Towards the automatic invention of simple mixed reality games

Robin Baumgarten, Maria Nika, Jeremy Gow and **Simon Colton**[1]

Abstract. The invention of mixed reality games that combine virtual and physical play offers a rich and challenging application area for AI techniques. We look at the possibility of using descriptive machine learning to automatically invent simple mixed reality games. Specifically, we demonstrate that the HR learning system can generate coherent domain knowledge from the noisy play data gathered from a number of simple physical games. We describe how this could be used to support mixed reality game invention, and discuss the prospects for further work in this area.

1 Introduction

Using AI techniques for game design is not nearly as well researched as using AI for avatars and for non-player characters, even though there is clearly potential to enhance the creativity of game designers. We look here at the possibility of using a descriptive learning approach to automatically invent simple mixed reality games.

Descriptive learning allows interesting concepts and properties of a domain to be discovered from observations, without being restricted to any particular learning goal. Applied to games, it has the potential to automatically discover game-specific domain knowledge (rules, strategies etc.) from observed play. This knowledge could help artificial agents fill a number of roles, without the necessity for providing game knowledge to the agent ahead of play. These include:

Game Player So-called *general game playing* agents can play unseen games without being told game rules or strategies [8].
Game Mediator Agents could mediate play between humans, e.g. taking on the role of a referee or coach.
Game Inventor Domain knowledge could be used as a basis for constructing new rule sets.

The advantage of using descriptive rather than predictive machine learning (see section 2) is that there is no specific goal, and we can simultaneously find hypotheses describing the environment and the particular game being played, which allows a greater understanding to be developed, e.g. to support game invention.

In addition, descriptive learning systems can start with background concepts but no data, and — through the use of third party systems such as constraint solvers, model generators and computer algebra systems — can invent concepts and flesh them out with examples [1]. Such abilities would allow agents to operate in a wider range of applications, e.g. social environments where humans are creating, playing and developing their own games. We envision agents that can join in such social play as a player, mediator or inventor.

[1] The Computational Creativity Group, Department of Computing, Imperial College London, UK. Contact email: sgc@doc.ic.ac.uk

Mixed reality games present a suitable domain for this approach, because the computer is already a natural part of the game, and would not have to be artificially introduced into play. However, they also present a challenging domain, partly because the physical data can be complex and noisy, and partly because of the typical complexity of the game mechanics

We demonstrate here that the HR system [2] is capable of extracting sensible domain knowledge from physical play data, obtained from location tracking of two players engaged in relatively simple physical games, such as Tag (sections 3 and 4). Applying descriptive learning to physical data is an essential first step for the invention of mixed reality games. We then argue (in section 5) that this knowledge can be used to invent new games, as well as discussing other directions for this work.

2 Background

2.1 Descriptive learning & HR

In a *descriptive* machine learning setting, an agent attempts to discover a theory that describes a data set. The theory can consist of example objects, concepts which categorise examples, conjectures which make claims about concepts, and explanations which support conjectures. This exploratory behaviour lacks a specific goal, and can be contrasted with predictive learning where the goal is to solve a specific categorisation problem. Logic-based descriptive learning systems include HR [2], CLAUDIEN [13] and WARMR [7].

HR is a theory formation system which generates a theory starting from an initial collection of example objects, in addition to a set of initial concepts and a set of axioms which relate the concepts, usually expressed in first-order logic. New concepts are constructed from the existing set using production rules, employing heuristic search based on various measures of interestingness [5] to control exploration of the concept space. HR has 17 production rules which each form a new concept by various syntactic manipulations and combinations of existing concept definitions. The production rules that we used in the application here were:

Compose Conjoins the literals of two input concepts.
Exists Abstracts ground values to existential variables.
Match Unifies distinct variables within a single input concept.
Negate Negates literals within a definition.
Size Counts the size of the success set of a clause.
Split Instantiates variables in a definition.

Conjectures are formed by HR during the concept search, by observing patterns in the sets of known examples that the concepts apply to. For instance, if HR noticed that the example set of a newly-formed concept was exactly the same as that of a previously defined

concept, it would make an *equivalence* conjecture stating that the two definitions are logically equivalent. Conjectures can be proved from known axioms and theorems either internally or using a third-party automated theorem prover: this can add theorems to the theory or, if the proof is based on a very simple subset of background knowledge, it can be used to remove trivial conjectures from the theory.

Note that in domains where the data may be noisy, HR is able to make *near-equivalences*, i.e., equivalence conjectures where the truth of the conjecture is only partially supported by the data. The user is able to set a parameter for the minimum fidelity of conjectures (usually in the range 60-80%). For instance, if the user set the value to be 75% and HR reported the conjecture that $A \leftrightarrow B$, then this means that, of all examples which satisfy either property A or property B, at least 75% of them satisfy both properties. The user is also able to specify that the calculation of the fidelity is carried out on only the positive examples of the concepts. This tends to avoid the reporting of near-equivalences between concepts for which the examples are mainly negative, for instance the false conjecture in number theory that the concept of square numbers is equivalent to the concept of prime numbers. While there is no overlap in the positive examples of these concepts, the sparsity of examples on the number line mean that this conjecture has 65% fidelity over the numbers 1 to 100.

As mentioned above, HR's search is driven by heuristic measures of interestingness. These measures can also be used to filter and sort the concepts and conjectures in HR's output. The two measures we use here are *applicability* and *fidelity* of conjectures. Applicability is defined as the number of examples that a conjecture relates. Hence, conjecture about even prime numbers score very low for applicability, as they only describe the number 2. Fidelity is measured as the proportion of examples that support a conjecture, for instance the conjecture that prime numbers are odd, while false, scores highly for fidelity, because it is nearly true — with only one exception.

HR has been applied to a variety of domains, most notably mathematical domains where it has been used to make some interesting discoveries [6]. Of particular relevance to mixed reality games is the extension of HR to work with noisy data in order to learn about the rules of a dice game from vision data [14].

2.2 Mixed reality games

Mixed reality games combine physical and virtual elements in gameplay, and research interest in them has been growing over many years as supporting technologies become more sophisticated and readily available. Early examples included ARQuake [15], an augmented reality version of the first-person shooter Quake, and Mixed Reality Pong [10], where a virtual ball is project onto a tabletop and any physical object can be used as a bat.

Research has expanded to cover mobile mixed reality games (e.g. Phone Tennis [9]) and serious games like Virus Life [12], in which players 'clean' a room to defend territory against a spreading virtual virus that simulates hospital infections. Commercial mixed reality games have now begun to be released, such as Eye of Judgement for Playstation 3. Modern consumer hardware, like the Wii, is better able to support mixed reality games with movement sensors and cameras. Other supporting technologies, such as interactive displays, are gradually becoming more commonplace, e.g. Microsoft Surface [11]. Hence there is great potential for the popularity of mixed-reality games to grow over the next few years.

3 Mining conjectures from physical data

We took a three stage approach to generating domain knowledge from observations of physical game playing:

Data gathering Play data was collected from multiple rounds of several games (section 3.1).

Data encoding Logic based descriptive learning systems, like HR, require input in the form of logical statements. For each game, the play data was encoded as a set of ground first order predicates. These predicates were hand-crafted to describe the physical domain (e.g. relative positions in physical space), but are not game-specific (section 3.2).

Descriptive learning We used HR to form a theory about the data in the given encoding. HR's theory investigation tools to help us identify the most interesting conjectures which described the actions of the players in the game (section 3.3).

The approach is independent of the games studied here, and could be generalised to other game domains — providing suitable data encodings can be designed.

3.1 Data gathering

The Ubisense location tracking system [16] was installed in a medium sized room (approx. 10m by 6m). Each player carried a tracking 'tag' with a single button which they could use to provide additional play data (see Figure 1). To increase the accuracy of the location tracking, the players walked rather than ran, and players remained in sight of the location sensors. Because of these artificial constraints, the games were more simulated than played, although they were still engaging physical activities for the players involved. A more sophisticated approach (e.g. with better tracking technology) might remove these artificial constraints.

We chose three simple physical games, plus one structured physical activity:

Tag One player attempts to catch the other, and when caught they swap roles. Both players constantly clicked their button to indicate they were still more than one metre apart. A 'tag' was indicated by the players temporarily ceasing the clicking.

Easter Egg Hunt A third tag was placed in the room, and the players 'searched' for it: for practical reasons it was actually in sight of the sensors. A player would click the button to indicate they had found the tag, bringing the game to an end.

Human Pong As a simulation of a mixed reality Pong [10], we used a third tagged person as a ball, with the two players acting as bats. The aim of the game is to keep batting the ball back to your opponent. The players used their tag buttons to indicate they were batting the ball, with the 'ball' person confirming this with their own tag button.

Walls Three players moved around the room, and whenever two were near a wall, the third person would click their tag button. This was a structured activity, rather than a game, but served as a good training domain for HR.

The Ubisense installation consisted of a network of four sensors connected to our existing standard network infrastructure, three palm-sized tracking tags (see Figure 1) and a PC running tracking server software to collect data from the sensors. The installation of the system requires careful distribution and calibration of the sensors, especially in complex indoor layouts.

Figure 1. A Ubisense tag. Players used the tag button to record game events.

The sensors record the time and angle of arrival of UWB radio pulses from specific tags, enabling the system to compute each tag's 3D position up to 20 times a second with up to 15cm accuracy. Ubisense does not require optical line-of-sight, but the radio signals are attenuated by water so the human body can cast a solid radio shadow. Thus, detecting and tracking humans effectively requires multiple well-positioned sensors.

Data from physical games is noisy due to the difficulties of accurate location tracking and the possibility that players may violate rules. For our setup, the level of tracking noise depends on the obstruction of the transmitter units from the four receivers. While the system has an average resolution of 15cm, occasional accuracy fluctuations affect the system. The largest inaccuracy we recorded was a movement of about 2 meters within one second. To reduce the impact of these spikes, we applied a central moving average to the data, weighted by the temporal proximity to the current data point.

3.2 Data encoding

For each tag, the location tracking system records a series of points in a real-valued 3D coordinate system with timings and orientation, along with timings of the tag button presses. For our analysis, we ignored the height from the ground as well as the orientation of the sensor.

In order to have HR learn about the games, we discretised this play data and encoded it as sets of predicates from which the system could make conjectures about common patterns. The axioms only represent information about relative locations, which has two advantages over an absolute approach: a) it is independent from the dimensions of the environment, and b) we do not have to worry about the numerical representation of locations in first order logic.

Each game session is encoded using the following predicates:

- `event(E)`: A point in time when a player presses a tag button.
- `player(P)`: Name of a player.
- `wall(W)`: Name of a wall.
- `event_of_player(E,P)`: Identifies the player who caused an event.
- `near_time(E1,E2)`: These two events were less than 2 seconds apart.
- `near_player(E,P1,P2)`: At the event the two players were less than 1.5 metres apart.
- `near_wall(E,P,W)`: At the event the player was within 1.5 metres of the wall.
- `happens_before(E1,E2)`: The first event happened before the second.

3.3 Descriptive learning

We used data from the Walls activity to determine the correct combination of production rules, measures of interestingness and search parameters to maximise HR's chances of finding conjectures of interest. We then used the same setup for the other three physical games. This adds some credence to our claim that our approach can mine interesting conjectures from physical play data for a range of different games.

The data recorded by the location tracking system is not perfect, so we configured HR to form near-equivalence conjectures with a fidelity threshold of 80% correctness. We established this threshold by running a couple of test runs of the tracking system: it was a good balance between reducing noise and preventing HR from excluding less common events. We employed HR's *compose*, *negate*, *exists*, *size* and *split* production rules. The exact choice of rules determines the concepts that HR will generate, and this is a typical initial selection. However, other configurations of the 17 rules are possible which might generate richer concepts at the expense of a larger search space. For example, we could have used the *match* rule that equates two previously distinct objects (e.g. a concept about two objects becomes a concept about one), or the *forall* rule that establishes a relation between an object and all other objects (e.g. something that is larger than everything else). Further work could explore the effectiveness of different rule sets in this domain.

We ran HR for 2000 theory formation steps, each of which results in either a concept or a conjecture being formed, and we examined the resulting conjectures. In particular, we first sorted the conjectures in terms of an equally weighted sum of their applicability and fidelity. We then cross-referenced the conjectures, so that we could identify conjectures which related particular concepts, for instance concepts which include the `event_of_player(E,P)` predicate in their definition. While we still had to look through a number of conjectures which were not interesting, we found that we were able to fairly easily identify some conjectures which captured aspects of the physical games.

4 Illustrative Results

In Figure 2, we present visualisations of the tracking data for an individual round of the four games. The noise in the player's path data is apparent in the fact that the lines are not smooth. Note that some of the larger features of the lines are also due to tracking inaccuracies rather than player movement. HR creates a large number of conjectures, depending on the number of theory formation steps employed. However, after sorting them using the weighted sum described above, the results we present below were usually found near the top of the list, mixed with less interesting (i.e. more obvious) results. However, in a few cases, a more prolonged search was required.

Noise in the test data set caused conjectures to be generated that did not reflect the (intended) rules of the game or were artifacts of too little training data. This could be mitigated by increasing the sample

size and the accuracy of the tracking system, e.g. one could make sure the sensors are not blocked by objects such as furniture.

In the following sections the two players are denoted a and b. For bound variables, we use p and q to denote players, e and f to denote events and u and w to denote walls.

4.1 Tag

In this game, a difficulty for HR is that the player is only caught ('tagged') once at the end of the round. This may be seen as an error in the data by HR when it formulates approximate conjectures. For example, it conjectures that all events were caused by the hunting player. Nevertheless, some interesting conjectures were found:

- When the event is caused by the hunted (player b), there are two players near each other. As there are only two players in the game, this means that the hunted has been caught:

$$\forall e.(event\_of\_player(e, b) \leftrightarrow \exists p, q.near\_player(e, p, q))$$

- When two events happen nearly at the same time, one of them is caused by the hunted (b). This means the hunted has been tagged:

$$\forall e.(\exists f.near\_time(e, f) \leftrightarrow event\_of\_player(e, b))$$

4.2 Easter Egg Hunt

As with Tag, the game structure was difficult for HR to work with, as it is only over once the player reaches the egg. This event only happens once, while the hunt takes longer. Thus there are a lot of negative examples and only very few positives, which result in near-equivalence conjectures that state that the egg is (almost) never found. Once we ignore these however, useful conjectures can be found:

- Whenever the event is caused by the egg, it is near a player (that holds because an event can only be caused by one player):

$$\forall e, p.(event\_of\_player(e, egg) \rightarrow (event\_of\_player(e, p) \leftrightarrow \exists q.near\_player(e, q, p)))$$

- Whenever the egg is detected by player b there is also another event at nearly the same time. This indicates that a player pressed the button because the egg was found:

$$\forall e.(event\_of\_player(e, b) \rightarrow \exists f.near\_time(e, f))$$

4.3 Human Pong

A difficulty here was the exceptionally high rate of trivially true conjectures. For example, for all events e, there exists an event f such that e happens before f. While this is not true for the last event, it is valid for all other events and thus matches most of the experimental data. Note that the ball is actually a player in this experiment, which is reflected in the formalisation. We found the following conjectures in HR's output:

- Whenever there is an event, nobody is near player a if and only if somebody (i.e. the ball) is near b. In other words, whenever a button is pressed the ball is near one of the players:

$$\forall e.(\nexists p.near\_player(e, p, a) \leftrightarrow \exists q.near\_player(e, q, b))$$

- Successive events are not both caused by the ball. This is due to both the ball and the player close to it pressing their buttons at the same time:

$$\forall e, f.(happens\_before(e, f) \rightarrow \neg(event\_of\_player(e, ball) \wedge event\_of\_player(f, ball)))$$

- When two events happen at nearly the same time, the first event will be caused by the ball. This is an effect of the discretisation:

$$\forall e, f.(happens\_before(e, f) \rightarrow (event\_of\_player(e, ball) \leftrightarrow near\_time(e, f)))$$

- When two events happen at nearly the same time, either a or b is close to another player (which must be the ball, according to the first conjecture):

$$\forall e, f.(near\_time(e, f) \rightarrow \nexists p.near\_player(f, p, a) \leftrightarrow \exists q.near\_player(f, q, b))$$

- Whenever a button is pressed, two players are close to each other (recall that one of the 'players' represents the ball):

$$\forall e.\exists p, q.near\_player(e, p, q)$$

4.4 Walls

HR found two conjectures that are very close to the 'rules' of the Walls activity. Firstly, one player clicks (i.e. there is no second event at nearly the same time) iff exactly two people are at the wall:

$$\forall e.(|\{p : \exists w.near\_wall(e, p, w)\}| = 2 \leftrightarrow \nexists f.near\_time(e, f))$$

Secondly: multiple players click iff three players are at the wall:

$$\forall e.(|\{p : \exists w.near\_wall(e, p, w)\}| = 3 \leftrightarrow \nexists f.near\_time(e, f))$$

Other conjectures HR found (with similar applicability and matching examples) describe side-effects of the above rules, or coincidences in the data. For example:

- At all events, there was never exactly one person at a wall:

$$\forall e.|\{(p, w) : near\_wall(e, p, w)\}| \neq 1$$

- Whenever a player presses a button, he is only standing at one wall or at no wall (u denotes a wall):

$$\forall e.|\{p : event\_of\_player(e, p) \wedge near\_wall(e, p, w)\}| = |\{(b, u) : near\_wall(e, b, u) \wedge event\_of\_player(e, b)\}|$$

- Whenever more than one player presses the button, each player stands near exactly one wall:

$$\forall e.|\{(p, w) : near\_wall(e, p, w) \wedge event\_of\_player(e, p)\}| = 1 \leftrightarrow \exists p.near\_time(e, p)$$

The latter two are coincidences of the play data, as it is also possible for players to stand in corners.

5 Future work

5.1 Inventing mixed reality games

The rules recognized by HR for these games can also be used to create games for players. By combining the output from HR when applied to the analysis of different games and using this as input data for a new HR session, we speculate that the system can be used to create new games. This could be achieved by forming new theories of games using the existing conjectures and concepts as background knowledge. The created game rules can then be used to guide movements of human players, with the goal of the human players being to guess the intentions of the computer, reversing the role of creator and learner. In particular, we envisage the following approach to the invention of guessing games:

- HR is used to produce theories about various mixed reality games, given data about the movement and actions of players (as above).
- From these theories, we extract two types of conjectures which are supported (at least partially) by the data. Firstly, conjectures which are true in multiple games. These are likely to be axioms of the physical environment, e.g. that a person cannot be close to three or more walls at the same time. Secondly, conjectures which are true only of an individual game. These are likely to contain concepts which can be used as ingredients in novel games, e.g., being near a wall, or clapping twice, etc.
- A new HR session is started, with the same background concepts as in the previous sessions, but without data for any of the concepts. In this mode, HR requires a mechanism for generating data to illustrate new concepts. (E.g. in [3] number theory concepts are given as background information without data and the Maple computer algebra system is used to generate data for new concepts.) In our context, we could use a constraint solver (as in [1]): whenever HR invents a concept, the constraint solver will be employed to generate data for it. The conjectures extracted from individual games will hopefully provide interesting ingredients for novel games.
- We can use our axioms of the physical environment to preprogam the solver with appropriate automatically generated constraints (see [1]) about the physical world, to prevent it from inventing physically impossible concepts.
- HR will produce a general theory of mixed-reality games, which will include various concepts which are physically possible. As in [4], we will enable HR to extract from the theory a set of mutually-possible concepts. The conjunction of these concepts will express a pattern of movements/actions for players in a mixed-reality game which is large enough to embed a pattern which is neither too obvious nor too convoluted. We envisage much experimentation in order to determine a suitable balance.
- Given the concept to embed in a series of movements and actions, we will employ a constraint solver to generate such a series which upholds both the physical axioms and the properties expressed in the concept. These will be given concretely as a set of timings for movements and actions for a set of players. The purpose of the game will be for the players to attempt to determine the underlying pattern that they are expressing, i.e., a guessing game.

Obviously, there is much work to do to in order to achieve such invention of mixed-reality guessing games. However, we believe that such an approach to inventing guessing games is certainly possible, and we plan to implement the methods required to achieve it.

5.2 Improvements to descriptive learning

A common challenge faced when applying HR to a new domain is the large number of uninteresting conjectures made along with the more relevant results. As mentioned above, we encountered a similar problems in this work. For example, HR correctly picks up uninteresting physical constraints that are independent of the game being played, e.g. a player cannot be near three walls at the same time. A typical solution that we could try is to filter trivial conjectures using a theorem prover: if a conjecture can be easily proved from a basic domain knowledge base (e.g. about players and walls) then it is removed, as per the application to number theory described in [3]. Alternatively, as described above, we could compare the conjectures generated from multiple different games, and assume that any conjectures appearing in multiple games actually describe axioms of the physical world (hence are not particularly interesting) rather than aspects of the game being played.

It is unclear how results from our approach could be more formally evaluated, other than simply reflecting on how well they describe the games. One approach might be to compare generated knowledge with first order logic versions of the intended game rules and known player heuristics. A theorem prover could be used to establish implication or equivalence between subsets of human- and machine-generated domain knowledge. This raises more general questions about how the system itself could distinguish between rules, player heuristics and coincidences.

5.3 Other applications

Apart from invention, mixed-reality games could benefit in other ways from knowledge about rules and theories concerning the behaviour of human players. Such knowledge could be used to guide computer players or computer-mediated play between humans. Learning from physical game data is also of interest itself, and descriptive learning in this domain has potentially interesting applications, e.g. coaching in sports education and training. The presented approach could be applied to a wider spectrum of physical games.

Having demonstrated early results in the physical domain, we are hopeful that our approach will have applications in other game domains. We are currently working on applying descriptive learning to combinatorial board games, where generated domain knowledge can be used to improve the performance of General Game Playing agents which can play unseen games without needing to be told domain-specific strategies [8]. Another potential application is video games, to facilitate intelligent game adaptation based on automatic analysis of player behaviour.

6 Conclusion

We have shown that it is possible to learn domain knowledge about physical games through a combination of a location tracking system, a discretisation algorithm and a descriptive machine learning system. HR was able to describe the physical activities after the data had been filtered with a discretisation process, which abstracted the data into a relative and environment-independent form. The setup of HR to handle noisy data is not trivial — for Pong and Easter Egg Hunt especially we encountered a large number of conjectures that were not interesting with respect to the games themselves. We discussed how to alleviate this by modifying the data representation and by introducing additional filtering mechanisms that use background information about the environment to remove unwanted conjectures.

We also discussed a way to use these uninteresting conjectures as axioms of the physical world in which the games are played.

While the work presented here is somewhat preliminary, and the automatic invention of mixed reality guessing games will require much additional work, we hope to have demonstrated the principal that raw data from a physical games can be turned into descriptions of the environment and the games being played. Looking at the nature of the raw data in Figure 2, we believe it is an achievement — albeit modest — to have succeeded in the first stage. In the second stage, we hope to demonstrate the potential for descriptive machine learning systems to innovate in game design — firstly with simple physical games, and eventually with fully featured mixed reality games.

ACKNOWLEDGEMENTS

We would like to thank the anonymous reviewers for their comments. This work is partly funded by EPSRC grant TS/G002835.

REFERENCES

[1] John Charnley, Simon Colton, and Ian Miguel, 'Automatic generation of implied constraints', in *ECAI 2006, 17th European Conference on Artificial Intelligence*, eds., G. Brewka, S. Coradeschi, A. Perini, and P. Traverso, pp. 73–77. IOS Press, (2006).

[2] Simon Colton, *Automated Theory Formation in Pure Mathematics*, Springer-Verlag, 2002.

[3] Simon Colton, 'Automated conjecture making in number theory using HR, Otter and Maple', *J. Symbolic Computation*, **39**(5), 593–615, (2005).

[4] Simon Colton, 'Automatic invention of fitness functions with application to scene generation', in *Applications of Evolutionary Computing, EvoWorkshops 2008*, ed., M.Giacobini et al., volume 4974 of *LNCS*, pp. 381–391. Springer, (2008).

[5] Simon Colton, Alan Bundy, and Toby Walsh, 'On the notion of interestingness in automated mathematical discovery', *Int. J. Human-Computer Studies*, **53**(3), 351–375, (2000).

[6] Simon Colton and Stephen Muggleton, 'Mathematical applications of Inductive Logic Programming', *Machine Learning*, **64**(1–3), 25–64, (2006).

[7] Luc Dehaspe and Hannu Toivonen, 'Discovery of frequent DATALOG patterns', *Data Min. Knowl. Discov.*, **3**(1), 7–36, (1999).

[8] Michael R. Genesereth, Nathaniel Love, and Barney Pell, 'General game playing: Overview of the AAAI competition', *AI Magazine*, **26**(2), 62–72, (2005).

[9] Anders Henrysson, Mark Billinghurst, and Mark Ollila, 'Face to face collaborative AR on mobile phones', in *ISMAR '05: Proceedings of the 4th IEEE/ACM International Symposium on Mixed and Augmented Reality*, pp. 80–89, Washington, DC, USA, (2005). IEEE.

[10] K. Kallio. Mixed reality pong, 2001. Web page, accessed Feb 2009: `http://www.mlab.uiah.fi/~kkallio/mr-pong/`.

[11] Microsoft Corporation. Microsoft surface, 2009. Web page, accessed Feb 2009: `http://www.microsoft.com/surface/`.

[12] Maria Nika, *Virus Life: An infection spreading game with location tracking*, Master's thesis, Department of Computing, Imperial College London, 2008.

[13] Luc De Raedt and Luc Dehaspe, 'Clausal discovery', *Machine Learning*, **26**(2–3), 99–146, (1997).

[14] Paulo Santos, Simon Colton, and Derek R. Magee, 'Predictive and descriptive approaches to learning game rules from vision data', in *Advances in Artificial Intelligence: IBERAMIA-SBIA 2006, 2nd Int. Joint Conf., 10th Ibero-American Conf. on AI, 18th Brazilian AI Symposium, Brazil*, eds., J.Šimão Sichman, H. Coelho, and S.Õ. Rezende, volume 4140 of *LNCS*, pp. 349–359. Springer, (2006).

[15] Bruce H. Thomas, Ben Close, John Donoghue, John Squires, Phillip De Bondi, and Wayne Piekarski, 'First person indoor/outdoor augmented reality application: ARQuake', *Personal & Ubiquitous Computing*, **6**(2), 75–86, (2002).

[16] Ubisense. Ubisense system overview. Available from Ubisense website `http://www.ubisense.net/` (accessed Feb 2009).

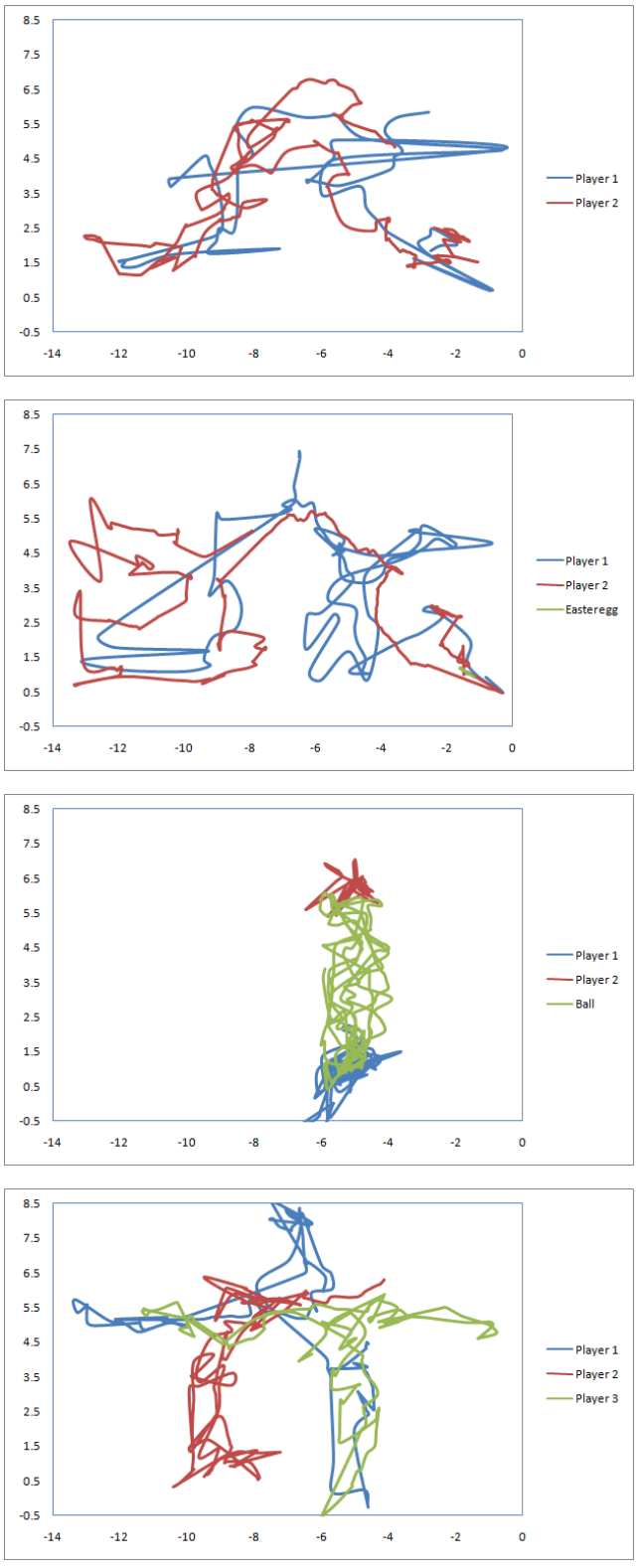

Figure 2. Tracking data showing player paths during a round of: (a) Tag (b) Easter Egg Hunt [with the egg located in the bottom right hand corner] (c) Human Pong, and (d) Wall.