Software Survey 2026

Team name

WolverBot Kickers

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?

Our software is not OpenSource at the moment.

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.

Yes, we are using the UT Austin Villa (simulation league) code release from 2016 as the base for our robot's higher-level software. https://github.com/LARG/utaustinvilla3d

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

Yes, we use the public BitBots TORSO dataset. https://github.com/bit-bots/TORSO_21_dataset

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

None, since we are a new team looking to begin participating this year.

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

As we are a new team, there is not anything software-wise which is significantly innovative or different.

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

We are developing control policies that output joint torques given reference velocity command inputs. Low level controls are being developed primarily using a reinforcement learning environment on Isaac Lab inspired by Unitree’s RL Lab (unitree_rl_lab).

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

We will be using keyframes to generate these motions, using mostly the same methods as the UTAustin simulation code release.

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)?

Yes, we used our CAD model to create a URDF of the robot in addition to some tweaks (like replacing linkages with direct constraints).

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

We utilize a custom-trained YOLOv11 model, optimized with TensorRT on an Nvidia Jetson Orin, to achieve real-time detection of game elements like the ball and goalposts. For localization, we apply a Pinhole Camera Model that estimates depth by comparing the pixel dimensions of detected objects against their known physical sizes and the camera’s focal length. This combination allows us to robustly track targets and calculate their 3D coordinates using a single monocular camera setup.

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. Since our camera has an insignificant amount of distortion, we convert between image space and cartesian space using trigonometry and the depth estimated by our model.

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

Yes, we have a simple approach of tracking the ball as long as it is in view; otherwise, we scan around the field based on its last known location.

What approach are you using to localize your robot?

We use a particle filter for localization, based on data from computer vision and our IMU.

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

No, we are not performing team communication. We will implement this if we have time before the competition, but it is not a priority for us at the moment.

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

We use the A* algorithm for navigation and avoiding obstacles.

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

We first use data from our world model to decide a role for each robot, which is reassessed regularly. Then, each role acts according to a behavior tree. For instance, our goalie robot scans for and tracks the ball, and if the ball is outside of a certain range, it will always put itself on the line between the ball and the center of the goal.

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

We use the Robocup Soccer Simulation

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

We use Ubuntu 22.04 and ROS Humble

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