Software Survey 2026

Team name

Kyung Hee University Biomedical Engineering Robotics (KHUBER)

Which division(s) are you applying for? If your used software differs between divisions, please fill out the survey once per division.

Small Size (height < 110 cm, weight < 15 kg)

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

At the time of submission, the software is still under active development and no software components are publicly released yet.
However, the KHUBER software stack is planned to be partially open-sourced.

Core components, including the robot control framework, simulation environment configurations, and key modules for learning-based locomotion, are planned to be released after RoboCup 2026.

The software is developed based on ROS 2, and all open-source components will be distributed via GitHub upon release.

Are you using any software developed by other teams? If so, list every component that you are reusing and the team that originally developed it.

KHUBER is not directly reusing robot control software developed by other RoboCup teams.
All robot control logic, walking control policies, and hardware interface code are developed internally by the team.

However, the system is built on top of the following publicly available, general-purpose robotics frameworks and algorithms:

- ROS 2 (Robot Operating System 2)
Used as the main software architecture and communication framework for the robot system.

- Nav2 (Navigation 2) package
Used to construct a SLAM-based navigation and localization stack.
Robot localization is performed using AMCL (Adaptive Monte Carlo Localization) to estimate the robot pose on a pre-built map.

- YOLOv8n-based visual perception pipeline
Used for object detection and segmentation to extract semantic information from the environment.
The YOLO model is employed for feature detection and semantic information extraction, and the resulting outputs are used as auxiliary inputs for robot localization and high-level behavior decision-making.
(No code developed by other RoboCup teams is directly reused.)

- Madgwick Filter
Used for IMU sensor fusion by combining accelerometer and gyroscope measurements to estimate the robot's orientation.

- MuJoCo physics simulator
Used as a dynamics simulation environment for training and validating reinforcement learning–based walking policies.

- PyTorch
Used as the deep learning framework for training and inference of reinforcement learning models.

All listed components are publicly available libraries or frameworks, and the overall robot control architecture and system integration are designed and implemented by the KHUBER team.

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

KHUBER uses a combination of simulation-generated datasets and publicly available motion and vision datasets for reinforcement learning–based humanoid locomotion research.

During the reinforcement learning process, the team generates custom simulation datasets using a MuJoCo-based physics simulation environment.
These datasets are created by recording the robot’s state, action, and reward information during simulation rollouts.
They consist of state–action–reward sequences including joint states, IMU measurements, foot contact sensor values, action commands, and reward signals.
The datasets are used for training and analyzing reinforcement learning policies and are currently used for internal research purposes.

To generate reference motions that mimic human walking patterns, KHUBER uses the AMASS (Archive of Motion Capture as Surface Shapes) dataset (https://amass.is.tue.mpg.de/).
Human walking motions are extracted from the AMASS dataset and transformed into the robot’s joint space.
These reference motions are used for reference motion tracking and for the design of reward functions in reinforcement learning.

For visual perception and robot localization research, KHUBER uses the TORSO-21 public dataset (https://github.com/bit-bots/TORSO_21_dataset).
This dataset is used to train and validate a YOLOv8n-based object detection and segmentation pipeline.
Visual information extracted from YOLO is used for environment perception and serves as auxiliary input to support AMCL-based robot localization within the ROS 2 Nav2 package.

In addition, for sim-to-real transfer, sensor log data collected from the real robot—including IMU data, joint states, and foot contact sensor measurements—are continuously recorded.
These datasets are currently used internally, and public release will be considered after further refinement.

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

KHUBER is a first-time applicant to RoboCup 2026 and has not yet published scientific papers directly associated with RoboCup activities.

The team is currently conducting research on reinforcement learning–based humanoid locomotion and embedded control, with the goal of publishing scientific results after RoboCup 2026.

Are there any other contributions you would like to share with the RoboCup community?

KHUBER aims to contribute to the RoboCup community in the following ways:

- Sharing a reproducible end-to-end pipeline for reinforcement learning–based humanoid locomotion

- Releasing modular hardware designs and embedded control architectures for small-size humanoid robot platforms

- Sharing practical design experiences and lessons learned related to simulation-to-real transfer

The team views RoboCup not only as a competition, but also as an open collaborative platform for advancing research and education in humanoid robotics.

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

KHUBER generates humanoid walking motions using a reference motion–based residual reinforcement learning approach.
Human walking data are first extracted from the AMASS motion capture dataset and converted into joint-space reference motions that match the kinematic structure of the robot.

During reinforcement learning, the policy does not directly track the reference motion.
Instead, a residual correction is learned on top of the reference motion.
This approach preserves the natural characteristics of human walking while allowing the policy to compensate for the robot’s dynamic constraints, contact conditions, and external disturbances, resulting in robust walking behavior on the real robot.

Which approach are you using to generate other motions of the robot (e.g. kicking, standing up)?

For kicking motions, KHUBER plans to adopt a similar approach to walking.
Human kicking motions are used to generate reference motions, and a residual reinforcement learning framework is applied to produce dynamically consistent and smoothly connected kicking behaviors during walking.

For standing-up motions after a fall, reinforcement learning is considered inefficient due to the large state space and sparse success conditions.
Therefore, standing-up behaviors are implemented using predefined script-based motion sequences.
Separate motion scripts are designed for forward and backward falls to ensure stable and reliable recovery.

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

KHUBER maintains both kinematic and dynamic models of the robot.
All mechanical components are designed using the 3D CAD software SolidWorks, and internal design reports are created based on the geometry, mass, and material properties of the manufactured parts.

Using this information, URDF and MuJoCo models are generated with mass distribution and inertia parameters closely matching the real robot.
This modeling process minimizes the sim-to-real gap and enables effective use of the models for reinforcement learning–based motion generation.

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

KHUBER uses an Intel RealSense D405 stereo vision camera to directly measure distances to objects in the environment.
Stereo vision provides depth information that is used for distance estimation and supports robot localization and behavior decision-making.

For object recognition and semantic perception, a YOLOv8n-based object detection and

segmentation pipeline is employed.
The detected objects and scene features are combined with depth information to construct a structured understanding of the environment.

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?

KHUBER performs planning in Cartesian space.
YOLO-based perception is used to distinguish between fixed elements of the soccer field (such as field lines and goal structures) and dynamic elements (such as robots and the ball).
The system exploits the fact that the soccer field geometry is predefined and standardized.

Feature points are extracted from fixed field elements and matched to known reference coordinates in the field map.
Image-space detections are converted into distance measurements using stereo depth information, and then mapped into the field coordinate system.
This process enables consistent transformation from image space to Cartesian space for object-level planning.

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

At present, the robot employs a passive vision approach using a fixed camera viewpoint.

The introduction of active vision, where the robot actively adjusts its camera
pose to improve perception, is planned as a future extension after the stabilization
of upper-body or head actuation.

What approach are you using to localize your robot?

Robot localization in KHUBER combines vision-based feature detection with probabilistic localization using AMCL.

First, YOLO is used to detect feature points corresponding to fixed field elements.
Relative distances to multiple detected features are used to estimate a coarse robot position

within the field.

Second, field lines are extracted through segmentation and converted into distance-based point observations.
These observations are integrated with a field map and processed using the AMCL algorithm from the ROS 2 Nav2 package to refine the robot's pose estimate.

By combining visual cues with map-based probabilistic estimation, the system achieves accurate and robust localization.

Is your team performing team communication? Which communication protocol are you using?

At the current development stage, KHUBER focuses on single-robot operation, and team communication has not yet been implemented.

For future multi-robot operation, the team plans to use ROS 2–based DDS communication to share robot states and enable cooperative behaviors among multiple robots.

What approach are you using for navigation? Are you avoiding obstacles?

Navigation and obstacle avoidance are not yet implemented in the current system.
KHUBER plans to implement navigation using the ROS 2 Nav2 stack in future development stages.

The planned system includes global and local path planning components to enable goal-directed navigation and basic obstacle avoidance in the soccer field environment.

How is the behavior of your robots structured? (e.g. Behaviour Trees)

The robot behavior is organized using a state-based structure,
centered around fundamental motion states such as walking, standing, and turning.

State transitions are determined based on posture stability using roll and pitch information from an IMU sensor, as well as the distance to dynamic objects obtained through YOLO-based object detection.

In future work, we plan to introduce a Behavior Tree or hierarchical behavior structure to systematically manage more complex soccer-specific behaviors.

Are you simulating your robot? If so, which simulator are you using and for what purpose do you use simulations?

KHUBER simulates the humanoid robot using the MuJoCo physics simulator.
Simulation is primarily used for training reinforcement learning–based walking policies, validating motion behaviors, and performing safety verification before deployment on the real robot.

By using simulation, potentially unsafe behaviors can be identified and corrected in advance, reducing the risk of hardware damage during real-world experiments.

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

The main onboard computer of the robot runs Ubuntu 22.04.
As the robot software middleware, ROS 2 Humble is used for node communication, system integration, and real-time data exchange.

Is there anything else you would like to share that did not fit any previous question?

KHUBER views RoboCup not only as a competition, but also as an experimental platform for humanoid robotics research and education.

The team plans to share design insights and practical experience gained from reinforcement learning–based locomotion, embedded control, and sim-to-real transfer processes.
These contributions are intended to be released in an open-source form in the future to support and strengthen the RoboCup humanoid robotics community.