Survey response 22

Software

Team Name

KURA

Is your software fully or partially OpenSource. If so, where can it be found:

Our software is not opensource yet

Do you have a kinematic or dynamic model of your robot(s)? If so, how did you create it (e.g. measure physical robot, export from CAD model)?

We export kinematic and dynamic model as URDF from SolidWorks

Are you using Inverse Kinematics? If so what solution (analytic, (pseudo)inverse jabcobian, etc...) are you using?

We use inverse kinematics with analytic solution based on Rhoban IKWalk

Are you simulating your robot? If so what are you using simulation for?

We have robot simulation in Webots, PyBullet and MuJoCo

What approach are you using to generate the robot walking motion?

We use Rhoban IKWalk but also use IMU for PD step position regulation for hip, ankle roll, pitch joints and pressure sensors for step time adaptation.

What approach are you using to generate motions for standing up?

Hand tuned keypoint trajectory interpolation (keypoint animation)

What approach are you using to generate kicking motions?

Hand tuned keypoint trajectory interpolation + some PD regulations for stabilization in ankle and hip

Do you use any other motions than the previously mentioned? If so, what approaches are you using to generate them?

Which datasets are you using in your research? If you are using your own datasets, are they public?

We use our own dataset to train YOLO. Currently, it's not public

What approaches are you using in your robot's visual perception?

A custom stereo vision arrangement is implemented to generate a point cloud. This process involves OpenCV-based rectification and disparity estimation, followed by a bespoke method for converting disparity to a point cloud. Ground plane estimation is done using the RANSAC algorithm. Obstacle detection uses Point Cloud Library's KD-Tree to identify objects protruding above the ground plane by a certain threshold. We also use OpenVINO optimized YOLOv5 neural network. To detect white lines we use sliding window convolutions to create "heatmaps" identifying potential line parts in images. These heatmaps undergo NMS to pinpoint line centers. The OpenCV Probabilistic Hough Transform then identifies line segments, which are filtered for length. Intersections are analyzed to detect corners, classifying them as either L-corners, T-, or X-crossings (2 or 4 L-corners respectively)

Are you planning with objects in Cartesian or image space? If you are using Cartesian space, how do you transform between the image space and cartesian space?

We plan with objects in Cartesian space. If feature comes from stereo vision, we use its position as a bottom center of point cloud cluster, if it comes from NN we use inverse perspective mapping.

How is your robot localizing?

Robot is able to localize itself on the field using Particle Filter approach. Vision system takes into account objects like white lines (very low orientation/high position ambiguity), corners (very low orientation/very low position ambiguity) and goalposts (high orientation/moderate position ambiguity) to perform fast and stable localization

Is your robot planning a path for navigation? Is it avoiding obstacles? How is the plan executed by the robot (e.g. dynamic window approach)?

Walking trajectory calculates each step using Teb Local Planner algorithm https://github.com/rst-tu-dortmund/teb_local_planner and is able to avoid obstacles such as other robots or goalposts provided by vision system.

How is the behavior of your robot's structured (e.g. Behavior Trees)? What additional approaches are you using?

We use hierarchal FSM. Each our motion has state and highlevel behaviors control lowlevel motions.

Do you have some form of active vision (i.e. moving the robots camera based on information known about the world)?

We have ball tracking

Do you apply some form of filtering on the detected objects (e. g. Kalman filter for ball position)?

Kalman filter for ball and robots got from YOLO. Size, distance and height threshold for 3d obstacles.

Is your team performing team communication? Are you using the standard RoboCup Humanoid League protocol? If not, why (e.g. it is missing something you need)?

Yes, our robots share their, ball and obstacles positions and robot current state. We use Protobuf with UDP broadcast.

Please list contributions your team has made to RoboCup

We're a new team

Please list the scientific publications your team has made since the last application to RoboCup (or if not applicable in the last 2 years).

We're a new team

Please list the approaches, hardware designs, or code your team is using which were developed by other teams.

Our walking is based on IKWalk, robot kinematics and lowlevel part of software is based on Rhoban team solutions.
We also use DXL Board from Rhoban team.

What operating system is running on your robot and which middleware are you using (for example Ubuntu 22.04 and ROS2 Galactic)?

Ubuntu 18.04 + ROS Melodic for visualization

Is there anything else you would like to share that did not fit to the previous questions?