Falcons team update

MSL workshop 2016 Kassel

Jaap Vos, Edwin Schreuder, Erik Kouters, Jan Feitsma

Content

FALCONS
supported by ASML

- Team update
- Hardware
- Robustness
- Motion
- Reasoning (teamplay)
- WorldModel+vision
- BaseStation
- Simulator

Team update

- 4th position at European open
- 3rd position at Portuguese open
- 8th position at World Championships Leipzig.
- Team has about 30 team members both on tech and non tech.
- All work is done in the evening hours, mostly Tuesday and Thursday evening.
- Based on spent hours we have about 4.45 FTE.
- 90% of the current workload is now on Software
- Started new activity: robot test and integration to test the robots on pre-defined KPI’s
- Monthly test match against VDL

Hardware updates 2016

- Improved motor board communication via USB and USB hubs i.s.o. RS-485-RS-232.
- Due to unexpected current loops and erroneous communication, had to build in USB isolators.
- Compass issues with new compass, revert back to old compass.
- Added front facing camera for the goalkeeper to enhance vision.
- New ball handler arms with wider capture angle.

Upcoming new hardware

FALCONS
supported by ASML

- Front facing camera’s on all robots
- New, improved ball handle angle encoders
- Increased shooting power
- Improved shooting lever.
- Moving keeper frame.
- Working on new platform with new CPU box ,Beagle Bone Black, Motor drivers, and new motors and vision system.

Robustness (problem)

FALCONS
supported by ASML

- USB connectivity instable (ground loops, usb re-enumeration)
- USB serial ports would disappear/reappear in OS.

- Hardware abstraction layer not able to cope.
- Insufficient diagnostics and code maintenance difficult.

Effect:

- Erratic / no movement
- Reboot of driver/system necessary

Robustness (solution)

FALCONS
supported by ASML

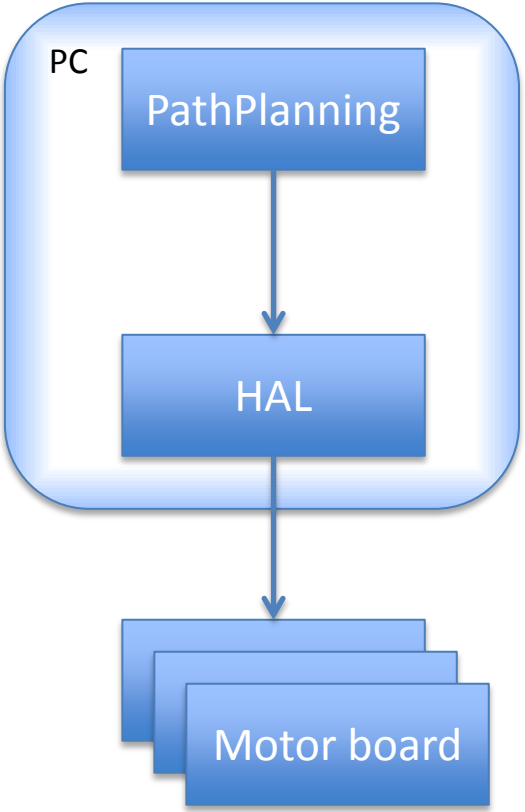
- Partial rewrite of current HAL.
- Robust for device reboots/hickups/emergency button.
- More diagnostics for errors to visualizer.

- Easier to extend and less resource intensive

Motion (problem)

- KPI’s have shown inaccurate motion
  - “Banana”-shaped movement
  - Robot velocity does not reach desired setpoint
  - Deviation in velocity in Y direction is higher than in X direction

Motion



- Position -> Velocity (PID)
- RCS Velocity -> Velocity per motor

  → 3x3 matrix $(\dot{x}, \dot{y}, \dot{\theta} \rightarrow \omega_{Left}, \omega_{Right}, \omega_{Rear})$
- Velocity -> Motor current (PID)

Motion (solution?)

FALCONS
supported by ASML

- PID tuning motor boards
- Suspected errors in RCS conversion matrix
  - Initial measurements show matrix is (partially) incorrect
  - Matrix deviating between robots?
- Different PID settings for different setpoints / mode of operation (gain scheduling)?

Reasoning (teamplay)

FALCONS
supported by ASML

- First version intended for our first steps
  - Difficult to maintain
  - Difficult to debug
- Second version intended for longer use
  - Behavior and code fully separated
  - No recompile necessary -> realtime tactics update possible
  - Dynamic role assignment of robots

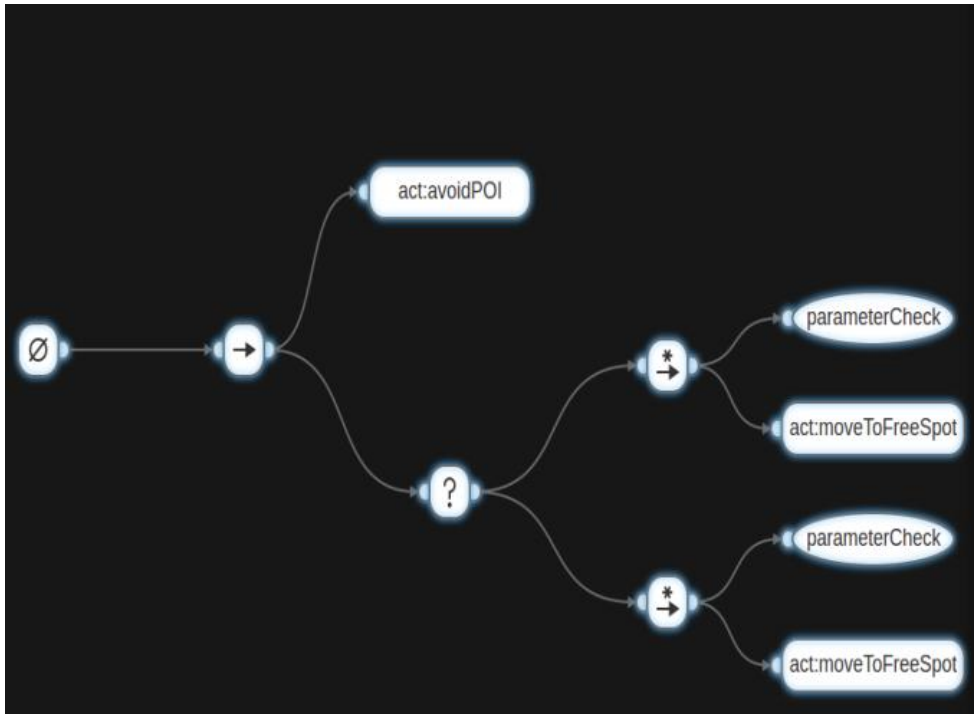
Reasoning (teamplay) GUI

FALCONS
supported by ASML

Nodes

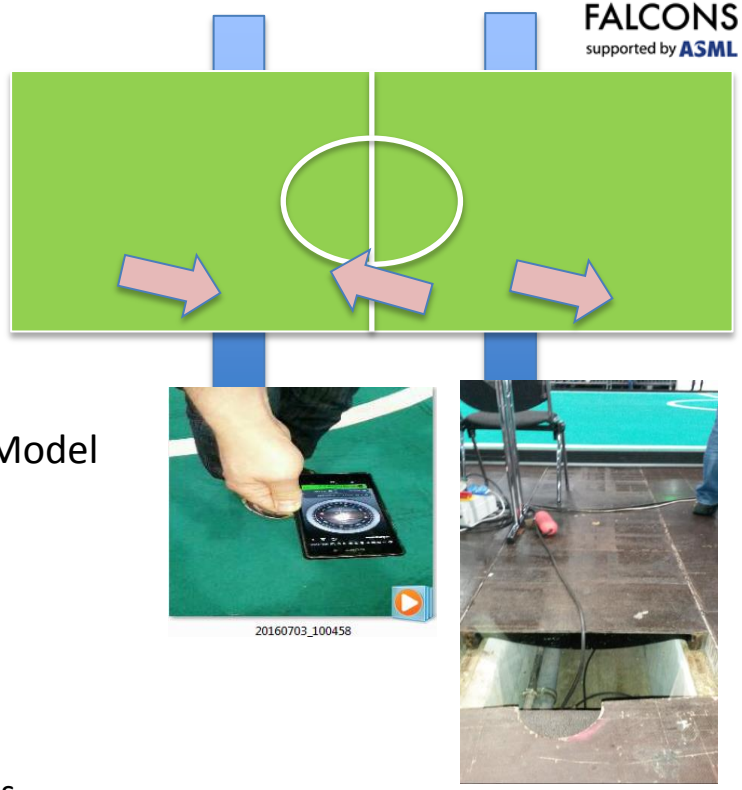
Composites
Sequence
Priority
MemSequence
MemPriority
wsf:isDroppedBallSetPiece
wsf:isPrepareSetPiece
wsf:isOwnSetPiece
wsf:isKickoffSetPiece
wsf:isCornerSetPiece
wsf:isBallAtOwnSide
wsf:doesTeamHaveBall
wsf:isBallAtOpponentSide
wsf:doesOwnRobotHaveBall
wsf:ballPickupOnOppHalf
wsf:isBallLocationKnown

act:shoot
act:stop
act:goalKeeper
act:getBall
act:success
act:avoidPOI
act:interceptBall
act:positionBeforePOI
act:positionBehindPOI
act:moveToFreeSpot
act:move



Worldmodel + Vision

- Leipzig:
  - major troubles with Compass
  - poor ball tracking
- ➔ decide to redesign worldModel+vision
- vision
  - split processing threads
  - communicate all localization candidates to worldModel
  - prepare for adding more camera's
- worldModel
  - localization:
    - 20% vision, 80% encoders
    - select best vision candidate
    - (slip detection / filtering)
    - no more compass, instead initialize to play forwards
  - improved ball- and obstacle tracking

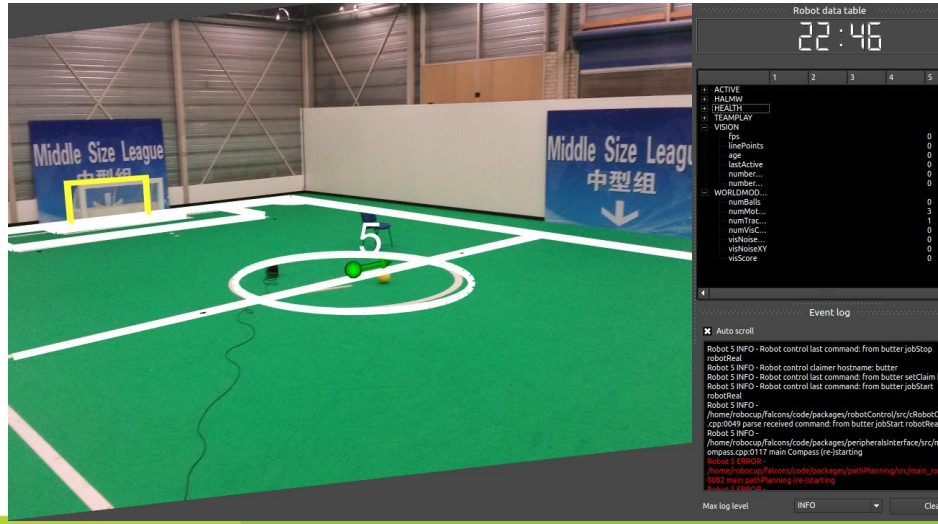


Base Station

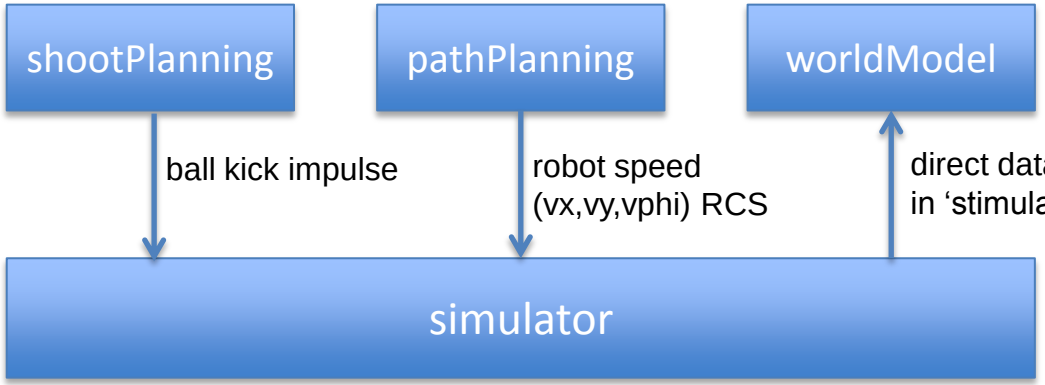
- old pygame visualizer end of life
- implemented new 3D C++ visualizer
  - VTK/QT
  - based on CarpeNoctem visualizer, which in turn is based on Cambada
- our ROS adapters are cleanly isolated in code, for reusability

- working on *augmented reality*: draw 3D data scene on top of video feed
  - building on Cambada scient. chall. 2016

- roadmap: merge other baseStation elements (commandGui, simulator, ...)



Simulator



- current design:
  - ROS interfacing
  - custom-built
  - robots can share the same space... no scrums
  - simple linear 2D ball model with friction and bouncing off objects
  - a second instance of our own team for opponents/obstacles
- intend to switch to Gazebo (for more realism) – on hold

- simulator link discussion
  - due to each team having their own SW environment, it cannot be on single laptop
  - so we need some network communication protocol (like worldModel mixed packets)
  - many things to agree on (interfaces, deployment, timing), lot of work
  - **why would we pursue this? is this worth the effort**
  - (why not re-boot mixed-team protocol, e.g. incorporate in technical challenge?)

SW arch. diagram