RoboDragons 2020 Extended Team Description

Masahide Ito, Mayu Nakayama, Yuta Ando, Yusuke Adachi,
Jiale Du, Ryoto Suzuki, Yuki Ono, Fusuke Kashiwamori, Kazutoshi Ban,
Shogo Isokawa, Takateru Yamada, and Tadashi Naruse

School of Information Science and Technology, Aichi Prefectural University
1522-3 Ibaragabasama, Nagakute, Aichi 480-1198, JAPAN
Email: `masa-ito@ist.aichi-pu.ac.jp` and `ssl.robodragons@gmail.com`
URL: `https://robodragons.github.io/`

Abstract. *RoboDragons* is a team of the RoboCup Soccer Small Size League from Aichi Prefectural University. Similarly to last three years, the seventh-generation robots will be used in RoboCup 2020. This paper presents main changes of our robot system from 2019 to 2020. In the hardware part, we replaced the dribbling roller to be able to apply backspin into the ball; in the software part, for a Technical Challenge, we developed a local vision system and also control algorithms.

1 Introduction

RoboDragons is a team of Aichi Prefectural University (APU) participating in the Small Size League (SSL) of RoboCup Soccer. This team originated from *Owaribito*—a joint team between APU and Chubu University—which was founded in 1997. In 2002, since two universities have been ready to manage each individual team, APU built a new team, RoboDragons. After that, RoboDragons has been participating in the SSL for more than 18 years, including activities as *CMRoboDragons*—a joint team with Carnegie Mellon University in 2004 and 2005. Our best record was the second place in 2009. We also finished twice in the third place (2007 and 2014) and four times in the fourth place (2004, 2005, 2013, and 2016). In RoboCup 2019, we placed five–six out of eight teams in Division A.

Similarly to last three years, the seventh-generation (7G) robots (Fig. 1) will be used in RoboCup 2020. This generation has developed in 2016. See our ETDP 2017 [1] for the specification of the hardware and software of the 7G robot. RoboCup 2017 was the first time for the 7G robots to compete the official SSL games with the other teams. After that, based on the issues found in some official/friendly matches and daily development, we have tried to improve the hardware and software. From 2017 to 2018, for widening the ball-touchable area of the dribbling roller, we whittled down some spaces on the side brackets of the dribbler; to improve motion control of the robots, we introduced a trajectory tracking controller based on linear model predictive control [2]. From 2018 to 2019, we changed the small wheels of the omni-wheels for their more smooth mobility and less maintenance; to increase the successful rate of ball placement

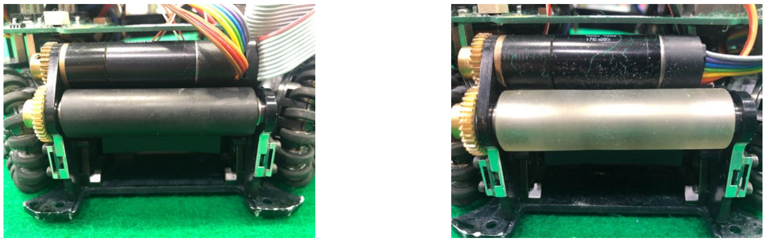
Fig. 1: The seventh-generation RoboDragons robots

starting near the wall even if the dribbler does not work for keeping the ball, we introduced a skill to kick a ball to the wall diagonally [3].

This paper summarizes the technical information of RoboDragons 2020, which includes the main changes from 2019 to 2020. There are a few updates of this year for solving a technical issue and for trying Technical Challenges. In the hardware part, we changed the material of the dribbling roller to be able to apply backspin into the ball for keeping it; in the software part, we developed a local vision system and also control algorithms for last SSL-Vision Blackout Challenge.

2 Change of the Dribbling Roller and Its Evaluation

The development of the dribbler has been attracting attention since OP-AmP has introduced an auto centering roller [4]. In fact, as two-year results of ZJUNlict [5,6] have showed, a function to strongly keep a ball by using the dribbler is important to dominant a game. The dribbling roller that we used in the previous year, however, did not work to apply backspin into a ball. The reason is that the surface of the dribbling roller is very slippery against the ball. One of the solutions is to change the material of the dribbling roller. This year we changed it from "non-repulsive rubber (Shore A70)" (Fig. 2 (a)) to "Ester Polyurethane (Shore A50)" (Fig. 2 (b)).



(a) A dribbler w/ a non-repulsive roller (b) A dribbler w/ an urethane roller

Fig. 2: A dribbling roller and a dribbler

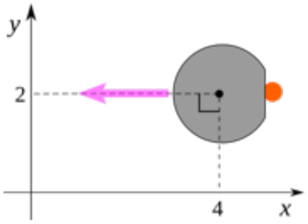


Fig. 3: The initial configuration in the experiments

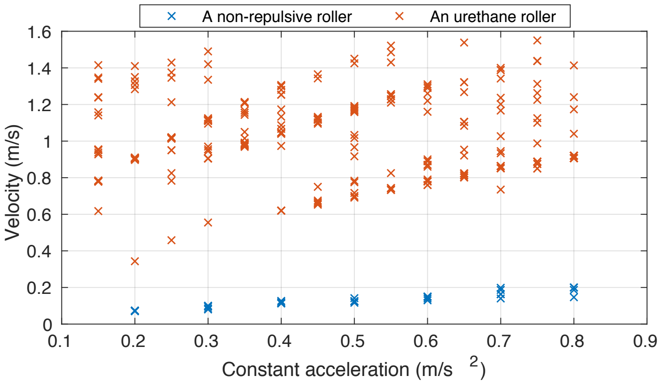


Fig. 4: Ball-leaving velocity against linear motion of constant acceleration

The dribbling roller can be evaluated by how much backspin to the ball is applied, how much shock from a coming ball is reduced, and so on. To compare the performance of the new and old rollers, we evaluated the effectiveness of backspin by the following experiment.

(1) Place a robot at $(4\,\mathrm{m}, 2\,\mathrm{m})$ on the world frame and also set the orientation of the robot to $0\,\mathrm{rad}$ with respect to the x-axis (Fig. 3).
(2) Apply backspin to the ball while keeping the initial configuration.
(3) Move back the robot parallel to the x-axis at a constant acceleration a_0 when one second passes, where the value of a_0 is chosen from $0.15\,\mathrm{m/s}^2$ to $0.8\,\mathrm{m/s}^2$.
(4) Record the value of the translational velocity when the ball left from the robot.

Figure 4 summarizes experimental results. The x-axis indicates the values of the constant acceleration; the y-axis indicates the values of the velocity when the ball left from the robot. Figure 4 shows that the new dribbling roller of Urethane works better than the non-repulsive one. As for the non-repulsive roller, we can find that the ball-leaving velocity is a proportional relationship with the constant acceleration; as for the Urethane roller, it is hard to find out a certain relationship from the data.

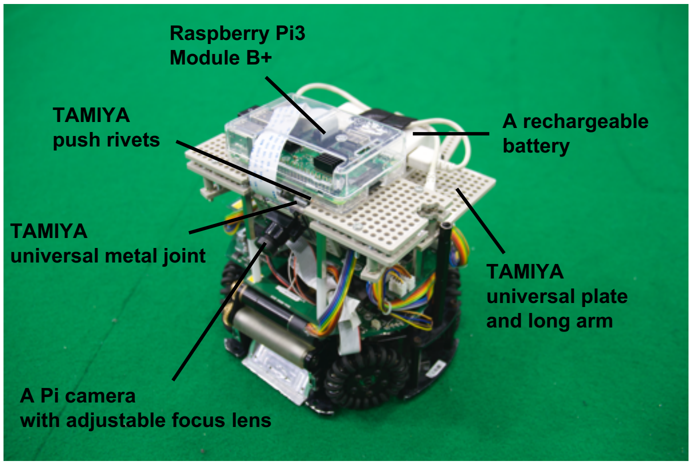


Fig. 5: A 7G RoboDragons robot with a local vision system

3 A Control Algorithm Based on Local Vision for SSL-Vision Blackout Challenge

SSL-Vision Blackout Challenge [8] is one of SSL Technical Challenges in RoboCup 2019. The SSL focuses on fast-paced soccer games with actual omni-wheeled mobile robots. Their autonomous soccer games are maintained by a global vision system called as *SSL-Vision*, wireless network, and a centralized system. There, however, is a technical issue that the game has to be interrupted if the global vision runs into any trouble; as taking the ultimate goal of RoboCup into account, it would not be special to shift the global vision system to a local one or to merge the global vision with a local one. Furthermore, Vision Blackout Challenge [8] was composed of the following two challenges:

- **1st challenge: Grab a stationary ball**
  Find and grab a ball within a square area of 1 m on each side.
- **2nd challenge: Intercept a moving ball**
  Intercept a ball passing through an area within 0.5 m around a robot at a speed of 6.5 m/s or less.

For these challenges, only local vision can be used to detect the ball and to control the robot.

3.1 Development of a local vision system

The SSL rules [7] allow participators use local vision under some restriction, but almost all teams have not been used so far for the main soccer competition to the best of the authors' knowledge. Our local vision system is mainly composed of a Raspberry Pi3 Module B+ and a Pi camera with adjustable focus lens as

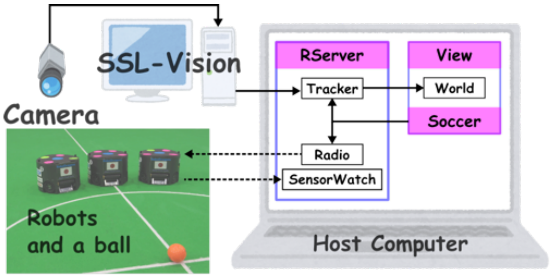


Fig. 6: Overview of the software system with global vision [3]

shown in Fig. 5. Note that the lens can be replaced with fisheye lens if we want to enlarge the field of view. There is no enough space inside the cover. Hence, we took the cover, and created a mounter by using Tamiya's universal plate, long arms, and so on. A Raspberry Pi, a Pi Camera, and a rechargeable battery are fixed on the mounter.

On the local vision system, a series of image processing extracts an orange ball from a captured (or distortion-corrected) image as a feature point (u, v). The main process is 1) to detect the orange domain of the image in the HSV color space, 2) to binarize the image by the orange and non-orange domains, and 3) to calculate the area of the orange domain and its center of mass. The depth z of the ball cannot be normally measured from an image of a single camera. We here calculate the estimated depth $\hat{z}$ by associating the area of the ball on the image plane with the actual depth z in advance. We implemented the image processing in Python with OpenCV [9]. The obtained data of (u, v) and $\hat{z}$ can be used for controlling the robot.

Under SSL-Vision, RoboDragons have a structure of the software system as in Fig. 6. The three main modules play the following roles, respectively:

RServer This module receives the data from *SSL-Vision*, and then the Kalman filter in the *Tracker* submodule estimates the states of a ball and robots. The estimated states are shared among all modules. Rserver sends a command packet to all robots via the *Radio* submodule; the *SensorWatch* submodule receives the information from the robots.

View This module gives a graphical user interface where a human operator can know and also can command the game situation.

Soccer This module chooses the best strategy for a given situation, assigns each robot a role based on the chosen strategy, and computes velocity commands to perform the role for each robot.

The host computer can bidirectionally communicate with each robot through packets between the *Radio* and *SensorWatch* submodules. See our ETDP 2017 [1] for the details.

We set a basic policy of development for SSL-Vision Blackout Challenge as exploiting the above-mentioned software system maximally. Based on it, we

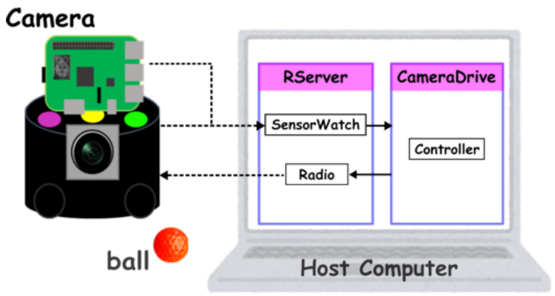


Fig. 7: Overview of the software system with local vision

designed the software system with local vision as shown in Fig. 7. The data of (u, v) and $\hat{z}$ is sent as a packet from the Raspberry Pi to the host computer by wireless communication.

3.2 A control algorithm based on local vision

For the 1st and 2nd challenges, we can design a controller mainly on the basis of image-based visual servoing [10]. Thanks to the camera mounting and the omni-directional motion (see Fig. 5), we can simply associate the horizontal motion on the image plane with changing the robot's orientation (angle). The vertical motion on the image plane can be also associated with the robot's forwarding/backwarding. The use of these relationships does not require a Jacobian matrix between the feature point velocity and robot velocity command to derive a visual servo controller. Therefore, we designed the following control algorithm for the 1st challenge:

Step 1 Turn the robot to the left direction. Stop to turn if detecting a ball (i.e., obtaining (u, v) and $\hat{z}$).

Step 2 Let $u^\star$ be the u component at the center of the image plane. Control the turning angular velocity of the robot, ω, by the following controller so that u converges to $u^\star$:

$$\omega[i] = k_u(u[i] - u^\star), \tag{1}$$

where i and k_u are discrete time and a tuning parameter, respectively.

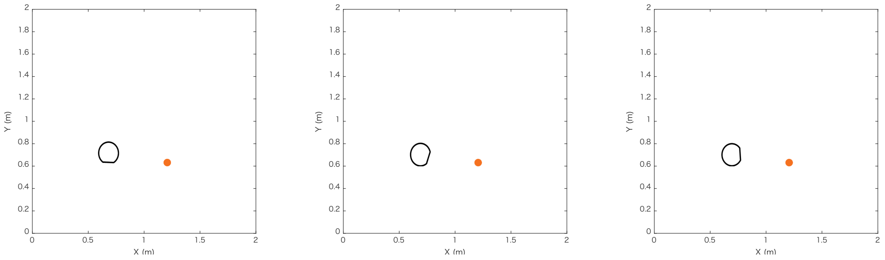
Step 3 Move the robot forward at a forwarding velocity v_x, where the value of v_x is linearly decreased from a certain initial value depending on $\hat{z}$.

Step 4 Stop the robot when the IR sensor responds.

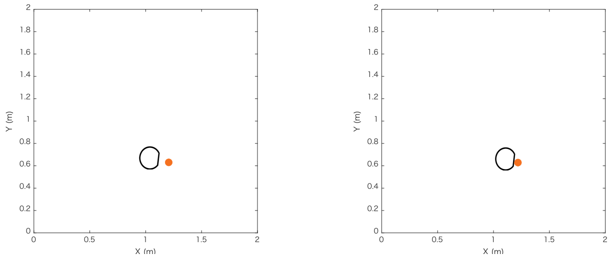
For the 2nd challenge, we designed the following control algorithm:

Step 1 Let $u^\star$ be the u component at the center of the image plane. Control the translational velocity of the robot's side direction v_y so that u converges to $u^\star$ by using the following equation:

$$v_y[i] = k_u(u[i] - u^\star), \tag{2}$$



(a) at $t = 0.000\,\mathrm{s}$ (b) at $t = 0.883\,\mathrm{s}$ (c) at $t = 1.167\,\mathrm{s}$



(d) at $t = 3.067\,\mathrm{s}$ (e) at $t = 3.817\,\mathrm{s}$

Fig. 8: Motion history of robot on the field

where k_u and k_{uc} are tuning parameters, respectively.

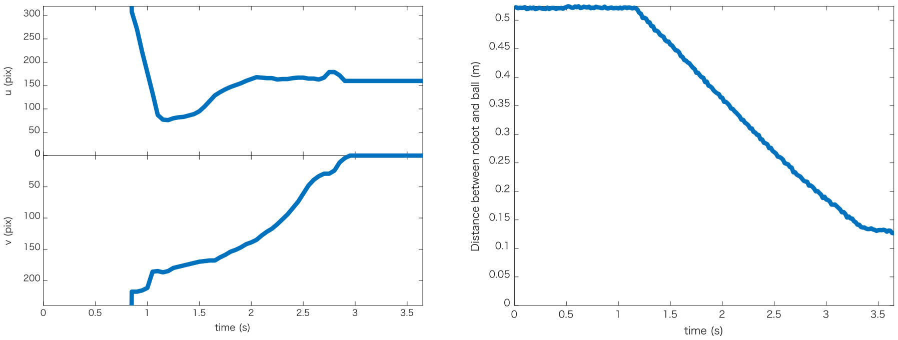
Step 2 Stop the robot when the IR sensor responds.

3.3 Experimental verification

To verify the effectiveness of the control algorithms, some experiments were performed.

Figures 8 and 9 show the experimental results for the 1st challenge. Note that the camera with adjustable focus lens was used in this experiment. Figure 8 insists that the control algorithm achieved the 1st challenge. The robot and ball were initially placed at $(0.6915, 0.7181)$ and $(1.207, 0.6318)$, respectively. The robot started to explore a ball (Fig. 8 (a)), detected it (Fig. 8 (b)), approached to it (Figs. 8 (c) and (d)), and finally touched it (Fig. 8 (e)). Figure 9 depicts time responses of two data during the experiment: the coordinates of the feature point (u, v) and the distance between the robot and ball $\sqrt{(x_r - x_b)^2 + (y_r - y_b)^2}$, where (x_r, y_r) and (x_b, y_b) represents the positions of the robot and ball, respectively.

An experiment for the 2nd challenge was performed as follows. There were a controlled robot and a ball. The distance between them was about $5\,\mathrm{m}$. Another robot kicked the ball to pass the area within $0.5\,\mathrm{m}$ around the controlled robot. The initial velocity of the ball was about $6\,\mathrm{m/s}$, and the velocity when approaching the robot was about $4.2\,\mathrm{m/s}$. Note that the camera with fisheye lens was used in this experiment. Figures 10 shows the experimental result. Figure 10 insists



(a) the coordinates of the feature point (b) the distance betw. the robot and ball

Fig. 9: Time responses

that the control algorithm failed the 2nd challenge. The robot moved with some delay against a moving ball. It can be considered that the controller (2) needs motion compensation of the target object such as in [11] and [12].

4 Concluding Remarks

This paper has presented the technical information of RoboDragons 2020. The main changes described in this ETDP are to change the material of the dribbling roller and to develop control algorithms based on local vision for last SSL-Vision Blackout Challenge.

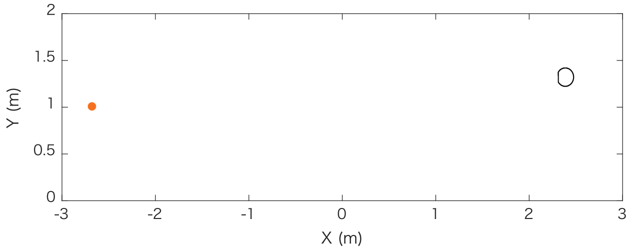
Acknowledgement.

This work was supported by The Nitto Foundation, JSPS KAKENHI Grant Number JP16K00430, Aichi Prefecture, and Aichi Prefectural University.

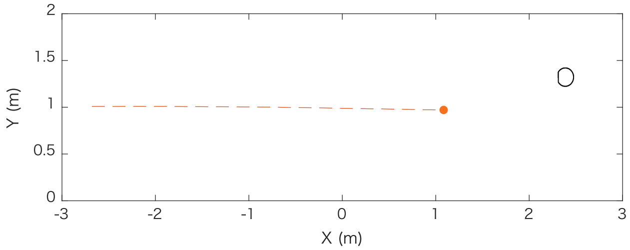
References

1. Adachi, Y., Kusakabe, H., Suzuki, R., Du, J., Ito, M., and Naruse, T.: “RoboDragons 2017 extended team description,” RoboCup Soccer Small Size League, 2017.
2. Ito, M., Kusakabe, H., Adachi, Y., Suzuki, R., Du, J., Ando, Y., Izawa, Y., Isokawa, S., Kato, T., and Naruse, T.: “RoboDragons 2018 extended team description,” RoboCup Soccer Small Size League, 2018.
3. Ito, M., Suzuki, R., Isokawa, S., Du, J., Suzuki, R., Nakayama, M., Ando, Y., Umeda, Y., Ono, Y., Kashiwamori, F., Kishi, F., Ban, K., Yamada, T., Adachi, Y., and Naruse, T.: “RoboDragons 2019 extended team Description,” RoboCup Soccer Small Size League, 2019.

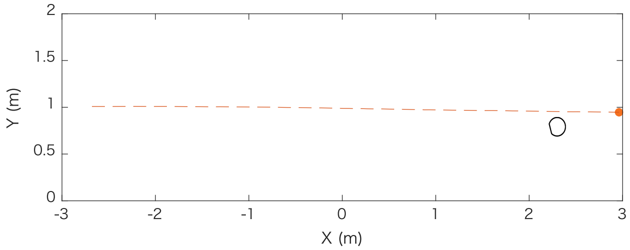
4. Yoshimoto, T., Horii, T., Mizutani, S., Iwauchi, Y., Yamada, Y., Baba, K., and Zenji, S.: “OP-AmP 2017 team description paper,” RoboCup Soccer Small Size League, 2017.
5. Chen, L., Jin, L., Lon, C.W., Wen, L., Gu, J., Li, J., Fang, T., and Xiong, R.: “ZJUNlict extended team description paper for RoboCup 2018,” RoboCup Soccer Small Size League, 2018.
6. Huang, Z., Chen, L., Li, J., Wang, Y., Chen, Z., Wen, L., Gu, J., Hu, P., and Xiong, R.: “ZJUNlict extended team description paper for RoboCup 2019,” RoboCup Soccer Small Size League, 2019.
7. RoboCup Soccer Small Size League: “Rules of the RoboCup Small Size League 2019,” 2020-01-20 version (accessed on 21 Jan 2020); `https://robocup-ssl.github.io/ssl-rules/sslrules.html`
8. RoboCup Soccer Small Size League Technical Committee: “SSL-Vision Blackout Technical Challenge,” 2018-10-16; `https://github.com/RoboCup-SSL/technical-challenge-rules/releases/download/2019-v1.0/ssl-vision-blackout-technical-challenge.pdf` (accessed on 2020-01-21)
9. Bradski, G.: “The OpenCV library,” *Dr. Dobb’s Journal of Software Tools*, 2000.
10. Chaumette, F., Hutchinson, S., and Corke, P.: “Visual servoing,” *Handbook of Robotics*, 2nd edition, Chapter 34, pp. 841–866, Springer, 2016.
11. Ito, M. and Shibata, M.: “Visual tracking of a hand-eye robot for a moving target object with multiple feature points: Translational motion compensation approach,” *Advanced Robotics*, Vol. 25, No. 3, pp. 355–369, 2011; doi:10.1163/016918610X552150
12. Ito, M., Hiratsuka, T., and Shibata, M.: “Image-based visual target following for a nonholonomic wheeled mobile robot with a single fixed camera,” *IEEJ Transactions on Electrical and Electronic Engineering*, Vol. 7, No. S1, pp. S160–S166, 2012; doi:10.1002/tee.21818



(a) at $t = 0.000\,\mathrm{s}$



(b) at $t = 0.800\,\mathrm{s}$



(c) at $t = 1.333\,\mathrm{s}$

Fig. 10: Motion history of robot on the field