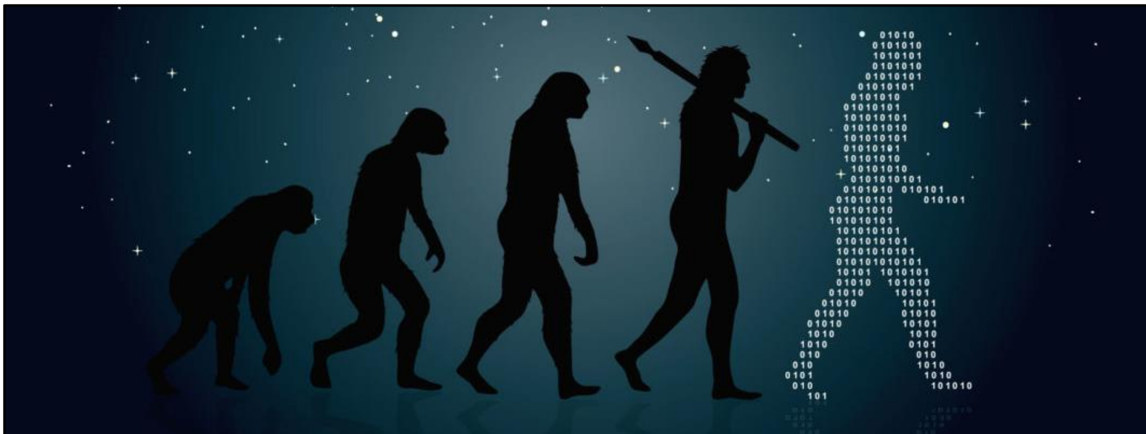


Game-Playing Perspectives: Planning for Success

Raluca D. Gaina

Talk outline

- Who am I?
- Why Game AI Research
- My PhD: Rolling Horizon Evolutionary Algorithms
- Next steps, impact

I'll start by introducing myself, and talking a little bit about where I work as well as my motivation for what I do. Then I'll be talking about what I actually do in my PhD. I'll be covering a lot of different topics and ideas, hopefully you'll find something of interest in there! And I'll conclude by talking about my next steps in approaching the end of my PhD, some visionary thoughts and what potential meaningful applications of all this can be! The hope here would be that you and young students would find these ideas interesting, exciting, inspiring. Games are a wonderful topic to study and I hope I'll be able to convey this in my talk.

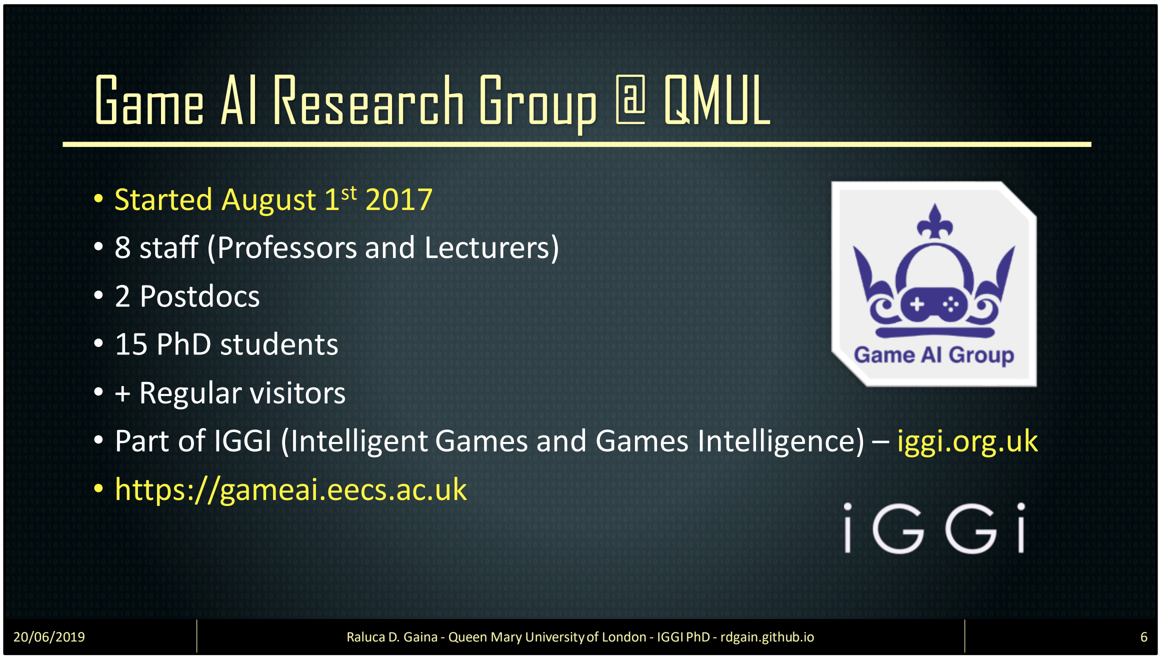
Introduction

My name is Raluca Gaina and I am originally from Romania. I've been studying for a long time in the UK, though. I've done my Bachelors in Computer Games at the University of Essex, then I continued with a Masters in Computer Games at the University of Essex and I'm currently in the 3rd year of my PhD in Artificial Intelligence in Games at Queen Mary University of London.

You may see a trend there (and I'm not talking about the many years I've spent at the University of Essex). Although the only woman in my degree up until the PhD level, I've kept pursuing Computer Games- specific degrees. And my main motivation behind it when I first started was fairly simple: I liked games. But not only that, I also liked programming. Yet I didn't want to do a generic Computer Science degree, I wanted to do * fun programming *! And learning to make games was a way to achieve this.

The degrees I've taken have been quite heavily programming-focused, about the specific game development part. But I've also learned about game design which is very similar to story writing, except you're defining the world in all its finest details. And although I am very much a programmer myself, it's worth emphasizing that there's so much more to games than code! There's design, arts, narrative and character and sound design, marketing and project management and so many more areas! With such a wide coverage of topics, I believe there’s something for anyone in the games industry and I find that fascinating.

I currently work at the Game AI Research Group at Queen Mary University of London, filled with all of these wonderful people.



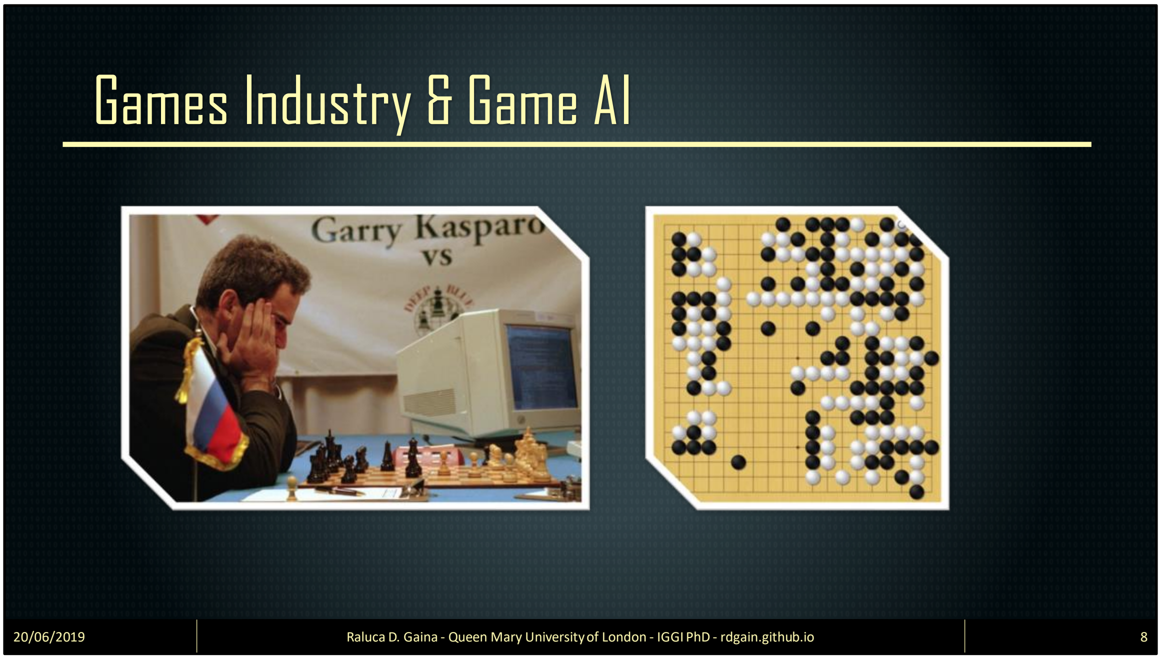
This is a fairly new research group at QMUL, started only 2 years ago, yet it's grown very quickly. We currently have about 8 staff (professors and lecturers), 2 postdoctoral research associates and 15 PhD students! We regularly receive visitors from all over the world as well who engage in our projects.

The group is also part of the Intelligent games and games intelligence programme, or IGGI, which funds PhD students (including myself) with the big goal of bringing the games industry and academic research closer together through various initiatives, including our very own games conference happening in September every year.

In IGGI we regard games research from two different perspectives: Games for AI and AI for games. On one hand, we can study artificial intelligence technology which would improve games and player experience. And on the other hand, we can study games as environments to develop Artificial Intelligence in and push the boundaries of what current methods can do. In my research I mostly focus on the second side of the coin.

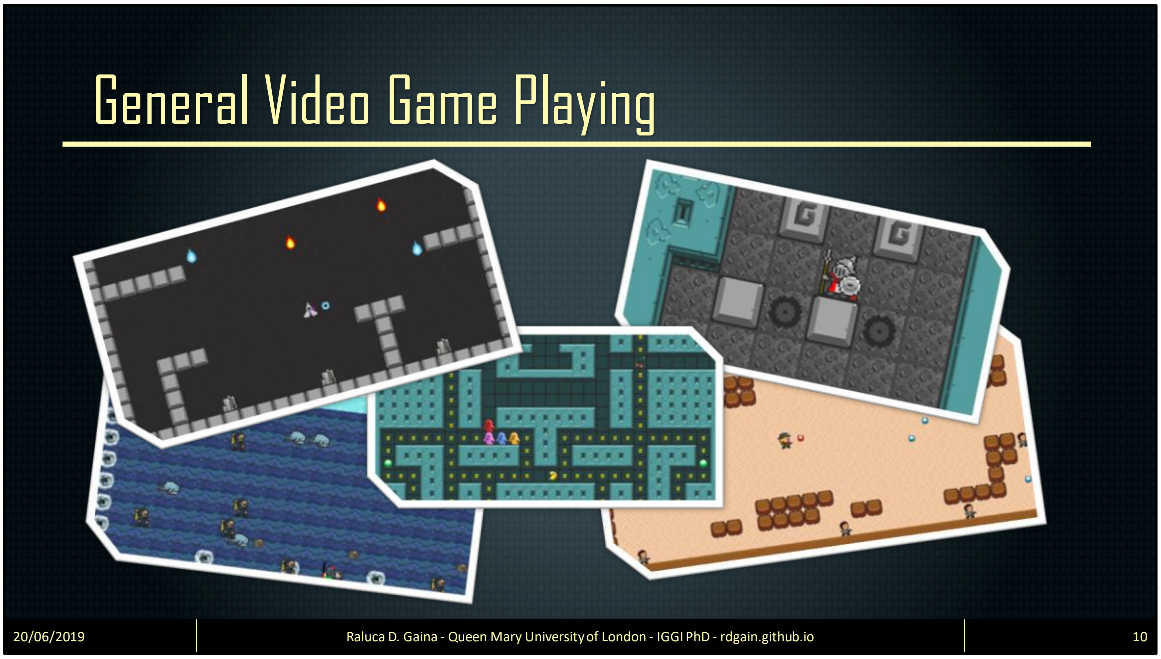
Games model the world as we see it, or we may wish to see it. Sometimes they break it down into simpler problems so that the whole concept can be more easily digested - like making bread! But if we develop artificial entities that behave intelligently in games, in these simulations of life, the step of bringing them into the real world would be trivial.

Video games involve real-time decisions and a need for adaptive behaviour: some may require in-depth strategic planning, while others may be simply all about shooting the enemies as fast as possible. But it is not only the wide variety of challenges that is interesting and useful for research. Games are cheap to run, as opposed to experiments with robots where you may have to buy new parts if things go wrong. Games are fast to run and millions of simulations can be run very quickly to provide quick feedback on the researcher’s work. And the complexity of games can be varied as needed, from very simple tasks, to complex simulations of real-life.



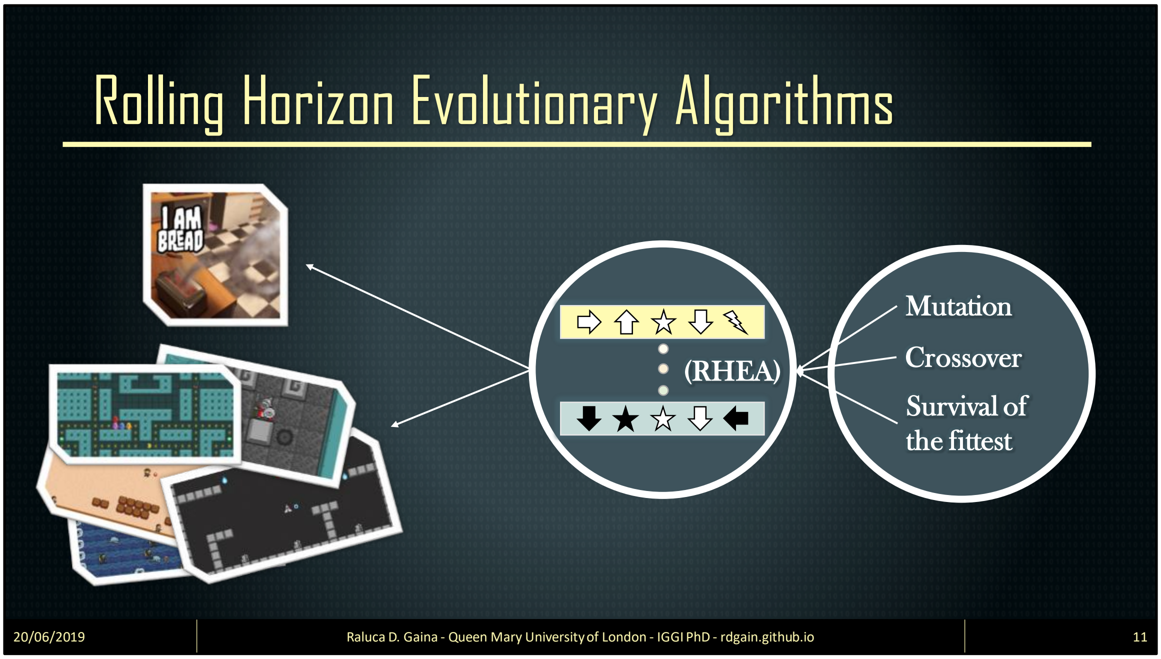
Using games as benchmarks for AI has been done for many years.
Highlights: Chess and Go.
They have allowed advancement of AI on multiple fronts and algorithms: alpha-beta, monte carlo tree search and deep learning.

My PhD



There are many problems in the real world, that we learn to deal with as we go through life. And the variety of games matches the variety of real-life problems or tasks. A lot of game AI research is focused on developing artificial players that get really good at a particular game. But, take that one player out of its comfort zone, and it becomes relatively useless. For example, asking an AI player that is super-human at chess, to play a car racing game, would most likely result in a crash of either the program, or the car.

And that's where the quest of general intelligence comes in. In the games world, in its simplest form, that can refer to an AI player that is skillful at a range of different games, even ones it hasn't seen before. And that is what my focus is on: I would like the programs and algorithms I develop to be widely applicable to a range of tasks, from puzzle games like Sokoban, to shooter games like Space Invaders, and to adventure games like Zelda.



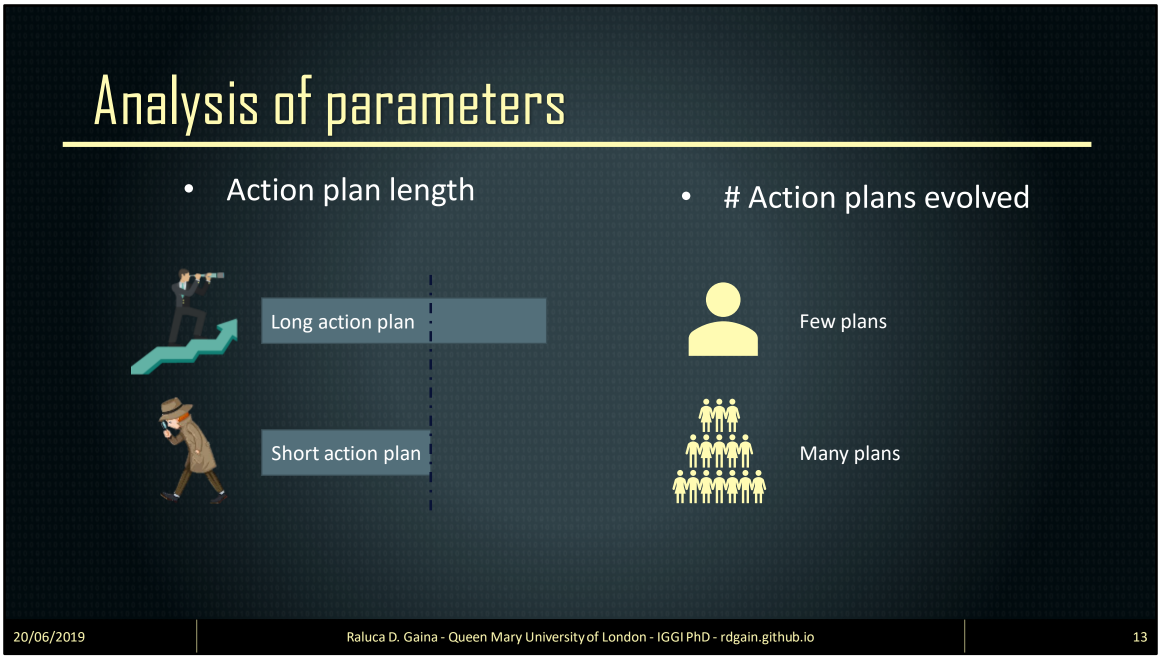
So my focus is on being adaptive to the different games being played, and even to the different scenarios a player may encounter while playing one game. Therefore, I use evolutionary algorithms which create action plans while playing games. We call this form of applying these algorithms as Rolling Horizon Evolutionary Algorithms (or RHEA for short), as the plan of actions the AI creates rolls forward into the future as the gameplay unfolds. They use the same concepts of evolution (survival of the fittest, mutations, combinations of genes) to create better and better plans of actions to be able to react to the unexpected and even come up with clever solutions to puzzle games. And unlike traditional applications of evolutionary algorithms, all the computation happens in real-time, while playing the game – so the program is actually given very little time to come up with meaningful decisions with no knowledge about its environment.

My PhD so far

- 5 core papers published + demo

- Analysis of parameters
- Seeding
- Enhancements and hybrids
- Simple macro-actions in physics-based games
- Dynamic action plan length adjustment

So far in my PhD I've published 5 core papers, as well as a demonstration of the variations of this method on a wide range of different games. I'll talk a little bit about the different ideas involved in each of the projects.

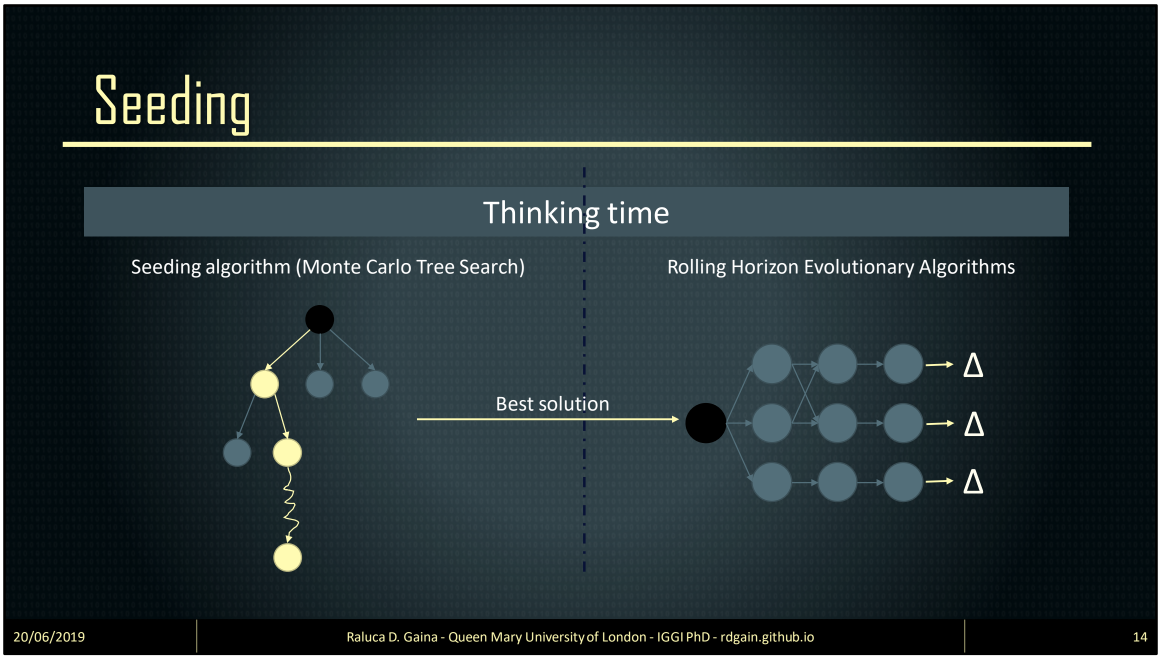


The first study I did focused on an analysis of the most basic form of the method, which evolves a population of individuals, where each individual is one action plan (so it keeps a collection of action plans to choose from and combine in order to find the best combination of actions). An action plan here can be seen as 'move up, move right, shoot, move left', so it is all composed of primitive actions and the planning does not happen in the high-level goal space (which would instead be 'get the key, go to the door while avoiding all enemies on the way'). Humans do think in high-level goal terms, as well as how to achieve each small individual goal, but automatic methods for identification of these high-level goals are not yet proficient or general enough for use in this context.

RHEA has two main things that tell it what to do: the length of the action plans it evolves, and the number of action plans it considers in its collection at once.

We can observe a tradeoff here, given a limited thinking time: the method can either evolve multiple short action plans at once, or fewer longer action plans. The benefits of the first option is that it is able to gather more statistics about what is good and what is not good in its immediate vicinity. However, it may ignore good things that may happen further into the future, and that is why sometimes it is actually better to try out fewer action plans that are longer. I've tested various combinations of these parameters on a range of different games and generally, the more action plans you can consider, and the longer they are, the better. But this is not always the case, sometimes less means more!

But at the end of this study, the best combination found was still not as good as the dominant algorithm in the general video game playing area, called Monte Carlo Tree Search.

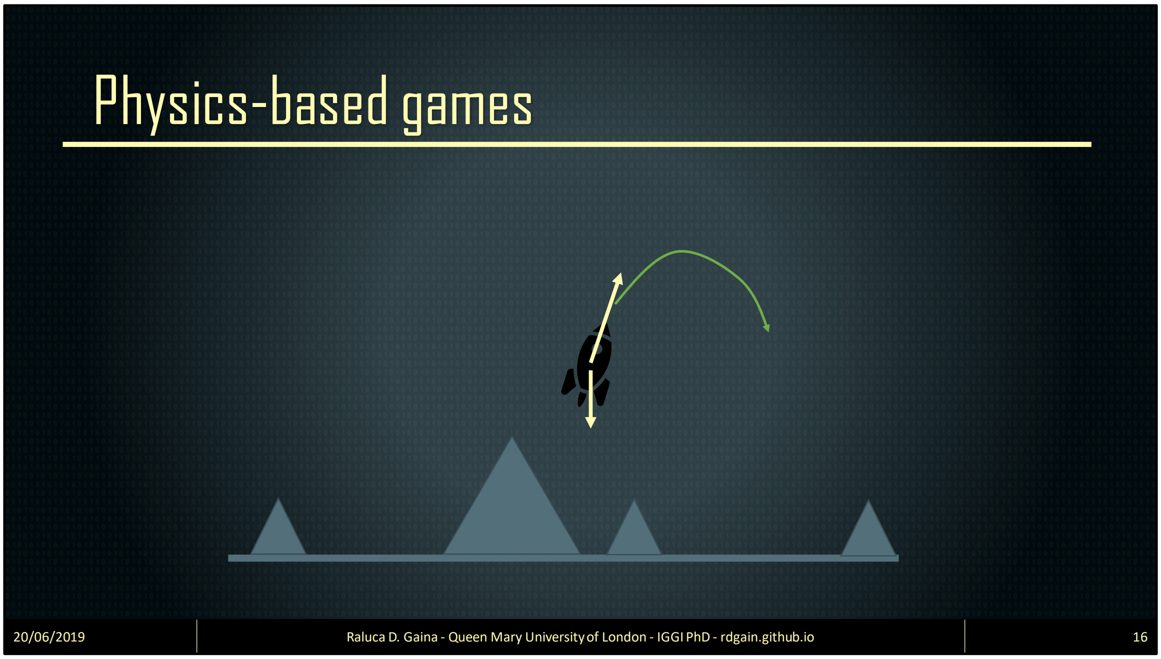


So I tried to make it better by testing what if we start the evolution from an action plan given by another algorithm. The basic method I've mentioned before starts evolving from random action sequences: so the theory is that if we start from an action plan that's already considered to be good (like what Monte Carlo Tree Search could offer), our final plan would be even better! And this does indeed hold true, even if we keep the same thinking time for the method, but we divide this equally between the two methods we use.

Enhancements and hybrids

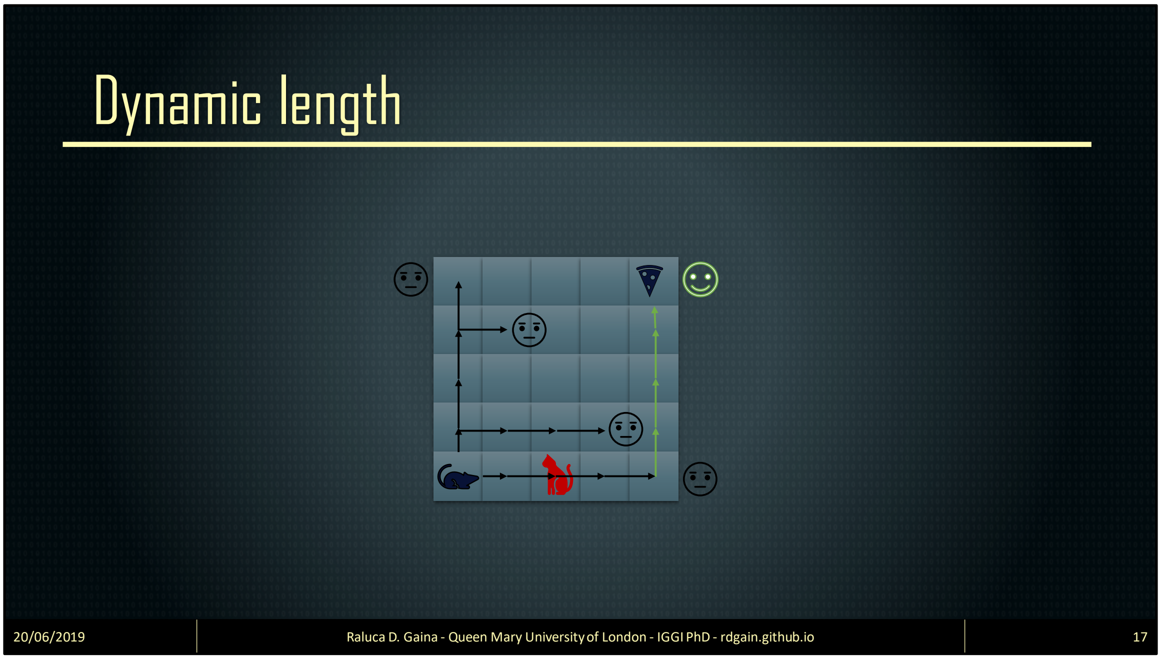
- Integration into the Evolutionary Algorithm of other methods
- Simple but efficient: shift buffer
  - Save knowledge while playing through the game
- Good performance in different types of games

Since we saw that combining methods works so well, we've tried more of these combinations, or hybrids. So we took various concepts from other areas and integrated them into our evolutionary algorithm. The best one that we've found to work great in most situations is also the simplest idea, which we call a 'shift buffer': instead of starting all the computation from scratch at every game step, why not save the best action plan we've found before and keep evolving and improving it? This saves knowledge acquired in the past, as well as some of our thinking time!



So far, all the projects I've mentioned tested the algorithms in grid-based games, that is, games where levels can be represented in a two-dimensional grid. A next challenge we dealt with from there is physics-based games. Although still two-dimensional, all the objects in these games were affected by physics and moving right, for example, did not mean simply move one square to the right, but actually apply a force in that direction, which may be affected by other forces in play at the time. As an example, take this rocket landing game, where the aim would be to land the rocket on a flat surface, so avoiding the spikes. Gravity acts on the rocket constantly, the downward force, but the player can also apply a counter movement force, say upwards, which would actually result in a trajectory similar to that drawn in green.

The AI players, therefore, have to deal with more complex challenges and ideas about how the world works. And because movement actions have very small effects, we've added something we call macro-actions: this means that each action in the plan the algorithm evolves is actually a repetition of the same action several times. So, instead of having a plan like 'move up, then right, then down' – it would now be 'move up 5 times, then right 5 times, then down 5 times'. This not only has an effect on the environment that the AI can better distinguish, but it also gives the AI more thinking time, as it can better plan while it knows that it's only supposed to repeat an action for a few game steps. And this did turn out to work very nicely in several games we've tested, although some do require more precise movement and a more dynamic approach should be taken.



So we've explored one dynamic approach, returning to grid-based games and with a particular scenario in mind: in a lot of games used for research, there is some form a score system that tells the AI player if it's doing good things or not (for example, collect a coin, get a point!). But in a lot of games, and a lot of commercial and modern games, this is not the case: some games are about exploring the world, or having a particular experience. And in some games it may be the case that in one part of the level there are a lot of rewards (things that give points to the algorithm), while in other parts the world looks empty from the AI's perspectives.

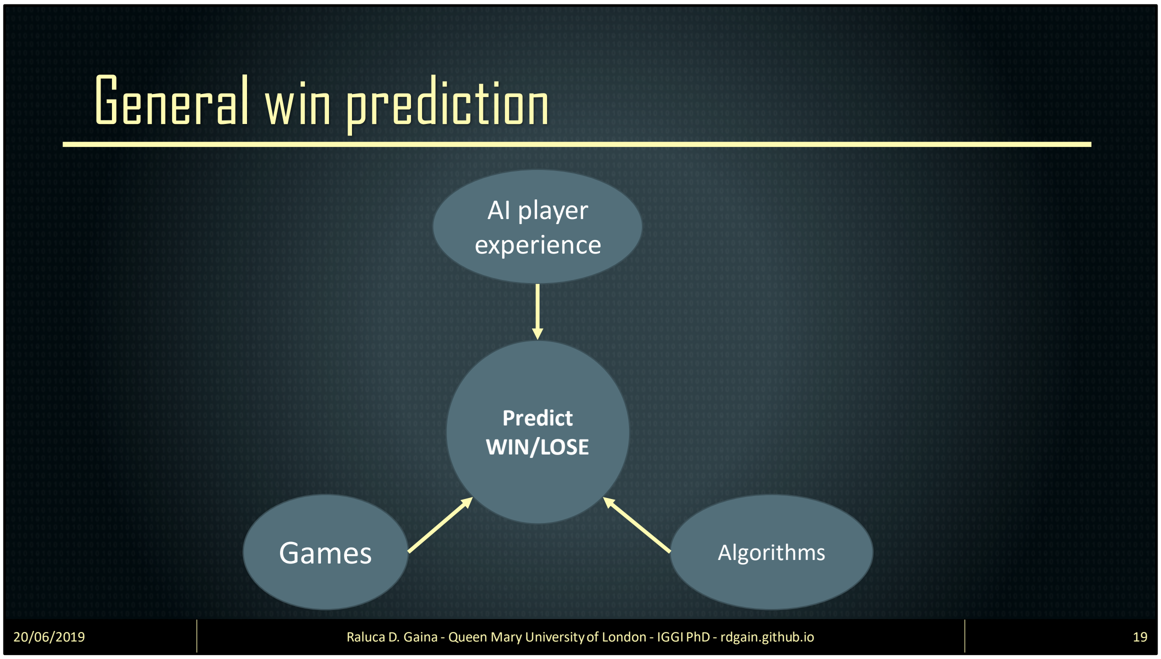
A very simple example scenario is presented on the slide, where the mouse aims to get to the cheese, but due to its short action plans, it doesn't really know where the cheese is. If we made the action plan very long, it might be able to find the cheese, but it could also accidentally ignore the immediate threat of the red cat. So, ideally, it should be able to avoid the nearby threat while also looking for the further ahead reward. And that is what we've done by using the agent's perception of the environment to dynamically adjust the length of the action plans it evolves: these would be shorter if it needs more statistics about immediate rewards, or longer if the world looks empty and it needs to find something interesting that's currently out of reach.

As a wrap on this part, my results so far have shown my technique, with various add-ons, is able to perform better in many games. Yet many still remain too difficult for any Artificial Intelligence algorithm to solve, let alone understand how they work.

Other projects and interests

- General win prediction
- Game design
- AI player and game parameter tuning
- General multi-player game learning: MARLO (Microsoft Research)
- Multi-action card games
- Multi-player partial observable games: Pommerman
- Learning world models
- General level generation
- Learning to play video games from sounds

I've also been very lucky to be involved in various other side projects, from automatic game design, to various other types of games (multi-player, multi-action, card games, hidden information). I'm not going to talk about all of them, but if you spot something that catches your eye, I'll be happy to talk more about it later. I would like, however, to touch on two of the projects mentioned here.



The first project I'd like to talk a bit more in detail about addresses an interesting aspect of games research. Usually, the question of whether the correct algorithm is used for the problem at hand comes at the end of an experiment, when the algorithm's ability to solve the problem (or not) can be verified. But what if this question could be answered in advance, with enough notice to make changes in the approach in order for it to be more successful? Predicting AI player performance before it even starts playing, or very early on in its run through a game, would be very useful: if the player looks like it's going to win, that's great! If it doesn't, then we can consider other alternatives available for obtaining better results. Consider the problem of complex modern games, which take hours or days of human play to complete. Even if sped up for the AI, the execution of a single run through the game would not only take a long time, but it might also not explore any meaningful game spaces or paths. Therefore, having an idea about how the player is going to do without waiting for it to complete a game could speed up the testing-feedback loop.

Several approaches have tried to do this from a game analysis perspective: looking at what sort of objects exist in the game, and their properties, and deciding if the current approach is being effective or not. For example, in Starcraft you could analyse the resources gathered by players or the units they're building and controlling to decide which player is going to win. But when you switch games, you'd need to design a completely new system that takes into account the specific properties of the new game.

In this work, we've tried to base our prediction exclusively on the agent's experiences, and its decision making process, extracting some information that is applicable to many intelligent algorithms. And we've found that this is actually informative enough to give a fairly good indication of what's going to happen in the game, even when using very simple methods with no customization. Our system uses three different sources of information to base its predictions on, namely the AI player's experience, various games from which we collected this data and various algorithms that we used to play the games and extract data from. We've also seen that his sort of system is good enough to even predict results for games or algorithms that were not used in its initial training, and it would be exciting to observe what happens if more complex methods are applied with these concepts in mind.

And one other exciting project I'd like to talk about is learning to play video games from sounds. One idea that's come out of a PhD breakdown of 'I want to change my topic!' which I've been trying to pitch this idea to different researchers for quite a while now and someone finally got on board, so we've written up our concepts into a first short paper, recently accepted for publication.

The idea behind this is that game-playing AI research has focused for a long time on learning to play video games from visual input or symbolic information. However, humans benefit from a wider array of sensors which we utilise in order to navigate the world around us. In particular, sounds and music are key to how many of us perceive the world and influence the decisions we make.

More so, sound and music has long been an important aspect of video game development and play. Not only can audio greatly influence our engagement and emotional investment in a game, but it can also provide important environmental information or gameplay cues. Sounds within games can be used to alert the player to a nearby hazard (especially when in darkness), inform them they collected an item, or provide clues for solving certain puzzles.

So we did some initial experiments on AI players learning to play video games solely from audio cues, on 3 different audio games, including an analysis of the games and the audio game design process. Although very simple, they showed that having this type of input can be beneficial for AI players.

This line of research addresses issues of inclusivity and accessibility in games as well. People who may be partially or completely blind rely exclusively on audio to play a large number of video games effectively. Including audio as well as visual information within a game can make completing it much more plausible for visually impaired players. Additionally, individuals with hearing difficulties would find it hard to play games that are heavily reliant on sound. Intelligent artificial players can help to evaluate games for individuals with disabilities: if such a player is able to successfully play a game using only audio or visual input, then this could help validate the game for the corresponding player demographics.

So after all that, what's next?

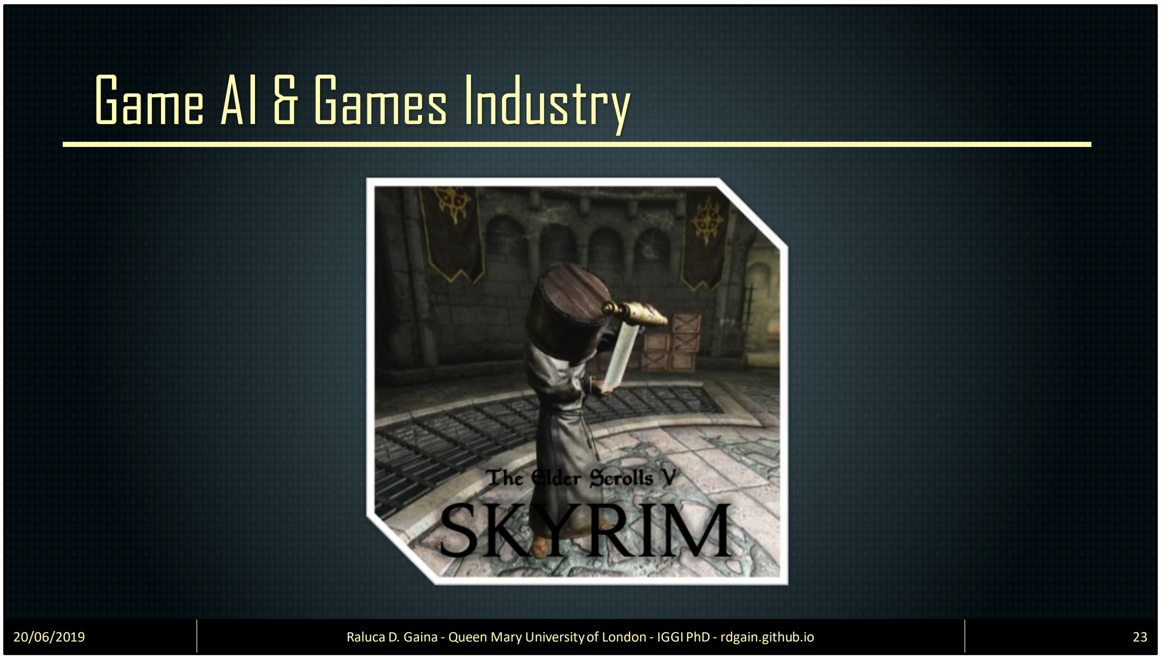
Next steps

- Artificial entities: Project Thyia
- Combining planning and learning
- Many games
- Optimisation
- Knowledge base
- Analytics
- Human interaction

https://rdgain.github.io/assets/pdf/thyia-cog.pdf

The space of Artificial Intelligence entities is dominated by conversational bots. Some of them fit in our pockets and we take them everywhere we go, or allow them to be a part of human homes. Siri, Alexa, they are recognized as present in our world. But a lot of games research is restricted to existing in the separate realm of software. We enter different worlds when playing games, but those worlds cease to exist once we quit. Similarly, AI game-players are run once on a game (or maybe for longer periods of time, in the case of learning algorithms which need some, still limited, period for training), and they cease to exist once the game ends. But what if they didn't? What if there existed artificial game-players that continuously played games, learned from their experiences and kept getting better? What if they interacted with the real world and us, humans: live-streaming games, chatting with viewers, accepting suggestions for strategies or games to play, forming opinions on popular game titles?

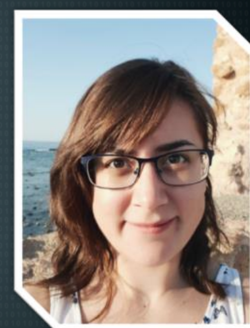
We've recently had a vision paper accepted for publication, where we introduce the vision behind a new project called Thyia, which focuses around creating a present, continuous, `always-on', interactive game-player. This system is a lot more complex than what I've worked with in the past. It aims to combine different methods, both planning and learning, to not only intelligently plan through the current game it's playing, but also learn over time from its mistakes and keep getting better, with no ‘end’ to its running time. This AI entity would be able to play any game given, as before, even ones shared with it by humans. It would change its behavior, parameters and structure to find the best version of itself. It would be building a knowledge base of the games it plays, strategies it uses and interesting events. It would analyse the games it plays, as well as its own gameplay. And it would interact with humans in meaningful ways: be it chatting, or sharing games with us, or playing games alongside humans. An ambitious vision which looks at AI players differently, as entities in their own right.



But we're not yet there, at the point of strong general Artificial Intelligence. Yet I believe we're getting close. And when we do get there, the potential applications are plentiful: we can not only create better games, with characters that interact with us in a meaningful way, instead of accepting buckets to be placed on their heads like all is normal. But we can also formulate any life problem as a "game" and apply these methods in other domains: transport, medicine, exploring Space!

Thank you!

- http://rdgain.github.io
- @b_gum22
- r.d.gaina@qmul.ac.uk



Resources & References

- http://gvgai.net
- http://iggi.org.uk
- http://gameai.eecs.qmul.ac.uk
- https://rdgain.github.io/publications