Software Survey 2026

Team name

ZETIN

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

Large Size (height < 190 cm, weight < 80 kg)

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

parially open source. In robocup hompage.

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.

We utilize the Kalman Filter implementation adapted from the codebase of UT Austin Villa for our state estimation.

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

We utilize a custom image dataset collected in our laboratory environment to train the deep learning-based object detector (YOLO) for ball recognition. And data set is not yet public.

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

Not Applicable (New Team)

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

We demonstrate how using commercial hardware (TRON1) can help new teams enter the league faster. We are willing to share our middleware code and integration experience to help others reduce development time.

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

We use the commercial model-based (ZMP) controller provided by the TRON1 manufacturer. We interface with this controller using a custom WebSocket bridge to execute dynamic soccer behaviors decided by our high-level software.

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

For standing up, we utilize the built-in recovery routines provided by the manufacturer. For kicking, we employ walk-kick strategy where we adjust the walking velocity and upper-body posture to strike the ball, rather than using static keyframe motions.

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)?

We utilize the standard URDF model provided by the manufacturer for the lower body. For our custom upper body, we generated the kinematic and inertial model by exporting directly from our CAD design.

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

We utilize a YOLO-based deep learning model for real-time object detection combined with RGB-D depth data for distance estimation. To handle visual occlusion, we implement a predictive tracking algorithm using an Exponential Moving Average (EMA) velocity estimator to robustly track the ball even when it is temporarily out of view.

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 primarily utilize Image Space for planning, adopting a Predictive Visual Servoing strategy. Instead of reconstructing a global Cartesian map, our system directly maps the ball's pixel centroid and RGB-D depth data to the robot's velocity commands. The head aligns with the target in image space, and the locomotion controller dynamically adjusts the approach vector based on the head's yaw angle and the target's distance.

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

Yes, we implement a state-based active vision system.
Search: The head executes a sinusoidal scanning pattern to locate the target.
Tracking: Upon detection, a PD controller minimizes the pixel error to center the target (Visual Servoing).
Context-Awareness: The head pitch is dynamically adjusted based on the distance to the target to keep it within the Field of View during close-range approaches.

What approach are you using to localize your robot?

Currently, we rely on odometry data from the walking engine. For the competition, we plan to implement a Monte Carlo Localization (MCL) system based on a Particle Filter. This system will estimate the robot's pose by matching visual field features—specifically L-junctions and T-junctions of the field lines—against a known map.

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

We plan to implement team communication using the UDP/IP protocol over Wi-Fi. Our strategy includes interfacing with the GameController to receive referee commands and establishing inter-robot communication via UDP broadcasting to share ball localization data and coordinate dynamic role assignments (e.g., Striker vs. Defender).

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

We plan to implement a Potential Field algorithm for reactive navigation. Obstacles detected via YOLO and depth data will generate repulsive forces, allowing the robot to autonomously steer around them while maintaining its path toward the ball.

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

We employ a Finite State Machine (FSM) structure. The robot transitions between discrete states such as Search, Approach, and Kick based on visual detection and game context.

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

Yes, we primarily utilize Gazebo to validate our kinematic model and control algorithms. Furthermore, we plan to use both Gazebo and Webots environments to train Deep Reinforcement Learning (DRL) agents for dynamic tasks, such as push recovery.

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

Ubuntu 20.04

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