Tech United Eindhoven Team Description 2022

Elise Verhees, Lars Nijland, Isabelle Franklin, Karin van Minnen, Jorrit Olthuis, Ruben Beumer, Wouter Kuijpers, Ainse Kokkelmans, Chuck Steijlen, Danny Hameeteman, Dennis Bruijnen, Emre Deniz, Ferry Schoenmakers, Harrie van de Loo, Joep Selten, Johan Kon, Koen Meessen, Matthias Briegel, Patrick van Brakel, Peter Teurlings, Peter van Lith, Ruud van den Bogaert, Stefan Kempers, Wouter Aangenent, Wouter Houtman, Yanick Douven, Aneesh Deogan, Javi Olucha Delgado, Youssof Nounou, and René van de Molengraft

Eindhoven University of Technology,
De Rondom 70, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
techunited@tue.nl,
Home page: www.techunited.nl

Abstract. The Tech United Eindhoven Middle-Size League (MSL) team is a five times world-champion, and achieved respectively a first and second place in the technical and scientific challenge of (online) RoboCup 2021. In the past year, despite two years of cancelled tournaments, the team has made a considerable amount of developments. For the sake of qualification for RoboCup 2022, this paper describes the most notable of those developments: shooting less predictable balls to score more goals, catching balls by the goalkeeper to increase ball possession, and simulating matches to realize more practicing time against other teams.

Keywords: RoboCup Soccer · Middle-Size League · multi-robot · effect balls · catching goalkeeper · simulating matches

1 Introduction

Tech United Eindhoven represents Eindhoven University of Technology (TU/e) in the RoboCup competitions. The team started participating in the Middle-Size League (MSL) in 2006, and achieved a spot in the final for twelve years in a row. The first place was achieved five times: in 2012, 2014, 2016, 2018 and 2019. At the moment of writing, the MSL team consists mainly of students (PhD, MSc and BSc), staff members and alumni of TU/e. This paper describes the major scientific improvements of the team over the past year. First, in Section 2, an introduction to Tech United's fifth generation of soccer robots is given. Then, Section 3 introduces a new mechanism to shoot a ball with effect. Next, Section 4 describes the efforts made to find a method that allows the goalkeeper to catch the ball. Furthermore, Section 5 describes the implementation of an architecture in which the existing simulation software of two teams can be used to realize simulated matches in MSL. Lastly, Section 6 gives some concluding remarks.

2 Robot Platform

Tech United's soccer robots are called TURTLEs, an acronym for "Tech United RoboCup Team: Limited Edition". Currently, the fifth generation hardware TURTLE is used; see Section 2 of the description paper of 2014 [1] for a schematic representation covering the main outline of the robot design. A detailed list of hardware specifications, along with CAD files of the base, upper-body, ball handling and shooting mechanism, can furthermore be found on the ROP Wiki[1]. The software[2] controlling the robots is divided into four main processes: Vision, Worldmodel, Strategy and Motion. More detailed information on this can be found in previous team description papers [5,6].

3 Towards the Shooting of Effect Balls

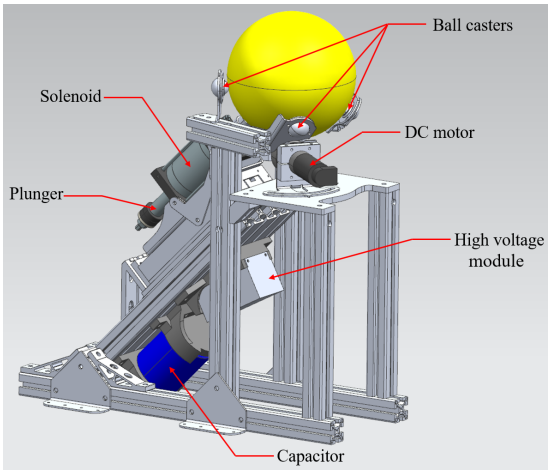
The field of robotics is advancing and soccer robots, especially goalkeepers, are improving their ball-blocking skills. Tech United thus aims to shoot balls in a way that makes it hard to predict where the ball will enter the goal, preventing the opponent's goalkeeper from blocking the shot. This section elaborates on the introduction of a new mechanism, that will allow the TURTLEs to intentionally shoot a ball with effect, leveraging models describing the ball's impact dynamics.

If the ball has a side spin when it hits the ground, it will obtain linear velocity components out of the plane of movement, causing the ball to bounce sideways. To calculate how the ball should be shot to obtain the desired bounce-effect, its initial spin and linear velocity need to be calculated. The equations for the path of a bouncing ball are derived and validated in [7], and are a function of these parameters. When the coordinates of the first bounce and the target of the ball are known, the equations can be inverted and solved by means of optimization, to find the spin and velocity. Here, the spin is minimized because the higher the spin, the more complicated it is to realize.

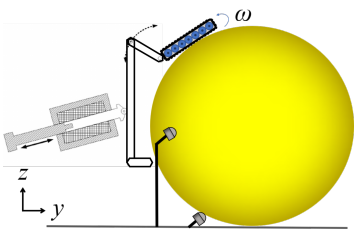
The current shooting mechanism of the TURTLEs it not capable of shooting an effect ball, because it cannot deliver the required spin and velocity in one moment of impact. Therefore, a test setup that decouples the spin and linear velocity was built, of which a schematic can be seen in Figure 1a. The ball is rotated by a separate mechanism, a small DC motor below the ball, and it is shot with the solenoid that the TURTLEs normally use. In Figure 2, some measurements from shots from the test setup are plotted. Here, it can be seen that it is possible to achieve the desired bounce angle and target by using the right amount of spin and linear velocity, with only a small deviation between the predicted and actual bounce locations.

In the future, the working principle of the test setup should be integrated in the TURTLEs. The rules set by the RoboCup federation should be adhered to [2]. The ball may thus not be lifted off the ground and may not be held over one third of its surface. However, it is allowed to exert forces on the ball that

[1] `http://www.roboticopenplatform.org/wiki/TURTLE`
[2] `https://gitlab.tue.nl/tech-united-eindhoven`



(a) 3D view of the test setup



(b) Giving spin using a small rotating belt in blue

Fig. 1: Setups for shooting effect balls

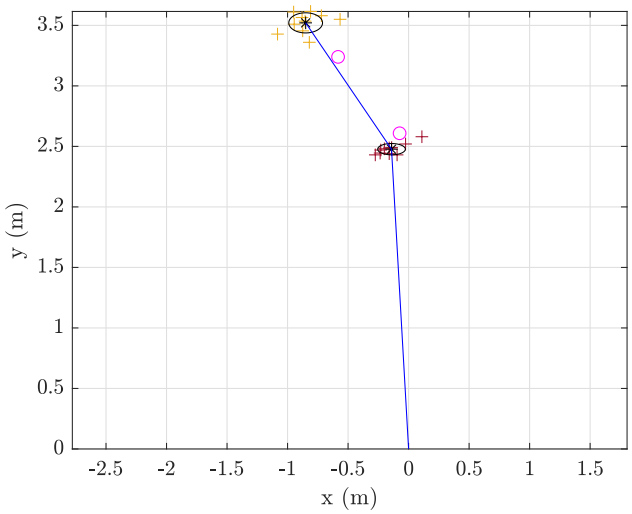


Fig. 2: Measurement of eight effect shots with the same settings. The predicted bounce locations are represented by ○, the actual first and second bounce locations are indicated by respectively + and +, and the path of the ball is represented by (—) . The mean actual bounce locations are shown as ∗. The spin of the ball is 35.3 rad/s and the duty cycle is 22.2%.

hinder it from rotating in its natural direction of rotation, for a duration of one second. An idea for realizing this on a TURTLE is presented in Figure 1b. Here, a cross-section of the ball handling system is depicted, with the solenoid, the ball, ball casters and a small rotating belt.

4 Catching Goalkeeper

The goalkeeper of Tech United has proven to be a reliable blocker throughout previous competitions. However, a big difference with human soccer players is that the TURTLE is not able to catch the ball. Due to this deficiency, the ball bounces back in a seemingly random direction after blocking. For example outside the field or towards an opponent. To maintain ball possession, the goalkeeper could be optimized with a ball catching system that, after catching, can pass the ball towards a specific team player. This section focuses on finding a method that allows the goalkeeper to catch the ball.

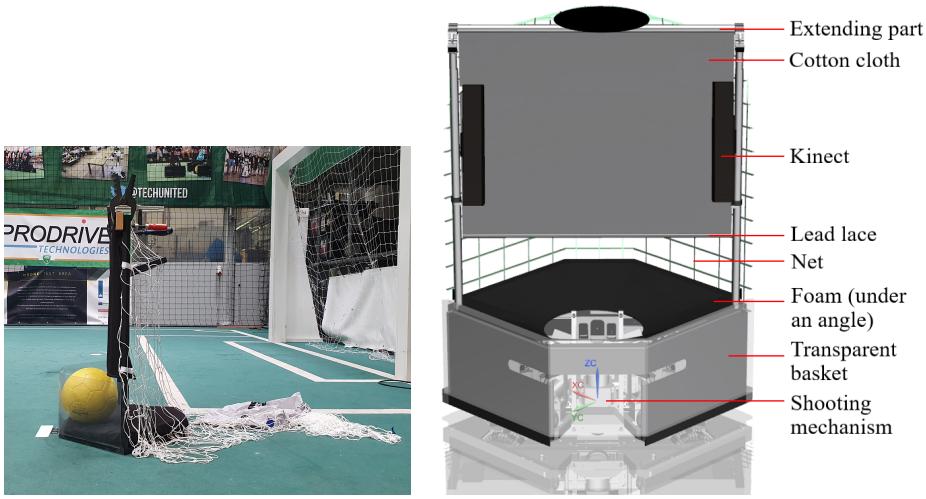
To be able to catch a ball, the kinetic energy of the arriving ball should be dissipated. This kinetic energy needs to be converted within a very short duration. This requires a (controlled) large force, otherwise the ball will bounce back before the energy is dissipated. By increasing the interaction time between the robot and the ball, more time to convert energy is available. This could be realized by receiving the ball using a fabric under a slack tension. Inspired by the rugby robots of the University of Tokyo RoboTech[3], the decision was made to use respectively a cloth and a net at the front and the back of the TURTLE, which are used to dissipate most of the kinetic energy of the ball before catching the ball in a basket.

The cloth at the front is hanging loose and is only fixed at the top. Experiments showed that adding mass to the cloth, especially at the bottom, is the most effective way to decrease the velocity of the ball. Therefore, a lead lace is added. The cloth gives the ball an upwards motion, due to the fact that the ball is not immediately able to separate itself from the cloth, reducing the velocity of the ball. Next, the net at the back, which is fixed all around the goalkeeper's frame, is used to bounce the ball into the basket. The net's tightness has influence on whether this is successful: the ball should not be caught in the net (too much slack), but it should also not bounce outside the basket (too tight). The latter can happen either directly when the ball is bouncing out of the net (bounce too far) or after bouncing in the basket (bounce high enough to leave the basket again). The tightness of the net is determined experimentally.

During a first set of experiments, combining the cloth and the net resulted in a catching percentage of 82.6% (out of 26 shots). The ball always bounced in the basket, but sometimes bounced too high within the basket, and left it again. Therefore, during a second set of experiments, a foam layer was added at the bottom of the setup, to decrease the ball's energy and prevent this from happening. The final setup, see Figure 3a, achieved a catch percentage of 100%

[3] `https://tuk.t.u-tokyo.ac.jp/robotech/?p=1260`

(out of 39 shots). It can be concluded that this concept, with only a mass of 0.96 kg, has proven itself to be useful, while being cheap, light and uncomplicated.



(a) Final test setup with a caught ball (b) Ball catching goalkeeper

Fig. 3: Ball catching design

To implement the system in the present goalkeeper, a system for an upwards moving basket should be made, the Kinect cameras should be re-positioned, the OmniVision camera should be removed, and a way to transport the ball from the basket towards the shooting mechanism should be designed. The adjusted goalkeeper is shown in Figure 3b and works as depicted in Figure 4.

5 Realizing Simulated Matches in the RoboCup MSL

As in human football, practice is of utmost importance in order to improve in the RoboCup MSL. Ideally, practice would include playing matches against other teams regularly. However, in reality matches are often only played during tournaments. Additionally, testing on real robots is costly, difficult and time-consuming due to (i) the highly dynamic environment the robots operate in and (ii) the fact that the teams are based worldwide. Simulation of matches is thus highly valuable in order to stimulate progression in the MSL. An example of software to realize this is Simatch, as introduced in [8]. However, in Simatch both teams are obliged to use the same simulation software. This forces them to abandon their own work, which is customary made for their robots, and therefore most likely a better fit. Therefore, based on a series of meetings with various MSL teams (RobotSports, Falcons, CAMBADA and Tech United Eindhoven), an architecture was defined in which the existing simulation software (simulator

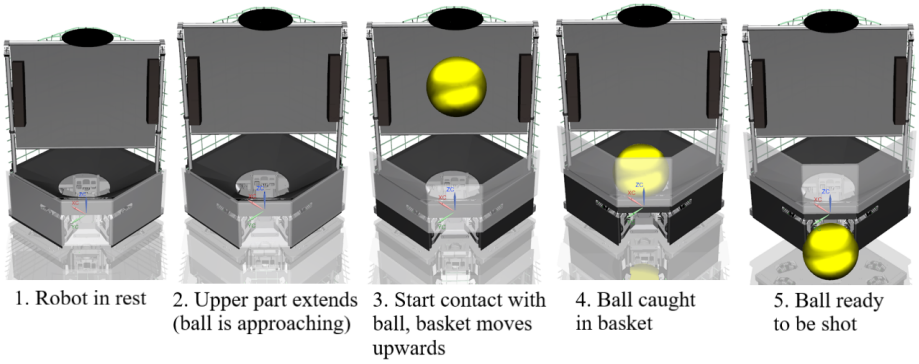


Fig. 4: Visualization of the working of the new designed robot in five stages

and visualizer) of two opposing teams are connected by means of an application referred to as middleware. The most important requirements behind this architecture are that (i) it should cost minimal effort for the teams to participate, (ii) the teams can use their own visualizer and (iii) simulations should not depend on the availability of another team. This section elaborates on this architecture.

Communication and Time Synchronization Protocol
The solution consists of two protocols: a communication and a time synchronization protocol. The communication protocol defines what information, e.g. robot and ball information, the teams must share with the middleware. The format that is defined only uses data that is readily available, because it is based on the data structure for world model sharing as proposed during the MSL Workshop of 2015, Aveiro, Portugal [3]. This minimizes effort to participate. Furthermore, the communication protocol is equipped with a timestamped data buffer with linear interpolation capability. Multiple sets of information of each team with a corresponding timestamp representing the simulation time at which the information was acquired are stored. The reason for this is that the simulators of both teams can advance through time at different rates. Moreover, since the robots and ball can attain high velocities, the corresponding information loses its relevance quickly.

The time synchronization protocol uses the aforementioned timestamps to ensure that both teams stay closely synchronized with respect to simulation time. If the difference in simulation time exceeds a certain threshold, the simulation software of the team that is ahead in time is paused. Once the other team has caught up and the difference in simulation time is below another threshold, the paused simulation software can continue.

Final Architecture
The solution chosen relies on teams sharing a binary version of their simulation software. Therefore, it is important that, regardless of the combination of the OS and programming language used, a team can access and run the software of

another team easily. Docker[4] is used for this. Teams produce a Docker image that includes everything, i.e. executables, scripts and dependencies, needed to run their simulation software. Other teams can easily access this Docker image and run it as a Docker container. The middleware will also be run from within such a container.

Communication between the application within various Docker containers is achieved by means of a TCP/IP socket connection. Additionally, since teams use different programming languages, all information is shared in JSON format. To facilitate the communication with the middleware, software has been written that can be easily integrated into the existing software of the teams. The teams only need to fill in where this piece of software can find the information that is sent to the middleware and where it should store information on the opponents robots obtained from the middleware.

The final architecture is shown in Figure 5. Here, two additional Docker containers, including two pauser applications, are shown. These pausers execute the time synchronization protocol upon command of the middleware and are required if the various containers are running on multiple computers. It is expected that this will be the case, since running the simulation software of two teams is computationally expensive.

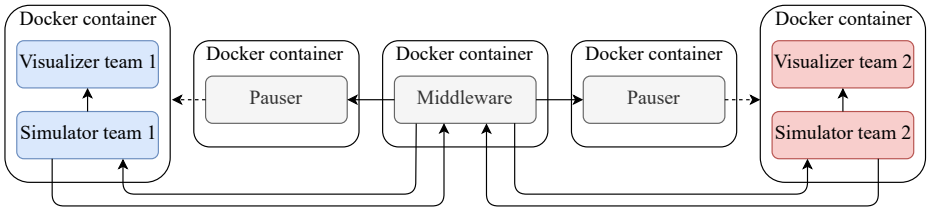


Fig. 5: Final architecture that is used to realize simulated matches consisting of five Docker containers that all run their own application or group of applications. Communication is represented by means of a solid arrow; the execution of a Docker (un)pause command is shown as a dashed arrow.

Results

Initially, the designed architecture was tested by simulating a match between two copies of the Tech United software. Next, in cooperation with RobotSports, their software was made compatible to function in the architecture as well. It has been verified that the proposed solution is suitable to achieve the goal of simulating matches to practice against one another, while both teams can use their own simulation software. For more information, the reader is referred to the full thesis on this section's subject [4].

[4] `https://www.docker.com/`

6 Conclusions

This paper described the major scientific improvements on the Tech United soccer robots over the past year. Using a new mechanism for shooting balls, complemented by exploiting ball impact dynamics, has proved to be an effective way to shoot less predictable balls, which makes it harder for the opponent's goalkeeper to block the shot. Furthermore, allowing the goalkeeper to catch balls through a new method that increases interaction time with the ball was validated, in an effort to increase ball possession. Lastly, the implementation of an architecture to simulate matches between two opposing teams, each using their own simulation software, was validated, which can be used to increase (simulated) practice possibilities. Hopefully, the presented progress contributes to an even higher level of dynamic and scientifically challenging soccer competitions during RoboCup 2022 in Bangkok, Thailand.

References

1. Lopez Martinez, C., Schoenmakers, F., Naus, G., Meessen, K., Douven, Y., Van De Loo, H., Bruijnen, D., Aangenent, W., Groenen, J., Van Ninhuijs, B., Briegel, M., Hoogendijk, R., Van Brakel, P., Van Den Berg, R., Hendriks, O., Arts, R., Botden, F., Houtman, W., Van t Klooster, M., Van Der Velden, J., Beeren, C., De Koning, L., Klooster, O., Soetens, R., Van De Molengraft, R.: Tech united eindhoven team description 2014 (2014)
2. MSL Techinal Committee 1997-2020: Middle size robot league rules and regulation for 2022 (2022), `https://msl.robocup.org/wp-content/uploads/2022/01/Rulebook_MSL2022_v23.0.pdf`
3. MSL Workshop 2015, Aveiro, Portugal: Data structure for world model sharing in msl (2020), `https://msl.robocup.org/wp-content/uploads/2020/01/Rulebook_MSL2020_v21.4.pdf`, last accessed 14 March 2022
4. Nijland, L.: Realizing simulated matches in the robocup msl (2021), `https://pure.tue.nl/ws/portalfiles/portal/188083876/0958546_Nijland.pdf`
5. Schoenmakers, F., Meessen, K., Douven, Y., van de Loo, H., Bruijnen, D., Aangenent, W., van Ninhuijs, B., Briegel, M., van Brakel, P., Senden, J., Soetens, R., Kuijpers, W., Olthuis, J., van Lith, P., van t Klooster, M., de Koning, L., van de Molengraft, R.: Tech united eindhoven team description 2017 (2017)
6. Schoenmakers, F., Meessen, K., Douven, Y., van de Loo, H., Bruijnen, D., Aangenent, W., Olthuis, J., Houtman, W., de Groot, C., Dolatabadi Farahani, M., van Lith, P., Scheers, P., Sommer, R., van Ninhuijs, B., van Brakel, P., Senden, J., van t Klooster, M., Kuijpers, W., van de Molengraft, R.: Tech united eindhoven team description 2018 (2018)
7. Senden, J., Douven, Y., van de Molengraft, R.: A model-based approach to reach a 3d target with a soccer ball, kicked by a soccer robot (2016), `https://www.techunited.nl/media/images/Publications/StudentReports/July2016/0716549-Jordy.pdf`
8. Zhou, Z., Yao, W., Ma, J., Lu, H., Xiao, J., Zheng, Z.: Simatch: A simulation system for highly dynamic confrontations between multi-robot systems. 2018 Chinese Automation Congress (CAC) pp. 3934–3939 (2018). https://doi.org/10.1109/CAC.2018.8623698