Team *Edinferno* Description Paper for RoboCup 2015 SPL

Alejandro Bordallo Micó, Nantas Nardelli, Svetlin Penkov, Stanislav Manilov, Subramanian Ramamoorthy, Sethu Vijayakumar

School of Informatics
The University of Edinburgh
United Kingdom

Abstract. This paper summarizes progress made by our robotic soccer team, *Team Edinferno*, towards participation in the 2015 RoboCup Standard Platform League competition held in Hefei, China. During 2013-2014 our focus was on improving modules including optimised kicks, player behaviour, team coordination/communication and probabilistic localisation. Starting with the 2015 competition, we plan to develop and field a new code framework based on the Robot Operating System (ROS), building up our team again from the ground up. Partly because of this, our participation this year will be restricted to the drop-in competition and all of the technical challenges.

1 Team Introduction and Focus

Team Edinferno is a team consisting primarily of three postgraduate students, one undergraduate student and one faculty member, all coming from the robotics research groups within the School of Informatics at the University of Edinburgh. As members of the Edinburgh Centre for Robotics, we have an active and productive interest in humanoid robotics and robot learning, situated within a diverse community of AI and computer science researchers - the largest and top ranked in the UK.

The team leader is Dr. S. Ramamoorthy, Reader(Associate Professor) in Robotics, who has extensive background in robotics and machine learning, in academia and industry. Research within our group is organized around the theme of developing interactive and autonomous decision making mechanisms in continually changing and strategically rich environments, while also leveraging our established strengths in robot control and motion synthesis. RoboCup fits nicely within this robotics agenda, allowing us to explore the practical side of the problem of fielding a multi-robot system in a dynamic environment.

Our team made its RoboCup début in the 2011 world cup in Istanbul, Turkey, where we entered both in the Standard Platform League and the 2D Simulation League. In our début in the SPL, using a home-grown framework, we did not manage to get past the first round. We returned to the 2012 Mexico City RoboCup competition, where our team reached the quarter-finals, losing to the defending champions and eventual finalists, B-Human. This version of our codebase leveraged the publicly available B-Human framework to provide us a software base and modules for walking and low-level vision. We continually improved upon this code for 2013-2014 based on B-Human' s 2011 release.

Based on these few years' experience, we observed that our primary research activities with larger robot platforms and our RoboCup efforts were not really coming together as it was difficult to transfer ideas between these domains given the disparate code bases. Following the dominant trend in the rest of the robotics community, we are interested in experimenting with the possibility of constructing our RoboCup SPL team using the Robot Operating System (ROS) framework.

Therefore our main objective for this year is to re-construct our code infrastructure using ROS. This should allow us to work on components in a much more modular fashion, implementing state of the art improvements while keeping code manageable for a *small* team such as ours. We hope that this would also facilitate inter-team code sharing in the future, anticipating to some extent potential future developments in the drop-in challenge, but at this stage these issues are beyond our scope. We do expect, however, that using this framework would allow us to avoid reinventing the wheel where the broader robotics community has already solved problems well, e.g., in aspects of low-level vision and probabilistic localisation. We are well aware of the processor and memory limitations on the NAO, which makes what we are trying challenging, which is why we are taking a cautious approach by first focussing on just developing a single drop-in player with this new software - albeit one who has to solve all the component problems of motion and perception in order to function at all.

That said, our research group has made significant contributions to the problems of ad-hoc coordination in multi-agent systems [1], and other technical problems that feature in challenges. We have also been early stage developers [1] of the coach robot, using it already during the 2014 competition to advise our players. This agent was already constructed using ROS, serving as our very first experiment in exploring the use of these tools on the NAO platform. Our aspiration is to transfer some of this know-how into our SPL codebase.

Figure 1 - 2014 Edinferno SPL core team. *Left to right:* Alejandro Bordallo, Nantas Nardelli and Svetlin Penkov. Also an important part of the team: Stanislav Manilov and Subramanian Ramamoorthy.

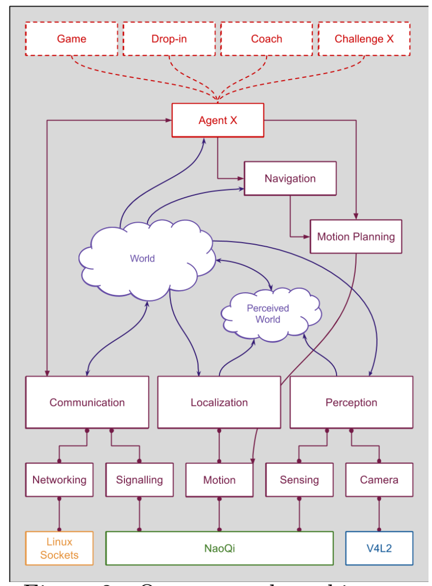

Figure 2 - Our new code architecture

2 Robot Hardware

Our team currently has 7 H21 V4 NAOs used for developing and testing the code in our university's small-scale pitch. We also own 5 new H25 V4+ NAOs, 2 of which will be used for the competition. We hope this will minimise some of the hardware problems arising from using older worn down NAOs for the football matches. Given our intention to only focus on drop-in and challenge competitions during 2015, our plan is to carry two H25 (Drop-in and reserve) and one H25 NAO (coach) robots.

3 SPL Participation

Our team would like to participate in the SPL *drop-in* competition and all technical challenges, including the open challenge where we expect to demonstrate some of the research advances mentioned in this document.

4 Basic Components

4.1 Software Architecture

Our earliest attempt at robot soccer was based on a completely home-grown implementation based on Aldebaran's NAOqi. However, limited resources and manpower meant our efforts were most effective by focusing our attention on specific modules where we had the expertise to innovate, utilising existing technology from elsewhere at the lowest software infrastructure levels. After a season of trials and after careful consideration of our long term needs, we decided to switch to the B-Human framework [2] [2], which gave us a fast, energy-efficient walk.

[1] To the best of our knowledge, we were the only team in the 2014 competition in Brazil to actively use the coach during the main games.

[2] http://www.b-human.de/downloads/bhuman11_coderelease.pdf

To increase the sustainability of our development process, we are now making the transition towards having a framework fully built around ROS, in which the NAOqi module will ultimately only be used as an API to gain access to the hardware of the robots, re-writing the majority of higher-level algorithms from scratch or through other sources. This will also allow us to directly address our dissatisfaction in not being able to connect our RoboCup activities with the much larger body of work within our centre for robotics - in particular, being able to quickly experiment with modules developed elsewhere for robot learning and other such functionalities.

4.2 The Robot Operating System (ROS)

ROS is a modular system implemented as a collection of tools and libraries that form an extremely flexible framework for building robotic software. It relies on modular code packages, which create *nodes* that manage *publishers* and *subscribers*, which serve to interface multiple *nodes*. Two nodes can share data through the network, regardless whether they run on-board or on a separate computer. This enables trivial connectivity between unrelated code infrastructures that need only agree on the type of message to be shared. Nowadays most state-of-the-art packages in visual servoing, planning and motion are either distributed as part of its package system or are distributed with interfaces to communicate with its ecosystem. It follows that to reduce and offload the low-level work and focus on building an effective agent for our RoboCup team it is ideal to build the system around ROS. Moreover the ecosystem does not only consist in robotics libraries, but also in debugging, building, visualisation and virtualisation tools, which allow our team to also revamp our building and testing tools.

It must also be noted that in general RoboCup and other robotics communities are converging towards ROS as the de facto standard framework to use, which implies broader applicability of the software that we develop for Edinferno and more chances for our work to be easily reusable in other applications. Unfortunately Aldebaran does not currently provide official support for an installation of ROS on the OpenNAO operating system, but there has been enough work done by the community (including our own, as mentioned earlier) so that we feel confident in using ROS as the new base for our robots.

5 Contributions for the 2015 competition

5.1 ROS

Over the last few months, we have built our framework with the aim of interfacing ROS packages with the Nao SDK. Our priorities are code maintainability, flexibility, and efficiency; which are only achievable for a small team such as ours due to the modularity and open-source support innate to ROS. Our framework is arranged in a hierarchical set of ROS module layers (*Fig. 2*):

The *Wrapper* layer contains the required ROS *services* that in turn call the NaoQI methods necessary for requesting or sending data to the NaoQI server. We consider these packages (i.e. Networking, Signalling, Motion, Sensing and Camera) the foundation of our infrastructure. They are designed from the bottom-up to be as efficient as possible, while providing the interface to higher layers of our framework. This ROS wrapper can be easily transferred and reused for multiple projects, and we aim to maintain it and distribute it as an official ROS package eventually.

The *Package* layer is where most of our innovative work is introduced. It is composed of a selection of higher level packages (i.e. Communication, Localization, Perception, Motion Planning, Navigation) which utilise our ROS wrapper to interface with the Nao SDK methods. Since we have abstracted the Nao from our higher layer packages, we can import and re-use ROS packages produced by the open-source community (CameraCalibration, ROS navigation, PTAM Visual SLAM, MoveIt!) or our own state-of-the-art research developments.

The *World* represents the collective namespace of topics to which our state estimation nodes publish up-to-date filtered world information. This data is readily accessible by the planning nodes, which are constructed using ROS *actions*. *Actions* allow the real-time use of *services*, while monitoring the *World* topics. This enables the on-line planning and execution of motions and activities such as kicks, navigating towards a target or obstacle avoidance.

The *Agent* layer is a hierarchical state machine (Using ROS Smach), which transfers the agent between *actions* (Using ROS Actionlib), depending on the results of previous *actions* and the state of the world. We thus create behaviours with all of the inherent ROS benefits, such as introspection into the agent's decision process, and a modular framework for concurrently running behaviours for closed-loop control.

This year we are fully committed into showing that ROS is a powerful set of tools that enable the rapid implementation of complex code, and easy interfacing between packages from different origins. We aim to present our approach in the open challenge, and show our Coach and AdHoc player capabilities. Our plan is to finalise the core infrastructure and release it as open-source for other SPL teams to build upon. We plan to maintain it in the future while we prepare for the next year competition, where we aim to field a full team of robots for the main competition.

5.2 Drop-in player

We are working on an adaptive drop-in agent, able to perform basic forms of rational cooperative behaviours and providing useful data for other team members. The architecture currently is essentially a finite state machine, with transitions between states depending on factors such as the *state of the game* and *confidence* on other agents' abilities to advance the ball. These are parameters our robot attempts to learn during game play, possibly aided by past knowledge from previous games.

The *state of the game* affects the urgency required for strategic moves, which is determined by the score and the position estimates of all agents. This may affect for example the choice of quickly clearing the ball from a friendly goal or passing ahead with more care towards another team-mate.

The *confidence* on other agents is acquired during play, and is calculated based on a number of factors. These include the length of time each agent has been in control of the ball and how successful it was at advancing its position. This affects the choice of whether to pass to a more able agent, or to intervene if the agent in control of the ball is unsuccessful at advancing its position.

These features are still a work in progress, but we have reason to believe that this way of developing cooperative behaviours from an ad-hoc agent perspective will provide a strong basis for our team and bring interesting improvements to the drop-in competition.

5.3 Challenges

We believe that, although in the past we have concentrated our attention on the main games, the proposed challenges have always brought fresh new ideas to the league, and typically represented quite interesting research contributions. This year, we aim to implement the necessary functionalities to fulfil most of these challenges with our framework.

We intend to prepare for the Open challenge, demonstrating the use of our framework in tasks such as agent tracking and state of game analysis as performed by our Coach robot.

5.4 The Coach

The Edinferno coach robot, as deployed in 2014, was our first step in implementing ROS infrastructure on the NAO. At present, the coach is capable (as shown in the attachment video) of detecting and tracking the ball on the field, analysing the game state and sending simple strategies such as *attack* or *defence*. There are three main modules which are already implemented by utilising ROS and NAOqi libraries.

Image Processing This module is responsible for performing image segmentation in order to detect the objects of interest such as ball, goal and field in each camera frame. The image segmentation is based on colour lookup tables which are created during the coach calibration process. Despite its simplicity, this approach leads to robust results after addition morphological filtering. Global changes in illumination are handled by re-calibration before a game. In RoboCup 2015, we will move towards online adaptation for these colour tables.

Ball Tracking This module controls the robot's head in order to keep the ball in the visual field of the coach. This is achieved by a visual feedback loop entirely implemented as a ROS process. Despite the limited computational capabilities of the NAO we concluded that ROS can be used for real time control of the head in order to obtain the visual servoing behaviour. Additionally, the position of the coach with respect to the field is calibrated before each game in order to be able to determine which side the ball is in. This will be further improved in 2015 by depending more on the field lines.

Game Analysis This module takes into account all perceived information and determines the best strategy for the given game state. Currently, the coach can only determine whether the ball is in the opponent's side or not, which translates to only two high level tactical decisions - attack or defend. However, this base now allows us to explore more policies based on player formations.

6 Related Research

Our team is composed primarily of researchers interested in intelligent autonomous robotics. So, in addition to the thrill of adventurous competition, we participate in RoboCup to advance our scientific agenda. In this section, we give a brief glimpse into areas where this exchange is or could potentially work effectively, between our RoboCup team and broader robotics research.

6.1 Coach Module and Plan Recognition

One of the main improvements we plan is with perception, especially on the coach - who could use field lines and player positions. This improves field detection accuracy and allows for self-estimation of coach position and orientation with respect to the field by using a simple structure from motion algorithm. Secondly, by projecting the estimate of the field on the camera frame we will be able to adjust the colour tables, adapt them to the changing light conditions thus improving the image segmentation accuracy. We have also investigated the formalisation of the role of the coach, and consequently the rest of the game itself, with a view to eventually bringing in more sophisticated reasoning tools to implement strategic planning. This formalisation was presented at the RoboCup 2014 symposium [12].

6.2 Opponent modelling, Intention Prediction and Strategic Interaction

We have an active research interest in ad hoc coordination and intention prediction. We have studied inferring the behaviour of other players in terms of pre-computed models [8], estimation of finite-state opponent models [6], distributions over template plans [7, 5], learning from human demonstration [16] and related problems. Continuing this line of work, we are constructing algorithms for multi-agent learning in *ad hoc* team settings without prior coordination [3, 1] or models of opponent strategies [4]. By building a software framework that makes is easy for us to port these other modules in, we hope to able to bring in some of these ideas into our SPL agent, starting first with our drop-in and coach robots.

7 Conclusions

Team Edinferno is the only SPL team from the United Kingdom. Our research work builds on strong background in robot learning and aims to advance the state of the art of autonomous decision making in continually changing worlds. Although we are a relatively small team, we are building upon our prior work and have already demonstrated a performance level comparable to some of the league's more established teams. Our focus for 2015 is to take a new approach to our SPL codebase, using ROS, and to validate this in the drop-in games with competent cooperative play. Our aim is to use this as a learning experience towards coming back in force by fielding a complete new team in 2016.

8 Acknowledgements

Our work is supported by grants EP/F500385/1 and BB/F529254/1 for the University of Edinburgh School of Informatics Doctoral Training Centre in Neuroinformatics and Computational Neuroscience (www.anc.ac.uk/dtc) from the UK Engineering and Physical Sciences Research Council, UK Biotechnology and Biological Sciences Research Council, and the UK Medical Research Council. The work has made use of resources provided by the Edinburgh Compute and Data Facility (ECDF; www.ecdf.ed.ac.uk), which has support from the eDIKT initiative (www.edikt.org.uk). Our team has also benefited from a Royal Academy of Engineering *Ingenious* grant and a European Commission grant, TOMSY (agreement no. 270436, FP7-ICT-2009.2.1 Call 6), both of which funded the purchase of some of our robots.

References

1. S. Albrecht, S. Ramamoorthy, On convergence and optimality of best-response learning with policy types in multiagent systems, In Proc. Conference on Uncertainty in Artificial Intelligence (UAI), 2014.
2. T. Röfer, T. Laue, J. Müller, M. Bartsch, Team Report and Code Release 2011. b-human.de, 2011.
3. S. Albrecht, S. Ramamoorthy, Comparative evaluation of MAL algorithms in a diverse set of ad hoc team problems. In Proc. International Conference on Autonomous Agents and Multi-agent Systems, 2012.
4. M.M.H. Mahmud, S. Ramamoorthy, Learning in non-stationary MDPs as transfer learning (Extended Abstract), International Conference on Autonomous Agents and Multiagent Systems, 2013.
5. A. Valtazanos, S. Ramamoorthy, Bayesian Interaction Shaping: Learning to influence strategic interactions in mixed robotic domains, In Proc. International Conference on Autonomous Agents and Multiagent Systems, 2013.
6. M. Leonetti, L. Iocchi, S. Ramamoorthy, Induction and learning of finite-state controllers from simulation (Extended Abstract). In Proc. International Conference on Autonomous Agents and Multi-agent Systems, 2012.
7. A. Valtazanos, S. Ramamoorthy, Intent inference and strategic escape in multi-robot games with physical limitations and uncertainty, In Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, 2011.
8. A. Valtazanos, S. Ramamoorthy, Online motion planning for multi-robot interaction using composable reachable sets. In Proc. RoboCup International Symposium, 2011.
9. A. Valtazanos, S. Ramamoorthy, NaOISIS: A 3-D behavioural simulator for the NAO humanoid robot. In Proc. RoboCup International Symposium, 2011.
10. I. Havoutis, S.Ramamoorthy, Constrained geodesic trajectory generation on learnt skill manifolds. In *Proc. Int. Conf. Intelligent Robots and Systems*, 2010.
11. I. Havoutis, S. Ramamoorthy, Motion planning and reactive control on learnt skill manifolds, *International Journal of Robotics Research*, 2013.
12. G.Y.R. Schropp, J-J. Ch. Meyer, S. Ramamoorthy, A formalization of the coach problem, In Proc. RoboCup International Symposium, 2014.
13. S. Ramamoorthy, B.J. Kuipers, Trajectory generation for dynamic bipedal walking through qualitative model based manifold learning. In *Proc. Int. Conf. Robotics and Automation*, pp. 359-366, 2008.
14. I. Havoutis, S. Ramamoorthy, Motion synthesis through randomized exploration on submanifolds in configuration space. In J. Baltes et al. (Eds.): RoboCup 2009, Lecture Notes in Artificial Intelligence, Volume 5949, pp. 92-103. Springer Verlag, Heidelberg (2010).
15. D.J. Mankowitz, S. Ramamoorthy, BRISK-based visual feature extraction for resource constrained robots, In Proc. RoboCup International Symposium, 2013.
16. A. Valtazanos, S. Ramamoorthy, Evaluating the effects of limited perception on interactive decisions in mixed robotic environments, In Proc. ACM/IEEE International Conference on Human-Robot Interaction, 2013.