Towards Using ROS in the RoboCup Humanoid Soccer League

Marc Bestmann

Universität Hamburg
Fakultät für Mathematik, Informatik und Naturwissenschaften
Fachbereich Informatik

Technische Aspekte Multimodaler Systeme

09. Mai 2017

Table of Contents

Table of Contents

Motivation

Achieving exchangeable ROS [6] based software modules in the RoboCup Soccer context can lead to:

- Acceleration of development
- Increased comparability of different approaches
- Easier entry into the league for new teams
- Specialization of teams on certain sub areas
- Increased knowledge transfer with general research

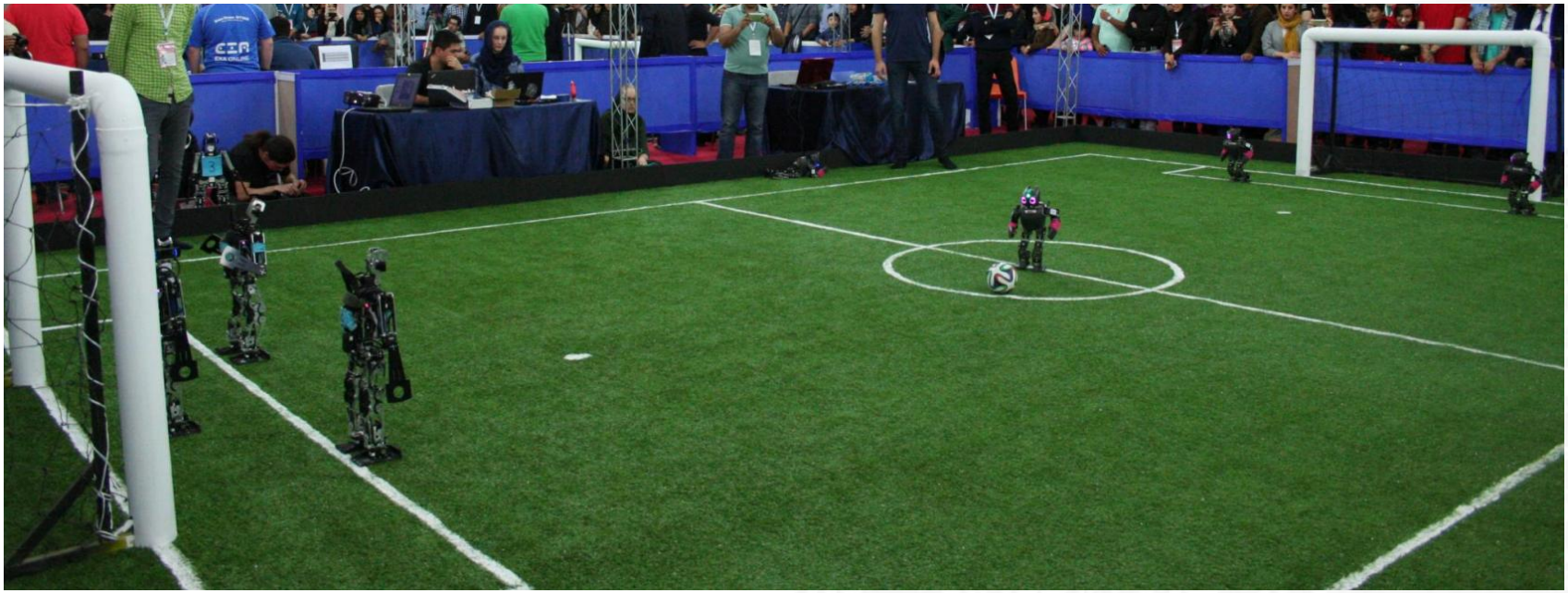
[5]

Table of Contents

RoboCup

- Official Goal
  - Win against the soccer world champion with a team of robots in 2050
- Actual Goals
  - Accelerate research in robotics
  - Test approaches under realistic conditions
  - Make different approaches comparable
  - Promote robotics research to the public
- Organization
  - Yearly world championship and symposium
  - Multiple other competitions, workshops and summer schools
  - Different Leagues for different purposes

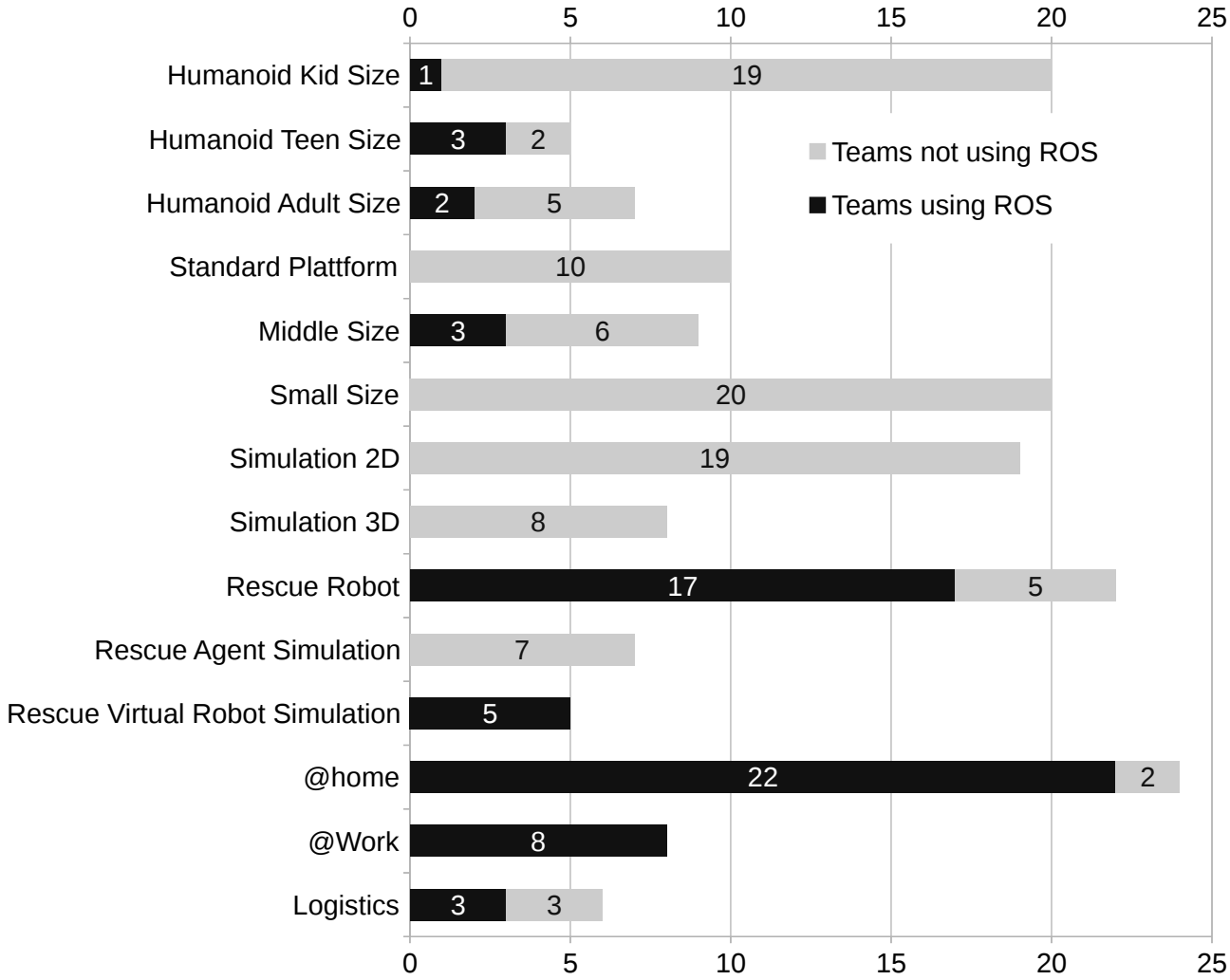
Humanoid Soccer League Field



[2]

Current State of Software Modularization

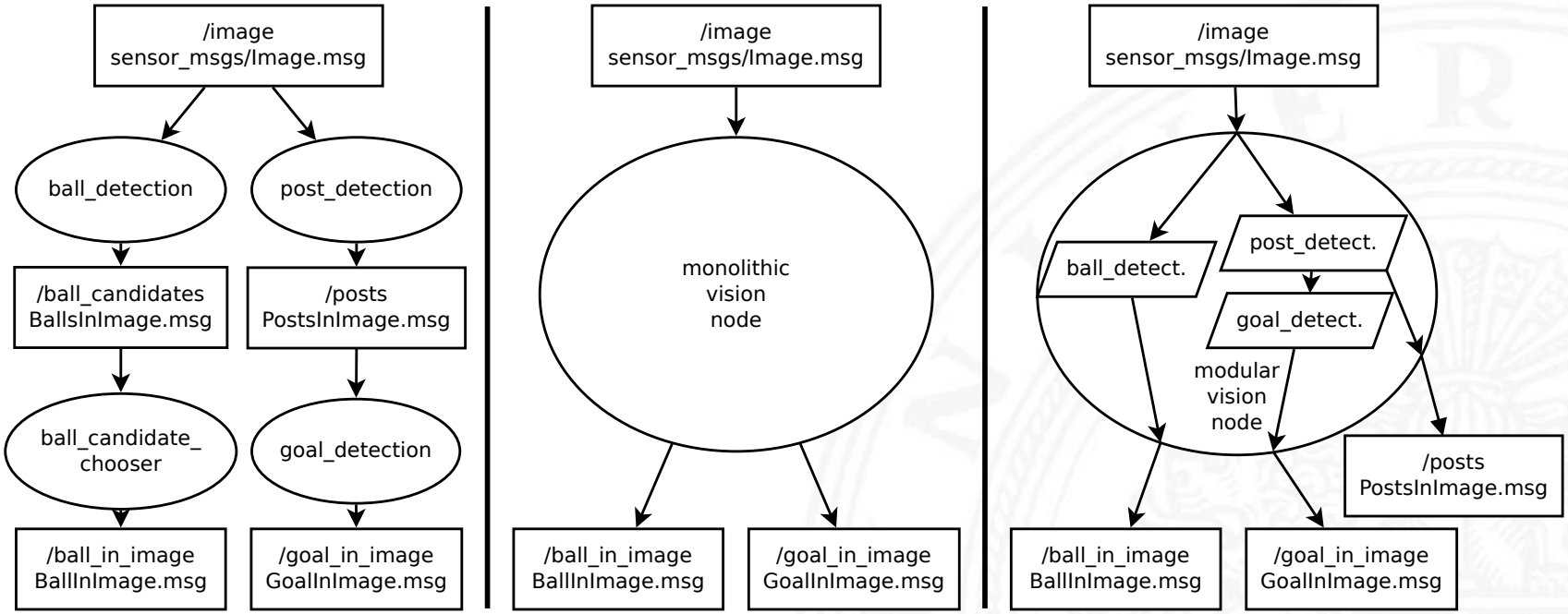
- Humanoid League
  - Almost only self made software frameworks
  - Low modularization
  - First teams starting to use ROS [4]
  - Exchange difficult due to different robot platforms
- Standard Platform League
  - Usage of the B-Human framework [7]
  - Higher level of games
  - ROS problematic on NAO
- Rescue Robot League
  - Initiative to use ROS invoked by team Hector
  - Now high use of ROS
  - Available software modules



Based on the team description papers [3]

Table of Contents

Approach


/image
sensor_msgs/Image.msg
ball_detection
post_detection
/ball_candidates
BallsInImage.msg
/posts
PostsInImage.msg
ball_candidate_
chooser
goal_detection
/ball_in_image
BallInImage.msg
/goal_in_image
GoalInImage.msg
/image
sensor_msgs/Image.msg
monolithic
vision
node
/ball_in_image
BallInImage.msg
/goal_in_image
GoalInImage.msg
/image
sensor_msgs/Image.msg
ball_detect.
post_detect.
goal_detect.
modular
vision
node
/posts
PostsInImage.msg
/ball_in_image
BallInImage.msg
/goal_in_image
GoalInImage.msg

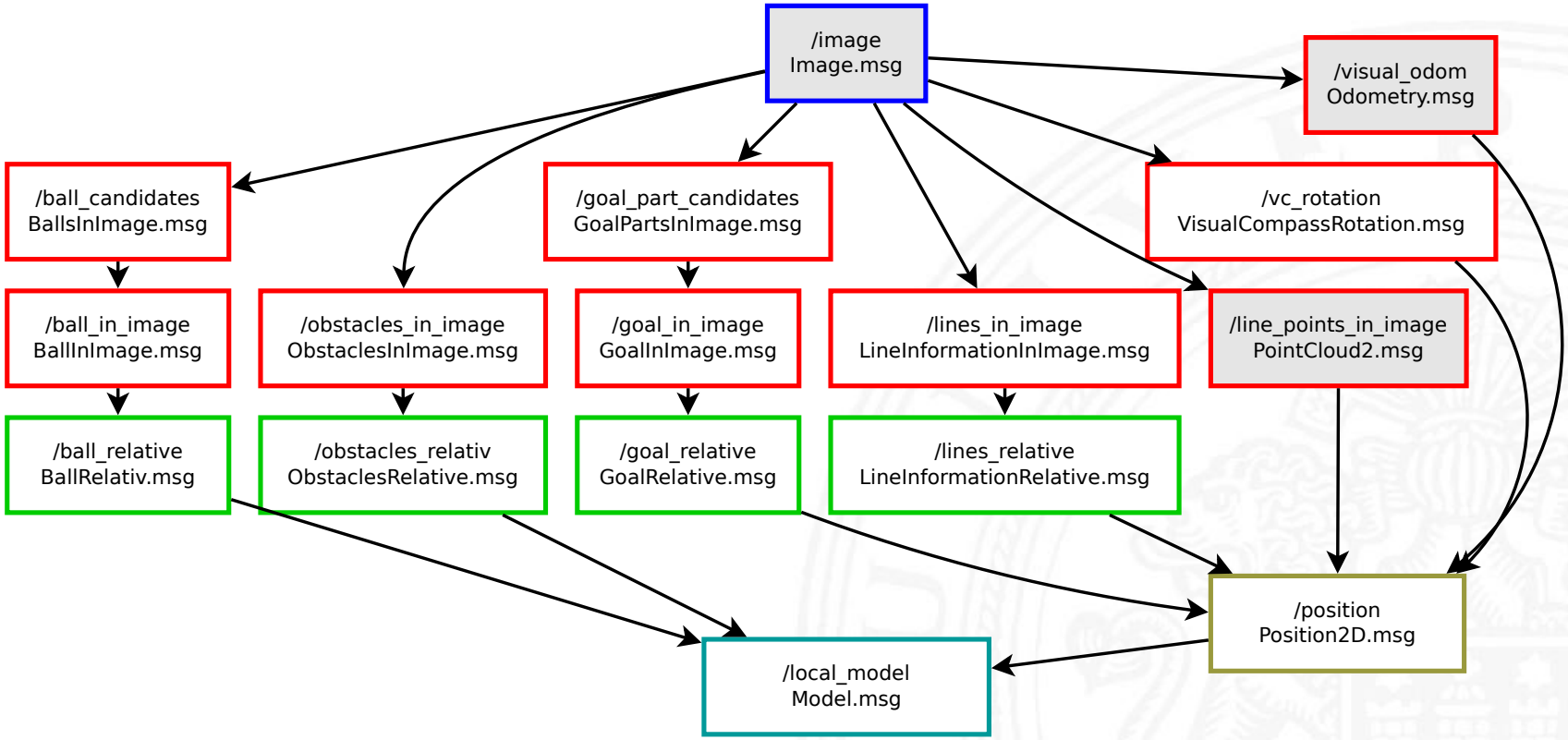
Goals

- Humanoid League
  - Definition of ROS messages for the RoboCup Soccer context
  - Provision of compatible software
    - Visualization tools
    - Simulation environment
    - League specific modules
- Hamburg Bit-Bots
  - Transfer of the robot control software to ROS
  - URDF model of the Minibot

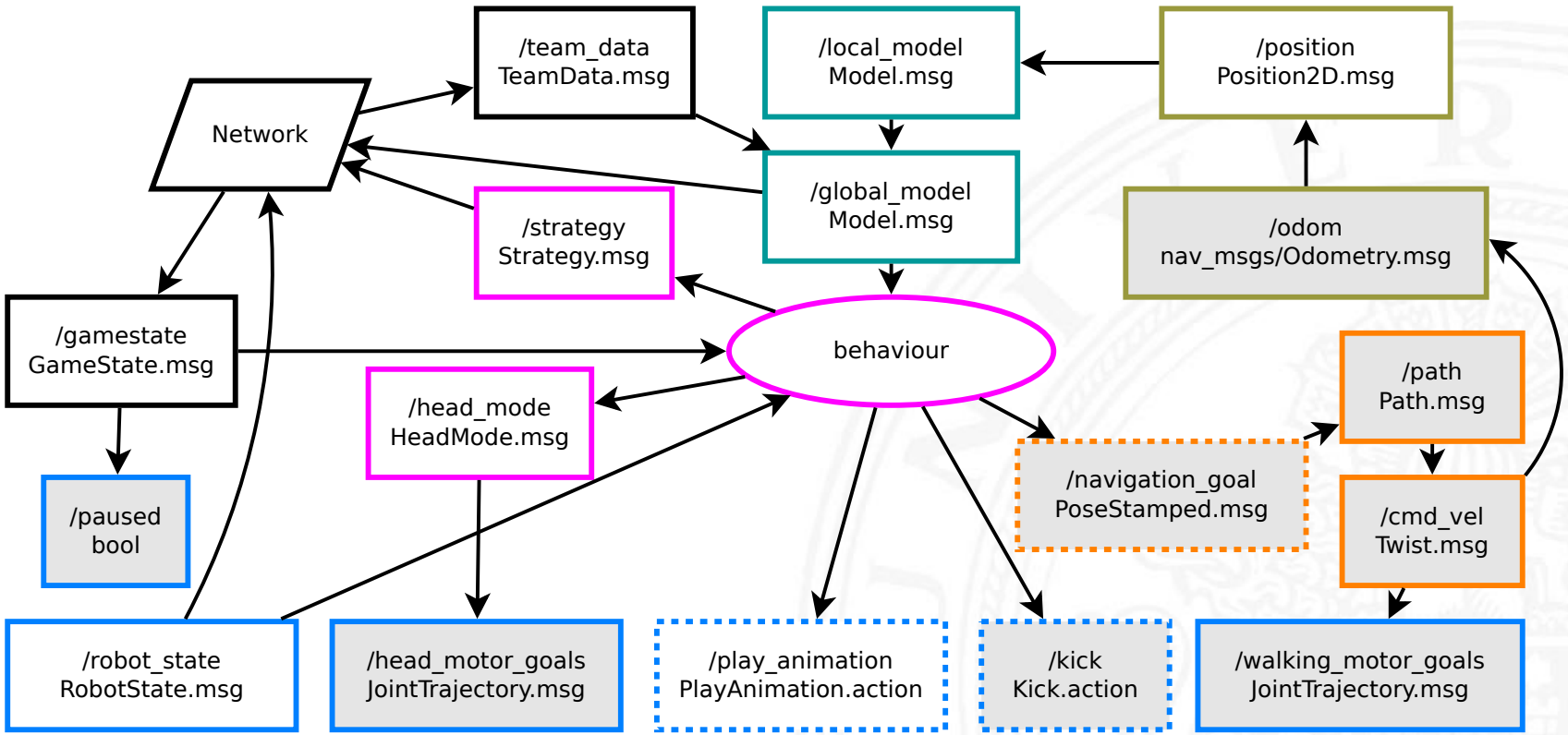
Architecture

- Architecture defined by ROS messages
- Abstraction from concrete robot platform
- Usage of standard ROS messages (grey) as far as possible
- Three parts: sense, plan and act

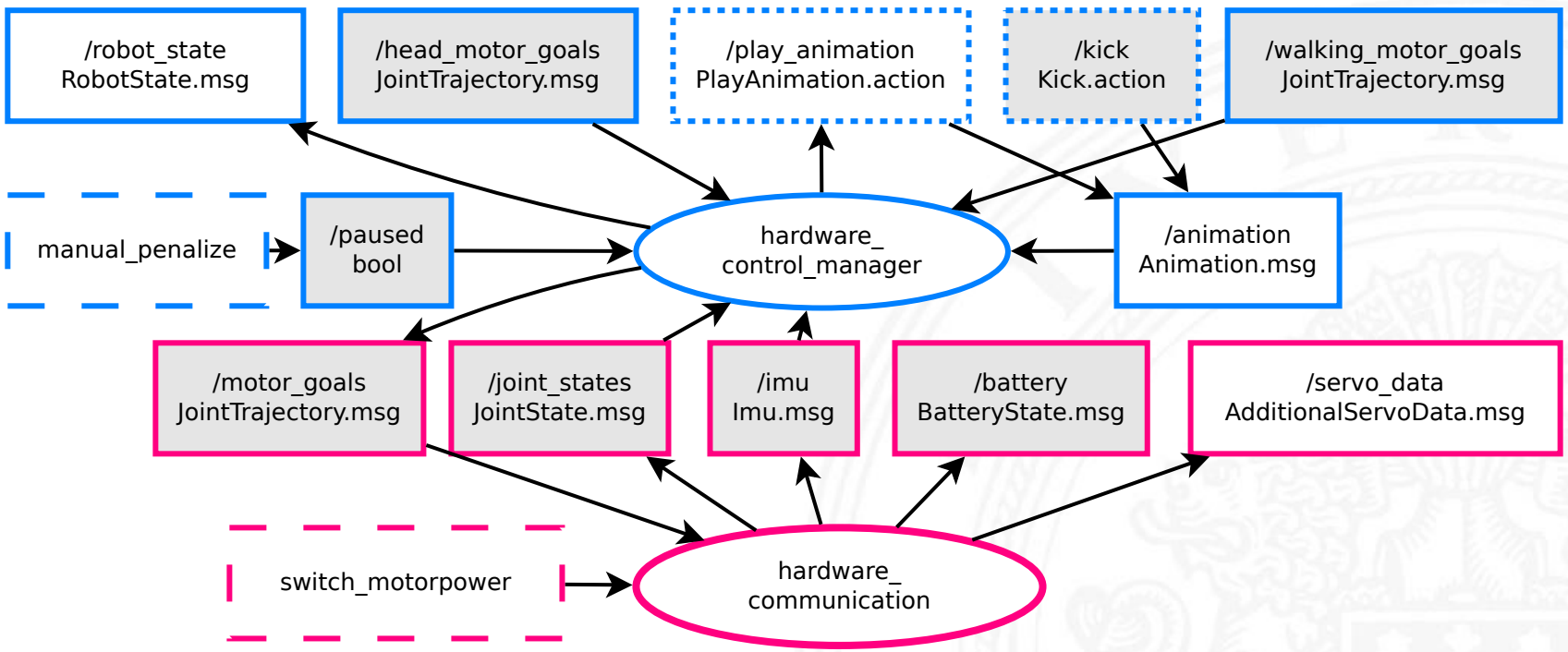
Vision


/image
Image.msg
/visual_odom
Odometry.msg
/ball_candidates
BallsInImage.msg
/goal_part_candidates
GoalPartsInImage.msg
/vc_rotation
VisualCompassRotation.msg
/ball_in_image
BallInImage.msg
/obstacles_in_image
ObstaclesInImage.msg
/goal_in_image
GoalInImage.msg
/lines_in_image
LineInformationInImage.msg
/line_points_in_image
PointCloud2.msg
/ball_relative
BallRelativ.msg
/obstacles_relativ
ObstaclesRelative.msg
/goal_relative
GoalRelative.msg
/lines_relative
LineInformationRelative.msg
/position
Position2D.msg
/local_model
Model.msg

Behaviour


/team_data
TeamData.msg
/local_model
Model.msg
/position
Position2D.msg
Network
/global_model
Model.msg
/strategy
Strategy.msg
/odom
nav_msgs/Odometry.msg
/gamestate
GameState.msg
behaviour
/path
Path.msg
/head_mode
HeadMode.msg
/navigation_goal
PoseStamped.msg
/paused
bool
/cmd_vel
Twist.msg
/robot_state
RobotState.msg
/head_motor_goals
JointTrajectory.msg
/play_animation
PlayAnimation.action
/kick
Kick.action
/walking_motor_goals
JointTrajectory.msg

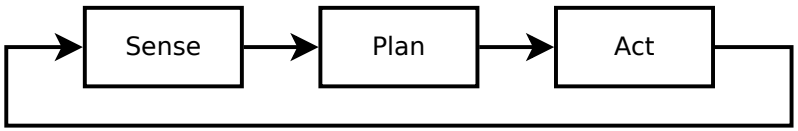
Robot Control



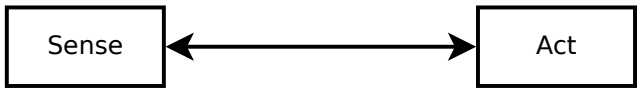
Robotic Paradigm

- Very similar to the hierarchical paradigm
- HCM can be seen as reactive behavior
- Behavior can consist of multiple reactive behaviors

Hierarchical/Deliberative Paradigm



Reactive Paradigm



Hybrid Deliberative/Reactive Paradigm

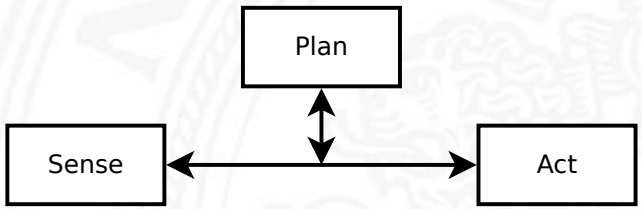
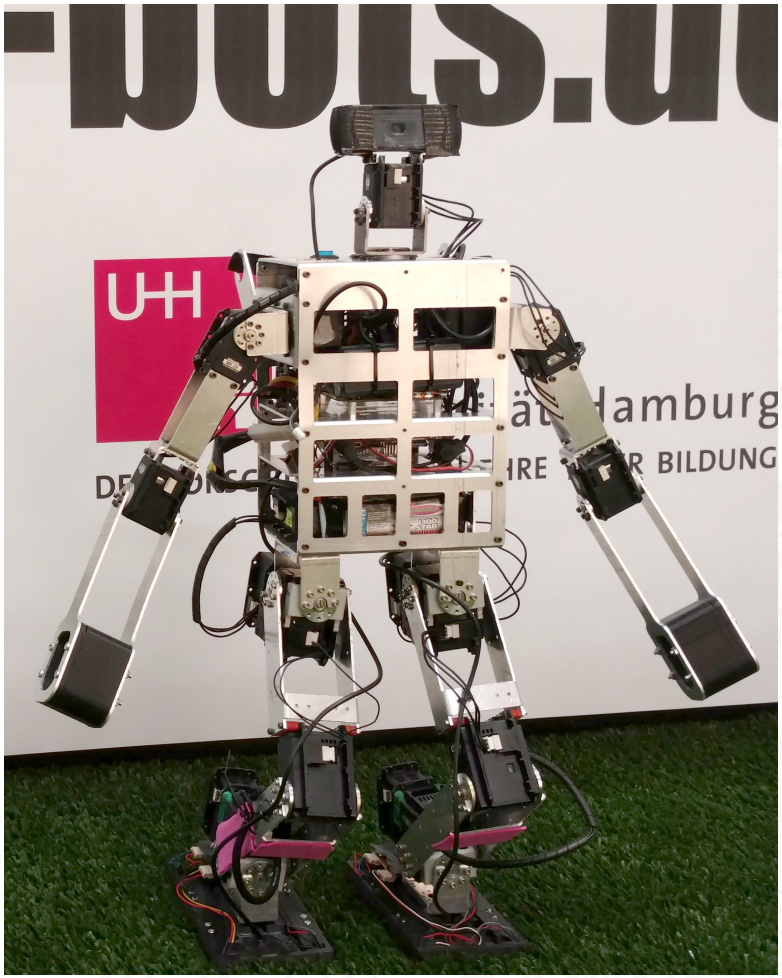


Table of Contents

Used Platform

- Minibot
- Own construction
- Upscaled kinematic layout of the Darwin-OP
- 20 degrees of freedom
- Typical design in the HL
- Odroid XU-3 (8 cores)



Implemented Software

- Humanoid League
  - Messages
  - Game state receiver
  - Team communication
  - rqt plug-ins
  - Gazebo world
- Hamburg Bit-Bots
  - Hardware control manager
  - Walking
  - Animation
  - Hardware communication
  - Minibot URDF

Live Demo

- Walking
- Animation
- URDF
- Gazebo
- Visualizations
  - image
  - relative
  - field

Table of Contents

Comparission

- Compared to the old framework
  - Usage of more than two cores
  - Better visualization and debug tools
  - Modules are more independent
  - Documentation
- Compared to the existing ROS frameworks in the HL
  - Closer to ROS standards (messages, tools, naming conventions)
  - Higher degree of modularization
  - Abstraction of used robot platform

Working with other teams

- WF Wolves already changed to the architecture
- Rhoban FC will until next year
- Standardized team communication will be obligatory next year
  - The mitecom protocol will be used
  - Usage of the implemented team com. package will be proposed
- Mixed team with WF Wolves during Iran Open
- Exchanged software modules with WF Wolves

Performance

	Standing	Animation	Walking
cm730	58%	58%	58%
joint_state_publisher	4%	4%	4%
hcm	10%	20%	20%
walking	12%	12%	40%
animation	7%	46%	7%
pause	0%	0%	0%
buttons	0%	0%	0%
Sum of all nodes	91%	140%	129%
previous motion	45%	50%	60%

Latencies

From	To	Message Type	Latency
cm730	hcm	Imu.msg	7.45 ms
animation	hcm	Animation.msg	17.74 ms
walking	hcm	JointTrajectory.msg	16.90 ms
hcm	cm730	JointTrajectory.msg	32.51 ms

- Latencies are higher than expected
  - rospy
  - ARM architecture
  - Old kernel (3.10)

Latency Comparission - Imu.msg with 100Hz

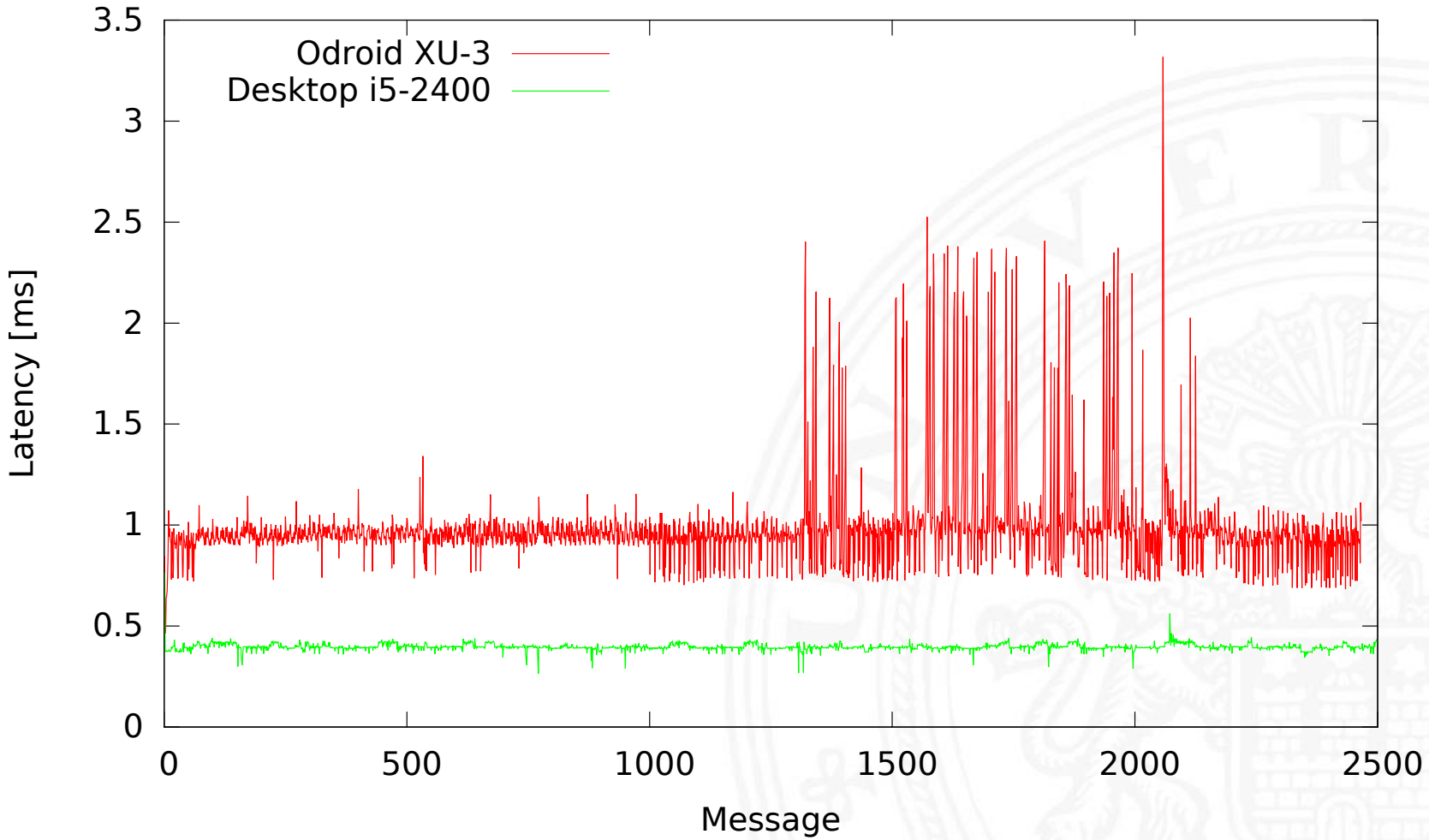


Table of Contents

Conclusion

- The proposed architecture was successfully implemented and tested
- Improvement to previous architecture
- Easier to adapt and closer to ROS standards as others
- Already actively used by two teams
- Possible advantages in the future for the HL

Further Work

- Further work on visualization tools
- Getting more teams to use the software
- Lowering latency
- One way UDP connection for games

References

[1] https://github.com/Pold87/academic-keyword-occurrence.

[2] Taken by Hamburg Bit-Bots during Iran Open 2016,.

[3] http://www.robocup2016.org/de/symposium/team-description-papers/.

[4] Philipp Allgeuer, Max Schwarz, Julio Pastrana, Sebastian Schueller, Marcell Missura, and Sven Behnke.
A ROS-based software framework for the NimbRo-OP humanoid open platform.
In *Proceedings of 8th Workshop on Humanoid Soccer Robots, IEEE-RAS Int. Conference on Humanoid Robots, Atlanta, USA*, 2013.

[5] Leonardo Leottau Forero, José Miguel Yánez, and Javier Ruiz-del Solar.
Integration of the ROS framework in soccer robotics: the NAO case.
In *Robot Soccer World Cup*, pages 664–671. Springer, 2013.

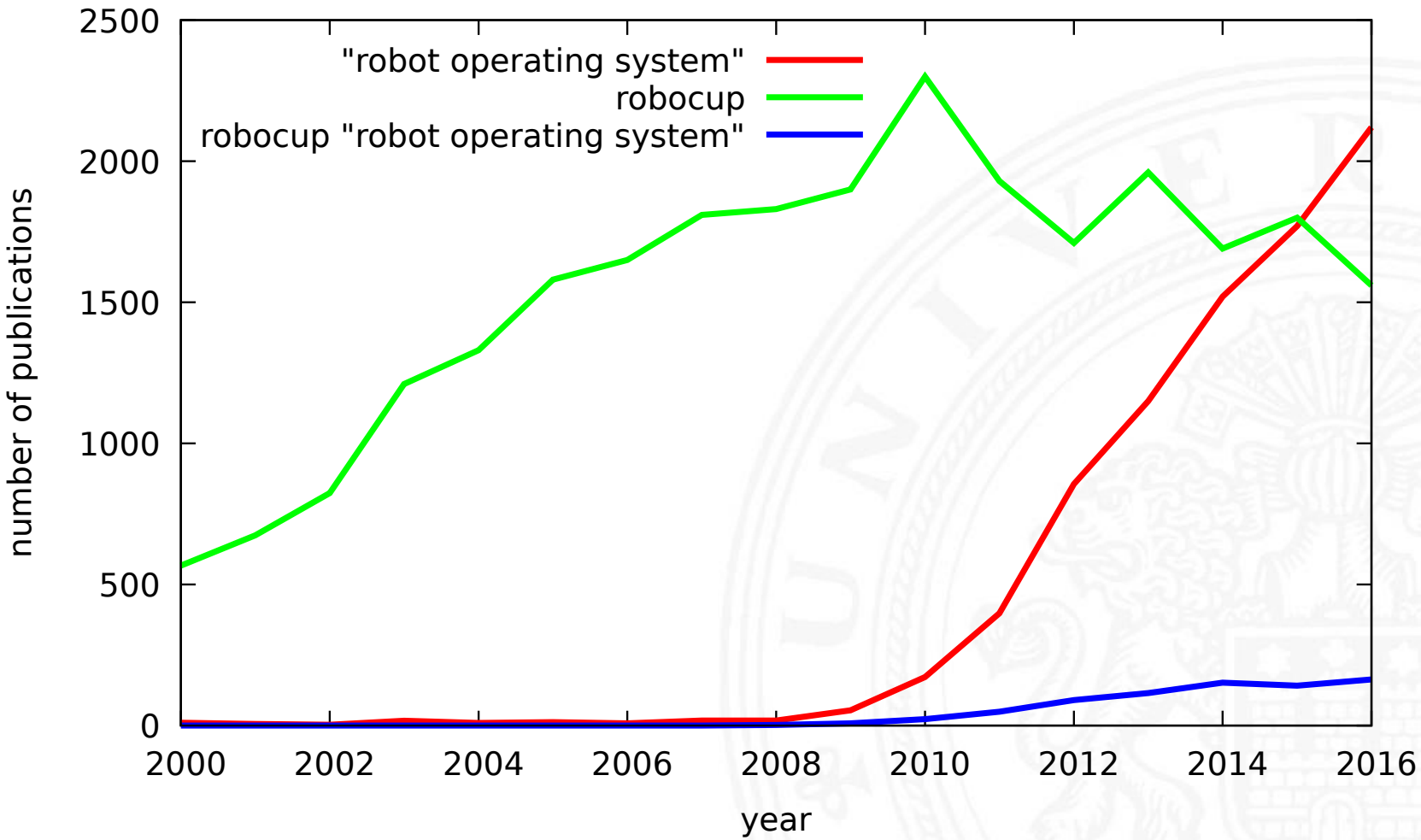
[6] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng.
ROS: an open-source Robot Operating System.
In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

[7] Thomas Röfer and Tim Laue.
On B-human's code releases in the standard platform league–software architecture and impact.
In *Robot Soccer World Cup*, pages 648–655. Springer, 2013.

Questions

Questions?

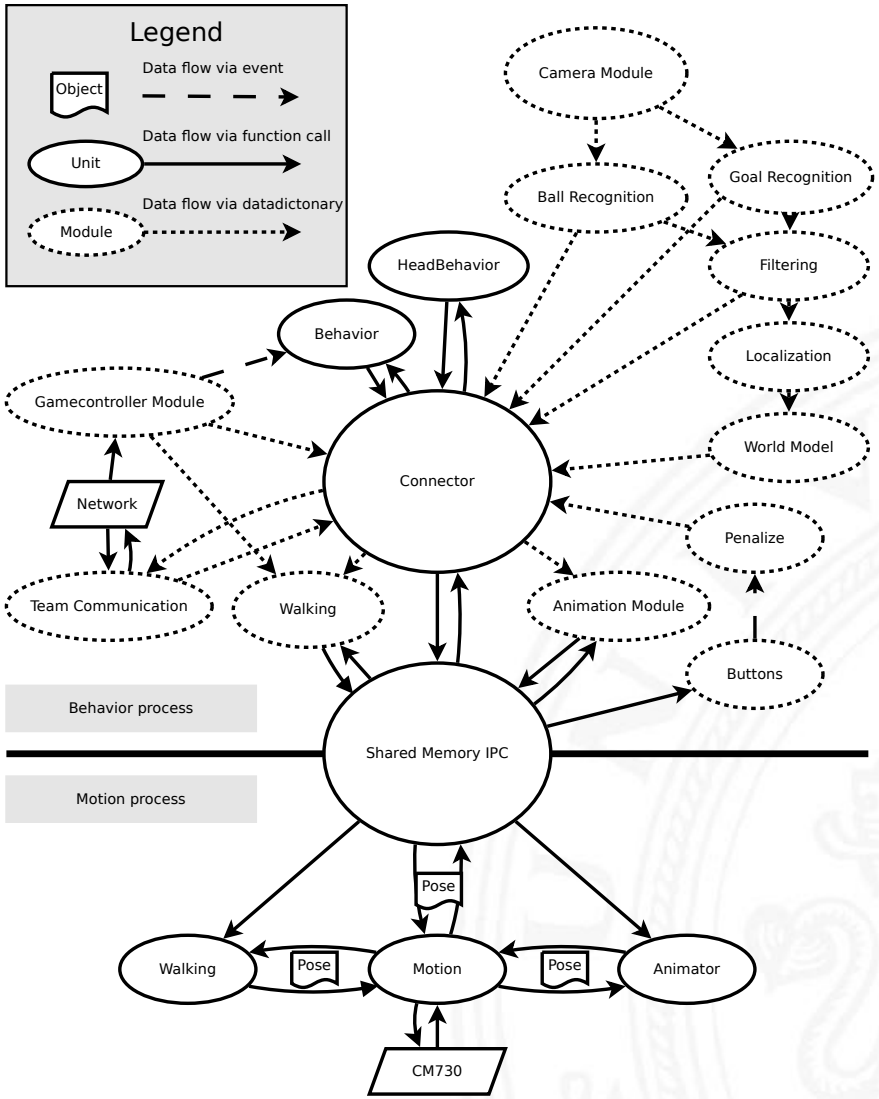
Relevance

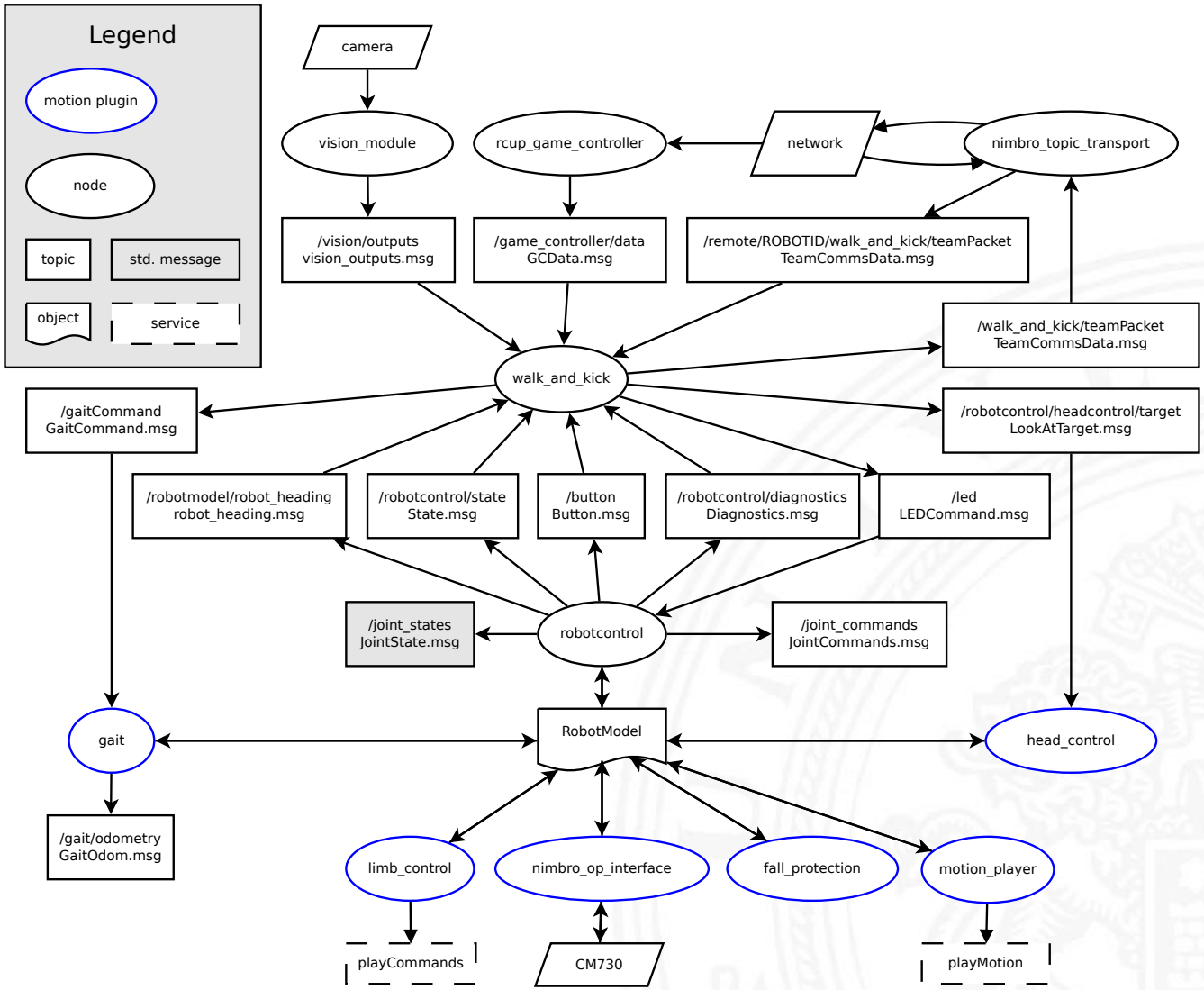

"robot operating system"
robocup
robocup "robot operating system"
number of publications
year
0
500
1000
1500
2000
2500
2000
2002
2004
2006
2008
2010
2012
2014
2016

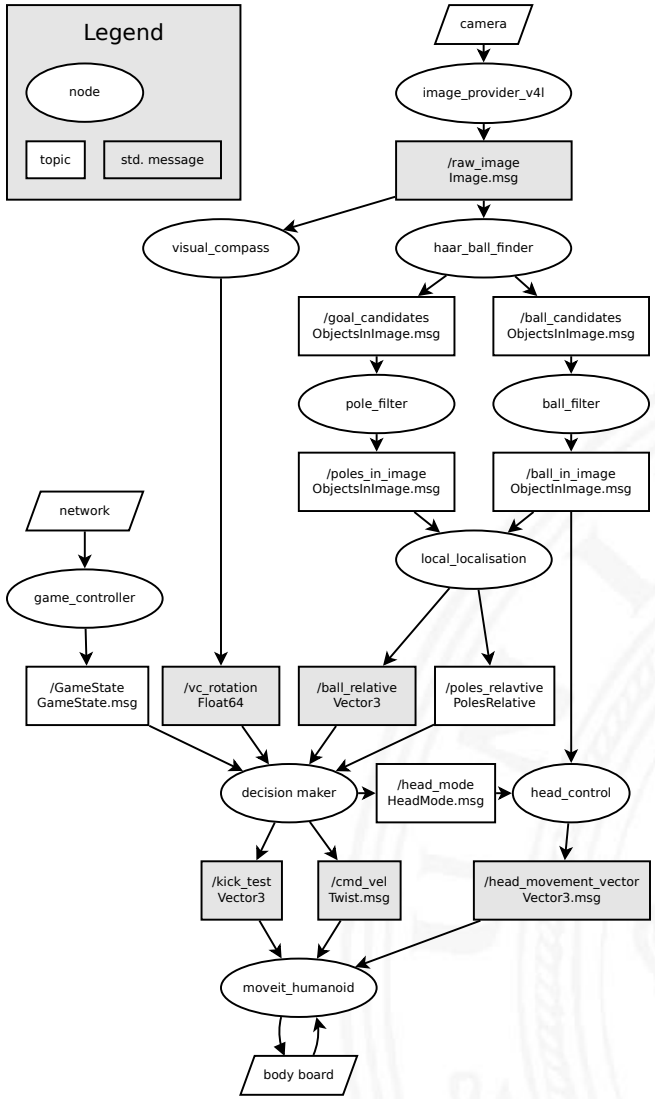
[1]

RoboCup Leagues

RoboCupSoccer	Humanoid Standard Platform Middle Size Small Size Simulation
RoboCupRescue	Robot Simulation
RoboCup@Home	Open Platform Domestic Standard Platform Social Standard Platform
RoboCupIndustial	@Work Logistics


Legend
Data flow via event
Object
Data flow via function call
Unit
Data flow via datadictonary
Module
Camera Module
Goal Recognition
Ball Recognition
Filtering
HeadBehavior
Behavior
Localization
Gamecontroller Module
World Model
Connector
Network
Penalize
Team Communication
Walking
Animation Module
Buttons
Behavior process
Shared Memory IPC
Motion process
Pose
Walking
Pose
Motion
Pose
Animator
CM730


Legend
motion plugin
node
topic
std. message
object
service
camera
vision_module
rcup_game_controller
network
nimbro_topic_transport
/vision/outputs
vision_outputs.msg
/game_controller/data
GCData.msg
/remote/ROBOTID/walk_and_kick/teamPacket
TeamCommsData.msg
/walk_and_kick/teamPacket
TeamCommsData.msg
walk_and_kick
/gaitCommand
GaitCommand.msg
/robotcontrol/headcontrol/target
LookAtTarget.msg
/robotmodel/robot_heading
robot_heading.msg
/robotcontrol/state
State.msg
/button
Button.msg
/robotcontrol/diagnostics
Diagnostics.msg
/led
LEDCommand.msg
/joint_states
JointState.msg
robotcontrol
/joint_commands
JointCommands.msg
gait
RobotModel
head_control
/gait/odometry
GaitOdom.msg
limb_control
nimbro_op_interface
fall_protection
motion_player
playCommands
CM730
playMotion


Legend
node
topic
std. message
camera
image_provider_v4l
/raw_image
Image.msg
visual_compass
haar_ball_finder
/goal_candidates
ObjectsInImage.msg
/ball_candidates
ObjectsInImage.msg
pole_filter
ball_filter
/poles_in_image
ObjectsInImage.msg
/ball_in_image
ObjectInImage.msg
network
local_localisation
game_controller
/GameState
GameState.msg
/vc_rotation
Float64
/ball_relative
Vector3
/poles_relavtive
PolesRelative
decision maker
/head_mode
HeadMode.msg
head_control
/kick_test
Vector3
/cmd_vel
Twist.msg
/head_movement_vector
Vector3.msg
moveit_humanoid
body board