Aalborg Universitet

AALBORG UNIVERSITY
DENMARK

A Software Framework for Multi Player Robot Games

Hansen, Søren Tranberg; Ontañón, Santiago

Published in:
Lecture Notes in Computer Science

DOI (link to publication from Publisher):
10.1007/978-3-642-17248-9_14

Publication date:
2010

Document Version
Også kaldet Forlagets PDF

Link to publication from Aalborg University

Citation for published version (APA):
Hansen, S. T., & Ontañón, S. (2010). A Software Framework for Multi Player Robot Games. *Lecture Notes in Computer Science*, *6414*, 131-140. https://doi.org/10.1007/978-3-642-17248-9_14

A Software Framework for Multi Player Robot Games

Søren Tranberg Hansen[1] and Santiago Ontañón[2]

[1] Danish Technological Institute,
Center For Robot Technology
Forskerparken 10, 5230 Odense M, Denmark
soren.tranberg@teknologisk.dk

[2] IIIA, Artificial Intelligence Research Institute
CSIC, Spanish Council for Scientific Research
Campus UAB, 08193 Bellaterra, Spain
santi@iiia.csic.es

Abstract. Robot games have been proposed as a way to motivate people to do physical exercises while playing. Although this area is very new, both commercial and scientific robot games have been developed mainly based on interaction with a single user and a robot. The goal of this paper is to describe a generic software framework which can be used to create games where multiple players can play against a mobile robot. The paper shows how an adaptive AI system (D2) developed for real-time strategy (RTS) computer games can be successfully applied in a robotics context using the robotics control framework Player/Stage. D2 is based on Case-Based Planning which learns from demonstration. Using the proposed framework, the paper shows how a robot learns a strategy for an implementation of a simple game.

Keywords: Human Robot Interaction, Games, Artificial Intelligence.

1 Introduction

Based on the demographic development in most western countries, it has been predicted that the number of people with mental and/or physical disabilities will increase while the amount of people to take care of them will decrease [26], [2]. Digital games hold a significant promise for enhancing the lives of seniors, potentially improving their mental and physical wellbeing, enhancing their social connectedness, and generally offering an enjoyable way of spending time [14]. It has been shown that mental and physical health can be improved through a small amount of physical exercises [24], [10], and e.g. Nintendo Wii has been suggested as a means to increase physical activity among elderly [4], [18].

In this paper we introduce a software framework for constructing robot games which motivates people to move physically when playing. The framework provides generic functionality which can be selectively overridden or specialized by user code providing specific functionality for creating a number of games using mobile robots. Many known games are derived from a pursuit-evasion scenario e.g. the child games robbers and cops and the game of tag [21], and in this paper we describe a concrete implementation of a game where multiple players compete against the robot.

The vision of robots participating in our day-to-day lives is the focus in the research field of Human Robot Interaction (HRI) [7]. The vision is supported by progress in the development of new sensor technology and computing, which open the door to a new generation of mobile robotic devices that see, hear, touch, manipulate, and interact with humans [11]. The development is reflected in the fact that computer games are gradually moving from the pure virtual domain into the physical world with the introduction of products like Nintendo Wii, Microsoft Kinect, Sony's EyeToy, etc. There are several examples of how games are constituted using a physical robot, e.g QRIO's routine of tracking and kicking a ball, and in [3] where the robot Leonardo is used to play an imitation game. In [23] we have described a game where a mobile robot can play a simple ball game with a human. Some commercial gaming robots exist, e.g. Tri-Bot from WowWee which is able to play three simple maze-based games with the user.

Although there has been a great advance in computer games in computer graphics, animation and audio, most of the games contain very basic artificial intelligence (AI) [17]. To deal with increasing complexity there has, however, been a push for the development of new AI algorithms, many of which shares requirements with those of robots operating in open-ended environments.

In this paper we will shortly summarize the background for making robot based games and outline some of the requirements. We introduce D2 which is an AI system based on case based planning. D2 is originally developed for real time strategy computer games and we show how it can be used in a robotics scenario also. This is done by implementing a game interlinking D2 with the robot control software Player/Stage.

2 Background

Motivating users to move physically by playing a game is related to persuasive technology which is defined as technology designed to change attitudes or behaviors of the users through persuasion and social influence, but not through coercion [8] and [9]. Robots offer advantages not found in on-screen agents or technology embedded in the environment, such as an increased sense of social presence in an interaction and the capacity for touch and physical interaction [15]. In [12] and [13] the concept of enjoyment as a possible factor influencing acceptance of robotic technology was investigated using the iCat Research Platform which is a robot research platform for studying human-robot interaction. The fact that sociable robots are more fun to play with was confirmed in [16]. Successful games are often characterized using the concept Flow, as proposed by Cskszentmihly [6] which is a mental state which can occur when there is an appropriate balance between challenge and skill. As the cognitive and physical capabilities of the users are expected to vary, the robot should adapt the difficulty of the game to the end user. The fact that intelligent agents working in real time domains need to adapt to changing circumstance has been recognized in computer game industry as well as in robotics. The ability to adapt is necessary in order for autonomous agents to improve their performance and avoid mistakes in a complex and dynamic environment.

As robot sensors gradually get more reliable and new types of sensors emerge, the robot's knowledge about the real world is getting more precise and informative. Computer games are on the other hand developing towards more complexity with an increasing high decision space, involving interactive users and real time performance.

This makes the AI requirements from these two distinct domains intersect and some of the common characteristics are:

- Real Time Nature: imposing constraints in terms of processing time that could be taken by AI approaches situated within these domains.
- Large Decision Spaces: most state of the art computer games have huge decision spaces [5], [1], and thus traditional search based AI techniques cannot be applied. The same applies for robots which should act in open ended environment relying on multiple sensors.
- Unanticipated Scenarios: it is not feasible to anticipate all possible situations and as result, it is hard to design behaviors that can handle all the possible situations and respond appropriately to all possible actions.
- Human in the loop: Games involve one or more interactive user(s) that can change the state instantaneously. Even small delays in AI decisions can be unacceptable.

A popular approach to deal with non-determinism is to use planners based on reinforcement learning, e.g. by modeling the problem as a Markov decision process and focus on learning a policy. These techniques, however, often require a large number of iterations to converge. In complex domains, they are intractable and generalize poorly [17]. While reinforcement learning approaches are good for low level control, symbolic approaches like Case Based Planning are good for high level strategic decisions which is the focus of D2 which we present here.

2.1 The D2 System

D2 [19] is a real-time case-based planning system designed to play Real-Time Strategy games (RTS). D2 implements the *on-line case-based planning* cycle (OLCBP) as introduced in [20]. The OLCBP cycle attempts to provide a high-level planning systems that operate on-line, i.e. that interleave planning and execution in real-time domains. The OLCBP cycle extends the traditional CBR cycle by adding two additional processes, namely *plan expansion* and *plan execution*. The main focus of D2 is to explore learning from human demonstrations, and the use of adversarial planning techniques. The most important characteristics of D2 are:

- It acquires cases by analyzing human demonstrations.
- It interleaves planning and execution.
- It uses an efficient transformational plan adaptation algorithm for allowing real-time plan adaptation.
- In case a simulator is available, D2 can make use of it to perform adversarial case-based planning.

For D2 to learn how to perform a given task, an expert has to provide demonstrations of such task. D2 can typically learn from a few demonstrations (1 or 2), but depending on the variability of the domain or the complexity of the task, many more traces might be needed.

D2 has been used used as an AI engine for several computer strategy games like Wargus (open source clone of Warcraft II), BattleCity, S2, Towers and Vanquish (clone of Risk), and is currently being applied to Starcraft. A previous version of D2 (Darmok) was applied to a game scenario where two virtual characters played a game of tag [17].

3 A Robot Game

In [23] we presented a game based on a simplified pursuit and evasion scenario. The human player should try to hand over a ball to the robot, while the robot should try to avoid receiving the ball. The robotic platform which formed the basis of these experiments, is shown in Figure 1. The robot platform from FESTO is equipped with a head having 126 red diodes (see Figure 1) which enables it to express different emotions. The robot is 1 meter high, and has mounted an URG-04LX line scan laser placed $35cm$ above ground level, scanning 220 degrees in front of the robot. In [23], an on/off switch was placed to find out when the robot had the ball.

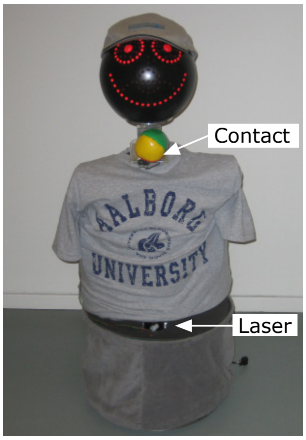

Fig. 1. The modified FESTO Robotino robotic platform

Based on the same robotic platform on which we have added a standard web camera, we suggest a similar simple pursuit and evasion game which allow multiple users to play with the robot. The game consists of a number of squares marked on the floor, a mobile robot and a number of human players. The winner of the game is the participant who has visited all squares first. During a game, the robot can tag a person which makes the person do a detour on the game board. The implementation of the game has been done using the following components (see also Figure 2):

1. A Robotics Environment which takes care of control of simulation and control of the robot (Player/Stage)
2. An AI Engine which learns and executes the robot's overall gaming decisions (D2).
3. A Game Application, which defines the frame and the purpose of the game an works as the interlink between the two former elements. Here we have chose an implementation of a variation of the Game of Tag, but many other games could be interesting.

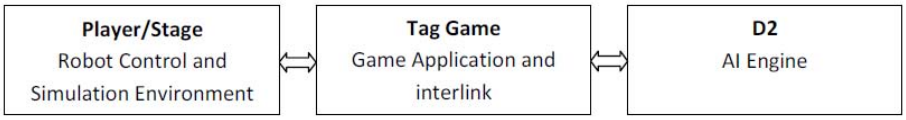
Player/Stage
Robot Control and Simulation Environment
Tag Game
Game Application and interlink
D2
AI Engine

Fig. 2. Illustration of the three components of the framework; a robot controller (Player/Stage), an interlinking game application (Tag Game) and an AI engine (D2)

3.1 The AI Engine

In order to implement D2 as the AI engine for a game, the following elements should be defined in a XML file following a specific D2 syntax: Entities, Sensors, Actions, Preconditions, Postconditions and Goals.

An Entity is a definition of the basic units of the game. In this case there are four: the human players, the robot, the squares marked on the floor and a wall entity. The latter is used to mark the barriers of the game area and obstacles in the game area. Each entity specification holds the name of the entity and a list of which actions are available to the entity.

The Sensor definition describe the robot's sensor input in any level of abstraction. In this case, this is the position values of the entities. Securing that the position of the player and the robot is actually correct is not in the scope D2 but is handled by Player/Stage and will be described later.

The Action definition is a list of all possible actions the entities can do. The low level implementation of these robot actions is not in the scope of D2 either but is handled by Player/Stage. D2 has the function of learning to select the right actions during a game in order for the robot to win the specified goals.

The Goal definition is specification of how to win the game, which in this case is when the robot has visited all squares. With all elements above properly defined, the D2 framework can auto generate a number classes which forms the basis of the game in the Game Controller. This can be interlinked to a robot controller and simulation environment like e.g. Player/Stage.

3.2 Control and Simulation Environment

Player/Stage handles a simulation environment and the low level robot control including position detection, path planning and collision avoidance. It also holds a pre-defined map of the environment which is read from a file.

The position of the robot is done using the standard Player/Stage amcl driver which implements the Adaptive Monte-Carlo Localization algorithm described by Dieter Fox. The amcl driver maintains a probability distribution over the set of all possible robot poses, and updates this distribution using data from odometry, laser range-finders and the pre-defined map of the environment. The detection of the players' position is done by the robot using the mounted laser scanner using a leg detection algorithm [25] and [22].

Collision avoidance is done automatically in Player/Stage by implementing the Vector Field Histogram Plus local navigation method by Ulrich and Borenstein. VFH+

provides real-time obstacle avoidance and path following capabilities for mobile robots and in order to obtain global navigation, the wavefront driver has been layered on top of that. When simulating the game, the human players have been modeled using the obstacle avoidance and path planning functionality offered in Player/Stage. Each player has been modeled having a different color and a standard Player/Stage blob-detection algorithm for web camera input is used to distinct the players from each other.

3.3 The Game Controller

The game controller is basically the game definition and the interlink between the AI Engine and the Simulation Environment. It constructs the game scenario by passing selected actions from D2 to the robot implementation and sensor data from the robot the other way. The game strategy of D2 is created using initial demonstrations, allowing a game instructor to control how the robot should act. In this implementation, demonstration is done by facilitating a GUI where the instructor can choose between the different robot actions. During this process, the Game Controller generates traces which are used by D2 to learn a strategy about the behavior of the robot and the players. When a strategy has been learned, the game controller can let D2 control the behavior of the robot without human interference. In a simulated environment, the Game Controller to some extent also specifies how the persons play the game, i.e. it controls to which position the persons should move. In the current implementation, the game controller can handle any number of squares and players but only one robot.

4 Experiments

We consider a simplified game setup to illustrate and validate the basic functionality of the proposed framework. In each game, the players, the robot and the squares are placed at random positions. The participant who first visits all squares, wins the game. While the simulated human players use a nearest neighbor algorithm to select where to move, the robot can choose between the following actions:

- Move To Square. Make the robot move to a specified square.
- Tag Player. Make the robot follow a specific player. If the distance between the robot and the player is under a specific threshold, the player is tagged and he/she should do a detour by moving to a specific position on the game board.
- Rotate left/right. Rotate the robot left or right from its point of view.
- Move forward or backwards.

In order for D2 to learn a strategy, two games have been completed using a human demonstrator. This is sufficient for D2 to learn a strategy, which now can be used to control the robot. Five different variations of the game have been tried out.

1. The simulated players play against each other without the robot. Each person is moving with a max speed of 0.5 m/s.
2. The speed of one player is set to move slower, having a max speed of 0.1 m/s.

3. The robot is introduced in the game, having a max speed of 2 m/s.
4. As before, but the game field now consists of 10 squares.
5. As 3, but now a goal has been added which states that all players should have visited at least 3 squares before the game ends. In this experiment, the robot has been retrained using a human demonstrator.

Each game has been repeated a total of 20 times for each experiment, and the number of won games by each player can be seen in table 1. The maximum, minimum and mean number of game rounds can be seen in table 2. Figure 3 shows a screenshot from a simulated instance of the implemented robot after a few seconds of playing time. On the figure, it can bee seen that each player including the robot has moved on to a square and is looking for a new square to visit.

Table 1. Number of won games by each player in 5 experiments with 20 games in each

Games won	Ex 1	Ex 2	Ex 3	Ex 4	Ex 5
Person 1	10	20	4	0	13
Person 2	10	0	0	0	2
Robot	0	0	14	13	2
N/A	0	0	2	7	3

Table 2. The minimum, maximum and mean number of game rounds in 5 experiments with 20 games in each

Game rounds	Ex 1	Ex 2	Ex 3	Ex 4	Ex 5
Min	105	101	59	150	221
Max	278	207	139	238	707
Mean	169	189	103	181	435

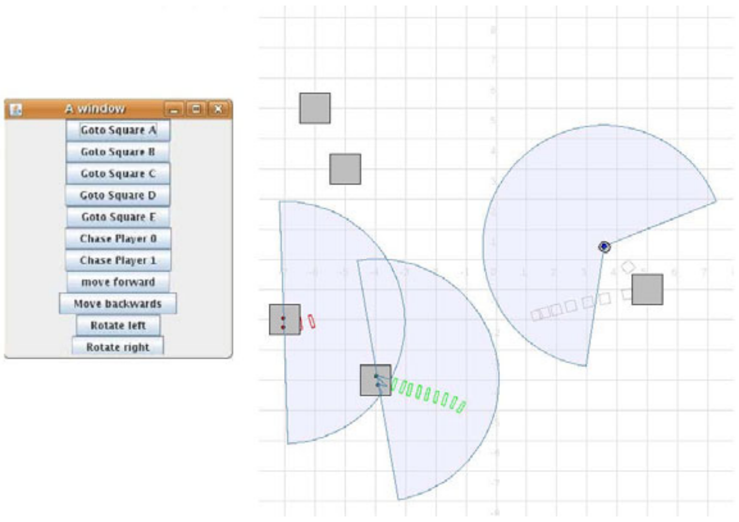

Fig. 3. The figure shows a screenshot of the implemented robot based game. Each player is moving from the initial position to a square marked on the floor. The window to the left is a GUI, which initially lets a human user demonstrate how the robot should behave in order for D2 to create a control strategy.

5 Discussion

In the first column in table 1, it can be seen that the two participating persons win an equal amount of times. This is as expected, as both persons use the nearest neighbor algorithm and move with the same speed. In experiment 2, person 1 wins all the games which is also as expected as this player is capable of moving faster than its competitor. In experiment 3 the robot is introduced and it wins 14 games. This shows that the robot has learned a valid strategy, and since it moves faster it is most likely to win. It is surprising that person 1 actually wins 4 of the games. This is considered to be due to the fact that the robot not has learned an optimal strategy, and to some degree due to a fortunate location of the squares in played games. Two of the games does not end with a winner, because the players are stuck in a deadlock where each one is blocking the way of the others. In experiment 4, the number of squares is increased. Now the robot wins 13 games, the persons win none, but 7 games ends without a winner because simulation has been aborted after a 1000 game rounds. In the last experiment, a goal has been added which states that all players should visit at least 3 squares. Although person 1 wins the majority of the games, person 2 actually manage to win 2 games although he/she is moving much slower than the other participants. Although the robot is capable of winning more often because of its higher speed, it only wins two of the games. This is due to the fact that a new goal has been added, and the robot now spends time on tagging person 1 to ensure that person 2 reaches at least 3 squares before the game finishes. Although more research is needed, this final experiment is included to illustrate that the robot can learn a strategy making all participants capable of playing although they have very different skills. Table 3 shows the maximum, minimum and mean number of game rounds for all experiments. It is worth noticing that the mean number of game rounds increases in experiment 5, because the robot now spends more time tagging the other players.

The presented gaming framework is still at a very early stage, and therefore results have been obtained through simulation only. This reduces the complexity of the game scenario, because getting precise data like e.g. the location of the robot and the players is not a simple problem in the real world. The actual behavior patterns of the human players are also simplified and should be elaborated through real world experiments. The results show that D2 can be used to learn a playing strategy for robot based games. It is important to notice that here the goal of D2 is not necessarily to make the robot play optimal, as this could have been archived with simpler means. The focus is to create a strategy for the robot which ensures a balance between skill and challenge for the participating players during the game. The contribution of this paper is to introduce a generic software framework applicable for multi player robot games. The framework is constructed by interlinking D2 with Player/Stage, each of which have been validated many times in other papers and therefore forms a good basis for further research.

6 Conclusion

In this paper we have outlined a generic software framework which can be used to implement robot based games where multiple human players can compete against a mobile robot. The framework consist in a Case Based Planner (D2) which is interlinked with Player/Stage which serves as a robot control and simulation environment. We have implemented a variation of the Game of Tag for a mobile robot competing against multiple human players. Using the framework, we have showed that the robot can learn different game strategies based on a few demonstrations. A possible application for robot based games is to motivate elderly to do a higher amount of physical exercises and thereby strengthen their mental and physical capabilities.

References

1. Aha, D., Molineaux, M., Ponsen, M.: Learning to win: Case-based plan selection in a real-time strategy game. In: Muñoz-Ávila, H., Ricci, F. (eds.) ICCBR 2005. LNCS (LNAI), vol. 3620, pp. 5–20. Springer, Heidelberg (2005)
2. A. Alzheimer's Disease International. The prevalence of dementia worldwide. Technical report, Alzheimer's Disease International, The International Federation of Alzheimer's Disease and Related Disorders Societies, Inc. (2008)
3. Brooks, A.G., Gray, J., Hoffman, G.: Robot's play: Interactive games with sociable machines. Computers in Entertainment 2, 10–10 (2004)
4. Brown, R., Sugarman, H., Burstin, A.: Use of the nintendo wii fit for the treatment of balance problems in an elderly patient with stroke: A case report. In: International Journal of Rehabilitation Research, Proceedings of the 10th Congress of the European Federation for Research in Rehabilitation, vol. 32, p. 109 (2009)
5. Buro, M.: Real-time strategy games: A new ai research challenge. In: IJCAI 2003, pp. 1534–1535. Morgan Kaufmann, San Francisco (2003)
6. Csikszentmihalyi, M.: Beyond boredom and anxiety. Jossey-Bass Publishers (1975)
7. Dautenhahn, K.: Methodology & themes of human-robot interaction: A growing research field. International Journal of Advanced Robotic Systems 4(1), 103–108 (2007)
8. Fogg, B.: Persuasive Technology. Using Computers to Change What We Think and Do. Morgan Kaufmann, San Francisco (2003)
9. Fogg, B.J.: Captology. the study of computers as persuasive technologies. In: Proceedings of the CHI 1997, Extended abstracts on Human factors in computing systems, p. 129. AMC Press, New York (1997)
10. Fox, D.K.R.: The influence of physical activity on mental well being. Public Health Nutrition 2, 411 (1999)
11. Gates, B.: A robot in every home. Scientific American 296(1), 58–65 (2007)
12. Heerink, M., Krose, B., Evers, V., Wielinga, B.: Observing conversational expressiveness of elderly users interacting with a robot and screen. In: IEEE 10th International Conference on Rehabilitation Robotics, ICORR 2007, June 13-15, pp. 751–756 (2007)
13. Heerink, M., Krose, B., Wielinga, B., Evers, V.: Enjoyment intention to use and actual use of a conversational robot by elderly people. In: ACM/IEEE International Conference on Human-Robot Interaction archive Proceedings of the 3rd ACM/IEEE international conference on Human Robot Interaction, pp. 113–120 (2008)
14. IJsselsteijn, W., Nap, H.H., de Kort, Y., Poels, K.: Digital game design for elderly users. In: Proceedings of the 2007 Conference on Future Play, Future Play 2007, Toronto, Canada, November 14 - 17, pp. 17–22 (2007)
15. Kidd, C., Breazeal, C.: Sociable robot systems for real-world problems. In: IEEE International Workshop on Robot and Human Interactive Communication, ROMAN 2005, pp. 353–358 (2005)

16. Leite, I., Pereira, A., Martinho, C., Paiva, A.: Are emotional robots more fun to play with? In: The 17th IEEE International Symposium on Robot and Human Interactive Communication, ROMAN 2008, pp. 77–82 (2008)
17. Mehta, M., Ram, A.: Runtime behavior adaptation for real time interactive games. IEEE Transactions On Computational Intelligence And AI In Games
18. Neufeldt, C.: Wii play with elderly people. International Reports on Socio-Informatics 6 (2009)
19. Ontañón, S., Bonnette, K., Mahindrakar, P., Gómez-Martín, M., Long, K., Radhakrishnan, J., Shah, R., Ram, A.: Learning from human demonstrations for real-time case-based planning. In: IJCAI 2009 Workshop on Learning Structural Knowledge From Observations, STRUCK 2009 (2009)
20. Ontañón, S., Mishra, K., Sugandh, N., Ram, A.: On-line case-based planning. Computational Intelligence Journal 26(1), 84–119 (2010)
21. Reynolds, C.W.: Competition, coevolution and the game of tag. In: Artificial Life IV (1994)
22. Svenstrup, M., Hansen, S.T., Andersen, H.J., Bak, T.: Pose estimation and adaptive robot behaviour for human-robot interaction. In: Proceedings of the 2009 IEEE International Conference on Robotics and Automation, ICRA 2009, Kobe, Japan (May 2009)
23. Tranberg, S., Svenstrup, M.: An adaptive robot game. In: ISR 2010 (2010)
24. Wendel-Vos, G., Schuit, A., Feskens, E., Boshuizen, H., Verschuren, W., Saris, W., Kromhout, D.: Physical activity and stroke. a meta-analysis of observational data. International Journal of Epidemiol. 33(4), 787–798 (2004)
25. Xavier, J., Pacheco, M., Castro, D., Ruano, A., Nunes, U.: Fast line, arc/circle and leg detection from laser scan data in a player driver. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, ICRA 2005, April 18-22, pp. 3930–3935 (2005)
26. Zlotnik, H. (ed.): World Population Prospects - The 2004 Revision, Highlights (2005), United Nations, Population Division/DESA at, `www.unpopulation.org`