AI Birds: Obstacles for Reaching Human-Level Performance and a New Role for Qualitative Reasoning

Diedrich Wolter[1], Felix Haase[1], Alexandra Kirsch[2]

[1] University of Bamberg, Germany [2] Independent Scientist

Abstract

Since 2012 the AI Birds competition hosted at major AI conferences sets out to challenge humans by fostering the development of autonomous agents that can outperform human players in a single-player physical simulation game. Unlike several other games, AI agents have not yet come close to human performance, let alone defeated average human players. In this paper we analyze what makes acting in physical environments hard and why computers show poor performance in open-world tasks. By evaluations performed on our agent that currently dominates the competition we aim to pinpoint to fundamental challenges which AI needs to face to ready itself for entering the open world. Our results show that the shortcomings are due to a lack of dynamics in common architectures. We then outline how qualitative reasoning can be applied to achieve a dynamic interplay of AI components.

1 Introduction

The AI Birds competition[1] (Renz et al. 2015) is carried out annually at major AI conferences since 2012. The aim of this competition is to assess the progress in AI towards problem solving in open domains whilst avoiding the challenges of working with technical systems such as robots and their limitations, thus putting a stronger focus on problem-solving skills (Renz et al. 2019). In short, the competition is based on the physical simulation game *Angry Birds* and requires an autonomous agent to catapult birds at structures protecting enemies in order to destroy them (see Figure 1). In a survey among AI researchers, AI Birds was estimated to be one of the next milestones of AI accomplishments in which an AI system will defeat humans by around 2022 (Grace et al. 2018). Since 2016, the BamBirds agent developed at the University of Bamberg participates in the competition and has won the competition three times so far. Like most agents participating, the code of the BamBirds agent is made publicly available.[2] We can thus use the BamBirds agent as a basis to discuss progress in the AI Birds competition. Also, the BamBirds agent can serve as a baseline to explore general shortcomings of today's AI approaches for open-world problem solving. The aim of this paper is to, first, give a description of the BamBirds agent and, second, to identify principle shortcomings that need to be addressed in order to make significant progress towards open-world problem solving in physical domains. A particular focus of this paper is to discuss possible contributions of qualitative reasoning. We also substantiate a claim that the problem areas identified encompass crucial gaps that need to bridged in order to reach for human-like performance in open words.

The remainder of this paper is structured as follows. In Section 2 we first introduce the AI Birds competition and discuss the challenges it encompasses for AI. Thereafter, Section 3 presents the BamBirds agent and discusses the contribution of distinct modules to successful performance in the competition. In Section 4 we then analyse principle limitations of the BamBirds agent that are symptomatic for current AI architectures. We identify research gaps and discuss means to overcome today's limitations. The paper concludes by summing up our key observations and claims.

2 The AI Birds Competition

In the AI competition, agents are confronted with a set of previously unseen levels. Within a set time limit, the agent has to gather as many points as possible by solving a level. The competition is run in several rounds: in the final, the two agents scoring highest in the semi-finals compete with one another; in the semi-final the four agents scoring highest in the quarter-finals compete; and so on, depending on the amount of agents participating. In each round a new set of unseen levels is used. In the finals, agents are typically given 20 minutes to solve 8 levels, allowing them to re-try each level about 2–5 times, depending on the complexity of the levels and agent speed. Each level (see Figure 1 for an example) comprises a set of target objects (green pigs), objects of different kinds, and a sequence of birds that can be launched from a slingshot by performing a drag-and-release operation. Once released, a bird is catapulted from the sling towards the area where pigs are positioned. By placing shots appropriately, all pigs must be destroyed, either by direct hit, or by any other physical impact of sufficient strength. When launching bird after bird once the physical scene has stabilised from the previous impact, a single level may take up to 2–3 minutes to play, depending on how many birds are available and shot. An agent is awarded points only if all pigs are destroyed. Points are determined by the game

[1] aibirds.org

[2] https://github.com/dwolter/BamBirds

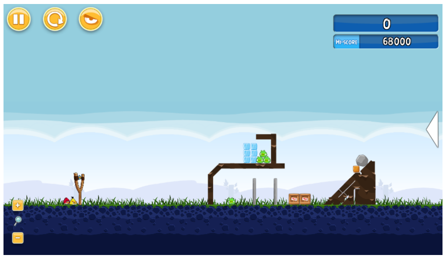


Figure 1: Example levels from AI Birds competition. To solve the level shown at the top, the blue bird (in the sling) has to be shot at the blue ice blocks in order to clear the path for the red bird (third in sequence). The yellow bird (second in sequence) must be shot at the wood structure that prevents the round stone on the right from rolling towards the TNT boxes on the ground. Explosion of the TNT boxes triggers a domino effect on the stone pillars, eventually destroying the bottom pig.

through some undisclosed formula that awards points to objects destroyed and a large amount of points to each unused bird when a level gets cleared. If an agent clears a level several times, it is awarded the maximum of points it has scored. The more damage is inflicted and the lower number of birds fired, the more points an agent receives. To make the game attractive to humans, several birds comprise special functions (e.g., blue birds can split into three smaller birds, allowing to hit multiple targets at once) and have unique effects when shot at particular game objects (e.g., yellow birds penetrate wood particularly well). Also, the game includes elements with special properties, in particular indestructible obstacles and explosives, which allow a great variety of levels to be constructed.

On a technical level, every agent communicates with the game through a network interface. Agents may request screenshots from the game and can issue click and click-drag-release (shot) actions. Moreover, agents can restart a level or select any of the levels from the current round. Also, agents can pan the view (for large levels) and control the zoom level. In later rounds of the competition, an agent may inquire the current best scores for each level. The AI Birds game is executed in a web browser window and can only be accessed via this interface. This setup has the following implications:

- game mechanics are concealed, i.e., physical simulation is performed with parameters unknown to the agent and can only be estimated from observations
- the interface is real-time, e.g., agents cannot quickly gather training data, even outside competitions

The competition challenges agents in two regards: solving individual levels and maximising the overall score.

2.1 Solving Levels

In order to solve a level, each agent has to interpret the screenshot and locate relevant objects. It is particularly important to identify the location of the slingshot and scale of the scene precisely in order to perform goal-directed shots as the game calculates flight trajectories with respect to the slingshot. If the pivot point of the sling is not estimated to within a few pixels accurately, no shot will be performed, or the bird drops off the sling. In order to clear a level, an agent has to plan a number of shots (two to five, typically) in an uncertain physical environment. Due to the lack of a reliable forward model and the sometimes chaotic reaction (e.g., how large structures collapse), uncertainty in action outcomes cannot be neglected.

2.2 Maximising Score

Typically, agents participating in the competition are not able to solve every level, at least not within the given time limit. As points are only awarded for levels solved, it is important to use some strategy for selecting which level to try next. Agents have to balance between re-playing a level already solved in order to improve the high-score, trying to crack a previously unsolved level, and not wasting time on levels unsolvable to them.

2.3 When Games are not Toys

AI has always considered games as benchmarks, be it for the public impact (like IBM's Deep Blue defeating Garri Kasparow or Google's AlphaGo defeating Lee Sedol), or for what Schaeffer called "microcosms of AI research" (Schaeffer 2014). Games may offer a convenient platform for conducting research as the rules of the game are fixed and clear—no bias by committing research to individual assumptions is at risk and results of different groups are easily comparable. Nevertheless, we believe one should reflect on a commitment to work on a game rather than a "real" problem that promises direct rewards for the society. As a physical simulation game, Angry Birds present a simplification of physical manipulation required for versatile service robots that eventually will assist humans in their everyday tasks, e.g., by getting dishes from cupboards, preparing meals for humans within an environment designed by humans for humans. For most labs, hardware for such versatile robots is beyond reach and even where such systems are available, technical challenges are manifold. For example, research on manipulation required for preparing meals like opening a bottle of milk, retrieving flour from their typical paper containers, etc. is hardly possible while contemporary progress in versatile robots is still involved with opening cupboards in kitchens, see (Kazhoyan et al. 2021). Above all, differences between robotic platforms used and the specific tasks considered hinder a direct comparison. In the light of the versatility of problems that can easily be constructed in a simple 2D physical simulation game (cp. (Stephenson, Renz, and Ge 2020)), the AI Birds competition thus constitutes a viable option for fundamental AI research that has prospects to improve future robotic applications. In particular, the physical nature of AI Birds is well-aligned with fundamental tasks and goals of qualitative reasoning (cp. (Forbus 2019)).

3 Synopsis of BamBirds Agent

The BamBirds agent is developed at the University of Bamberg, Germany. Its development is significantly supported by

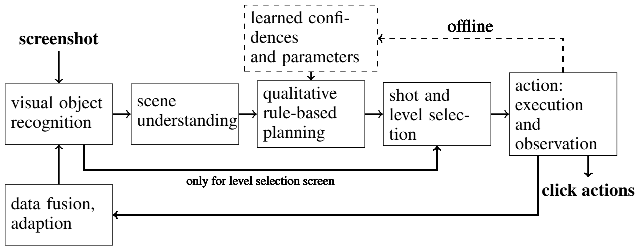


Figure 2: Architecture of BamBirds agents

student projects and thesis works. Within individual study groups, selected AI techniques that were expected to improve the agent are developed, implemented, and evaluated. Successful components are then integrated into the BamBirds agent. By design, BamBirds integrate GOFAI (Good Old-Fashioned AI) approaches like symbolic state space representation with probabilistic methods and lightweight machine learning. An explicit hybrid representation comprising quantitative and qualitative knowledge about levels is central to the design of the agent. BamBirds comprise the building blocks we detail in the following, and shown in Figure 2.

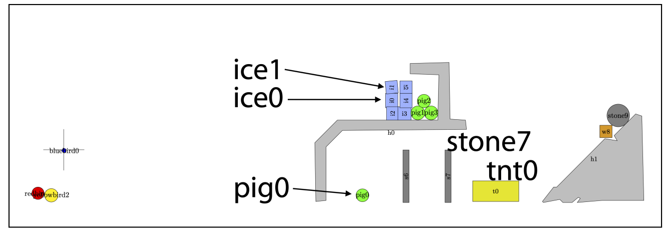
3.1 Visual Object Recognition and Scene Understanding

All planning hinges on a description of the situation the agent faces, in particular the objects within a level, their whereabouts, and the overall level scale (which can be derived from the size of the sling in pixels due to its constant size in spatial units). A precise representation is required for delivering precise shots at chosen targets.

Scene understanding itself is largely based on simple methods provided by the AI Birds organisers for visual object recognition to detect primitive objects; due to the graphic nature of the Angry Birds game, a simple approach already yields sufficiently good results in most cases. The visual recognition provided by the organisers and used in the BamBirds agents provides polygonal outlines of objects and a classification into the object types (wood, stone, ice, etc.). As an example, see Figure 3 for the example level from Figure 1 as seen by the BamBirds agent.

Visual object recognition is also responsible for detecting the game state, in particular to recognise that a ‘level won’ or ‘level lost’ screen appears and the agent is expected to select a new level.

From the geometric description of the scene, the agent derives qualitative spatial and physical relations that allow basic strategies to be grounded. To this end, two techniques are used. First, qualitative spatial relations (above, below, left of, etc.) are instantiated based on the semantics grounded in the location of objects. Second, a physical simulation using a 2D physics simulator is consulted to determine whether objects weigh on one another (for inferring stability) and to foresee selected effects of actions performed in the game. As physics simulation under uncertain start conditions is susceptible to noise and may easily yield wrong results, the component is only consulted for basic prediction of forces. In order to




```
belongsTo(ice0,struct2).
expl_affects(tnt0,stone7).
isOver(hill0,pig0).
protects(struct2,pig1).
slingshotPivot(145.2,332.8).
supports(ice0,ice1).
...
```

Figure 3: Example level from AI Birds competition depicted in Figure 1 as interpreted by BamBirds agent an excerpts from the respective scene descriptions.

Table 1: score of BamBirds vs. IHSEV agent per round in 2016 competition, level complexity increases towards final

round	BamBirds	IHSEV
quarter final	280, 390	470, 940
semi final	406, 200	562, 820
final	451,250	288, 720

construct this component, we consulted the IHSEV agent which also includes physical simulation and used regression to fit parameters to the game (M. and Buche 2013). Generally, physical simulation is not robust due the nearly chaotic nature of how complex arrangements response to impact in conjunction with inevitable uncertainty in parameter estimation and visual object recognition. To illustrate, we point to the score of the IHSEV agent relying on physical simulation from the 2016 competition (AI Birds 2016) as reported in Table 1. As can be seen in the table, advancing from the quarter final to the final, the increasing level complexity correlated with the performance of IHSEV wrt. BamBirds decreasing (we note that absolute points are not comparable due to different amount of points that can be reached). Similar numbers can be observed in the 2019 competition where BamBirds defeated the simulation-based agent SimbaDD.

In short, the competition taught us that physical simulation is not reliable beyond simple inferences in semi-complex and complex environments. We therefore critically assess a much recognised argumentation for physical simulation in scene understanding (Battaglia, Hamrick, and Tenenbaum 2013), beyond grounding qualitative primitives on simple force calculations, e.g., `rests_on`.

As the output of the scene understanding module, a scene description in Prolog syntax is generated which contains both quantitative information (in pixel coordinates) and qualitative relations (see Figure 3 for an excerpt).

3.2 Qualitative Rule-Based Planning

The second and most involved component is responsible for determining possible shot candidates, given a scene description. By obtaining an explicit representation of qualitative relationships such as, for example, from `isOver(pig,ice)` and `supports(ice,stone)`, it is possible to design rules that serve as heuristics for identifying (potentially) useful shots. One of these rules states that by destroying an object that supports another, the now unsupported object will fall down and be likely destroyed. In the example above, aiming at the ice object could thus be a viable plan since the unsupported stone will fall onto the pig, destroying it. Until now, the rule base of BamBirds has been designed manually. Although BamBirds does not perform physical simulation for shots, the symbolic method is augmented with a quantitative estimator, e.g., to estimate the likelihood of penetrating objects by a single shot or the likelihood that a tower of objects will collapse when shooting at it. Also, an estimate is given whether the shot is expected to succeed. For example, a direct shot at a freely reachable pig is given full confidence, whereas a shot against a wall of objects to bounce off into the direction of some goal is given low confidence. To obtain functions for estimations, machine learning and regression has been performed on selected parameters from recorded games.

As a last step in the shot heuristic, a simple partial order planning is performed. In particular, shots are decreased in confidence if a later bird will be better suited to reach the goal, and shots are increased if the current bird is more useful for achieving some (intermediate) goal than forthcoming birds. A level taken from Stephenson and Renz (2018) that challenges lookahead planning is depicted in Figure 4. The player has a blue and a yellow bird, the blue bird must be shot first. The yellow bird can penetrate both wooden (yellowish) pillars, directly hitting the pig. The blue bird can only destroy one pillar, making the stones fall down and render the level unsolvable. Here, the agent has to waste the blue bird (e.g., shooting it over the structure or against the stone blocks) in order to finish the level with the yellow bird.

3.3 Shot Planning and Level Selection

The third component of our agent implements the shot selection from the set of candidates computed by the shot heuristic module. We approach the problem as heuristic search in a tree whose edges represent shots. For every shot performed we monitor its effect (e.g., the points awarded, pigs destroyed). When retrying a previously unsolved level the algorithm aims to find an alternative to a previously tried shot sequence. Consider again the example from Figure 4. Our agent lacks a forward model to anticipate that shooting the blue bird at the wooden pillars or the ice bar is a bad idea. However, once it has tried that shot (and noticed it has no plan for finishing the level with the yellow bird), the shot is discarded and an alternative is tried when revisiting the level. In absence of promising alternatives, the agent soon tries shooting at one of the stone objects (without much effect, if any) and is then able to finish the level with the yellow bird. Unlike classic game settings previously studied in AI, it is not possible to explore a significant portion of the search tree since exploration requires to engage in the game; only very few retries are possible during the competition.

In the 2021 competition, a clone of the BamBirds 2019 agent has won the competition which has chosen a parameter in favour of more exploration. Although winning 257,330 to 168,290 against BamBirds 2021 in the grand final[3], the BamBirds 2021 agent defeated its clone 312,910 to 270,200 in the previous round. We may therefore conclude that shot selection is critical but not sufficiently well evaluated in a single competition.

In a fourth and last step, once a level has been played, we decide which level to try next. We select the level that is expected to yield the largest reward considering information about the type of level, the number of previous attempts, the points that might be earned, the set of shot candidates not yet tried. We apply machine learning (offline) to obtain an estimator function that predicts the probability distribution for the performance of our agent based on previous attempt and features of the level. We then apply a randomised selection balancing potential gain with probability of success.

3.4 Action Execution and Monitoring

For all shots performed in the game, we monitor the effects to collect data and to determine when a level has stabilised after a shot, allowing the agent to plan its next shot.

3.5 Modul Interconnections and Adaption

For the most part, modules are executed one after another along the main horizontal axis shown in Figure 2. However, there is one notable exception to the linear flow, which is found in the module "adaption".

Precise visual object localisation is required for delivering precise shots at chosen targets. We found our agent to be too limited when relying on the visual object recognition techniques available. Therefore, we use data from the observed flight parabolas to improve estimates of scene scaling and sling position using regression on a per-level basis while the agent is playing. Most importantly, we trace the flight parabolas of each bird shot and, using regression, we adapt slingshot location and scaling parameters to align predicted shot parabolas to the observed one.

4 What is Missing in AI Problem Solving?

As we have seen, the typical "AI" approach of transforming real-life problems into optimisation problems fails for all but the simplest configurations in AI birds. On the level of physical processes alone, it seems to be unsolvable with current techniques. Of agents relying on machine learning, the agent DQ-Birds (Nikonova and Gemrot 2021) using a Deep Q-Network trained from about 115,000 situations was the best-performing agent so far with being able to solve 3 out of 8 levels scoring 185,869 points in the 2017 quarter finals, the last round it participated. Other learning-based agents have performed worse and teams decided to quit participating in the competition. By contrast, BamBirds scored 290,020

[3]http://aibirds.org/past-competitions/2021-competition/results.html

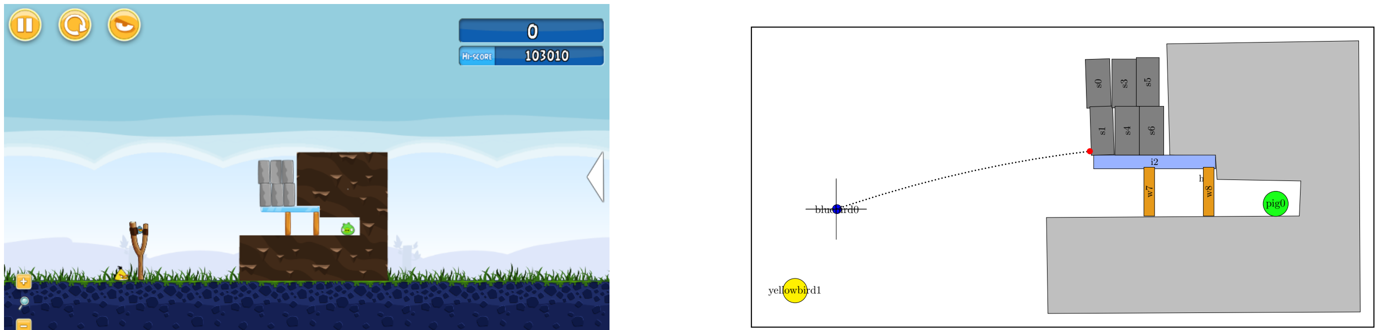


Figure 4: Example of level that requires lookahead planning

points in the 2017 quarter finals and the best-performing agent in that round 405,260 points.

Approaches relying on rules, inference, or planning on an abstract symbolic representation work for specific cases, but despite the fragile basis of symbols grounded in perception, they are still missing (and that regards all computational systems today) the ability to switch strategies, to "step out of the system" (Hofstadter 1979) and reconsider one's own understanding of the situation and the strategy to be applied.

AI has a long tradition on abstract representations: logic (Nebel 2001; McCarthy 2000), frame-based representations (Minsky 1974), and qualitative abstraction (Forbus 2019). Together with the representations we have powerful reasoning mechanism for, e.g., qualitative reasoning (Forbus 2019), analogical reasoning (Falkenhainer, D.Forbus, and Gentner 1989), and planning (Ghallab, Nau, and Traverso 2016).

It has turned out that none of these representations alone can represent and solve realistic problems (Forbus, Nielsen, and Faltings 1991). Ideas of how to combine different representations on a problem have been around for a long time as well, such as the mental image of a society of mind (Minsky 1986). Blackboard systems (Engelmore and Morgan 1988) have tried to provide an architectural basis for combining different representation and reasoning strategies.

Such approaches have been around for a while now and it feels this must be the direction to go. Still, none of them have made the step from hand-crafted systems to self-aware systems that understand the situation and act according to it.

4.1 Knowledge Representation and Transformation

From our experience, the main bottleneck is the interaction between different representations, especially symbolic and subsymbolic representations, often referred to as *symbol grounding*. Of course, there have also been attempts to do this, especially in robotics (see contributions in special issue (Coradeschi, Loutfi, and Wrede 2013)).

The fundamental flaw with all of these approaches is that they treat the task as a mathematical mapping from one representation to another. But there is no such mapping. When we transform subsymbolic information to a symbolic representation, we lose detail (usually numerical information), but we add interpretation. This interpretation always comes with some arbitrariness. The same numerical state representation can be part of different situations. Sitting in front of a black screen can mean that the computer is switched off, or that the screen is broken, or that the computer simply shows a fully black screen to name just a few possible interpretations. At this moment, there is no way to fully understand the situation based on perception. Some disambiguation can be done by including memory (remembering having switched on computer and screen), some may be possible after waiting some time (the computer showing something else than black), but others may need some interaction between reasoning, action and re-observation, like switching the screen off and on again.

The other way around we encounter the same problem. When some reasoning process has come to a conclusion like a "good" shot, it has done so with impoverished information, since it had abstracted from numerical values. Abstraction is great to focus a problem and keep the state space small. The problem is just that settling on one specific solution makes the whole process extremely fragile. It is up to luck whether the one solution settled on will really fit the situation. And when we transform the one abstract solution into a lower-level command, we again have to guess, this time the numerical values that are necessary for action execution, but that are not part of the output of the abstract solution process. If we then observe an action to fail, it may be due to a poor plan or a poor transformation.

4.2 Dynamics

We propose that the way out of the dilemma lies in a more dynamic view on computation. Even if the problem itself is static (as is the case for Angry Birds, at the moment the agent has to decide on an action, nothing changes in the setup), the solution process needs to be dynamic.

The basic idea is that we should replace the computational pipelines that are used today with a network of interacting modules, each of which is running its own decision-making process in a way described in (Kirsch 2019): the module would continually 1) consider a set of alternatives (which may be the output of some other module) and 2) evaluate and rank those alternatives (again a service that may be provided by other modules). Differently from the pipeline approach, no module would have to settle on one single solution. Of course, at some point an agent should act. This could be done if enough modules have converged to a stable solution, meaning that their most favoured alternative is not changing by any further decision iterations. Actions could also just be executed with a certain frequency, using the best-ranked alternative in some action module.

For example, during shot planning multiple shots are output which aim at the same object but at a different target point, each of them being a candidate for one specific goal (e.g., tossing over some object). If we know that one of these shots has succeeded, there is no need to consider the alternatives—if one shot goes terribly wrong by destroying the object, other shots at the same object will suffer from the same problem. The problem our agent is facing is that there are too many potential alternatives to consider. We could counter-act this problem by structuring the suggested shots and providing means to dynamically move within this structure. To this end, feedback information about a shot tried is required, revealing *how* it failed and *why* it might have failed. Qualitative relations allow us conveniently to describe how an action failed by comparing the actual outcome with the expected outcome. With respect to grasping causality, we again have to acknowledge that it will not be possible to single out the one reason, but only to rank alternative explanations.

In further steps, modules could even be added and removed (or switched on and off) depending on the situation (there would be a need for special modules to decide on the module configuration). The machine would not have to invent completely new modules, but it could decide to run modules with the same task, but being instantiated with different sets of parameters.

Why should this work? A clear argument for trying more dynamic processes is nature. There is no doubt that human thinking is dynamic, both on the neuronal level (Hawkins and Blakeslee 2004), on behavioral or problem-solving levels (Hayes-Roth and Hayes-Roth 1979; Newell and Simon 1972), also in the development of language and abstract concepts (Lakoff 1987). The reason why human thinking processes are dynamic, is surely the complex and dynamic world around us (Rittel and Webber 1973; Taleb 2010; Varela, Thompson, and Rosch 2016). Previous attempts by the authors in this direction have shown promising results (Kirsch 2017). We back up our claim by the following experiment: In Bambirds, we have a very simple form of dynamics implemented by proposing a certain type of risky shots (termed 'last resort shots') only if no other shots can be found. To study the effect of this simple form of dynamics, we compare it against a variant of Bambirds that always considers last resort shots. Running the agent on the 131 competition levels with a time limit of 5 minutes per level (about four tries per level), the agent in the dynamic condition was not able to solve 32 levels. When always considering last resort shots, 36 levels remained unsolved. Put differently, the agent performed better when dynamically increasing its set of plan candidates as compared to considering all plans at once. This experiment suggests that a dynamic interaction between shot planning and other modules is helpful.

Engineering Fears Interacting modules is about the last thing an engineer wishes for. While single modules are easy to control and debug, interaction always comes with uncertainty. A change in one module may break the whole system. It is exactly this kind of complexity that engineers try to avoid by a module pipeline.

Interaction, however, does not necessarily imply parallelism. In previous work we have explored interacting modules for robot navigation (Kirsch 2017). The modules were run sequentially, the resulting behavior was "rather" deterministic (since the study was run in a physical robot simulation, navigation tasks could be exactly reproduced, but the physical parameters still introduced some non-deterministic behavior in the actions).

Even with modules running in parallel, the behavior can be stable without extreme engineering overhead. In a retrospective of the Hearsay II blackboard architecture, the authors report: "A surprising result was that system performance, in terms of accuracy, was as good with the synchronization disabled as its performance with the full synchronization." (Lesser and Erman 1977, p. 797)

And there are theories around how to deal with dynamic systems, e.g., cellular automata (Wolfram 2002). It is just that the type of stability shown by dynamic systems is different than the exactly predictable input–output pairs we are used to from chaining functions in a processing pipeline. As soon as the environment exhibits uncertainty and dynamics, the pipe(line) dream comes to an end anyway. Instead of trying to force environments into our engineering wishes, we should rather accept the challenge and learn to deal with it.

Interfaces When different modules use their own representations, we need to find a way to combine them. Blackboard architectures (Engelmore and Morgan 1988) are an attempt to channel the complexity of interacting modules to a central memory where all modules communicate. This makes the information flow easy to track and to debug (all the relevant information is in the central blackboard memory). But it also makes it hard or almost impossible to find the one representation that fits all modules. The experiences described on the Hearsay II architecture (Lesser and Erman 1977) confirm what can be expected: at the end, one does construct special pieces of information that are only relevant and useful for some modules.

Therefore, instead of trying to put all pieces of information in one central memory in a unified language, we suggest to try networks of interconnected modules. Each module must support communication interfaces to its neighbors, in the way known from current pipeline structures. Such a network constrains the options for adding and removing modules, but as stated above, we do not expect to generate fully new modules any time soon. Additional modules could be clones from other modules and would have a matching place in the network of modules.

We want to emphasize that modules form a network, not a hierarchy. One module may use a more abstract representation than another, but that does not put it "above" the other module. An observation from a neuroscientist friend: When we look at graphical representations of modules in the human brain, each scientist will put the module she is working on in the center of any diagram, but if you were to draw the full picture, there is no 'upper module'. All the pieces are connected, and *in all directions*. A network only makes sense with information passing in both directions, otherwise we would be back at a processing pipeline.

4.3 A New Role for Qualitative Reasoning

The considerations above motivate us to propose a new approach to qualitative reasoning in agents. Rather than only using QR in the classical form of describing a process abstractly, we advocate to use QR to describe the interplay of modules. To give an example, the feedback loop in the Bambirds agent that adapts parameters for visual scene recognition from observations could be described using qualitative rules that explain how scene parameters must be changed (i.e, increased, or decreased), given how a shot missed the target anticipated. QR techniques can then be applied to govern the convergence process of modules, similar to how QR rules about throwing objects like "reduce launch speed if throwing too far" make action selection converge more quickly (Wolter and Kirsch 2015).

5 Summary and Conclusion

This paper presented the BamBirds agent, which has won the AI Birds competition three times. The agent is based on several modules that are involved with visual object recognition and scene understanding, shot planning, shot selection, an action module, and feedback components that allow the agent to improve (during gameplay by adapting parameters, during development by learning estimators from recorded data). We discuss why we believe a physical simulation game cannot be tackled with existing AI techniques such as machine learning or QR alone, but motivates basic AI research. Despite the survey among AI experts (Grace et al. 2018) that projected the arrival of an agent defeating humans in Angry Birds around the year 2022, we are pessimistic that an agent will come close to human performance in the near future. We argue that a severe limitation of today's approaches is due to static architectures of independent modules that lack the ability to reflect their decisions and to reach their output in a dynamic process of interacting with other modules. In order to let the modules step out of their static roles, researchers must also step out of their beaten path of higher degrees of specialisation in AI research and focus more on AI architectures and how they allow existing techniques to be integrated. Rather than aiming for a precise method to govern module interactivity, we argue that QR techniques are of interest which steer convergence processes, but allow components to interact in a dynamic manner.

References

AI Birds. 2016. Results from 2016 Competition. http://aibirds.org/past-competitions/2016-competition/competition-results.html. Last accessed 2022-05-22.

Battaglia, P. W.; Hamrick, J. B.; and Tenenbaum, J. B. 2013. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110(45): 18327–18332.

Coradeschi, S.; Loutfi, A.; and Wrede, B., eds. 2013. *KI special issue "Symbol Grounding"*, volume 2/2013. Springer.

Engelmore, R.; and Morgan, T., eds. 1988. *Blackboard Systems*. Addison-Wesley Publishing Company.

Falkenhainer, B.; D.Forbus, K.; and Gentner, D. 1989. The structure-mapping engine: Algorithm and examples. *Artificial Intelligence*, 41(1): 1–63.

Forbus, K. D. 2019. *Qualitative Representations: How People Reason and Learn about the Continuous World*. Cambridge (MA), USA: MIT Press.

Forbus, K. D.; Nielsen, P.; and Faltings, B. 1991. Qualitative Spatial Reasoning: The Clock Project. *Artificial Intelligence*, 51(1-3): 417–471.

Ghallab, M.; Nau, D. S.; and Traverso, P. 2016. *Automated planning and acting*. Cambridge University Press.

Grace, K.; Salvatier, J.; Defoe, A.; Zhang, B.; and Evans, O. 2018. When will AI exceed human performance? *Journal of Artificial Intelligence Research*, 62.

Hawkins, J.; and Blakeslee, S. 2004. *On Intelligence*. Times Books.

Hayes-Roth, B.; and Hayes-Roth, F. 1979. A Cognitive Model of Planning. *Cognitive Science*, 3(4): 275–310.

Hofstadter, D. R. 1979. *Gödel, Escher, Bach: an eternal golden braid*. VintageBooks.

Kazhoyan, G.; Stelter, S.; Kenfack, F. K.; Koralewski, S.; and Beetz, M. 2021. The Robot Household Marathon Experiment. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Kirsch, A. 2017. A Modular Approach of Decision-Making in the Context of Robot Navigation in Domestic Environments. In *3rd Global Conference on Artificial Intelligence (GCAI)*, 134–147.

Kirsch, A. 2019. A Unifying Computational Model of Decision Making. *Cognitive Processing*, 20(2): 243–259.

Lakoff, G. 1987. *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind*. The University of Chicago Press.

Lesser, V. R.; and Erman, L. D. 1977. A retrospective view of the Hearsay-II architecture. In *IJCAI'77: Proceedings of the 5th international joint conference on Artificial intelligence*.

M., M. P.; and Buche, C. 2013. Towards A Theory-Of-Mind-Inspired Generic Decision-Making Framework. In *IJCAI 2013 Symposium on AI in Angry Birds*.

McCarthy, J. 2000. concepts of logical AI. In Minker, J., ed., *Logic-Based Artificial Intelligence*, 37–58. Dordrecht, Holland: Kluwer.

Minsky, M. 1974. A Framework for Representing Knowledge. Technical report, MIT-AI Laboratory.

Minsky, M. L. 1986. *The society of mind*. Simon and Schuster.

Nebel, B. 2001. Logics for Knowledge Representation. In *International Encyclopedia of Social and Behavioral Sciences*. Elsevier.

Newell, A.; and Simon, H. 1972. *Human Problem Solving*. Upper Saddle River, New Jersey: Prentice Hall.

Nikonova, E.; and Gemrot, J. 2021. Deep Q-Network for Angry Birds. Technical Report 1910.01806v2 [cs.AI], arXiv.

Renz, J.; Ge, X.; Gould, S.; and Zhang, P. 2015. The Angry Birds AI Competition. *AI Magazine*, 36(2): 85–87.

Renz, J.; Ge, X. Y.; Stephenson, M.; and Zhang, P. 2019. AI meets Angry Birds. *Nature Machine Intelligence*, 1: 328.

Rittel, H. J. W.; and Webber, M. M. 1973. Dilemmas in a General Theory of Planning. *Policy Sciences*, 4.

Schaeffer, J. 2014. The Games Computers (and People) Play. In *Proceedngs of AAAI*.

Stephenson, M.; and Renz, J. 2018. Deceptive angry birds: towards smarter game-playing agents. In *FDG '18: Proceedings of the 13th International Conference on the Foundations of Digital Games*, 1–10. Article No. 13.

Stephenson, M.; Renz, J.; and Ge, X. 2020. The computational complexity of Angry Birds. *Artificial Intelligence*, 280: 103232.

Taleb, N. N. 2010. *The Black Swan*. Penguin Books.

Varela, F. J.; Thompson, E.; and Rosch, E. 2016. *The Embodied Mind: Cognitive Science and Human Experience*. MIT Press. Revised edition.

Wolfram, S. 2002. *A New Kind of Science*. Wolfram Media.

Wolter, D.; and Kirsch, A. 2015. Leveraging Qualitative Reasoning to Learning Manipulation Tasks. *Robotics*, 4: 253–283. Special Issue Representations and Reasoning for Robotics, N. Bellotto, N. Hawes, M. Sridharan, D. Nardi (eds).