RoboCup SPL 2017 Team Paper

Gary Bai, Sean Brady, Kenji Brameld, Amri Chamela, Jeremy Collette, Samuel Collis-Bird, Brad Hall, Kirsten Hendriks, Bernhard Hengst, Ethan Jones, Maurice Pagnucco, Claude Sammut, Peter Schmidt, Hayden Smith, Timothy Wiley, Addo Wondo, and Victor Wong

School of Computer Science and Engineering
UNSW Sydney
Sydney 2052 Australia
http://www.cse.unsw.edu.au

Abstract. RoboCup continues to inspire and motivate our research interests in cognitive robotics and machine learning, including vision, state-estimation, locomotion, layered hybrid architectures, and high-level programming languages. The 2017 UNSW Sydney team consists of undergraduate students, Master and PhD students, past RoboCup students and supervisors who have been involved in RoboCup for over a decade. Major developments in 2017 include a region of interest finder, an improved ball detector which is more robust to different lighting conditions, increased stability in locomotion and kicking to cope with an astro-turf playing surface, improved high level soccer strategies, contention strategies and improvements to the penalty shoot-out challenge.

1 Introduction

Team *UNSW Sydney* (*rUNSWift*), has been competing in the Standard Platform League (SPL) since 1999.

The competition in 2016 introduced new challenges for the teams. The ball changed from being a distinct orange colour, to being a standard black and white, making it significantly less unique and harder to detect. Teams struggled to detect the new ball consistently in the RoboCup 2016 Competition as well as in the RoboCup 2017 Competition where the new ball still proved to be a major challenge. Furthermore, the rule changes for 2017 included a shift to natural lighting, making it difficult to continue using a static camera and colour calibration method for vision. Introducing artificial grass was another major change made in the rules, resulting in increased instability in the Nao robot's locomotion.

The 2017 UNSW Sydney team members are Gary Bai, Sean Brady, Kenji Brameld, Amri Chamela, Jeremy Collette, Samuel Collis-Bird, Kirsten Hendriks, Ethan Jones, Peter Schmidt, Hayden Smith, Addo Wondo, Victor Wong and faculty members Brad Hall, Bernhard Hengst, Maurice Pagnucco, Claude Sammut, and Timothy Wiley.

Fig. 1. A subset of the 2017 team.

This paper outlines the innovations of the 2017 team, including the vision restructure, the colour region of interest finder, ball detector, robot detector, obstacle avoidance and robot-specific motions.

2 Team Organisation and Development Methodologies

Consistent team organisation and routines have played a significant role over the past year and has allowed the team to successfully reach the quarter-finals at RoboCup 2017.

The team maintained regular weekly meetings and testing routines that have been used in past years. In addition, Slack was used to provide a platform for constant team communication.

The team would meet once a week, utilising Google Hangouts when geography made physical meetings challenging. At these meetings each team member would outline their progress over the previous week, ask questions about topics that were challenging and set goals for the next week. This process made all team members accountable for their actions over the past week and ensured everyone remained focused. It also allowed team members to ask for assistance on difficulties they had experienced, fostering a culture of support for other team members.

All members on the team communicated through Slack, a team-communication platform, which allows messages to be sent out to specific members through the use of channels, along with periodic stand-ups for disciplined updates on a more frequent bases. This encouraged clean, organised team communication, allowing team members to only listen in to conversations that is of significance to them.

The team also maintained regular testing schedules. The simplest of these is a 'striker test', where a single robot and the ball are placed in set positions around

the field, and the efficiency in scoring a goal is observed and evaluated against previous tests. This tests the integration of all our modules and improvements over the past week to ensure that changes are actually improving our overall soccer play. We also ran regular practice drills and matches involves multiple robots to test team play, positioning, contention and obstacle avoidance.

3 Motion

Competitive bipedal soccer playing robots need to move fast and react quickly to changes in direction while staying upright. An overview of the rUNSWift motion module is provided in an internal technical report [3]. This report describes our approach to reactive omni-directional locomotion for the Nao robot as used in the RoboCup Standard Platform League competitions from 2014 onward.

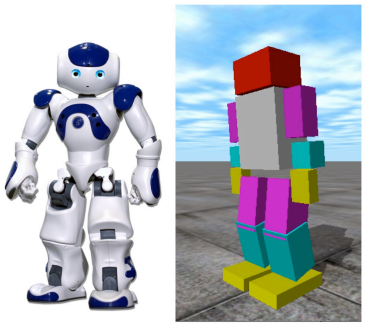

Fig. 2. Nao Robot and Simulated Version. The box-like rendition of the ODE simulated Nao has been programmed with the precise dimensions, masses, and joint-locations from the manufacturer's specification.

We have used reinforcement learning to learn a policy to stabilise a flat-footed humanoid robot using a physics simulator. The learned policy is supported theoretically and interpreted on a real robot as a linearised continuous control function [4]. For 2017, the walk engine did not require any significant changes, and was robust to the astroturf surface used at the competition.

The capability for individual configurations for get-ups and further static actions was integrated into our Action Generator, responsible for joint interpolations between pre-determined actions. This introduced more tuning into the rUNSWift system, however, also allowed for flexibility when determining the best set of joint angles and pose durations for each action when taking into account the inherent varied joint offsets in each robot, which would become more apparent as the hardware became more damaged through continued use.

Individual kick lean configurations were also integrated into the system to tackle varied hip angles required for balancing on the support foot when swinging

forward for the ball, as this remains our most unstable point during the kick. Reseach and testing was performed into changing the phases of the current kick to increase its speed, however a stable solution was not converged on by the time of the 2017 competition.

The get-ups themselves also underwent significant modifications to account for the increased friction in the surface change from carpet to astro-turf as part of the 2017 SPL rule changes. Rather than building upon the actions the rUNSWift team has been using over the past few competitions, majority of the poses in this get-up were modelled off of those in RoboCup SPL team B-human's front get-up[8]. Our use of get-ups largely targeted less dragging of limbs along the surface, which also prevented a commonly increasing problem of limbs being caught and causing the motors to go into overdrive and shutting down the system.

A turn dribble was re-introduced into the motion system this year that was frequently used during contention behaviours. The turn dribble is a simultaneous rotation and kick action, designed to get the ball away from an opponent robot when two robots are contesting for the ball. It entails a series of multiple walk commands involving aligning, turning with one foot, and then following through with the other. There was brief development into maximising the accuracy of the dribble with the black and white ball by striking the ball further up its arc for a more perpendicular trajectory, however the same angles were used as the orange ball in the 2017 competition which ultimately still proved to be successful.

4 Odometry

A visual odometry module was developed in 2012 that estimated the true change in robot's heading by monitoring changes in pixels between vision frames. This is important during external disturbances such as collision with another robot.

In 2015, z-axis gyroscope measurement was added to Nao v5, which reliably replaced the visual odometry module. The change in heading reported by the walk engine is subsituted with z-axis gyroscope measurement. To avoid drift that occurs when integrating biased gyroscope measurement, walk odometry heading is only replaced when the rates are sufficiently different. The resulting odometry is more accurate and robust to external disturbances.

In 2017 a simple system was added to help predict the current offset in order to compensate for the offset of the camera due to the robot rocking. This is important when estimating the position of visually detected objects, especially when they are relatively distant. The system records the gyro readings over a short period and compared the current gyro reading to this history. It then replaces the gyro reading with the reading that occured 2 frames after the best matching point in the history. The system essentially assumes a consistant walk pattern, and does not deal well with situations in which external forces are affecting the robot.

5 Localisation

The 2000 competition saw the initial use of a Kalman filter-based localisation method that continued to evolve in subsequent years [6]. The localisation system evolved to include a multi-modal filter and distributed data fusion across the networked robots. In 2006, we went from treating the robots as individuals sharing information, to treating them as one team with a single calculation spread over multiple robots. This introduced the ability to handle multiple hypotheses. It also allowed us to use the ball for localisation information.

A significant innovation in 2012 was the use of a variation of the Iterative Closest Point (ICP) algorithm extending previous work on field-line matching [7, 10] to other visual features and objects.

In 2014, the 2006 distributed multi-modal Kalman filter localisation innovations developed for the AIBOs were ported to the Naos. We track multiple hypothesis modes for the pose of the robots and ball on the field. Each hypothesis mode consists of the robot pose, ball position and velocity, and the poses of the teammate robots. The robots share information regarding the ball's state and the robot's pose so that teammates can incorporate this information into their own filters.

Previously when the competition was using the orange ball, the high reliability of observation meant that it could be used as a point of disambiguation between the two symmetric sides of the field as teammates could vote in a way which side of the field they believed the ball was on. However with the introduction of the black and white ball, the inconsistency of observation (in both accuracy and recall) meant that it was unlikely for more than two robots to (correctly) observe the ball at any one point in time, compromising its suitability as a landmark. The introduction of white goal posts also meant that if a set of goal posts were positioned in front of a white background (which we saw in Nagoya with the white field barrier), the reliability of measurement vastly decreased.

Thus in 2017, we removed goal posts and ball landmarks from our system and instead relied more heavily on field features. By using the relative position, heading and orientation of key field features (corners, T-junctions and centre circle), the robot is able to triangulate its position to a set of poses (we have more than one possible poses due to the aliased nature of the field feature). The pose innovations are passed through to the Kalman filter with measurement variance based on discrepancy with our current pose hypothesis. The performance at Nagoya suggests the system was successful in improving our robots' localisation with greatly improved ready skills and a reduction in the number of "flips".

6 Vision

In 2017 the vision system underwent a major overhaul. As in previous years when a frame is received, three downsampled representations of the image are generated - a colour saliency image from a pre-populated lookup table, a greyscale image and an edge image. The subsequent vision processing is done on these downsampled images, excepting where the Fovea system is used to zoom in.

The two key stages in the vision system are the region of interest (ROI) finding followed by a series of detectors that utilise the ROIs found. The abstraction of the region of interest finder to its own stage in the pipeline is a key change from 2016, where each detector attempted to do its own region of interest finding. These ROIs retain the Fovea system's zooming capability.

The system also underwent a number of optimisations to allow it to process the top camera image downsampled to 320x240, up from 160x120. This allows the ROI system to pick up on objects farther away.

The field features code from previous years still remain the same. An initial scan is performed over the edge image to generate candidate points in the image. We use RANSAC to fit circles and lines to these candidate points. Once lines are formed, complex features such as the T intersections and corners are created by looking at lines that intersect. A similar calculation is done with the centre circle and a line that passes through it which is used to generate the orientation. From 2016, we learnt that the system relied on goal detection too much for localisation so once this was turned off a greater weight was placed on the detection of corners and T junctions.

The team also explored options for dynamic colour calibration. The method involved sampling the frames for green during the initial state to build a definition of green. This definition for green was used in place of the colour table when a colour saliency image gets generated. We tried to use the YUV colour space, but found that the method was overfitted to the lab environment. Due to time constraints it was not explored further for 2017. Furthermore the majority of the vision system was colour independent, and green was only used to establish the field boundary. Instead, work was done to ensure that the other parts of the vision system was able to cope with varied lighting.

6.1 Colour ROI

At this stage everything of interest on the RoboCup Soccer SPL field is primarily white. Correspondingly in the 2017 vision system ROI are created by finding white areas in the image. These areas are located using a standard connected component analysis algorithm [9], with some modifications. The first modification takes advantage of the fact that only the bounds of the regions are needed. Only the first pass of the algorithm is performed, recording the axis aligned extents of each group of pixels. When this is done connected groups can be quickly merged by taking the maximums of these bounds.

This algorithm tends to produce large groups containing most of the objects in the frame due to field lines connecting other white objects, such as the ball. To handle this the system arbitrarily splits the scene into an n by n grid, and prevents components from being connected across grid lines. Of course, this may split useful objects like the ball. To handle this the system merges nearby groups where the bounding box formed after merging has a sufficient ratio of white to not white pixels. Together this produces bounding boxes that are reasonably likely to contain interesting objects, tightly bounded when those objects are not close to other white objects.

6.2 Ball Detector

A new black and white ball detector was written in 2017 [11]. The ball detector takes in a region of interest then runs several heuristic checks to analyse the pattern inside that region. Each check that passes adds some level of confidence, and if the confidence reaches a certain threshold we have detected a ball.

There are several stages to the ball detector. First we look for a ball candidate inside the region of interest. If the region is cropped well around the ball, this stage is trivial. In other cases (such as when the ball is positioned near a line so that from the robot's perspective there is no clear separation between the ball and the line) a ball candidate is narrowed by looking for the shadow of the ball. The shadow of the ball is generally the darkest spot on the image, allowing it to be classified as black in the initial saliency image. A candidate region is then formed around the shadow, with the size estimated based on the location in the frame. For cases where the ball is up close and no shadow is seen, we use a cascading circle fit. The best circle found is then used for further analysis.

Next the ball detector does some image preprocessing to make the data more consistent. The size of a region is adjusted to a square and its resolution for that ROI increased if it is far away. An initial circle fit is done to improve the accuracy of the heuristic checks. We also normalise the contrast inside the circle, to handle variance in lighting and shadows.

Finally there is the analysis stage that contains the heuristics to check and verify the ball. The main features that we look for are the black patches, identified using Otsu's method [5]. The ROI system already identifies clusters of white, so any internal regions made up of dark pixels inside should be the black patches of a ball. The ball detector requires at least one of those internal regions to be fully internal (where the patches of black cannot touch the edge of the circle) and at least three black regions. Additional sanity checks are done to ensure that these black patches are approximately the correct size and the circle fit is good.

An example of the ball detector features can be found in Figure 3.

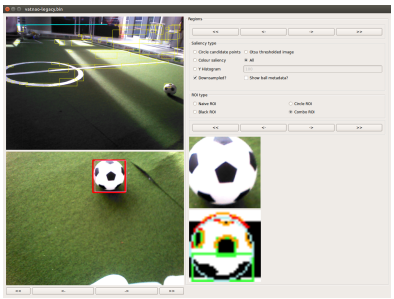

Fig. 3. Vatnao debugging and visualisation tool, ball detector visualisation. The bottom most ball image shows the thresholded image annotated with ball detector results. Red pixels are circle fit candidates, the circle chosen is in blue and black regions are indicated by green bounding boxes.

6.3 Robot Detector

Robot detection has been rewritten to make use of the new ROI system. It now searches for clusters of regions of interest and combines those that overlap. The clusters are classified as robots if they are of roughly the correct size, shape and have approximately the correct amount of white. These are still passed through the same hysteresis system used by the robot detector in previous years.

7 Behaviours

Behaviours follow instructions from the game controller. For game-play an ongoing objective is to localise each robot on the field and to team-track the ball. Only then can robots dynamically allocate their roles for Striker and the three supporter roles (Defender, Midfielder, and Upfielder). The Goalies role is static. Role switching relies on hysteresis to avoid role vacillation. Striker allocation is based on the time to approach a ball for a kick and supporter roles are filled in the order of Defender, Upfielder, then Midfielder, and depends on an estimated time it takes for a robot to reach a certain position and heading.

The 2017 striker uses a strategy to determine the appropriate action, namely whether to perform a straight kick, dribble the ball or dribble while turning. This strategy process is divided into two parts. Firstly, a high-level strategy is decided, such as clear the ball, kick up-field, dribble up-field, perform cross. Once the strategy is decided, the appropriate kick options are selected, which consists of kick mode, kick target, foot to approach ball with, and a flag indicating whether to kick the ball as hard as possible. According to these kick options, appropriate lower level skills, such as Kick, Dribble and ApproachBall are called.

The 2017 Striker demonstrates obstacle avoidance abilities from three separate information sources. Sonars are the primary sensors for obstacle detection, and are capable of detecting obstacles within a metre of the robot. The robot moves in the opposite direction to the side the obstacle is detected on by the sonars. Foot bumper sensors are utilised to detect extremely close range obstacles and obstacles lying on the ground, which are both cases where sonar proves ineffective. Robot (cluster) detection is also utilised for visual detection of obstacles

The FindBall behaviour for Striker2017 utilises a SprinklerTurn skill, which consists of a robot turning on the spot, while keeping the head stationary relative to the ground, by counter-rotating the head. This allows the head to stay stable and reduce motion blur.

Behaviours are written in Python and generally rewritten each year. The perception thread reads sensor information, updates the world model, and calls a Python function in an embedded interpreter which returns actions for lighting LEDs and to physically move the robot. Python is the choice of language due to its lack of compilation process and its ease to sync code changes to the robots during run-time, allowing rapid development of behaviours in the weeks leading up to competition.

8 Tooling

8.1 Vatnao

As part of the development of our new Vision System we developed a new tool to assist in development and debugging. The tool, Vatnao, runs the vision system on a local machine against dumps saved from the Nao Robots. It allows the developer to examine variables, annotate images based on how Vision processes the raw frames and even adjust configuration in real time. We used it heavily as part of development of the ball detector to reduce the development cycle, reducing testing time, allowing us to see the effects of tweaks and changes to our codebase.

The offline vision testing reduced the overhead of running the algorithms on a NAO. It also allowed us to break down the componenets of each detector. For example, in the case of the ball detector, each heuristic can be broken down and displayed separately to determine optimal values for the checks.

8.2 Simswift

Simswift[2] is a new build target of the rUNSWift codebase that runs on a Linux PC and controls a virtual NAO on the RoboCup 3D League (3DL) Simulator[1]. More specifically, Simswift is a rUNSWift build target with preprocessor definitions that disables components that interface with NAO hardware and replaces them with components that interface with the 3DL Simulator instead. The 3DL Simulator is the software which handles the physical simulation of NAOs and the SPL environment. Simswift is the simulator-friendly build target of the rUNSWift codebase, that is able to interact with the 3DL Simulator (that is to say, Simswift itself is not a simulator).

Fig. 4. Five copies of simswift running simultaneously simulating a team of robots, as visualised by *RoboViz*

Instead of interfacing with a NAO via libagent, Simswift interacts with the 3DL Simulator using the Simulation module. If activated, the Simulation module and 3DL Simulator effectively replace the physical NAO and its environment in

the system architecture. Instead of receiving sensory information from libagent, Simswift receives sensory information from the Simulation module. Instead of sending joint commands to libagent, they are sent to the Simulation module. In turn, the Simulation module receives sensory information from the 3DL Simulator via TCP connection, and in response, sends the joint commands to be actuated on the virtual NAO.

For compatibility, the Simulation module provides the same interface and exhibits the same behaviour that libagent does. That is, the Simulation module writes sensory information to the shared memory at the same location and in the same format as libagent. Further, the Simulation reads joint commands from the same location in shared memory and in the same format as libagent. This means that the rUNSWift code that interacts with libagent remains unchanged. In essence, the libagent or Simulation modules are interchangeable.

The rUNSWift source can be built and run interchangeably in both simulation and on a physical NAO. A feature can be developed and tested in simulation and then rebuilt and run on a physical NAO. This gives way to more efficient testing and easier application of machine learning approaches.

9 Concluding Discussion

In 2017, UNSW Sydney made significant improvements to the vision system, specifically for efficiently finding regions of interest in the camera image, ball detection, and robot detection. The team also made a range of improvements to the behaviours, with the introduction of a dribble tactic, camera stability to reduce motion blur, and a turn dribble for when competing for the ball against other robots. With these improvements, rUNSWift made the Quarter Finals of the Championship cup.

Behaviours, vision and motion are the three areas for improvement and future work leading into the 2018 competition in Montreal. The current behaviours use weak assumptions about the distance at which field features and the ball can be seen, and instability in kicking routing. The improvements to vision in 2017 meant that behaviours could be updated to take advantage of seeing the ball at further distances, and rely on a more accurate position from localisation. Vision requires improvements to the efficiency of field feature detection, so that the processing of images can occur at the camera frame-rate. Finally, the current kicking routine caused the robot to fall on the 2017 competition fields, despite working in the UNSW lab conditions. Thus, future work will investigate approaches to more easily adapt and tune our walk and kick engines to new fields.

In 2017, UNSW Sydney developed competitive software and a quality codebase. However, with a high student turnover, and a general increase in the quality of SPL teams, the task of winning the championship is always challenging. The continued achievements of the team cannot be attributed to a single factor, or to a single year, but to a combination of integrated software developed and team collaboration over several years.

Acknowledgements

The 2017 team wish to acknowledge the legacy left by previous rUNSWift teams and the considerable financial and administrative support from the School of Computer Science and Engineering, University of New South Wales. We wish to pay tribute to other SPL teams that inspired our innovations in the spirit of friendly competition.
References identified as UNSW CSE Robocup reports and other Robocup related references in this paper are available in chronological order from:
`http://cgi.cse.unsw.edu.au/~robocup/2014ChampionTeamPaperReports/`

References

1. Boedecker, J., Dorer, K., Rollmann, M., Xu, Y., Xue, F., Buchta, M., Vatankhah, H.: SimSpark User Manual. RoboCup Soccer Server 3D Maintenance Group (2010)
2. Collette, J.: Using 3d simulation to develop robot code (2017)
3. Hengst, B.: rUNSWift Walk2014 report. : http://cgi.cse.unsw.edu.au/'tilde'robocup/2014championteampaperreports/20140930-bernhard.hengst-walk2014report.pdf, University of New South Wales (2014)
4. Hengst, B.: Reinforcement Learning inspired disturbance rejection and Nao bipedal locomotion. In: 15th IEEE RAS Humanoids Conference (November 2015)
5. Otsu, N.: A threshold selection method from gray-level histograms. IEEE transactions on systems, man, and cybernetics 9(1), 62–66 (1979)
6. Pham, S.B., Hengst, B., Ibbotson, D., Sammut, C.: Stochastic gradient descent localisation in quadruped robots. In: RoboCup. pp. 447–452. Springer (2001)
7. Ratter, A., Claridge, D., Ashar, J., Hengst, B.: Fast object detection with foveated imaging and virtual saccades on resource limited robots. In: Australasian Joint Conference on Artificial Intelligence. pp. 560–569. Springer (2011)
8. Rofer, T., Laue, T., Kuball, J., Lubken, A., Maa, F., Muller, J., Post, L., Richter-Klug, J., Schulz, P., Stolpmann, A., Stowing, A., Thielke, F.: B Human team report and code release. Tech. rep., Deutsches Forschungszentrum fur Kunstliche Intelligenz and Universitat Bremen (2016)
9. Rosenfeld, A., Pfaltz, J.L.: Sequential operations in digital picture processing. Journal of the ACM (JACM) 13(4), 471–494 (1966)
10. Sheh, R., Hengst, B.: A fast vision sensor model: Matching edges with nightowl. In: Australian Conference on Robotics and Automation (2004)
11. Wondo, A.: Black and white ball detection using handcrafted heuristics (2017)

References identified as UNSW CSE Robocup reports and other Robocup related references in this paper are available in chronological order from:
`http://cgi.cse.unsw.edu.au/~robocup/2014ChampionTeamPaperReports/`