Falcons team presentation

MSL workshop Eindhoven, 22-24 nov 2019

Ronald van der Weide, Edwin Schreuder, Erik Kouters, Jan Feitsma, Stan Mertens, Lu Dai

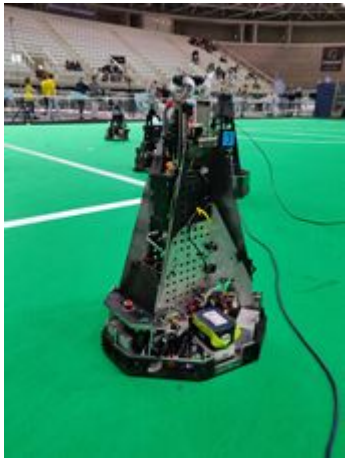
Contents

- General
- New hardware platform
- Software: usability & diagnostics
- “MultiCam” as OmniVision system
  - Machine Learning in Falcons software

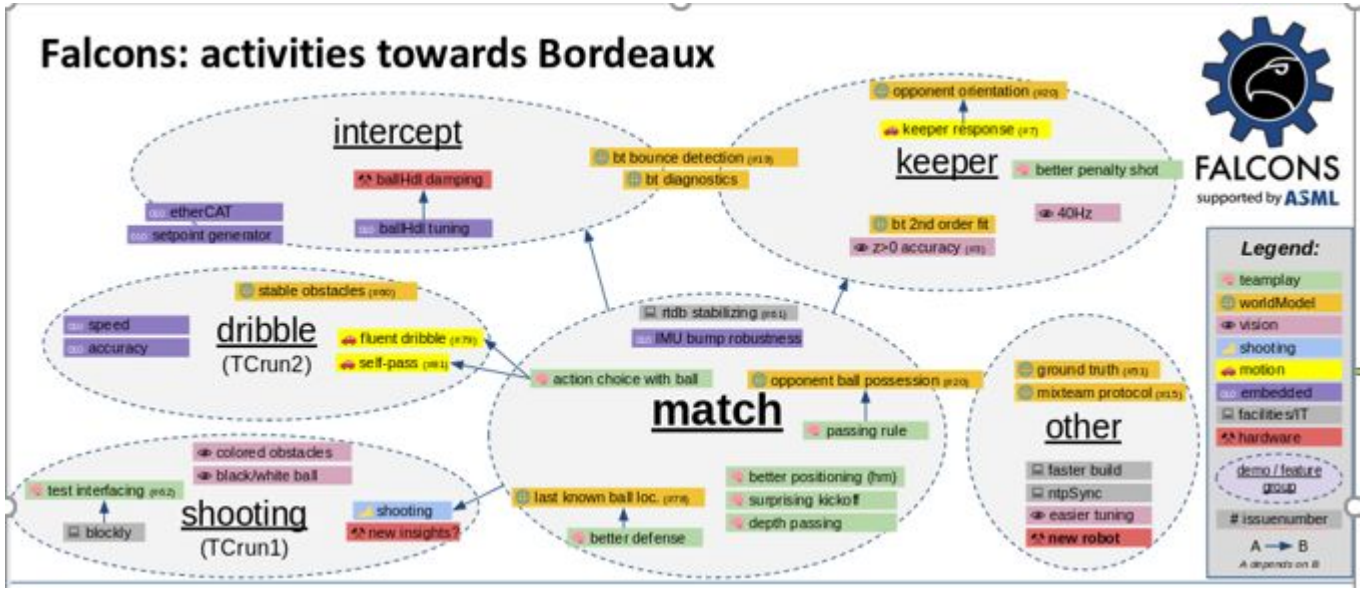
General

2018-2019:

- Introduction of new Proto Robot 7
- Introduction Extendable Keeper Frame
- 3rd place Robotica Portugal 2019
- 3rd Place Robocup 2019 Sydney
- Robot uptime very high
- Increase in manpower from 17 to 46 right now
- Creation of New Organization Structure



Road to Bordeaux



FALCONS
supported by ASML

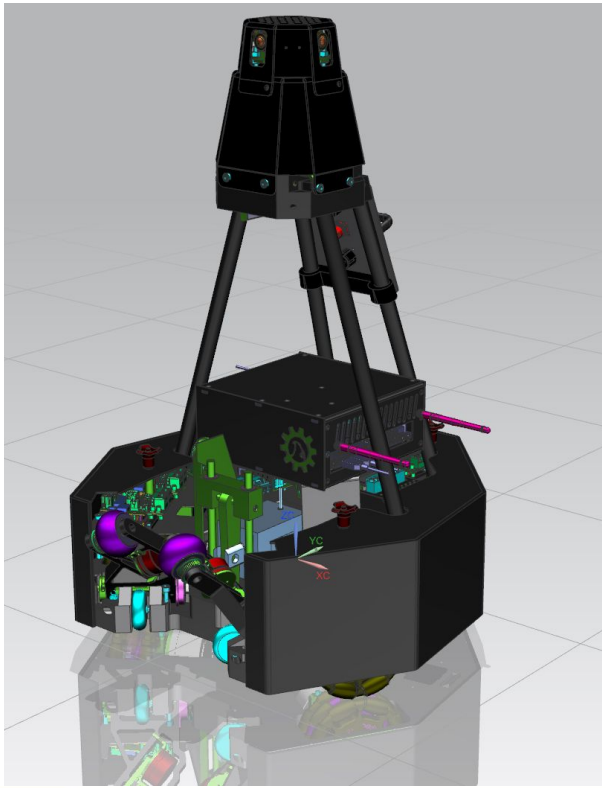
New hardware platform

New hardware platform

FALCONS
supported by ASML

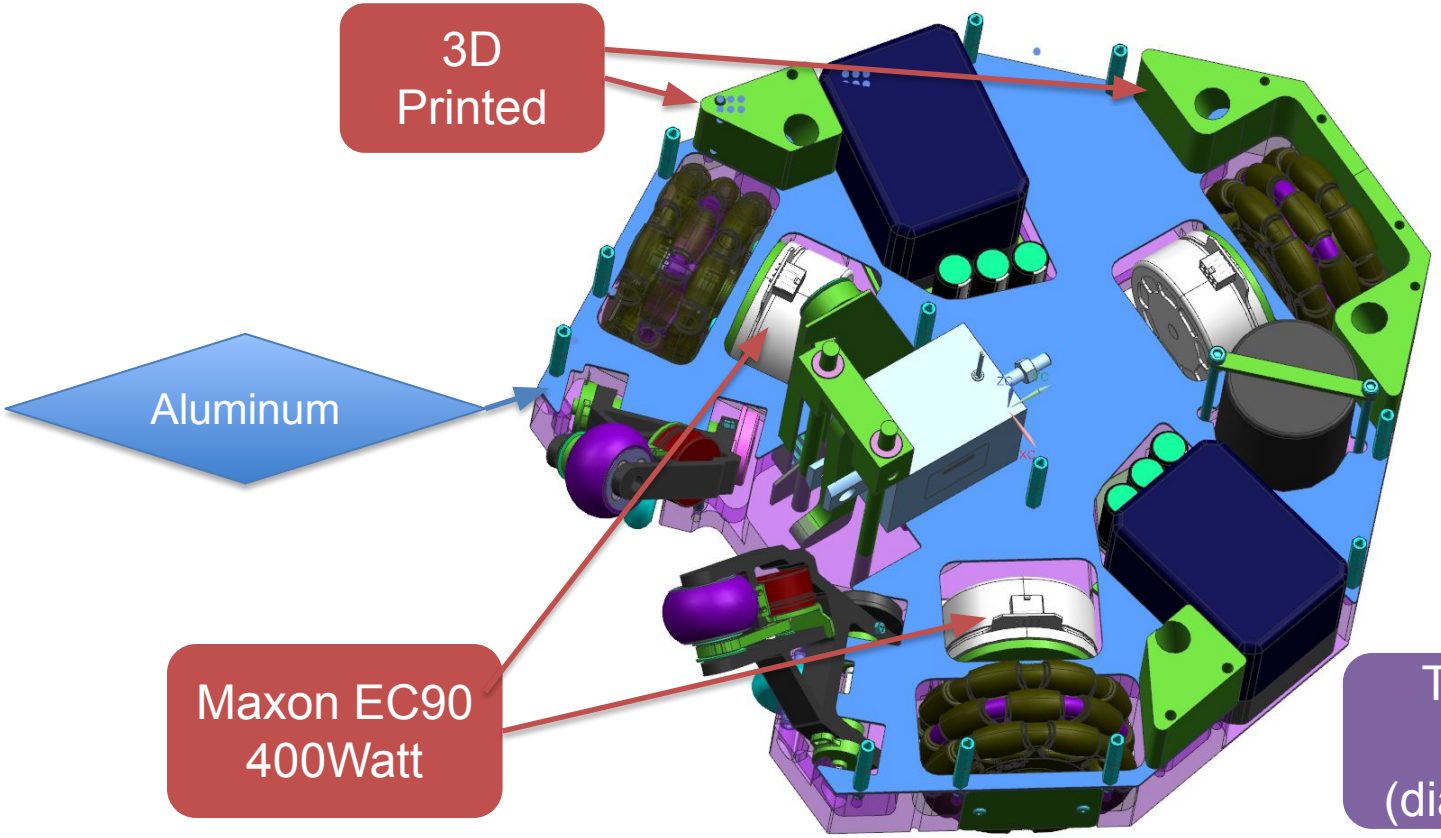
- reasons for redesign:
  - enable faster motion
  - improved serviceability (modularity)
  - stiffer bottom frame
  - . . .

- targeting 1 working robot Portugal 2020

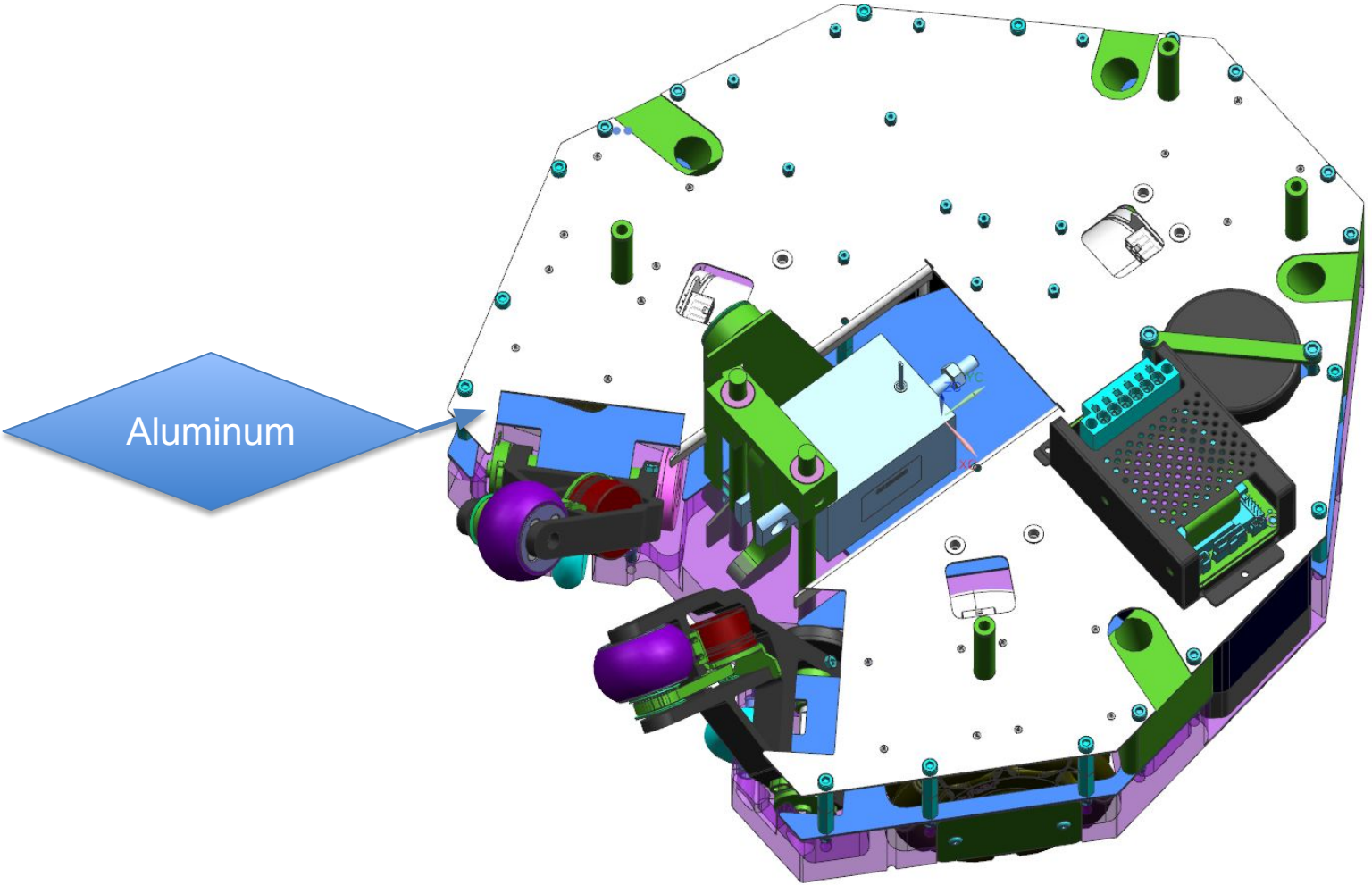


New hardware platform

FALCONS
supported by ASML

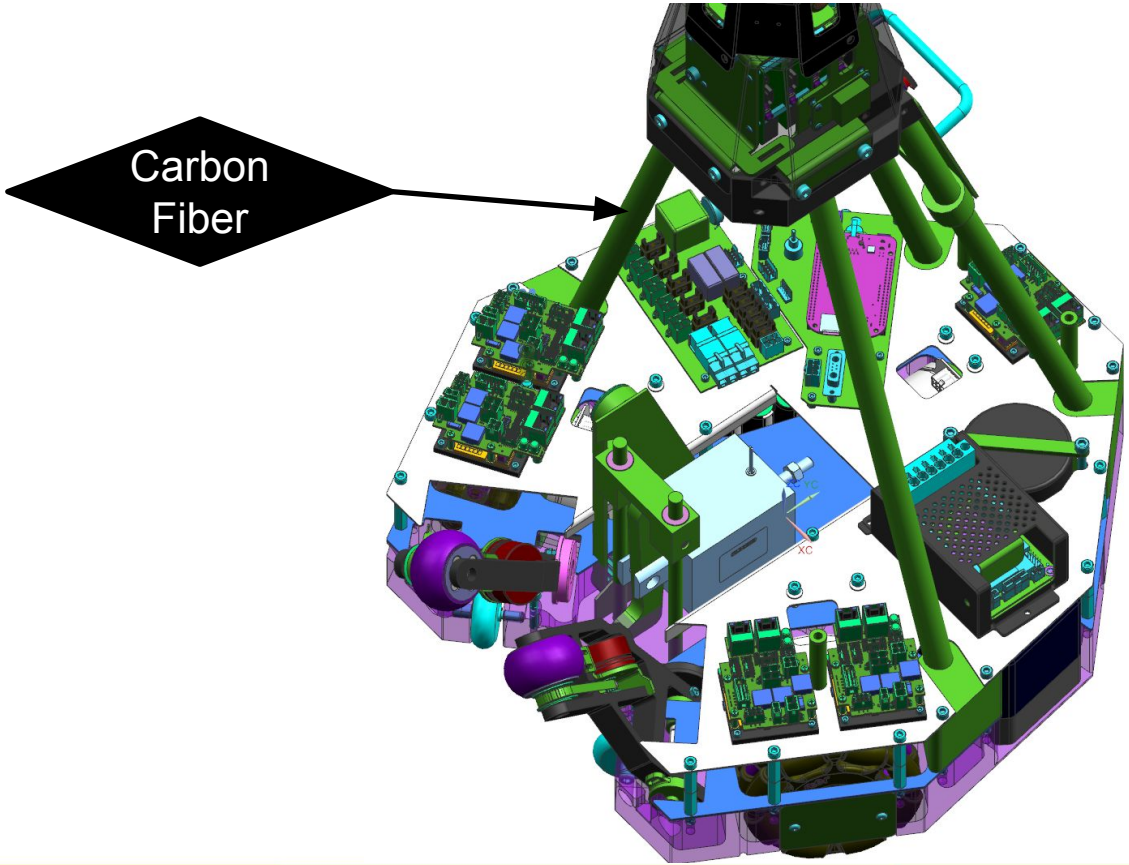


FALCONS
supported by ASML



New hardware platform

FALCONS
supported by ASML



Ballhandlers concept

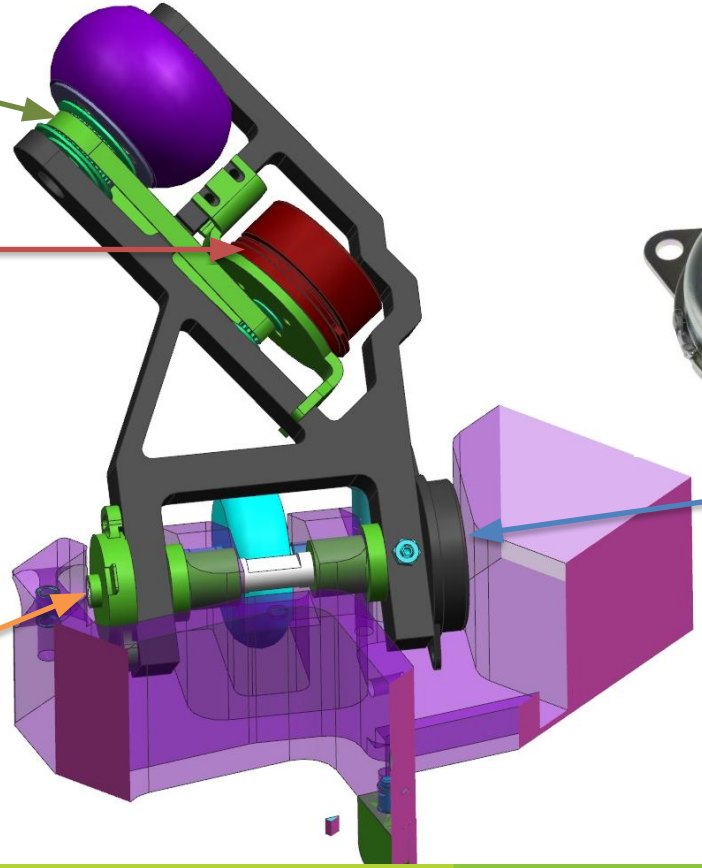
FALCONS
supported by ASML

Belt Driven

High Torque EC45

Rotary Damper

Linear Angle Sensor

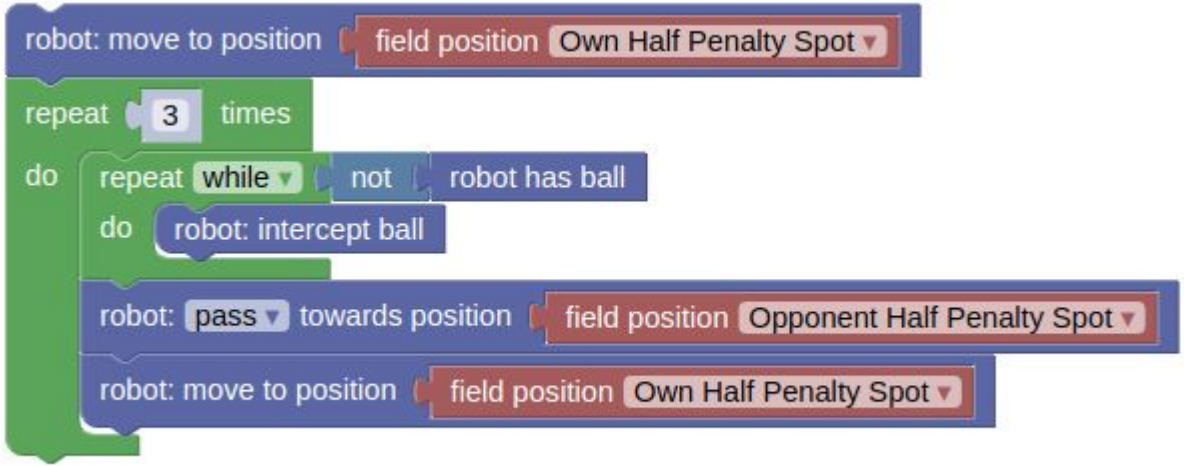


FALCONS
supported by ASML

Software: usability and diagnostics

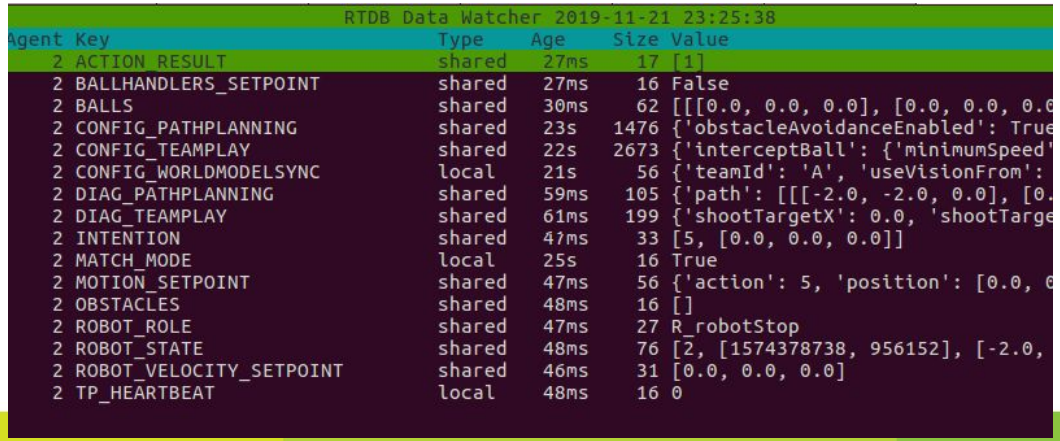
Usability

- Testing the robot made easy: Google Blockly
- Inspired by VDL RobotSports



Diagnostics

- The robot is not behaving as expected ⇒ How do you pinpoint the root cause?
- Step 1: Find the software component this is showing unexpected behavior
  - Playback what the robot was thinking
    - playback takes a **`.rdl`** file: RTDB Data Logging
      - can be from baseStation (shared data) or robot (more/local data)
      - is a sequence of zipped RTDB frames, similar to what **`comm2`** is using
    - during playback, one can inspect RTDB contents, or connect other diagnostics tools



RTDB Data Watcher 2019-11-21 23:25:38

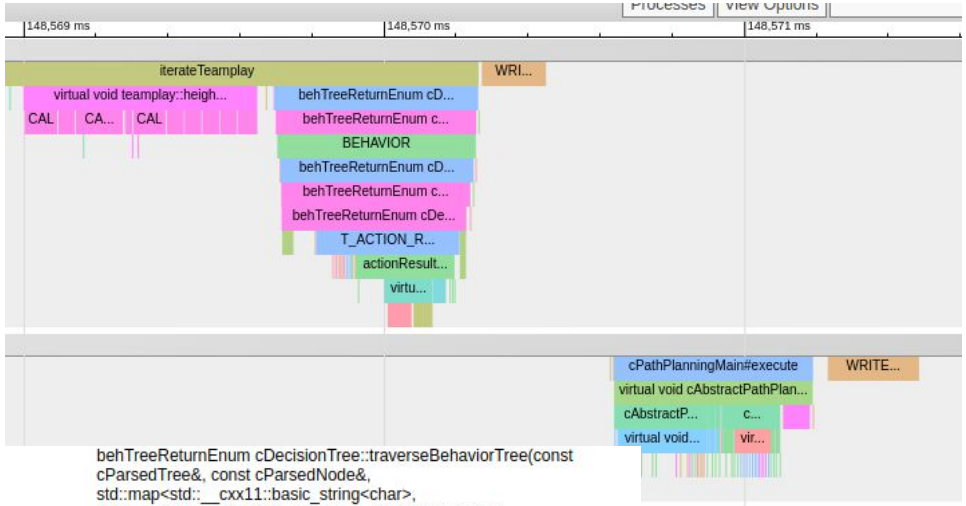
Agent	Key	Type	Age	Size	Value
2	ACTION_RESULT	shared	27ms	17	[1]
2	BALLHANDLERS_SETPOINT	shared	27ms	16	False
2	BALLS	shared	30ms	62	[[[0.0, 0.0, 0.0], [0.0, 0.0, 0.0
2	CONFIG_PATHPLANNING	shared	23s	1476	{'obstacleAvoidanceEnabled': True
2	CONFIG_TEAMPLAY	shared	22s	2673	{'interceptBall': {'minimumSpeed'
2	CONFIG_WORLDMODELSYNC	local	21s	56	{'teamId': 'A', 'useVisionFrom':
2	DIAG_PATHPLANNING	shared	59ms	105	{'path': [[[-2.0, -2.0, 0.0], [0.
2	DIAG_TEAMPLAY	shared	61ms	199	{'shootTargetX': 0.0, 'shootTarge
2	INTENTION	shared	47ms	33	[5, [0.0, 0.0, 0.0]]
2	MATCH_MODE	local	25s	16	True
2	MOTION_SETPOINT	shared	47ms	56	{'action': 5, 'position': [0.0, 0
2	OBSTACLES	shared	48ms	16	[]
2	ROBOT_ROLE	shared	47ms	27	R_robotStop
2	ROBOT_STATE	shared	48ms	76	[2, [1574378738, 956152], [-2.0,
2	ROBOT_VELOCITY_SETPOINT	shared	46ms	31	[0.0, 0.0, 0.0]
2	TP_HEARTBEAT	local	48ms	16	0

Playback merged with video

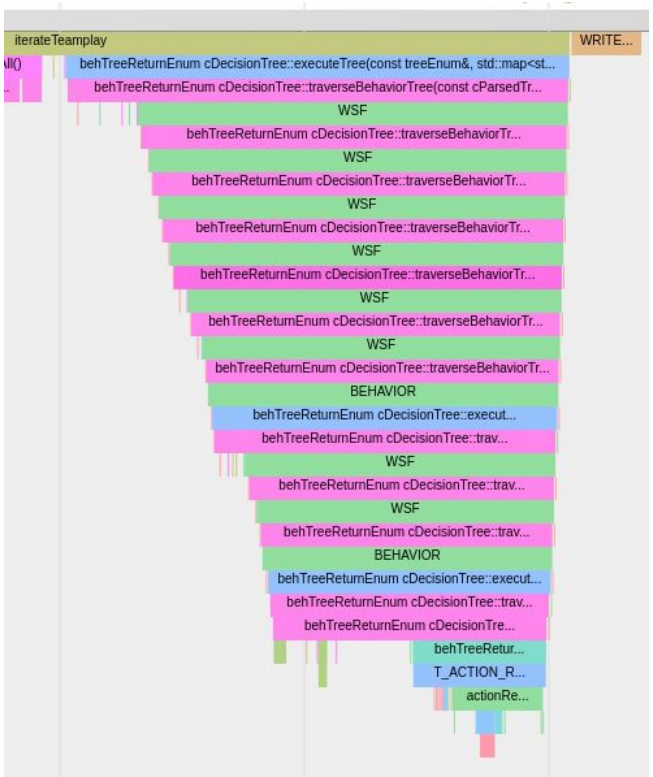
- camera on top of field as *“ground truth”*
- processing after match:
  - one-time action
  - dewarp, field border lines
  - time sync video w.r.t. RDL

Diagnostics

- Step 2: Zoom in on software component
  - Chromium Tracing Framework



Title	behTreeReturnEnum cDecisionTree::traverseBehaviorTree(const cParsedTree&, const cParsedNode&, std::map<std::__cxx11::basic_string<char>, std::__cxx11::basic_string<char> >&, const treeEnum&, memoryStackType&, int, memoryStackNodes&)
Category	/home/robocup/falcons/code/packages/teamplay/src/cDecisionTree.cpp
User Friendly Category	other
Start	20,896.468 ms
Wall Duration	1.588 ms
Self Time	0.057 ms
Args	
msg	"tree: defenderMain; node: wsf:doesOwnRobotHaveBall"



Diagnostics

FALCONS
supported by ASML

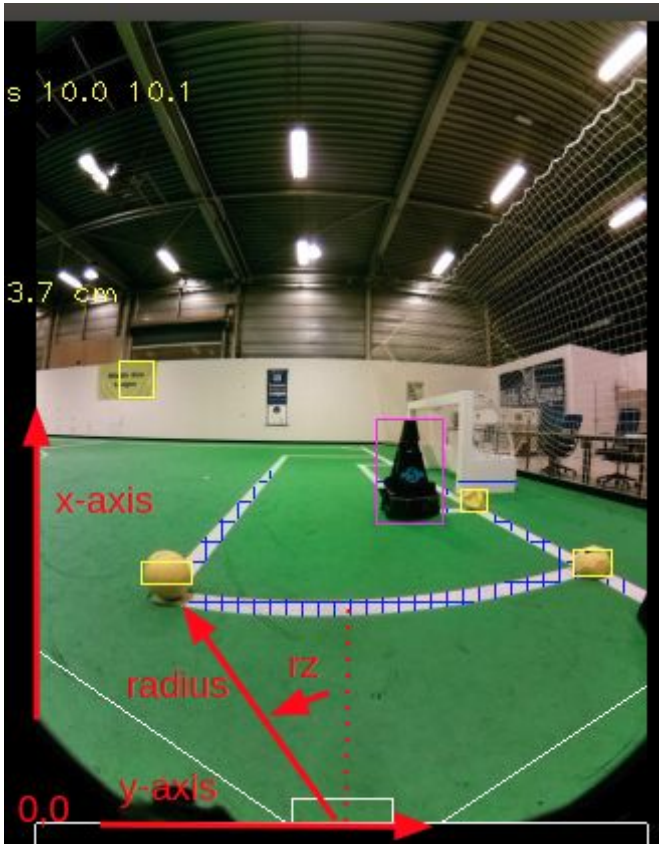
- BaseStation stores:
  - `.rdl` file containing shared RtDB data for all robots, allowing playback.

- Every robot stores:
  - `.rdl` file containing detailed local RtDB data for the robot itself, and shared RtDB data from other robots, allowing playback from detailed robot perspective.
  - Trace files for all software components, containing every single decision made during the match.

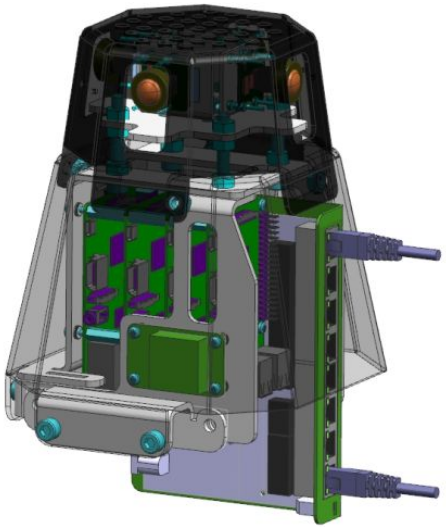
FALCONS
supported by ASML

“MultiCam” as OmniVision system

MultiCam front view



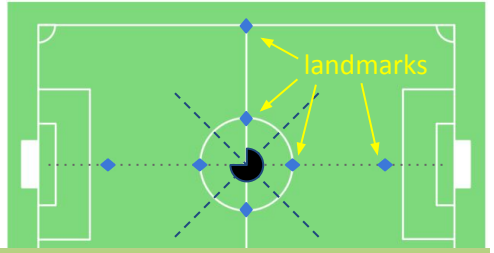
showing 1 out of 4 cameras
~100° horizontal (600px)
~135° vertical (800px)
large fisheye distortion



Localization using MultiCam

FALCONS
supported by ASML

- calibration:
  - fisheye per camera (intrinsic parameters)
  - mounting/pointing errors using field **landmarks** (extrinsic parameters)
  - quite accurate, **robust** and fast, using opencv3



- per camera, *line point detection* is done
  - using a **lookup table** to transform white pixel coordinates to robot coordinate system

- the result of 4 cameras is used in localization:
  a ***simplex*** algorithm determines position candidates using the selected white pixels in robot coordinates

- filtering & selection of candidates is done by worldModel
  - same architecture principle for localization, balls and obstacles

Machine learning @Falcons

- we see two main candidate applications for Machine Learning in Falcons software:
  - ***teamplay*** (= decision making, choice of robot action)
  - computer ***vision***

- teamplay
  - our SW stack behaves “deterministic”, using decision trees and tuned height maps as function of world state
  - mainly for ease of development and diagnostics purposes
  - this will probably remain the case for coming year(s), so no Machine Learning
- vision
  - current vision system is sensitive to lighting changes
  - furthermore, filter design and tuning is complex / labour-intensive
  - **this year, we are considering Machine Learning** for ball- and obstacle detection
    - regular black&white balls
    - obstacles of arbitrary shape/colors

FALCONS
supported by ASML

End - questions?

FALCONS
supported by ASML

Backup slides

Backup: MultiCam system features

- 4 cameras
  - ~100 degrees horizontal and ~135 degrees vertical
  - MIPI CSI-2 (raw data/low latency interface)
- 4 Raspberry 3B+
  - 4 x A53 CPU (used for pre-processing)
- X86_64
  - Detection, Localization, Synchronization, Control, deWarp,..
  - Ethernet connection between raspi's and x86_64
- Pro
  - Way better view than old omni camera (~6 meters vs 3.5m)
  - "Low" latency (complete vision ~50 ms, possibility to ~40ms)
  - Works pretty good!
- Cons
  - Cross platform, more work to maintain
  - **Same high dependency on color calibration**
  - Obstacles sometimes merged with border
  - Rather complex state machine / rules to handle exceptions