

INPUT 2022

Team Description Paper

Masaki Yasuhara, Tomoya Takahashi, Hiroki Maruta, Hiroyuki Saito, Shota Higuchi, Takaaki Nara, Keitaro Takeuchi, Yota Sakai, and Kazuki Ishibashi

Nagaoka Activation Zone of Energy, 4-1-9, Shinsan Nagaoka, Niigata, 9402127
JAPAN

Email: rcjinput@gmail.com
URL: <https://input-ssl.dev/>

Abstract. INPUT is a team participating in the RoboCup Soccer Small League (SSL). It aims to show the world the technological capabilities of the Nagaoka region of Niigata Prefecture, which is where the team members are from. For this purpose, we are working on one of the projects from the Nagaoka Activation Zone of Energy (NAZE). Herein, we introduce two robots, v2019 and v2022, as well as AI systems that will be used in RoboCup 2022. In addition, we describe our efforts to develop robots in collaboration with companies in the Nagaoka area.

1 Introduction

Team INPUT was formed in August 2018. The team is unique in the aspect that many of its members have prior experience of participation in the RoboCup Junior Soccer Open. Our team consists of some members from three junior teams: CatPot, INPUT, and CatBot. These teams have participated in world competitions, and INPUT in particular won the RoboCup 2017 Nagoya (Fig. 1). All members are graduates of the National Institute of Technology, Nagaoka College (NITNC). INPUT is one of NAZE's projects from the Nagaoka area, where NAZE and NITNC are located. Nagaoka is a manufacturing town. We are developing robots in collaboration with companies in the Nagaoka area. The RoboCup 2019 Japan Tournament held in Nagaoka City was INPUT's first official competition (Fig. 2). In the first game of the tournament, all eight robots worked and successfully passed the ball. We were able to win one game.

2 Introducing Robot Generation v2019

This chapter describes the configuration of the v2019 robot used in RoboCup 2019 Japan Open, Nagaoka. This robot was the first robot developed by team INPUT. Fig. 3 shows the overall view of the v2019 robot.



Fig. 1: World Championship in the Junior League.



Fig. 2: RoboCup Japan Open 2019, Nagaoka.

2.1 Mechanism Design

The drive system of this robot is composed of four motor units (Fig. 4). These mechanical units were designed with consideration for replaceability and ease of repair, such that the units can be easily removed by unscrewing three bolts. In this drive mechanism, the rotation is decelerated and transmitted by the internal gear attached to the omni-wheel. By using the internal gear, it is possible to reduce the distance between the motor and the rotation axis of the wheel, and thus possible to lower the mounting position of the motor and lower the center of gravity of the robot. The reduction ratio of the internal gear was approximately 3.33: 1, and the maximum tire speed was 1557 rpm.

This robot is equipped with a deformable dribble mechanism, as shown in Fig. 5. The dribble unit is composed of two parts: the motor and the roller. The roller part is movable around the rotation axis M. This is because when the ball comes into contact with the roller, the impact is released by the movement of the roller.

For the kick mechanism of this robot, we selected the Super Stroke Solenoid (S-1012SS, Shindengen Mechatronics Co., Ltd.), which has a sufficient output and a long stroke. A Teflon sheet was attached to the bottom of the contact part with the ball and was used as a guide (Fig. 6).

2.2 Circuit

The circuit of this robot consists of five parts: the main board, voltage boost circuit, motor driver for the dribbler, ball detector, and the Li-Po battery. The main board contains the motor driver for the omni-wheel and communication modules. The details are shown in Fig. 7 and Table 1.

The STM32 was developed using the hardware abstraction layer (HAL) library. An XBee 2.4 GHz band radio was used to communicate with the PC. When the XBee was unavailable, 5 GHz Wi-Fi was used to communicate with the PC. The moving motor does not have an external encoder, and the ESCON

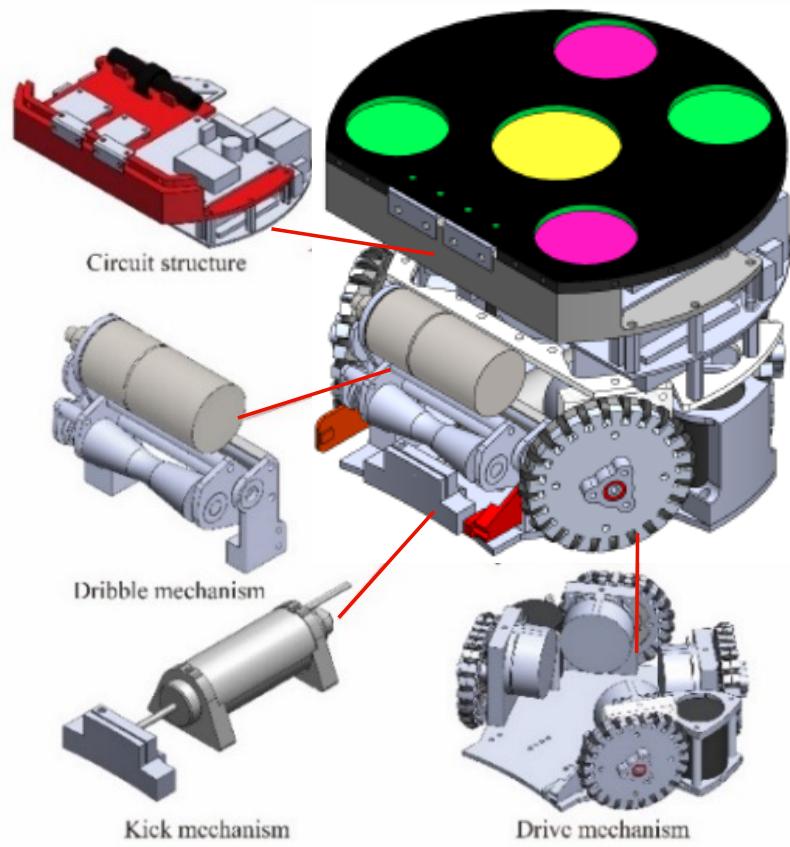


Fig. 3: Components of robot v2019.

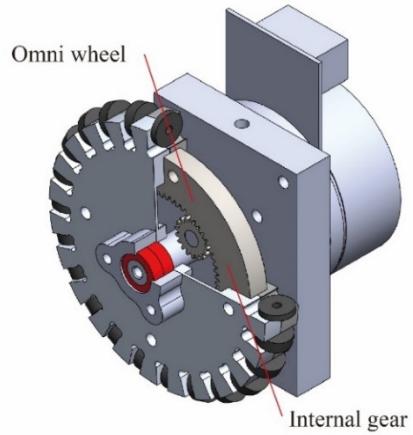


Fig. 4: Drive unit.

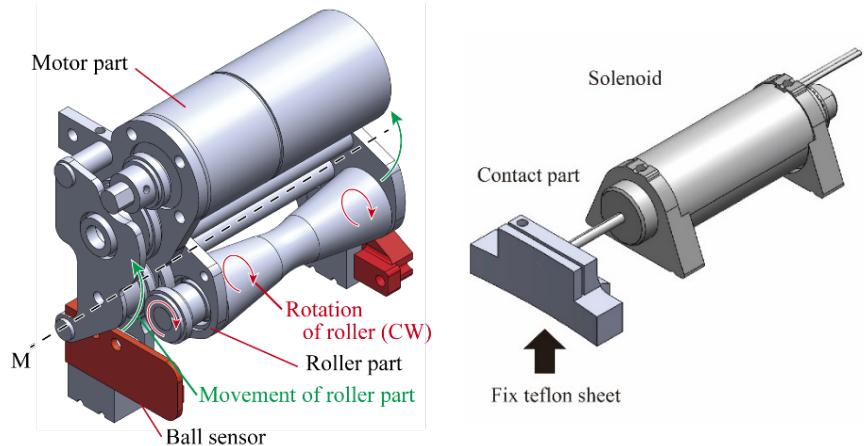


Fig. 5: Dribble mechanism.

Fig. 6: Kick mechanism

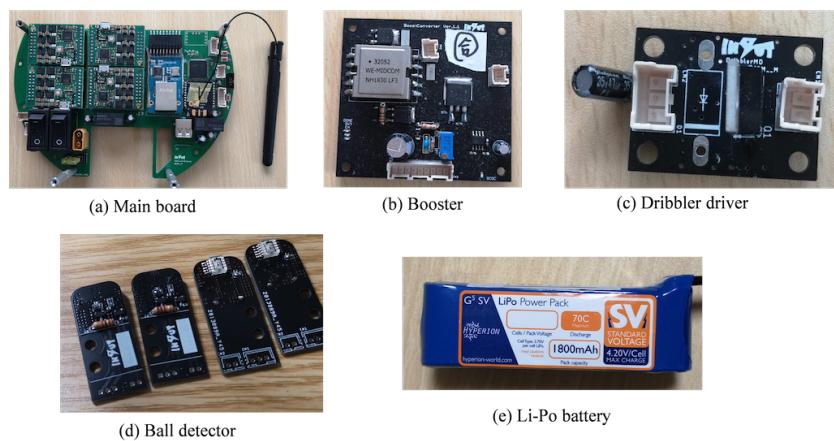


Fig. 7: Circuit components.

Table 1: List of circuit components and their details

Device	Description
Main board (Fig. 7a)	CPU is STM32F407VGT6 on 215 MHz. It contains 4 motor drivers ESCON50/5. An XBee3 is used for communication with the PC. DP83848 ethernet phi-module is included for using Wi-Fi router.
Booster (Fig. 7b)	Li-Po battery 4 cell is boosted to 175 V to use the kicker.
Motor driver for dribbler (Fig. 7c)	An Nch MOSFET (EKI04027) DC motor driver.
Ball detector (Fig. 7d)	Photo IC with optical switch functions (S8119) and Infrared LED.
5 GHz WIFI router	WRH-583BK2-S, IEEE 802.11abgn 2.4/5 GHz wireless LAN
Li-Po battery (Fig. 7e)	Hyperion LiPo 4-cell 1800 mAh

is voltage-controlled; the CPU conveys the speed of the motor to the ESCON through pulse width modulation (PWM) signals. The lack of an external encoder is due to space constraints inside the robot. The motor of the dribbler uses a single FET and can only rotate in the direction of the ball roll. The dribbler does not need to rotate in the counterclockwise (CCW) direction; hence, a very simple circuit is used to rotate only in the clockwise (CW) direction.

The voltage boost circuit of this robot is based on the voltage boost circuit of Team Roots [1], and it is boosted up to 175 V using LT3750. The kicking force is adjusted based on the switching speed of the IGBT.

The robot is equipped with a ball sensor in the dribbling mechanism, as shown in red in Fig. 5, which is used in order to determine if the ball was ready to be kicked. A ball sensor was mounted on the dribble mechanism. A photo IC (S8119) and an infrared LED were used to detect the presence of the ball in the dribbler mechanism.

The communication protocol for XBee is shown in Fig. 8. XBee operates in the AT mode with a baud rate of 115200 bps. The speeds v_x, v_y, v_θ are received as floats. The 5 GHz Wi-Fi is used to receive the User Datagram Protocol (UDP) using FreeRTOS and LwIP. The Wi-Fi router assigns a local IP address to each robot and sends a comma separated string to the specified IP address (each robot is assigned a local IP address by the router, and a comma-separated string is sent to the specified IP address (robot)). The contents of the packet are v_x, v_y, v_θ , kicker power, and dribbler power.

3 Introducing Robot Generation v2022

While the v2019 robot had sufficient functionality to work in a match, it had a high center of mass (COM) that caused the robot to nearly fall over when accel-

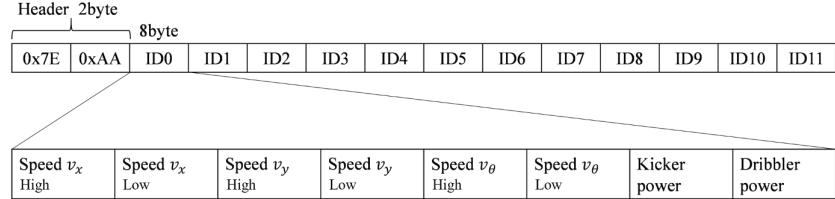


Fig. 8: Communication protocol for XBee.

erating from a stationary state. We developed v2022, which solves this problem by improving the hardware. In the following section, we explain the solutions for each problem.

3.1 Improved Wheels for a Lower COM of Robot

To lower the COM, we focused on enlarging the space closer to the ground in the configuration of the robot (ground area), as shown in Fig. 9(a). Heavy objects must be placed at a lower height to lower the COM, and sufficient space in the lower area is needed to achieve this. However, because this area is at a height where the ball can easily be touched, devices that directly touch the ball (solenoids for kicking, sensors, etc.) as well as the motor wheels for driving are placed here. In some teams, the wheel drive motor is placed at the top for space reasons, and gears are used to transmit rotation to the wheels, resulting in a high COM. Therefore, to reduce the COM, it is necessary to save space for these devices. Hence, we first made the omni-wheel thinner to increase the available space.

The bottleneck for making a thinner wheel lies in the process of fastening the wheel to the motor shaft. Many teams inserted a screw from the side of the shaft and pressed down on the shaft to fix the wheel. However, the thinner the wheel, the less fastening force is available because screws with a larger diameter cannot be used. This method also requires access to the screws on the side of the wheel when attaching and detaching it, which increases the assembly time. In Team-TIGERs [2], the shaft is fixed with glue, and the wheel is fixed with screws; however, the screws loosen due to vibration and rotation.

Therefore, we developed an omni-wheel equipped with a mecha-lock mechanism, as shown in Fig. 9(b). The mecha-lock is a mechanism to fix the shaft that generates a force in the direction of diameter contraction by tightening inner and outer tapered parts in the axial direction. The inner part has a straight hole that has the same diameter as the motor shaft and it has a slit to easily deform with changing hole diameter. The tapered shape of the inner part is the same as the tapered hole of the outer parts. Therefore, by tightening two-part, the inner parts can cause fixing force to shaft. It can also be detached repeatedly by applying a force in the axial direction with the removal screw. This enables

the use of fewer parts, high fastening force, and simple attachment and removal. The basic characteristics of the developed omni-wheels are listed in Table 2.

In v2022, the wheel was directly connected to the motor shaft to increase the space in the lower section (Fig. 9(a)). We used a Maxon motor EC 45 flat 70 W 24 V instead of the conventional Maxon motor EC 45 flat 50 W 18 V because it requires a large torque. The circuit changes due to the change in the motor are described in the next section.

Table 2: List of circuit components and their details

	v2019	v2022
Diameter [mm]	80	55
Thickness [mm]	16.5	11
Height of COM [mm]	49.2	39.7
Material	A2017	A2017

3.2 Change of battery and voltage boost circuit

The battery was changed to a 6-cell LiPo because the motor rating was changed to 24 V in v2022, as described in Section 4.1.

The v2019 booster circuit is designed for a 4-cell LiPo battery and, hence, cannot be used in v2022. In v2022, the voltage booster circuit is redesigned to allow a maximum input voltage of 40 V using LT3751 as the voltage booster control IC and an additional regulator. The input circuit to the solenoid and the adjustment of the kick force were the same as in v2019. Fig. 10 shows the v2022 boost circuit. The occupied area was reduced using a two-stage board.

4 Software

4.1 Structure

The structure of the AI software is illustrated in Fig. 11. It is divided into three major blocks: communication, strategy, and control. The system is event-driven, triggered by the frame information in the ssl-vision. The entire sequence of operations, including the transmission from the trigger to the robot is processed within one frame. First, the communication block receives the vision and referee information and sends a signal to the strategy block. This includes communication with robots. The strategy block determines the movement of the robot based on the situation of the game. The control block is responsible for path generation, acceleration/deceleration control, and yaw angle control to achieve the determined motion.

The strategy block consists of four modules. The GameManager module updates the status of the game from the vision and referee information. The strategy module determines the combination of roles assigned to the robot and the

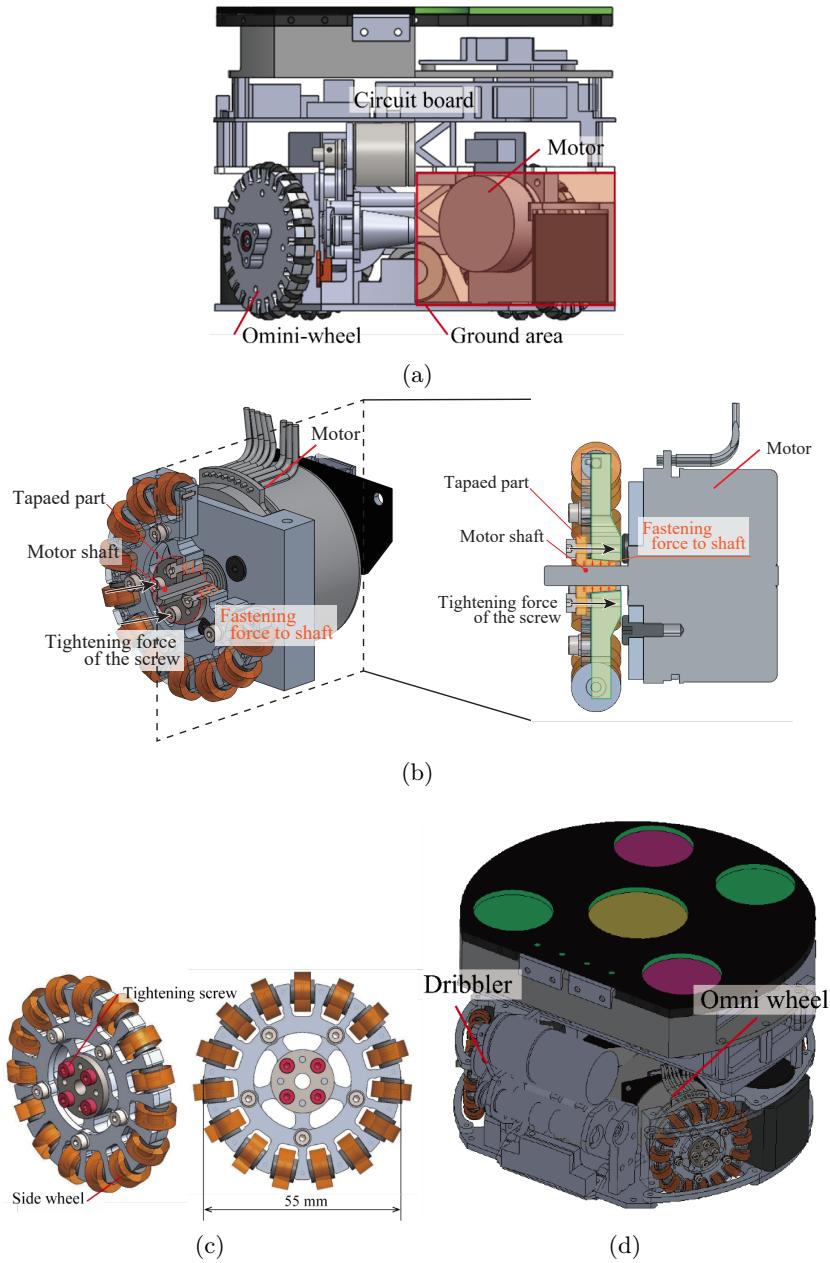


Fig. 9: v2022 robot with the new Omni-Wheel. (a) Ground area of robot, (b) Cross-section view of drive unit, (c) Wheel unit, (d) Isometric view of v2022.



Fig. 10: v2022 Booster Circuit (a) separated, (b) stacked.

priority of each role based on the situation of the match. The role module appropriately assigns roles to robots and determines the actions to be taken for each role. The skill module is a subdivision of roles and consists of the most basic descriptions.

The control block consists of the PathPlan module and the MotionControl module. The PathPlan module generates paths for collision avoidance. The MotionControl module performs acceleration/deceleration control and posture control.

The acceleration/deceleration control is based on the trapezoidal acceleration. The details are described in section 5.4. Yaw angle control was performed using PD control.

4.2 Role

One of the six roles is assigned to each robot. The number of roles assigned and their priorities depends on the situation of the match.

Goalie This role is always assigned the highest priority, and a robot assigned this role only works as a goalie.

Attacker This is a role that involves actively approaching the ball and aiming to pass, clear, and shoot; the attacker role is never assigned to more than one robot.

Defender The robots assigned this role move around the perimeter of their own penalty area to block the path of shots. The number of defenders assigned to each team varies depending on the situation of the game.

PassReceiver This is a role that involves waiting for a pass at a location where it is most likely to be received. The number of passengers assigned varies depending on the match situation.

PassInterrupter This player marks the opponent and intercepts the pass. The number of passengers assigned varies depending on the match situation.

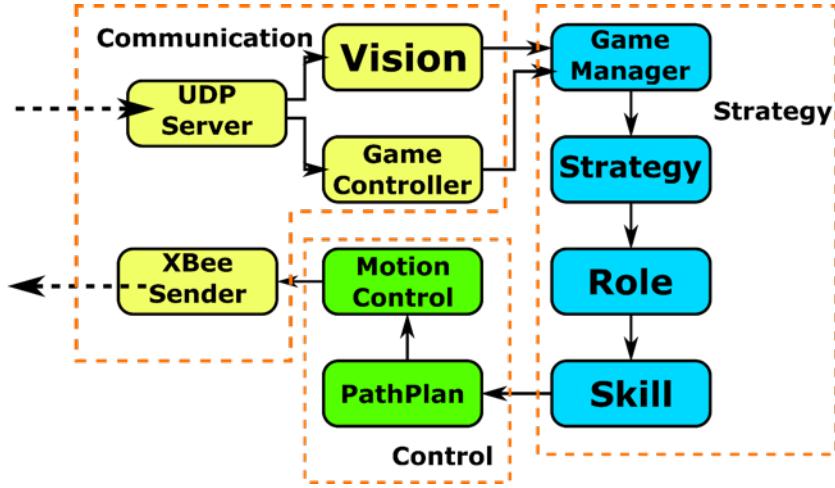


Fig. 11: Software structure overview.

Waiter This is the lowest priority role, and involves moving to a larger space for a smooth transition to another role.

4.3 Passing position

The best location for passing is determined by the potential method, with the field delimited by a grid. A detailed description is presented in TIGERs' 2017 TDP [3].

To determine the potential field, we computed the sum of several masks. A typical mask is shown in Fig. 12a, which considers the ball position as a point light source and excludes shadows cast by obstacles. The passing location is determined flexibly by computing the sum of masks with other simple gradients, as shown in Fig. 12b.

4.4 Acceleration/deceleration control

Speed control is based on the trapezoidal acceleration/deceleration. In the simple trapezoidal acceleration/deceleration, the speed is low in the low-speed range and short-distance movement. For this reason, the trapezoidal acceleration/deceleration was cut at both ends, as shown in Fig. 13.

4.5 Framework

Qt was used as the software framework. The user interface developed in Qt is shown in Fig. 14. Qt is used not only for the user interface, but also for UDP communication, serial communication for XBee, and the core "signal, slot."

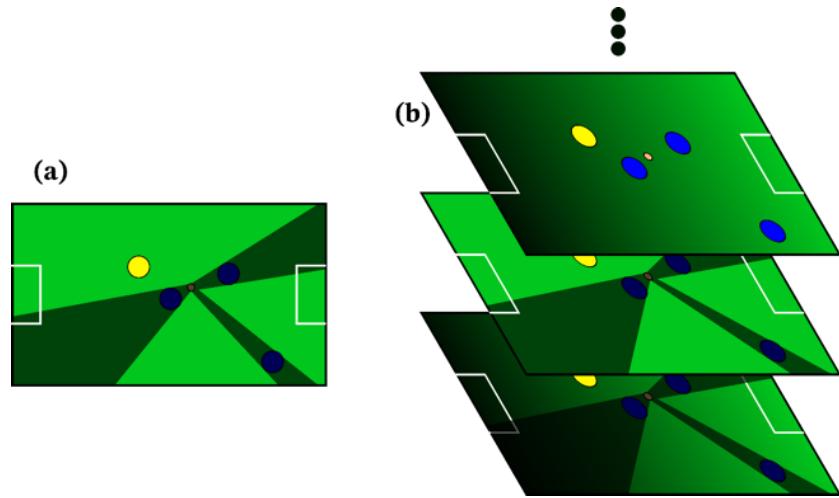


Fig. 12: Potential score mask for determining the passing position. (a) Example of a mask, and (b) applies multiple masks.

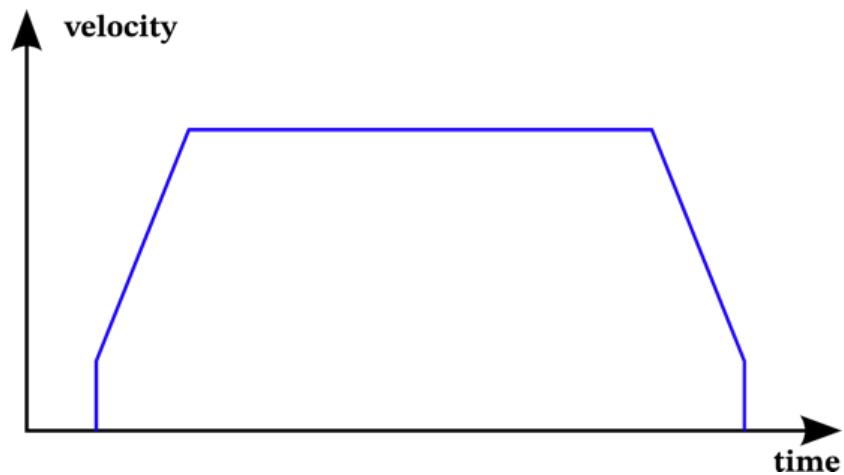


Fig. 13: Trapezoidal acceleration/deceleration with both ends cut.

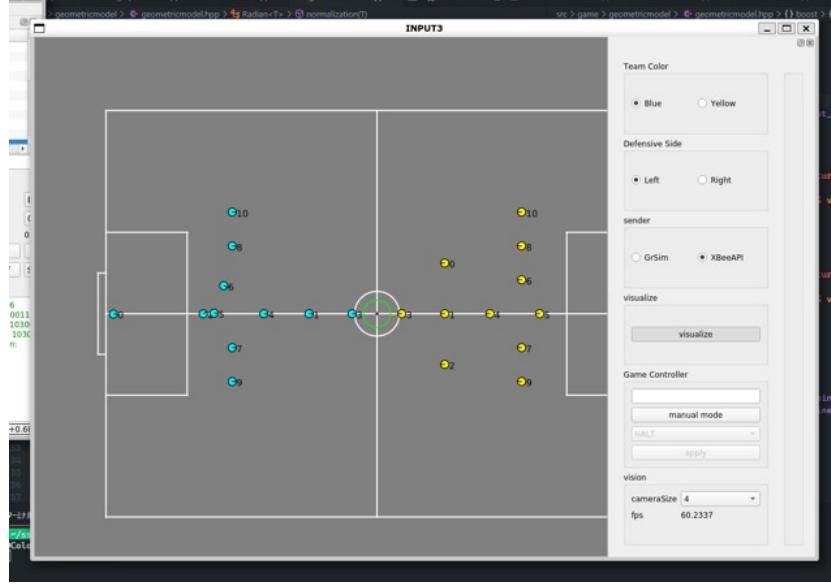


Fig. 14: Software user interface.

4.6 Development Environment

The software development was managed by GitLab. We used the GitLab CI to automatically verify software builds and tests. In addition, a system to output code documentation was configured using Doxygen.

5 Introduction of robot development in cooperation with companies in the Nagaoka area

Our team of students from Nagaoka is building a robot in collaboration with a company in Nagaoka. Fig. 15 shows a discussion between students and a company in Nagaoka. The new Omni-Wheel, described in Section 4.1, was developed through this collaboration.

We also received support from a company in the Nagaoka area for the installation of cameras in the practice field of our team. In general, SSL cameras are installed in a high position so that they can show a wide view of the field, but it is difficult to install a camera in a high position. Therefore, we set up a scaffold made of a single pipe right next to the field and set up the camera at a height of approximately 4 m as shown in Fig. 16.

6 Acknowledgement

This robot was developed with the generous support and cooperation of NAZE and its member companies, who provided funding, design guidance, parts pro-



Fig. 15: Discussions with partner companies.

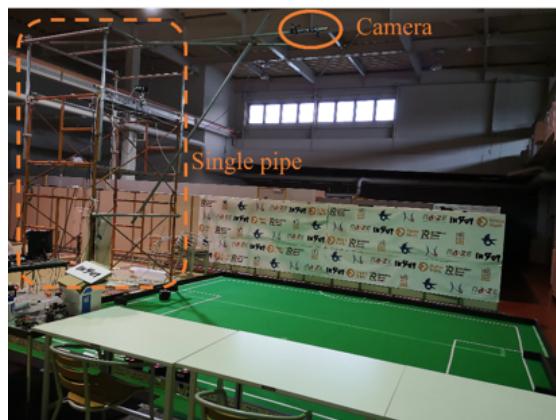


Fig. 16: Camera mounted using single pipe.

duction, and field installation. We would like to express our gratitude to Sanshin Co., NSS Co., and Ogawa Conveyor Corporation for their generous support. We would like to take this opportunity to express our deepest gratitude.

References

1. Team Roots, Detail booster PCB.
https://ssl-roots.github.io/Roots_home/roots/elec/detail_booster_pcb/ (access: 2021 Aug. 2)
2. TIGERs Mannheim, Ryll, A., & Jut, S. (2020). Extended Team Description for RoboCup 2020. RoboCup 2020, 1–16.
3. TIGERs Mannheim, Mark Geiger, Chris Carstensen, Andre Ryll, Nicolai Ommer, Dominik Engelhardt, Felix Bayer (2017). Extended Team Description for RoboCup 2017. RoboCup 2020, 1–16.