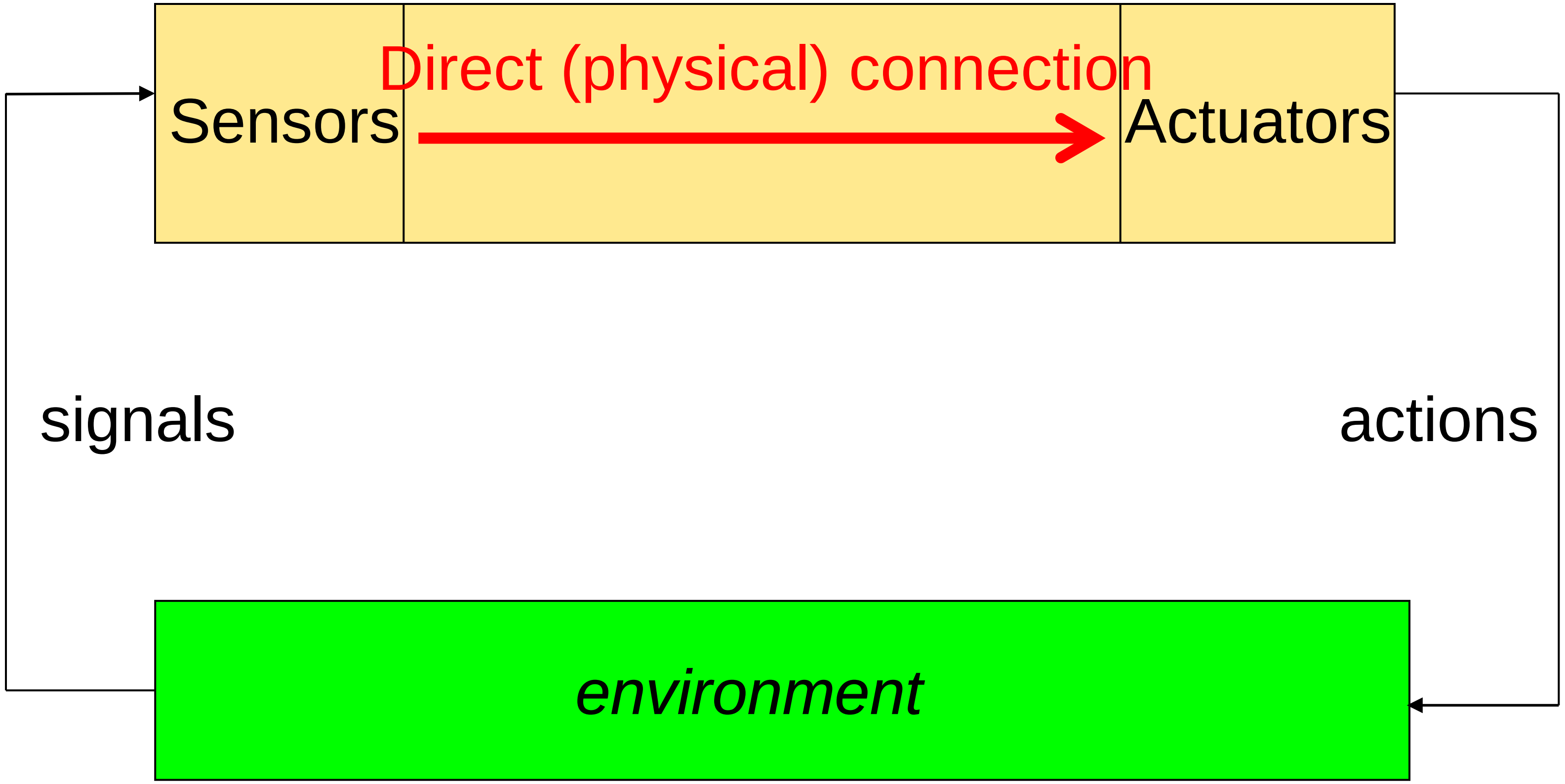


Braitenberg Vehicle
Cybernetic Turtle

Alternatively: „Conscious“ Acting

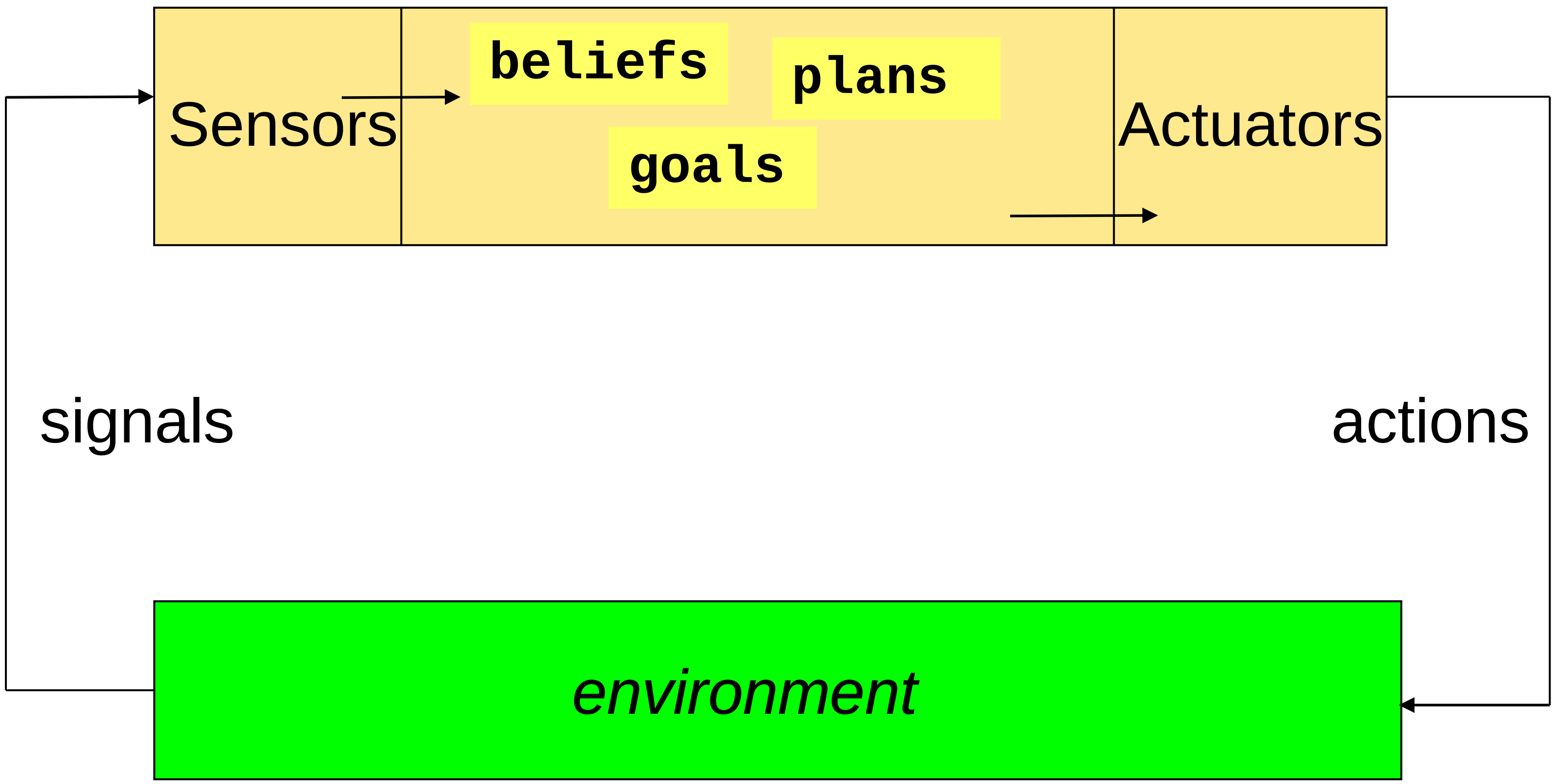
Sensor-Actor Coupling

- Simple design
- Immediate reaction



Deliberative Agents

- Complex design
- Long term planning



Robots

Robota = work (Czech, Karel Capek 1921)

- Artificial humans
- Manufacturing automata
- Mobile robots
- Science Fiction

Applications

- Industry (Mining, Architecture, ...)
- Agriculture
- Service (Transportation, Security, Cleaning, ...)
- Medicine
- Entertainment
- Military
- ...

Environments

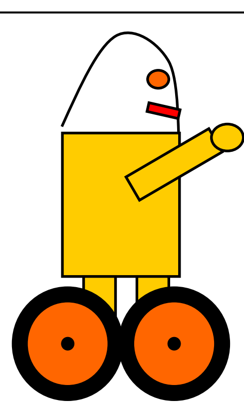
Indoor

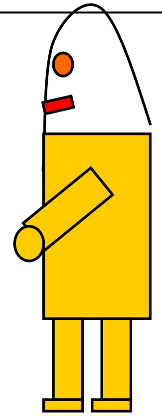
Earth (surface, subsurface)

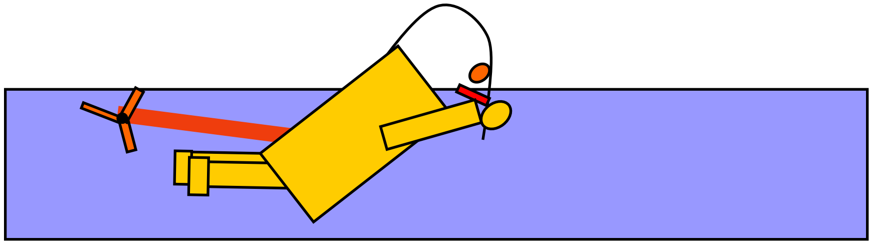
Water (surface, submarine)

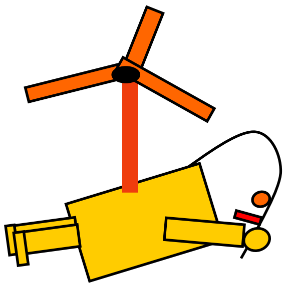
Air

Space









Special interest for Applications in

- Dangerous environments
- Non-accessible environments

Hardware

Sensors

Effectors/Actuators

Drives

Energy

Materials

Design

Processors

Communication

…

Software

Perception
Representations
Behaviors
Planning
Communication
Coordination
Adaptation
Learning
…

Outline

Introduction

Simple Example

RoboCup

RoboCup: 3D-Simulation League

Locomotion

Acting in SimSpark/RoboCup

Keyframe Motions

Perception

Perceptors in SimSpark

A Simple Example from RoboCup

1. search for the ball
2. approach to ball
3. kick the ball

Agent_SimpleSoccer in Simulation

Idea of the program:

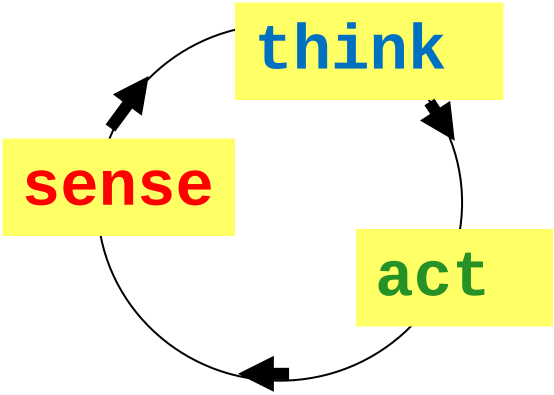
Repeat (whenever a motion is complete):

- If robot has fallen down: Stand up
- If position of ball is not known:
    - Search for ball by turning head (and body)
- else if if ball is far away: turn to ball, walk to ball
- else if ball not between player and goal: turn around ball
- else walk forward („dribbling“)

The implementation is very simple – what happens?
What could be improved?

Implementation of sense-think-act

```
public void run(){   ....
  for (int i = 0; i < totalServerCycles; i++)
  {   sense();  think();   act();
  }
}
private void sense() {
   percIn.update(); localView.update();
}
private void think(){
   soccerThinking.decide();
}
private void act(){
  kfMotion.executeKeyframeSequence(); lookAround.look(); effOut.sendAgentMessage();
}
```



Example
class Agent_SimpleSoccer
from program
agentSimpleSoccer

Implementation of think

```
public void decide() {
    if (motion.ready()) {
   // if the robot has fallen down …
   // if the robot has the actual ball coordinates …
          // if the ball is not in front of the robot …
          // if the robot is far away from the ball …
          // if the robot has the actual goal coordinates …
                // if the ball does not lie between the robot and the goal …
                // if the robot is in a good dribbling position …
         // if the robot cannot sense the goal coordinates from its actual position …
    // if the robot cannot sense the ball coordinates from its actual position …
```

Example
class SoccerThinking
from program
agentSimpleSoccer

Competition

Task:

Become the Soccer Champion of the Fast Scoring Competition!

The task is to score as soon as possible (as described below).

The example agentSimpleSoccer pushes the ball towards the goal. During 10 minutes it almost reaches the goal with the ball. You can use this program as an inspiration for your task.

You can modify and extend it with new motions, better perception and more intelligent behavior. You can even program a team of up to 4 robots which cooperatively perform the task.

Workshop Competition

Become the Soccer Champion of the Fast Scoring Competition!

Task continued:

- You have only one trial to score into the opponents goal.
- The Fast Scoring Competition ends if
  - the team has scored or
  - the ball is outside the playground or
  - 3 minutes have elapsed after start.

Final ranking by lowest scoring times followed by lowest distances.

Workshop Competition

Become the Soccer Champion of the Fast Scoring Competition!

- Competition between student groups.
- About 3-4 members per group
- Groups constituted on Thursday, June 19th

Finals will be on Wednesday, June 25th

- Each group gives 3-minutes explanation on trials and achievements.

Outline

Introduction

Simple Example

RoboCup

RoboCup: 3D-Simulation League

Locomotion

Acting in SimSpark/RoboCup

Keyframe Motions

Perception

Perceptors in SimSpark

Robot Soccer as Testbed

Annual world championships and conference

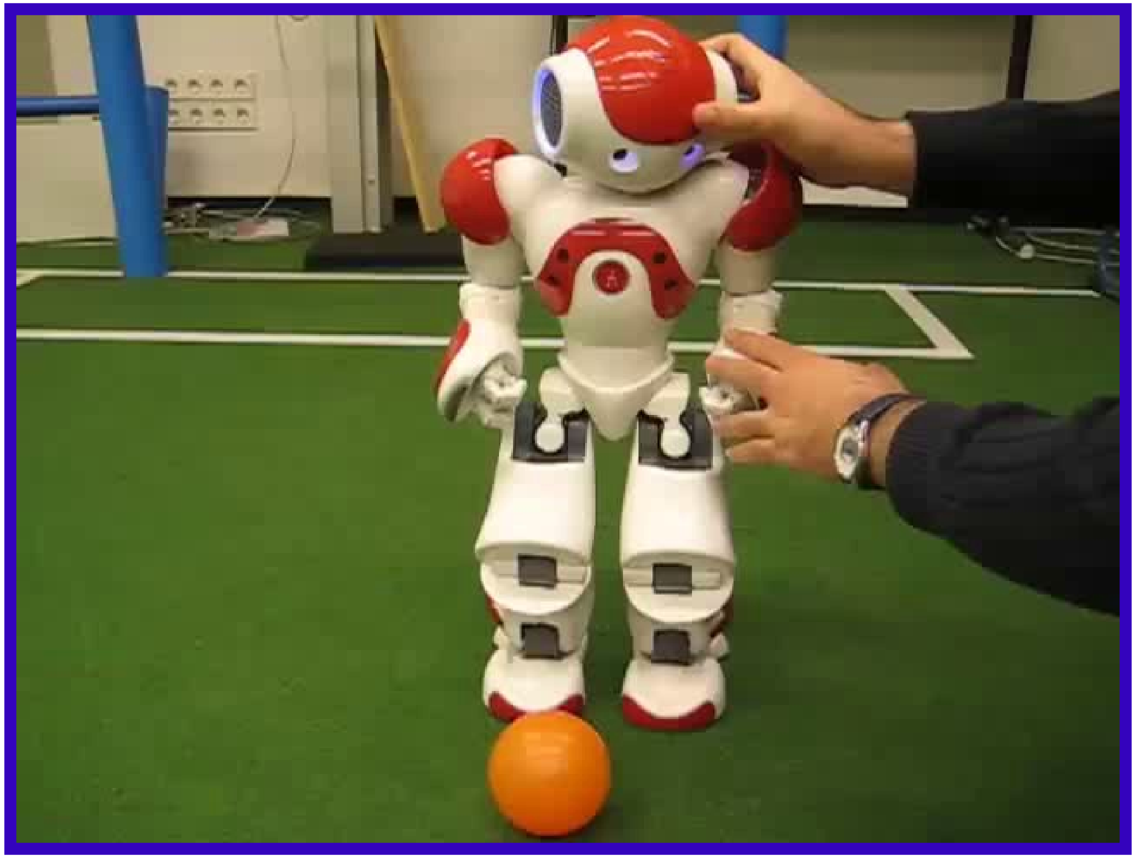
Long term goal: Play like FIFA champion in 2050

RoboCup

Different leagues with different real or simulated robots
for different challenges, e.g. human walking, coordinated play

„Standard Platform“: Robot Nao

Produced by the French Company Aldebaran



Real and Simulated Nao Robots

- Standard Platform League
  with NAO from Aldebaran

- **3D Simulation League** with simulated NAO robots
- Webots Simulation from Swiss Company Cyberbotics
- Simulation in our development tool Robot Control

Outline

Introduction

Simple Example

RoboCup

RoboCup: 3D-Simulation League

Locomotion

Acting in SimSpark/RoboCup

Keyframe Motions

Perception

Perceptors in SimSpark

Simulation

Communication
via protocols (TCP)

<u>Effector messages</u>

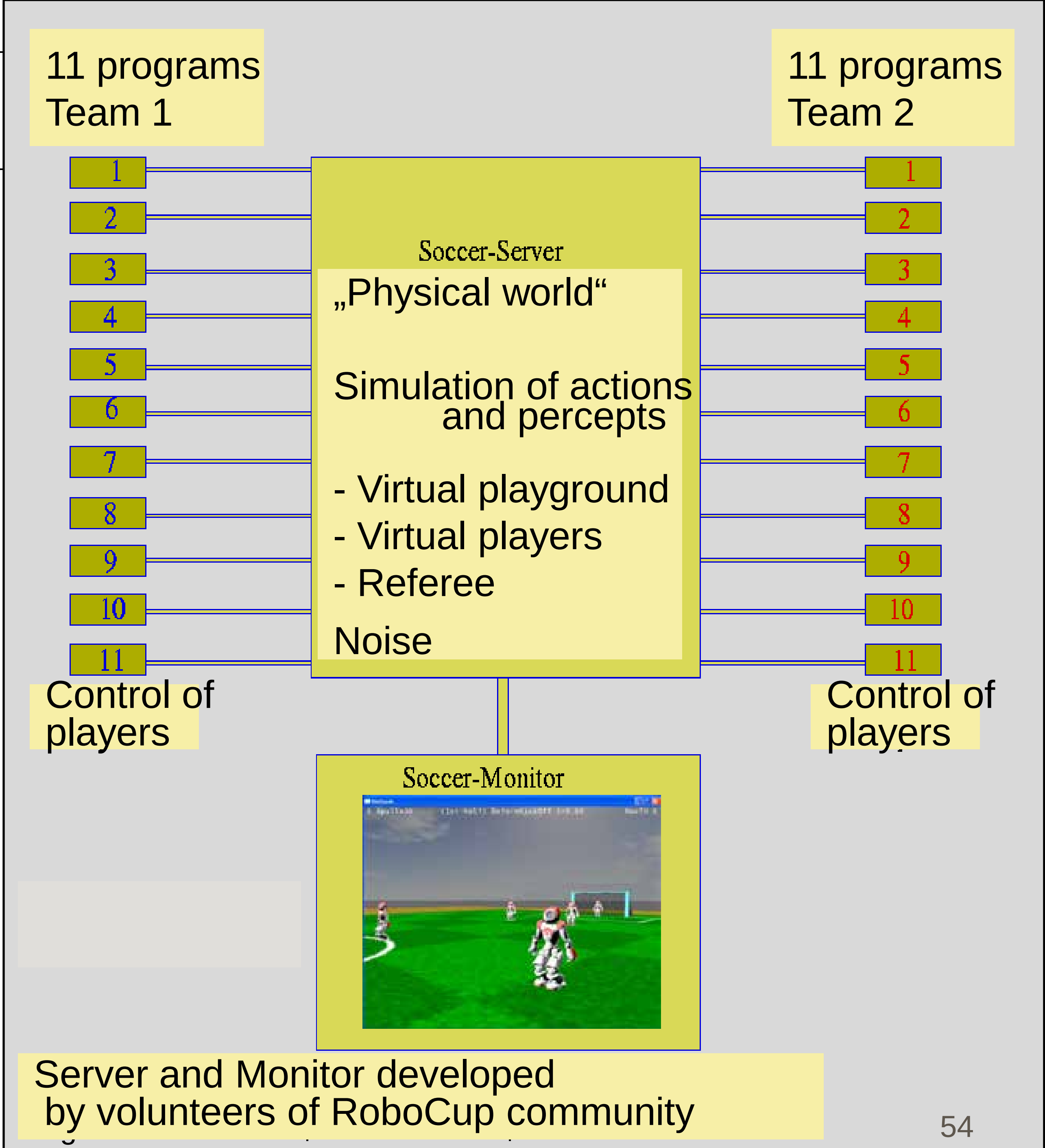
Motor commands

similar to real robot

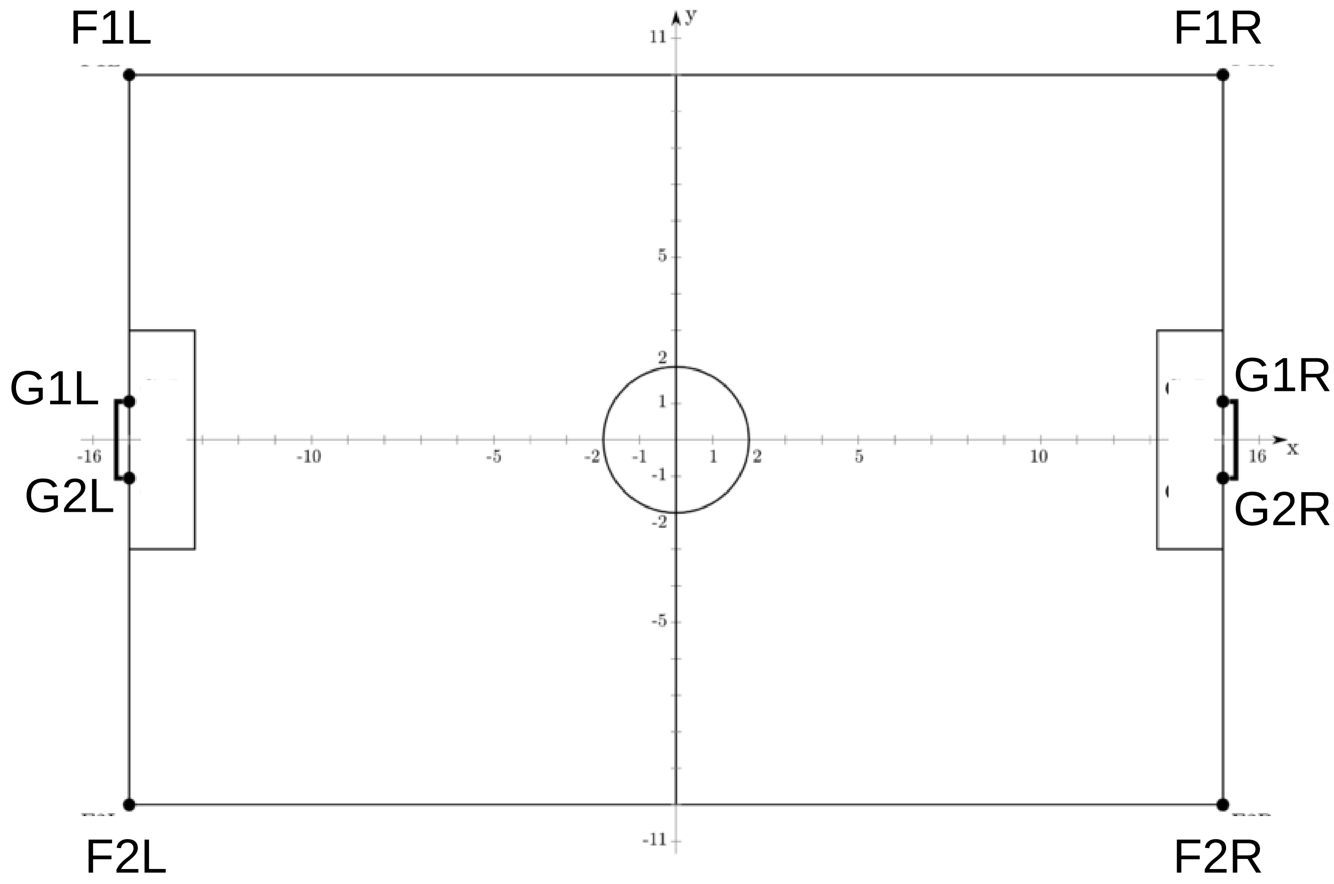
<u>Perceptor messages</u>

Vision, acoustic, inertial,

….



Playground of 3D Simulation League



Actual sizes in our distribution are 10x7 m

Components of Simulated Soccer

Environment:

Simulation of real soccer world

- field and ball
- bodies of players

regarding physical laws (using ODE)

and soccer rules (partially implemented "referee")

Common for all teams

Agents:

Simulation of player control ("brain")

Individual teams

Open Software

You can make your own experiences by using open software from RoboCup community (explore the internet):

- 3D-Simulation League:
  SimSpark (Server + Monitor)

Thanks to
RoboCup Community

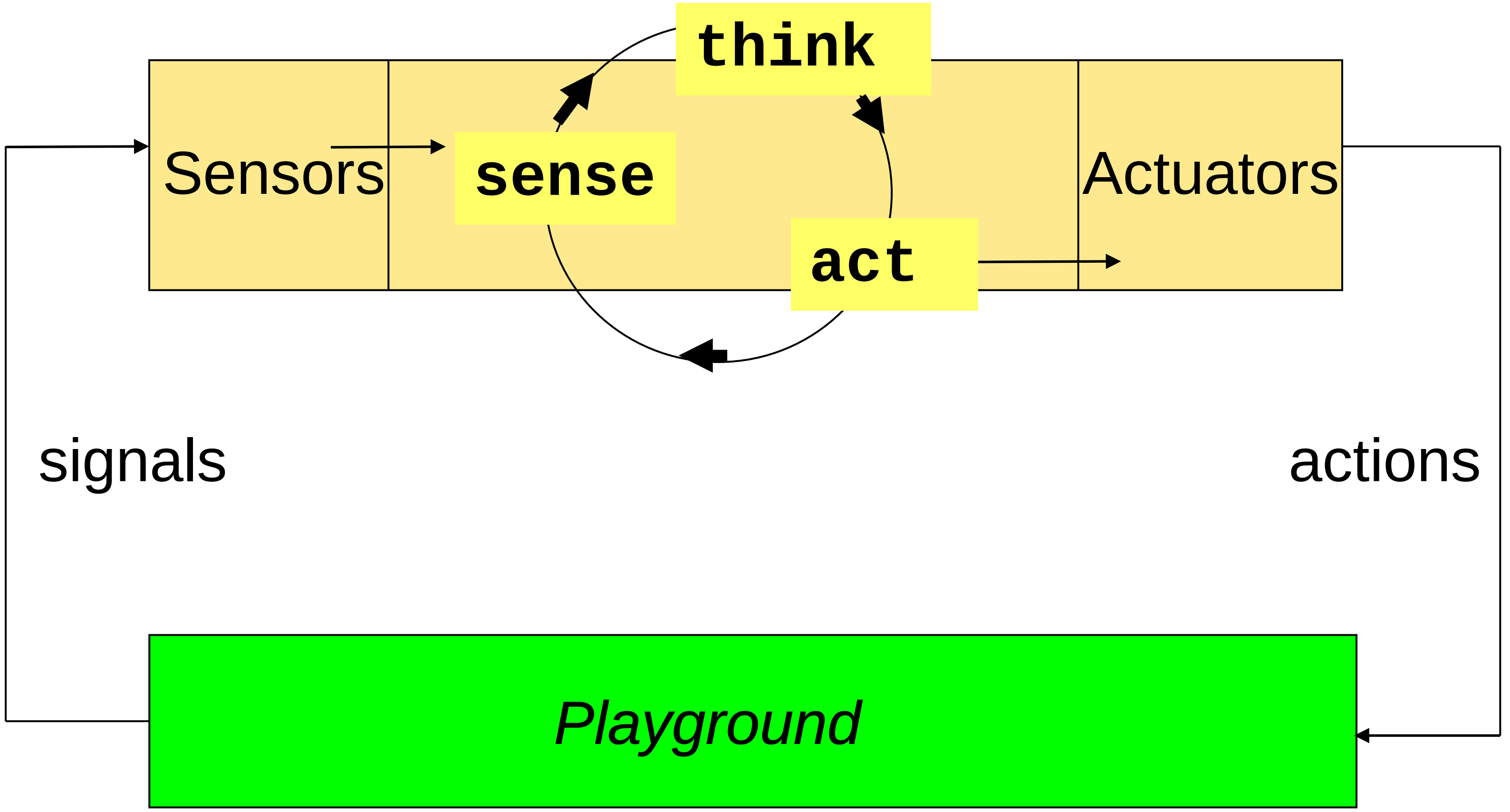
http://simspark.sourceforge.net/wiki

- RoboNewbie Agents of NaoTeam Humboldt

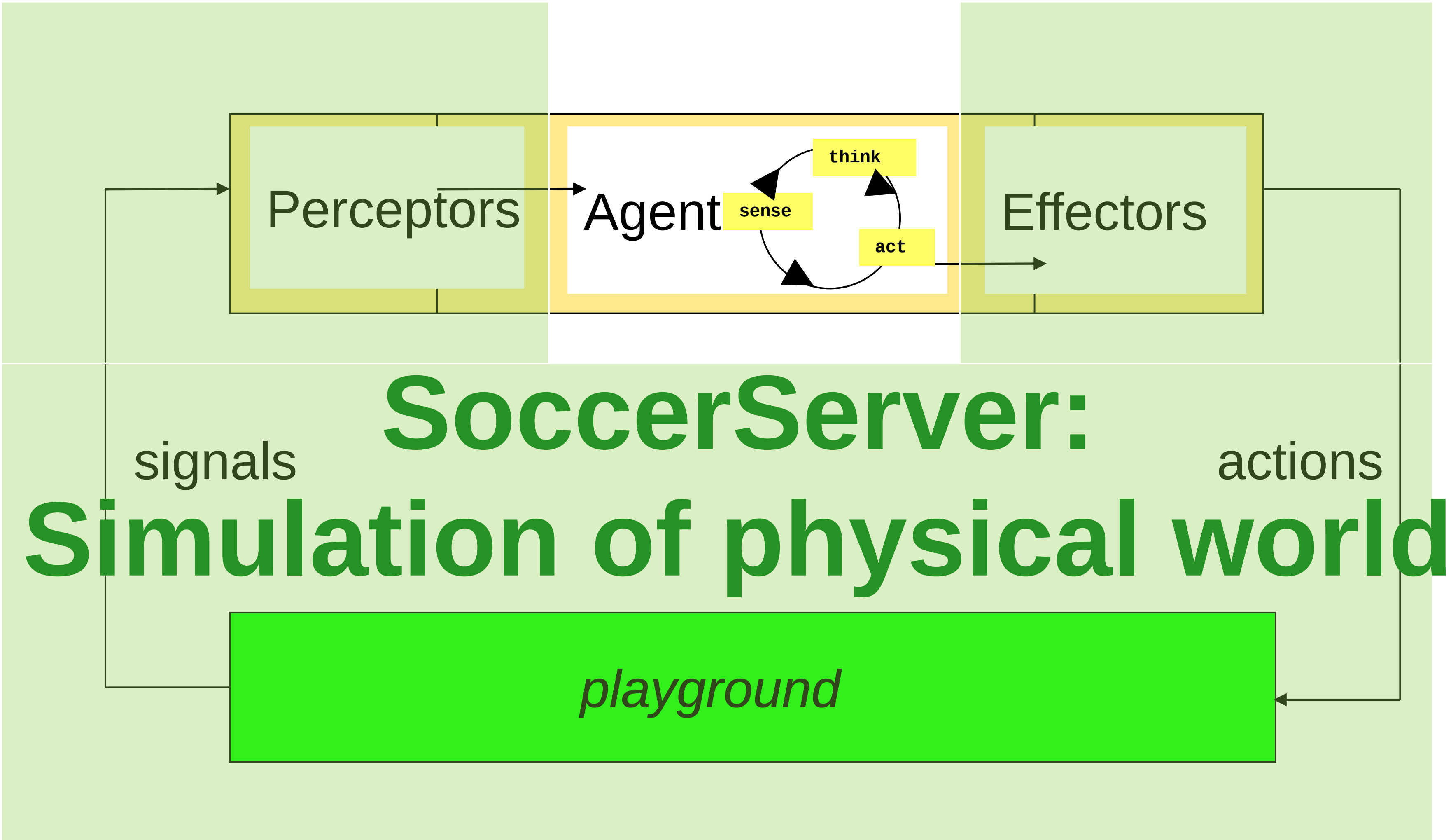
All resources are placed on
our web page (NaoTeam Humboldt)

Thanks to
NaoTeam Humboldt
Magma Offenburg

Inside the robot: Sense-think-act cycle



Agent in Simulation



Simulation Cycle

Cycles (basically 20 msec) with the following steps:

- server sends individual server message with perceptor values ("sensations") to the agents.
- agents can process perceptor values
- agents can make decisions for next actions
- agent can send agent messages with effector commands
- server collects the effector commands of all agents and calculates resulting new situations

Note that messages are interleaved (next slide)!

Synchronization Server/Agent

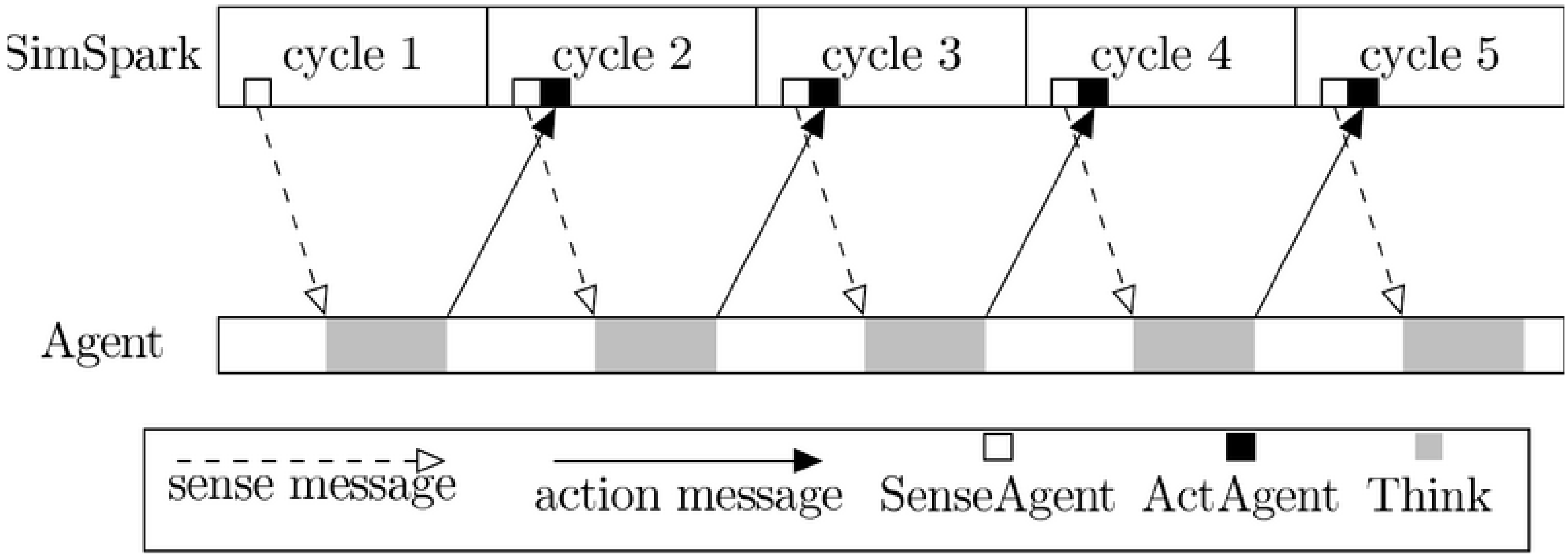


Figure from the SimSpark-Wiki :
http://simspark.sourceforge.net/wiki/i

Synchronization Server/Agent

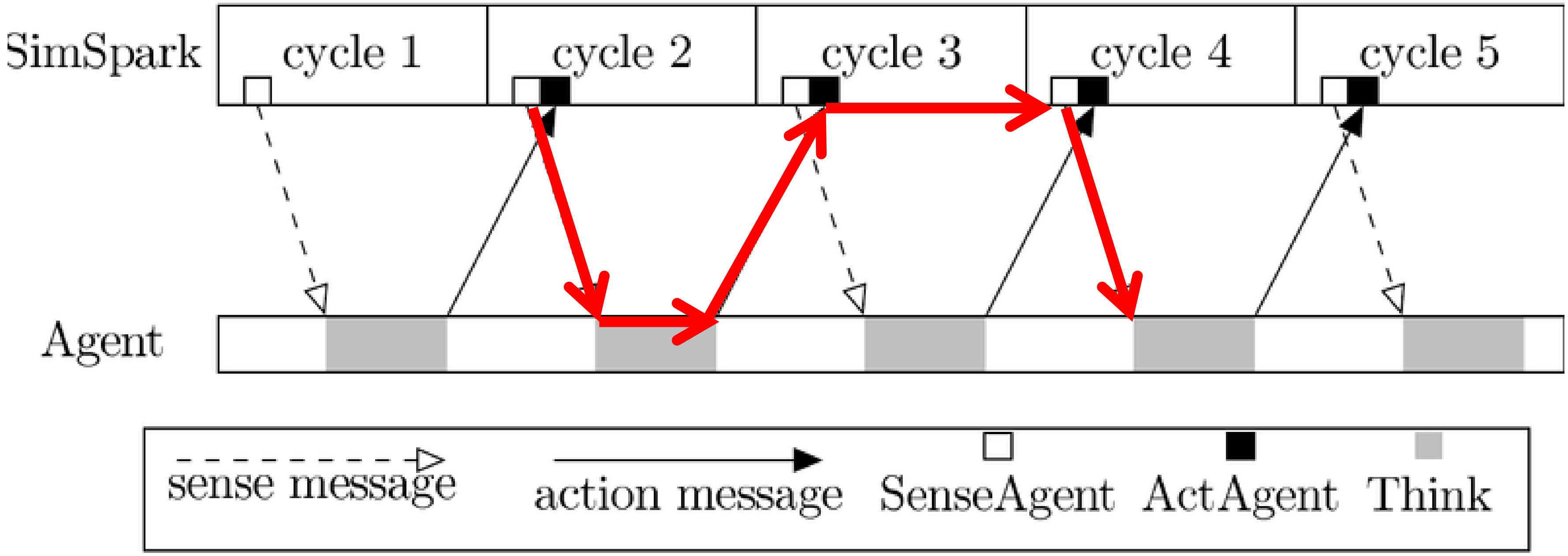


Figure from the SimSpark-Wiki :
http://simspark.sourceforge.net/wiki/i

Outline

Introduction

Simple Example

RoboCup

RoboCup: 3D-Simulation League

Locomotion

Acting in SimSpark/RoboCup

Keyframe Motions

Perception

Perceptors in SimSpark

Locomotion

Unmanned air/surface/underwater vehicles (UAV, USV, UUV):

- Simple design and control (despite obstacles)

Unmanned ground (UGV)

- More complex, depends on the environmental conditions:
  - wheels for (paved) roads
  - tracked vehicles for rough terrain
  - others

Locomotion

Vehicles have simpler actuation than legged robots

Vehicles:
- Accelerate
- Drive
- Turn
- Stop

Legged robots:
- Coordination of limbs
- Complex kinematics
- Stability maintenance (even in stop state)

Special designs

for rolling, snaking, crawling, creeping or jumping

Legged locomotion

Octavio.
Hild, M.: Neurodynamische Module zur Bewegungssteuerung autonomer mobiler Roboter. Dissertation 2007 Humboldt Universität zu Berlin

Now owned by Google

Examples from Boston Dynamics

BigDog

Rhex

RiSE

Boston Dynamics
http://www.bostondynamics.com/

Humanoid shape

- for acting in human environments (buildings, using machines, …)
- for interaction with humans

Outline

Introduction

Simple Example

RoboCup

RoboCup: 3D-Simulation League

Locomotion

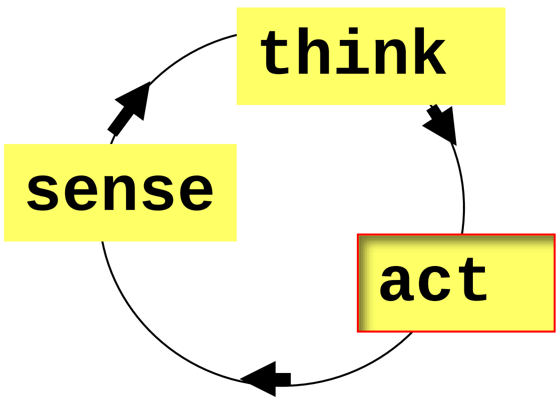
Acting in SimSpark/RoboCup

Keyframe Motions

Perception

Perceptors in SimSpark

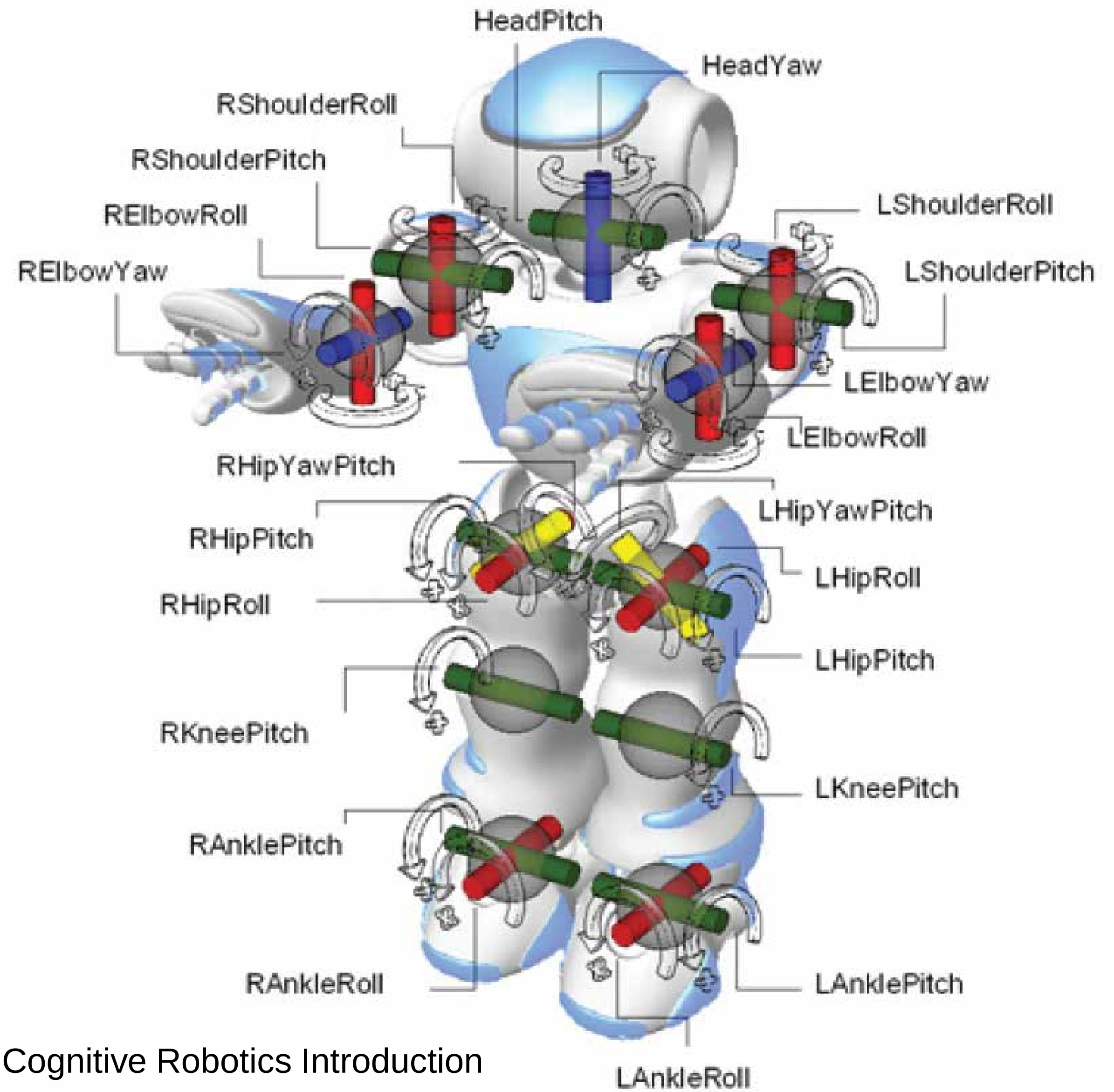
Acting in SimSpark/RoboCup



Effectors for

- Motion
- Speech
- …

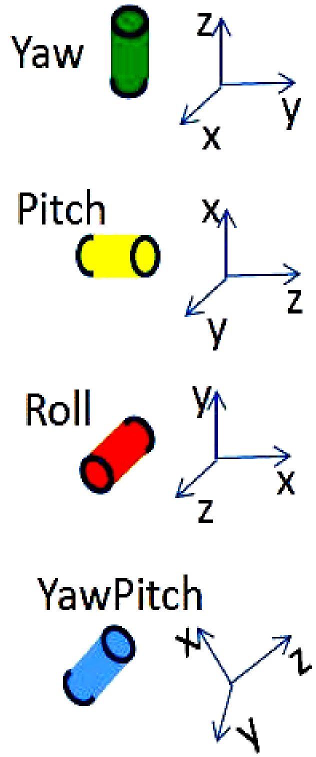
Joints of Nao from Aldebaran

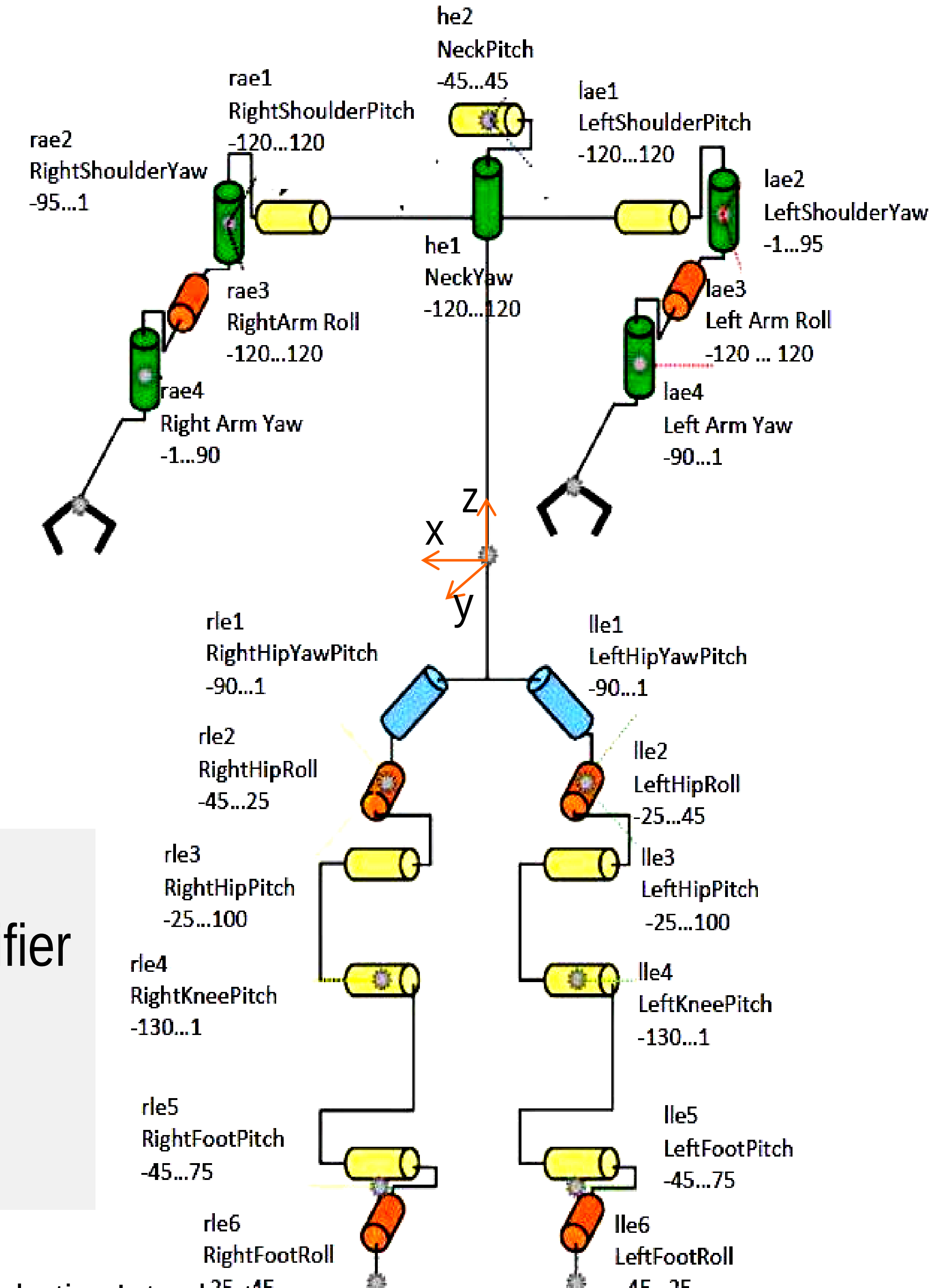


21 Servo-Motors:

- 2 head
- 4 per arm
- 5 per leg
- 1 hip

Nao in SimSpark Simulation





Joints revolve around the roles.
Abbreviations like `rae2` are identifier in effector messages.
Ranges of angles are given below the names of the joints.

Effector messages for Hinge Joints

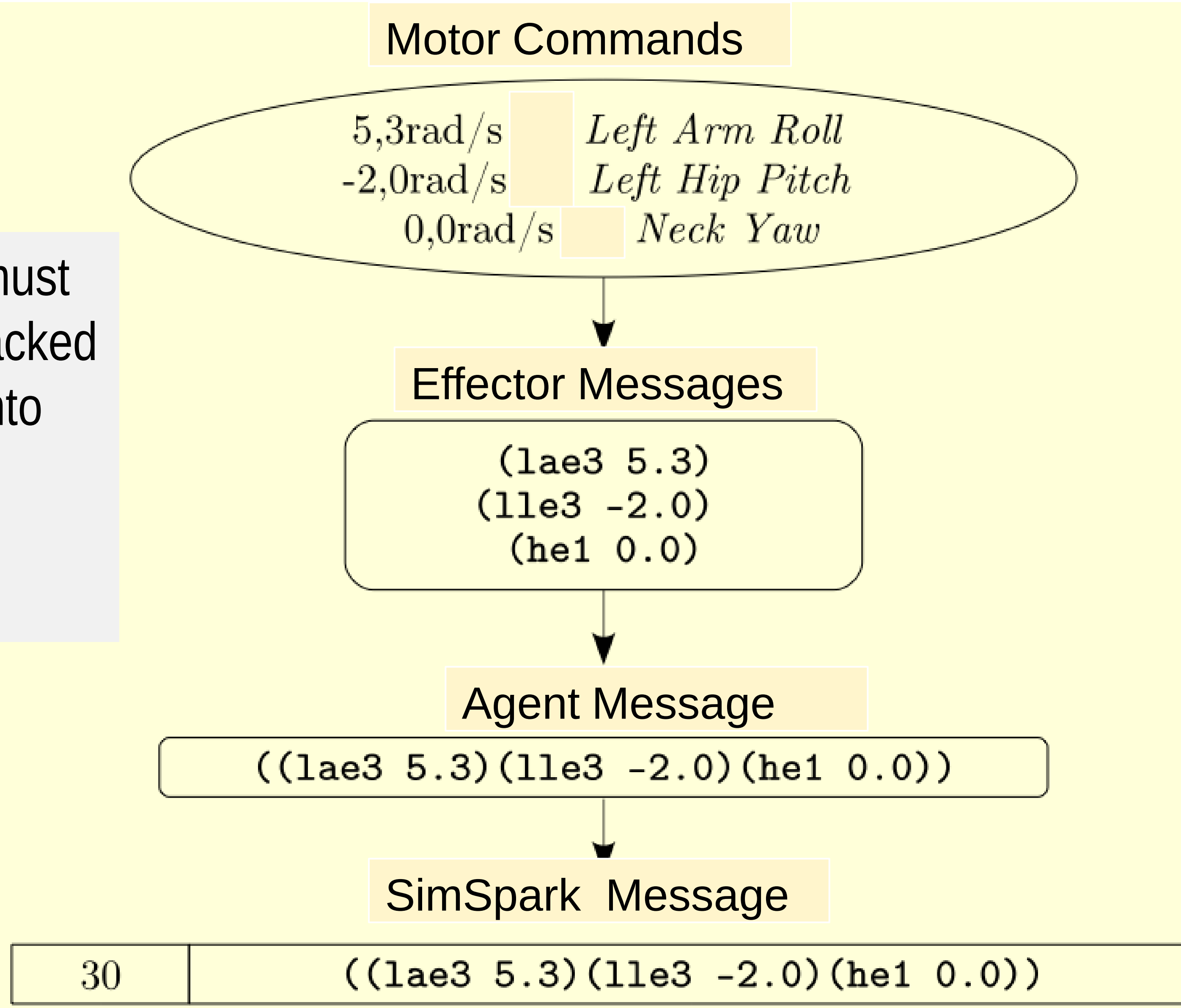
Format: (<joint> <speed>) , e.g. (rae2 2.3).

speed:

- angular speed in radians per second, range -p ... +p
- it is continuously (!) maintained until a new value is set (even if the joint meets its extremity)
- speed=0: no movement, joint holds its position.
- robot model has great stiffness, hence effects of other forces (e.g. gravity) have minor influence.

Effector messages for Hinge Joints

Motor commands must be collected and packed as S-expressions into an message.
Then they are sent to the simulator.



Effector messages for Hinge Joints

Motor Commands

5,3rad/s	*Left Arm Roll*
-2,0rad/s	*Left Hip Pitch*
0,0rad/s	*Neck Yaw*

Motor commands must
be
as
an
The
to t

effOut.setJointCommand(RobotConsts.LeftArmRoll, 5,3);
effOut.setJointCommand(RobotConsts.RightShoulderPitch, -2.0);
effOut.setJointCommand(RobotConsts.NeckYaw, 0.0);

RoboNewbie provides setter methods for each joint.

Users can address motors just like for real robots and need not to care about messages.

SimSpark Message

30	((lae3 5.3)(lle3 -2.0)(he1 0.0))

Programming Motor Commands

Every cycle (20 msec) new messages can be sent to 22 joints, i.e. 1100 messages have to be determined per second.

Different methods for efficient calculations, e.g.

- Keyframe motions
- Sensor controlled motions
- Model based motions
- Biological principles
- ….

Outline

Introduction

Simple Example

RoboCup

RoboCup: 3D-Simulation League

Locomotion

Acting in SimSpark/RoboCup

Keyframe Motions

Perception

Perceptors in SimSpark

Example: Keyframe Motions

Keyframes:

Characteristic poses during a motion (“like in a comic”).

Originally used in animated movies.

Transition times define speed to reach next pose.

Poses between keyframes are interpolated automatically.

(in our programs by package keyframeMotion)

Keyframe

Complete set of joint angles to be set in given time

Time 1000

HeadPitch HeadYaw 0

RShoulderPitch LShoulderPitch 120

RShoulder RollLShoulderRoll 0

RElbowRoll 90

LElbowRoll -90

RElbowYaw 90

LElbowYaw -90

RHipYawPitch LHipYawPitch 0

RHipPitch LHipPitch -31

RHipRoll LHipRoll 0

RKneePitch LKneePitch 63

RAnklePitch LAnklePitch -31

….

Motion Skill: Set of Keyframes

FILE walk_forward-flemming-nika.txt
in …/keyframes

300 0 -21 -62 32 -69 -59 0 -8

300 -5 -21 -62 46 -69 -59 0 0

300 0 -21 -62 60 -69 -59 0 8 -10 -0 12 -11 0 8 12 -0 -3 -11 -110 -32 69 59

300 0 -21 -75 60 -69 -59 0 8 6 -36 27 -11 0 8 12 -15 7 -11 -97 -32 69 59

300 0 -21 -86 60 -69 -59 0 8 42 -69 13 -11 0 8 12 -30 23 -11 -86 -32 69 59

300 0 -21 -110 60 -69 -59 0 8 12 -0 -9 -11 0 8 -10 -0 12 -14 -62 -32 69 59

300 -5 -21 -110 46 -69 -59 0 0 18 -0 -9 -4 0 0 -10 -0 17 -5 -62 -46 69 59

300 0 -21 -110 32 -69 -59 0 -8 12 -0 -3 11 0 -8 -10 -0 12 11 -62 -60 69 59

300 0 -21 -97 32 -69 -59 0 -8 12 -15 7 11 0 -8 6 -36 27 11 -75 -60 69 59

300 0 -21 -84 32 -69 -59 0 -8 12 -30 23 11 0 -8 42 -69 13 11 -84 -60 69 59

Each line starts with the transition time followed by the target angles of joints in a predefined order.

Keyframe sequences are "played" by class keyframeMotion.

Order of Joints in our Keyframes

NeckYaw = 0
NeckPitch = 1
LeftShoulderPitch =2
LeftShoulderYaw = 3
LeftArmRoll = 4
LeftArmYaw = 5
LeftHipYawPitch = 6
LeftHipRoll = 7
LeftHipPitch = 8
LeftKneePitch = 9
LeftFootPitch = 10
LeftFootRoll = 11
RightHipYawPitch = 12
RightHipRoll = 13
RightHipPitch = 14
RightKneePitch = 15
RightFootPitch = 16
RightFootRoll = 17
RightShoulderPitch = 18
RightShoulderYaw = 19
RightArmRoll = 20
RightArmYaw = 21

Development of Keyframe Motions

You can change the .txt-files of existing motions

in directory keyframes.

The new motion will then be used by the program.

You can develop new motions.

- Develop the new motion using MotionEditor for creation and keyframeDeveloper for test.
- Change the program KeyframeMotion as explained there.
- Use the new motion in your program.

  (as e.g. in Agent_SimpleWalkToBall)

Outline

Introduction

Simple Example

RoboCup

RoboCup: 3D-Simulation League

Locomotion

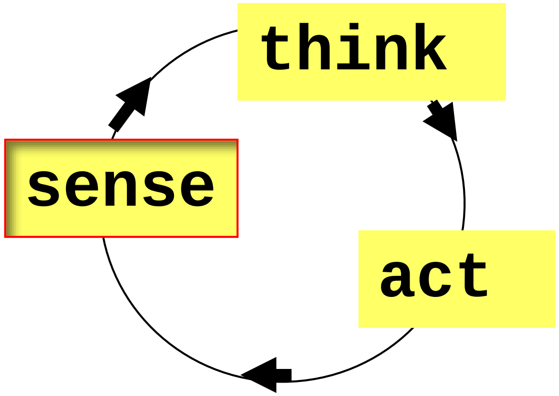
Acting in SimSpark/RoboCup

Keyframe Motions

Perception

Perceptors in SimSpark

Perception



Perceptors for

- Vision
- Speech
- Acceleration
- ...

Sensors in Robotics

Other sensor types in technique than in nature.

by Pollinator, by Anthere (Wikimedia Commons)

But:

Natural systems use more sensors than today robots.

Natural sensors are often more robust.

Technical systems have problems with data interpretation.

Redundancies

Information is usully noisy and incomplete

Much (redundant) information is available

- Vision data
- Audio data
- …
- Previous data of vision, audio,…
- World knowledge

But it may need extreme efforts to exploit it.

Sensors of Nao (Academic Version 2010)

4 Microphones
2 CMOS digital cameras
32 Hall effect sensors (joints)
2 axis gyro
3 axis accelerometer
2 Bumpers (feet)
2 channel sonar
2 Infrared
Tactile Sensor (touch sensor)
8 FRS (force sensors, feet)

Outline

Introduction

Simple Example

RoboCup

RoboCup: 3D-Simulation League

Locomotion

Acting in SimSpark/RoboCup

Keyframe Motions

Perception

Perceptors in SimSpark

Sensors in SimSpark

SimSpark provides preprocessed information,
so called „percepts“,
which are received by „perceptor messages“.

The RoboNewbie agents have
comfortable access methods for sensor values.

Example of Perceptor Message

(time (now 104.87))(GS (t 0.00) (pm BeforeKickOff))(GYR (n torso) (rt 0.24 -0.05 0.02))(ACC
(n torso) (a -0.01 0.05 9.80))(HJ (n hj1) (ax -0.00))(HJ (n hj2) (ax -0.00))(See (G2R (pol 20.11
-18.92 0.84)) (G1R (pol 19.53 -13.04 0.90)) (F1R (pol 19.08 4.58 -1.54)) (F2R (pol 22.73 -
33.49 -1.47)) (B (pol 10.12 -33.09 -2.94)) (L (pol 15.13 -55.78 -2.03) (pol 8.67 10.24 -3.34)) (L
(pol 22.78 -33.20 -1.23) (pol 19.05 4.32 -1.76)) (L (pol 19.08 4.57 -1.55) (pol 1.81 60.14 -
17.11)) (L (pol 22.77 -33.23 -1.26) (pol 14.49 -59.60 -1.79)) (L (pol 17.56 -11.77 -1.83) (pol
18.76 -23.38 -1.60)) (L (pol 17.58 -11.67 -1.74) (pol 19.35 -10.53 -1.53)) (L (pol 18.71 -23.82 -
1.97) (pol 20.43 -21.36 -1.45)) (L (pol 11.68 -28.23 -2.73) (pol 10.93 -23.90 -2.69)) (L (pol
10.91 -24.22 -2.95) (pol 9.84 -22.59 -3.02)) (L (pol 9.84 -22.64 -3.06) (pol 8.81 -25.74 -3.68))
(L (pol 8.83 -25.33 -3.34) (pol 8.35 -32.24 -3.68)) (L (pol 8.35 -32.20 -3.64) (pol 8.69 -39.32 -
3.48)) (L (pol 8.68 -39.59 -3.71) (pol 9.63 -43.18 -3.37)) (L (pol 9.65 -42.85 -3.10) (pol 10.75 -
42.17 -2.80)) (L (pol 10.75 -42.28 -2.89) (pol 11.61 -38.36 -2.50)) (L (pol 11.62 -38.15 -2.33)
(pol 11.94 -33.38 -2.58)) (L (pol 11.94 -33.31 -2.52) (pol 11.70 -28.03 -2.56)))(HJ (n raj1) (ax -
0.00))(HJ (n raj2) (ax 0.00))(HJ (n raj3) (ax 0.00))(HJ (n raj4) (ax 0.00))(HJ (n laj1) (ax -
0.01))(HJ (n laj2) (ax 0.00))(HJ (n laj3) (ax -0.00))(HJ (n laj4) (ax -0.00))(HJ (n rlj1) (ax
0.01))(HJ (n rlj2) (ax 0.00))(HJ (n rlj3) (ax 0.01))(HJ (n rlj4) (ax -0.00))(HJ (n rlj5) (ax
0.00))(FRP (n rf) (c -0.02 -0.00 -0.02) (f -0.02 -0.17 22.52))(HJ (n rlj6) (ax -0.00))(HJ (n llj1)
(ax -0.01))(HJ (n llj2) (ax 0.01))(HJ (n llj3) (ax 0.00))(HJ (n llj4) (ax -0.00))(HJ (n llj5) (ax
0.00))(FRP (n lf) (c 0.02 -0.01 -0.01) (f -0.08 -0.20 22.63))(HJ (n llj6) (ax 0.00))

Perceptors of SimSpark Soccer Simulator

- Hinge Joint Perceptors
- Vision Perceptor at the head
- Gyrometer in the torso
- Accelerometer in the torso
- Force Resistance Perceptor at the feets
- Hear Perceptor at the head
- Game State Perceptor

Example of Perceptor Message

(time (now 104.87))(GS (t 0.00) (pm BeforeKickOff))(GYR (n torso) (rt 0.24 -0.05 0.02))(ACC (n torso) (a -0.01 0.05 9.80))(HJ (n hj1) (ax -0.00))(HJ (n hj2) (ax -0.00))(See (G2R (pol 20.11 -18.92 0.84)) (G1R (pol 19.53 -13.04 0.90)) (F1R (pol 19.08 4.58 -1.54)) (F2R (pol 22.73 -33.49 -1.47)) (B (pol 10.12 -33.09 -2.94)) (L (pol 15.13 -55.78 -2.03) (pol 8.67 10.24 -3.34)) (L (pol 22.78 -33.20 -1.23) (pol 19.05 4.32 -1.76)) (L (pol 19.08 4.57 -1.55) (pol 1.81 60.14 -17.11)) (L (pol 22.77 -33.23 -1.26) (pol 14.49 -59.60 -1.79)) (L (pol 17.56 -11.77 -1.83) (pol 18.76 -23.38 -1.60)) (L (pol 17.58 -11.67 -1.74) (pol 19.35 -10.53 -1.53)) (L (pol 18.71 -23.82 -1.97) (pol 20.43 -21.36 -1.45)) (L (pol 11.68 -28.23 -2.73) (pol 10.93 -23.90 -2.69)) (L (pol 10.91 -24.22 -2.95) (pol 9.84 -22.59 -3.02)) (L (pol 9.84 -22.64 -3.06) (pol 8.81 -25.74 -3.68)) (L (pol 8.83 -25.33 -3.34) (pol 8.35 -32.24 -3.68)) (L (pol 8.35 -32.20 -3.64) (pol 8.69 -39.32 -3.48)) (L (pol 8.68 -39.59 -3.71) (pol 9.63 -43.18 -3.37)) (L (pol 9.65 -42.85 -3.10) (pol 10.75 -42.17 -2.80)) (L (pol 10.75 -42.28 -2.89) (pol 11.61 -38.36 -2.50)) (L (pol 11.62 -38.15 -2.33) (pol 11.94 -33.38 -2.58)) (L (pol 11.94 -33.31 -2.52) (pol 11.70 -28.03 -2.56)))(HJ (n raj1) (ax -0.00))(HJ (n raj2) (ax 0.00))(HJ (n raj3) (ax 0.00))(HJ (n raj4) (ax 0.00))(HJ (n laj1) (ax -0.01))(HJ (n laj2) (ax 0.00))(HJ (n laj3) (ax -0.00))(HJ (n laj4) (ax -0.00))(HJ (n rlj1) (ax 0.01))(HJ (n rlj2) (ax 0.00))(HJ (n rlj3) (ax 0.01))(HJ (n rlj4) (ax -0.00))(HJ (n rlj5) (ax 0.00))(FRP (n rf) (c -0.02 -0.00 -0.02) (f -0.02 -0.17 22.52))(HJ (n rlj6) (ax -0.00))(HJ (n llj1) (ax -0.01))(HJ (n llj2) (ax 0.01))(HJ (n llj3) (ax 0.00))(HJ (n llj4) (ax -0.00))(HJ (n llj5) (ax 0.00))(FRP (n lf) (c 0.02 -0.01 -0.01) (f -0.08 -0.20 22.63))(HJ (n llj6) (ax 0.00))

Gyrometer and Accelerometer

Accelerometer (acceleration in m/s^2 of torso relative to free fall).

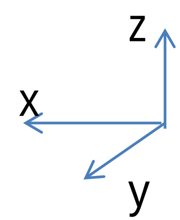
Format:

(ACC (n <name>) (a <x> <y> <z>))

Example:

(ACC (n torso) (a 0.00 0.00 9.81))

Orientation:
y-axis in facing direction



Measurements regard motion in last cycle.

Gyrometer (change rates in degrees/s for orientation of torso)

Format:

(GYR (n torso) (rt <x> <y> <z>))

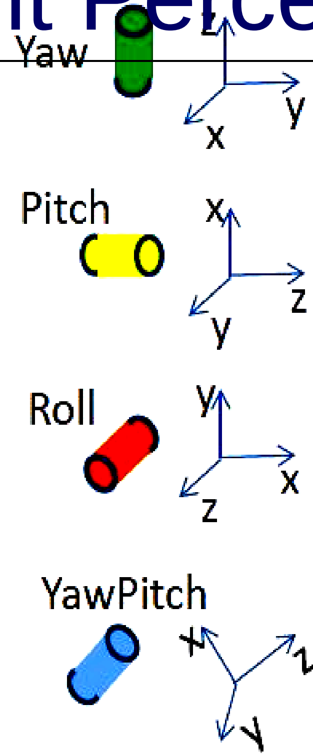
Example:

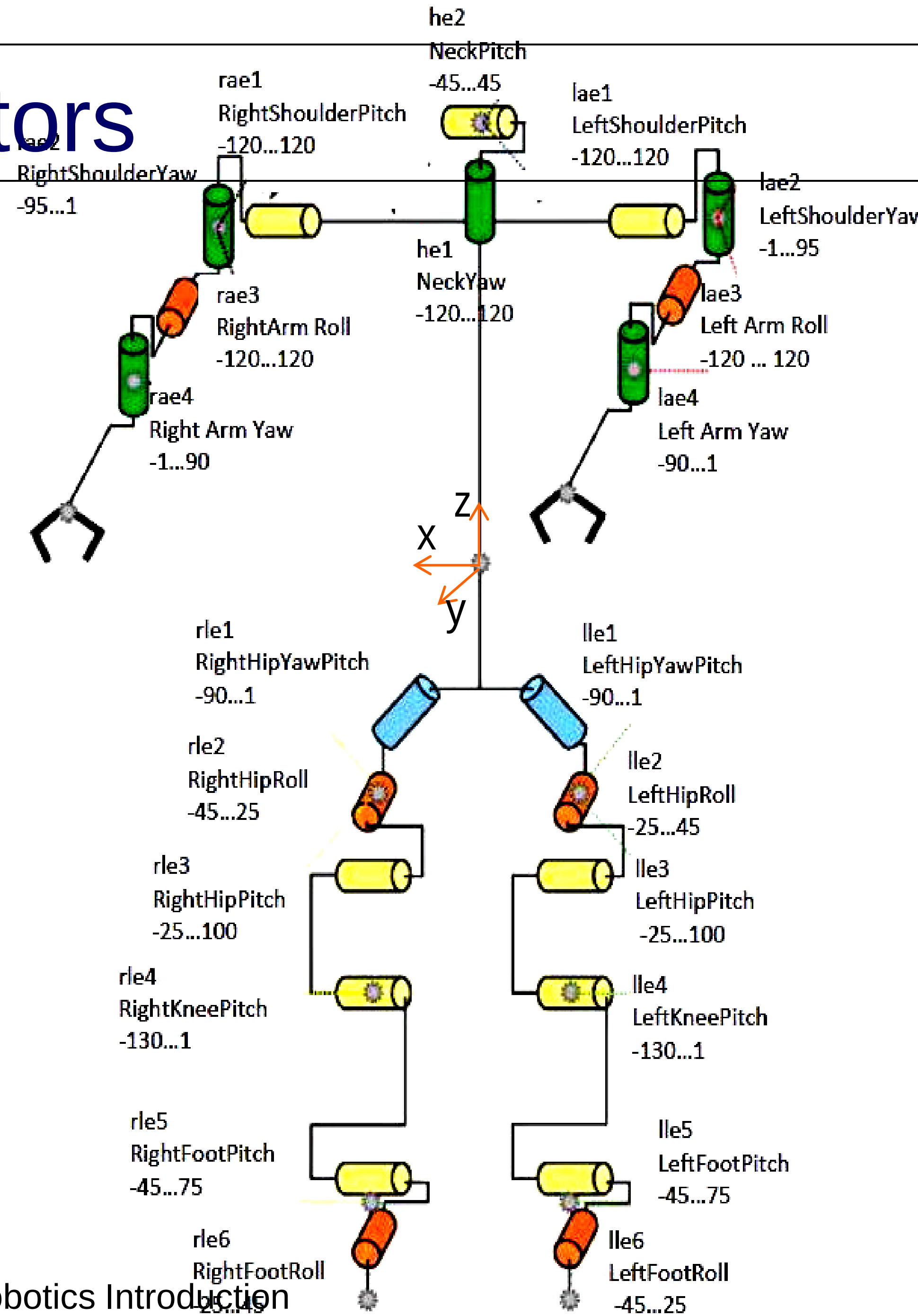
(GYR (n torso) (rt 0.01 0.07 0.46))

Hinge Joint Perceptors

(time (now 104.87))(GS (t 0.00) (pm BeforeKickOff))(GYR (n torso) (rt 0.24 -0.05 0.02))(ACC (n torso) (a -0.01 0.05 9.80))(HJ (n hj1) (ax -0.00))(HJ (n hj2) (ax -0.00))(See (G2R (pol 20.11 -18.92 0.84)) (G1R (pol 19.53 -13.04 0.90)) (F1R (pol 19.08 4.58 -1.54)) (F2R (pol 22.73 -33.49 -1.47)) (B (pol 10.12 -33.09 -2.94)) (L (pol 15.13 -55.78 -2.03) (pol 8.67 10.24 -3.34)) (L (pol 22.78 -33.20 -1.23) (pol 19.05 4.32 -1.76)) (L (pol 19.08 4.57 -1.55) (pol 1.81 60.14 -17.11)) (L (pol 22.77 -33.23 -1.26) (pol 14.49 -59.60 -1.79)) (L (pol 17.56 -11.77 -1.83) (pol 18.76 -23.38 -1.60)) (L (pol 17.58 -11.67 -1.74) (pol 19.35 -10.53 -1.53)) (L (pol 18.71 -23.82 -1.97) (pol 20.43 -21.36 -1.45)) (L (pol 11.68 -28.23 -2.73) (pol 10.93 -23.90 -2.69)) (L (pol 10.91 -24.22 -2.95) (pol 9.84 -22.59 -3.02)) (L (pol 9.84 -22.64 -3.06) (pol 8.81 -25.74 -3.68)) (L (pol 8.83 -25.33 -3.34) (pol 8.35 -32.24 -3.68)) (L (pol 8.35 -32.20 -3.64) (pol 8.69 -39.32 -3.48)) (L (pol 8.68 -39.59 -3.71) (pol 9.63 -43.18 -3.37)) (L (pol 9.65 -42.85 -3.10) (pol 10.75 -42.17 -2.80)) (L (pol 10.75 -42.28 -2.89) (pol 11.61 -38.36 -2.50)) (L (pol 11.62 -38.15 -2.33) (pol 11.94 -33.38 -2.58)) (L (pol 11.94 -33.31 -2.52) (pol 11.70 -28.03 -2.56)))(HJ (n raj1) (ax -0.00))(HJ (n raj2) (ax 0.00))(HJ (n raj3) (ax 0.00))(HJ (n raj4) (ax 0.00))(HJ (n laj1) (ax -0.01))(HJ (n laj2) (ax 0.00))(HJ (n laj3) (ax -0.00))(HJ (n laj4) (ax -0.00))(HJ (n rlj1) (ax 0.01))(HJ (n rlj2) (ax 0.00))(HJ (n rlj3) (ax 0.01))(HJ (n rlj4) (ax -0.00))(HJ (n rlj5) (ax 0.00))(FRP (n rf) (c -0.02 -0.00 -0.02) (f -0.02 -0.17 22.52))(HJ (n rlj6) (ax -0.00))(HJ (n llj1) (ax -0.01))(HJ (n llj2) (ax 0.01))(HJ (n llj3) (ax 0.00))(HJ (n llj4) (ax -0.00))(HJ (n llj5) (ax 0.00))(FRP (n lf) (c 0.02 -0.01 -0.01) (f -0.08 -0.20 22.63))(HJ (n llj6) (ax 0.00))

Hinge Joint Perceptors





Format:

(HJ (n <name>) (ax <ax>))

Example:

(HJ (n laj3) (ax -1.02))

Vision Perceptor

(time (now 104.87))(GS (t 0.00) (pm BeforeKickOff))(GYR (n torso) (rt 0.24 -0.05 0.02))(ACC (n torso) (a -0.01 0.05 9.80))(HJ (n hj1) (ax -0.00))(HJ (n hj2) (ax -0.00))(See (G2R (pol 20.11 -18.92 0.84)) (G1R (pol 19.53 -13.04 0.90)) (F1R (pol 19.08 4.58 -1.54)) (F2R (pol 22.73 -33.49 -1.47)) (B (pol 10.12 -33.09 -2.94)) (L (pol 15.13 -55.78 -2.03) (pol 8.67 10.24 -3.34)) (L (pol 22.78 -33.20 -1.23) (pol 19.05 4.32 -1.76)) (L (pol 19.08 4.57 -1.55) (pol 1.81 60.14 -17.11)) (L (pol 22.77 -33.23 -1.26) (pol 14.49 -59.60 -1.79)) (L (pol 17.56 -11.77 -1.83) (pol 18.76 -23.38 -1.60)) (L (pol 17.58 -11.67 -1.74) (pol 19.35 -10.53 -1.53)) (L (pol 18.71 -23.82 -1.97) (pol 20.43 -21.36 -1.45)) (L (pol 11.68 -28.23 -2.73) (pol 10.93 -23.90 -2.69)) (L (pol 10.91 -24.22 -2.95) (pol 9.84 -22.59 -3.02)) (L (pol 9.84 -22.64 -3.06) (pol 8.81 -25.74 -3.68)) (L (pol 8.83 -25.33 -3.34) (pol 8.35 -32.24 -3.68)) (L (pol 8.35 -32.20 -3.64) (pol 8.69 -39.32 -3.48)) (L (pol 8.68 -39.59 -3.71) (pol 9.63 -43.18 -3.37)) (L (pol 9.65 -42.85 -3.10) (pol 10.75 -42.17 -2.80)) (L (pol 10.75 -42.28 -2.89) (pol 11.61 -38.36 -2.50)) (L (pol 11.62 -38.15 -2.33) (pol 11.94 -33.38 -2.58)) (L (pol 11.94 -33.31 -2.52) (pol 11.70 -28.03 -2.56)))(HJ (n raj1) (ax -0.00))(HJ (n raj2) (ax 0.00))(HJ (n raj3) (ax 0.00))(HJ (n raj4) (ax 0.00))(HJ (n laj1) (ax -0.01))(HJ (n laj2) (ax 0.00))(HJ (n laj3) (ax -0.00))(HJ (n laj4) (ax -0.00))(HJ (n rlj1) (ax 0.01))(HJ (n rlj2) (ax 0.00))(HJ (n rlj3) (ax 0.01))(HJ (n rlj4) (ax -0.00))(HJ (n rlj5) (ax 0.00))(FRP (n rf) (c -0.02 -0.00 -0.02) (f -0.02 -0.17 22.52))(HJ (n rlj6) (ax -0.00))(HJ (n llj1) (ax -0.01))(HJ (n llj2) (ax 0.01))(HJ (n llj3) (ax 0.00))(HJ (n llj4) (ax -0.00))(HJ (n llj5) (ax 0.00))(FRP (n lf) (c 0.02 -0.01 -0.01) (f -0.08 -0.20 22.63))(HJ (n llj6) (ax 0.00))

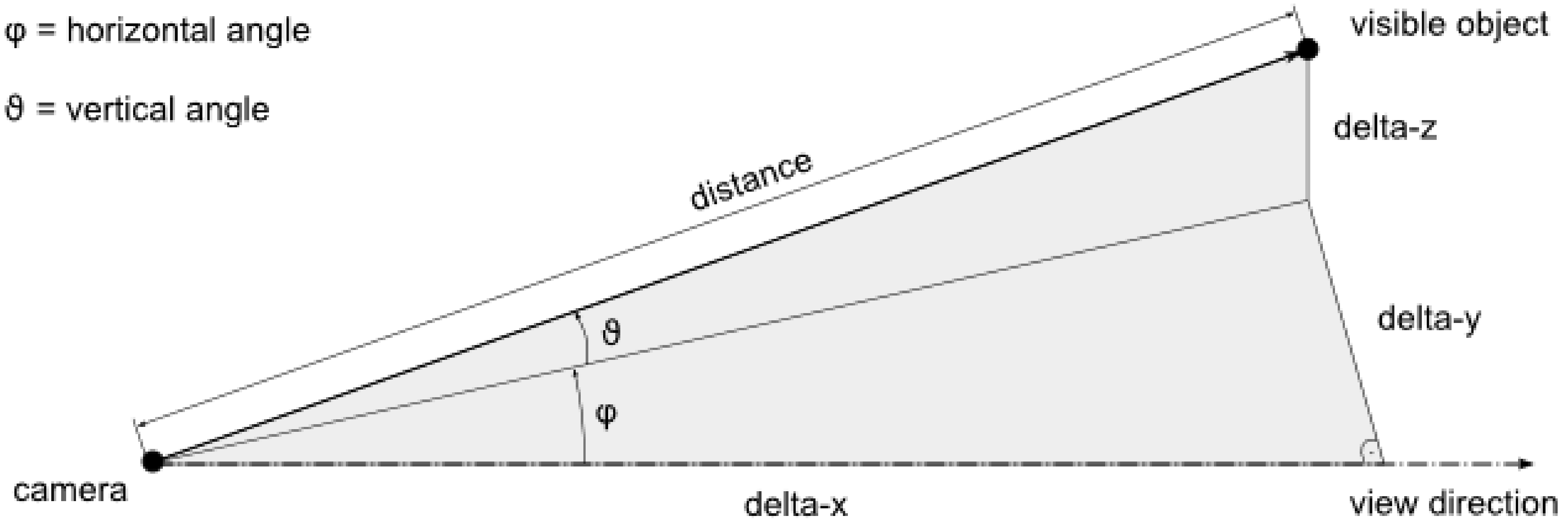
Vision Perceptor

Information comes only each 3rd cycle, i.e. each 60 msec.

No image processing.
Simulator provides correct perceptor values

View angle of camera is
120 degrees horizontally and vertically

Format:
(See
(<name> (pol <distance> <angle1> <angle2>))
(P (team <teamname>) (id <playerID>) (pol <distance> <angle1> <angle2>)))

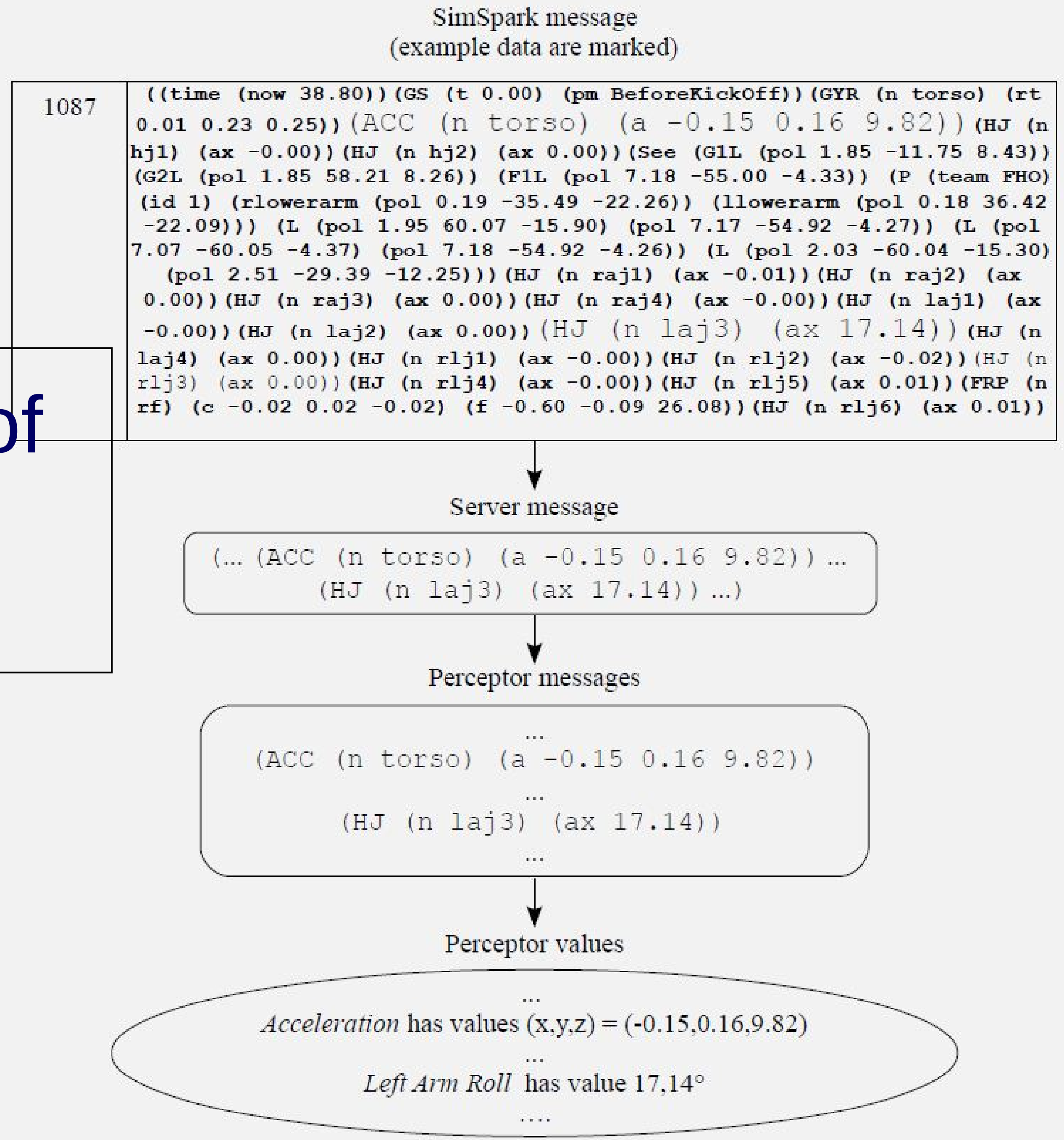


Visual Information SimSpark

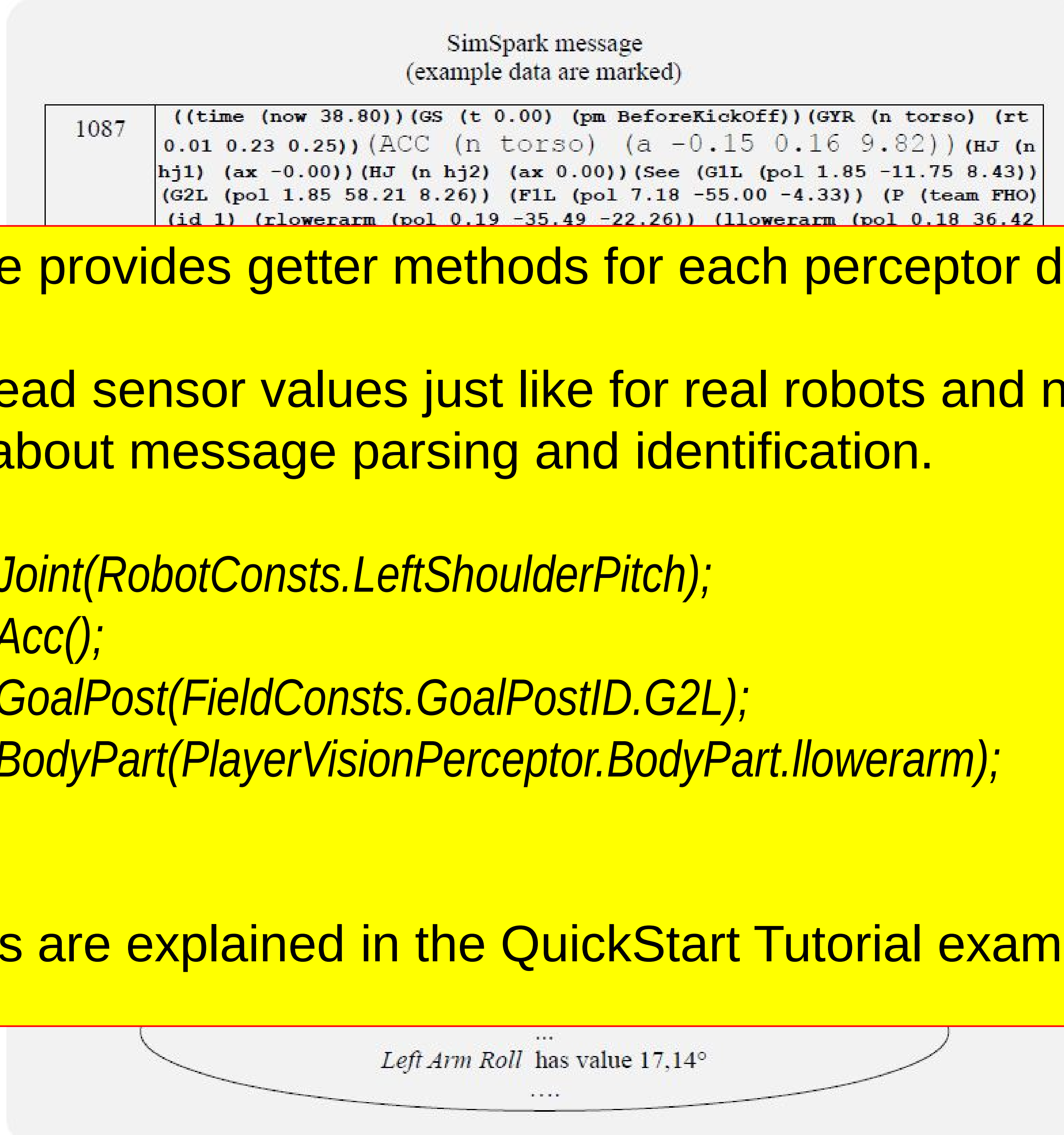
Example:

```
(See (G2R (pol 17.55 -3.33 4.31))
 (G1R (pol 17.52 3.27 4.07))
(F1R (pol 18.52 18.94 1.54))
(F2R (pol 18.52 -18.91 1.52))
 (B (pol 8.51 -0.21 -0.17))
(P (team teamRed) (id 1)  (head (pol 16.98 -0.21 3.19))
       (rlowerarm (pol 16.83 -0.06 2.80))    (llowerarm (pol 16.86 -0.36 3.10))
       (rfoot (pol 17.00 0.29 1.68))          (lfoot (pol 16.95 -0.51 1.32)))
(P (team teamBlue) (id 3)
       (rlowerarm (pol 0.18 -33.55 -20.16))    (llowerarm (pol 0.18 34.29 -19.80))))
 (L (pol 12.11 -40.77 -2.40) (pol 12.95 -37.76 -2.41))
 (L (pol 12.97 -37.56 -2.24) (pol 13.32 -32.98 -2.20))
```

Semantics of SimSpark Messages



Burkhard



RoboNewbie provides getter methods for each perceptor data.

Users can read sensor values just like for real robots and need not to care about message parsing and identification.

percIn.getJoint(RobotConsts.LeftShoulderPitch);
percIn.getAcc();
percIn.getGoalPost(FieldConsts.GoalPostID.G2L);
percIn.getBodyPart(PlayerVisionPerceptor.BodyPart.llowerarm);

Data formats are explained in the QuickStart Tutorial examples.