Behind The Corner: Using Qualitative Reasoning for Solving Angry Birds

Felix Haase and **Diedrich Wolter**
Otto-Friedrich-University Bamberg
felix.haase@stud.uni-bamberg.de, diedrich.wolter@uni-bamberg.de

Abstract

Physics-based logic puzzle games present diverse challenges for AI, primarily related to coping with uncertainty in complex continuous environments. As these challenges tie in with the goals of qualitative reasoning (QR), in particular to plan, predict and diagnose physical mechanisms, these games pose a natural benchmark for QR. In this paper we consider the projectile collision games *Angry Birds* by Rovio Entertainment on which the AI benchmark *AIBirds* is based. We investigate whether a qualitative approach to action planning under uncertainty proposed by Ge et al. (2016) can be adapted to the domain of Angry Birds in order to identify targets hittable by multiple rebounds. It turns out that the search space for solving Angry Birds is complexly structured and a fine-grained decomposition is required, leading to high computational costs. This paper presents an analysis of the problem and discusses means to overcome the problems faced. We propose an improved qualitative prediction method that is able to solve hard levels, yet it also suggests the need of complementary methods.

keywords: physical reasoning, qualitative simulation, action planning

1 Introduction

Understanding physics is still one of the most complicated tasks for computers. Humans are very fast at foreseeing consequences of actions and intuitively formulate plans to achieve some goal. Despite a solid understanding of mechanical systems and availability of numerical simulation of physical systems, there still does not exist a general algorithm to solve everyday problems on a near-human performance level. The challenges lie specifically in two areas: First, the world is continuous and therefore the action space is also infinite, it may even behave in a chaotic manner. Moreover, there is no generally applicable discretization of the action space, i.e., no obvious *place vocabulary* [Forbus *et al.*, 1991] that would allow search or planning techniques to be applied in a straightforward manner. For coping with infinite or large action spaces, sampling-based approaches are an attractive option [Elbanhawi and Simic, 2014; Venkatasubramanian *et al.*, 2003].

Second, many physical parameters needed for predicting action consequences according to the laws of physics remain either unknown or uncertain. This inhibits application of action planning techniques which rely on a so-called forward model that can reliably predict the direct consequences of an action, ruling out the sampling-based approaches mentioned above. Above all, physical effects that need to be exploited in planning may not be immediate but eventually occur at the end of a chain of causal consequences. For example, in the game of Angry Birds, we may need to shoot a projectile at a structure to make the structure collapse to make another structure collapse to clear the path for a later shot.

In the Qualitative Reasoning (QR) community several approaches have been developed that allow handling discrete and non-discrete values or intervals and defines state transitions, considering prediction and diagnosis of various scenarios related to everyday physics [Bobrow, 1984; Kuipers, 1986; Forbus, 1997; Forbus, 1984; Zhang and Renz, 2014; Ge *et al.*, 2016]. An interesting approach is to combine quantitative and qualitative aspects [Berleant and Kuipers, 1997] as for some parameters, it is easier to find qualitative representations than for others. Moreover, a purely qualitative simulation may yield a considerable number of results that are not realizable due to an inevitable over-generalization that occurs. The main challenge is thus to identify a suitable level of abstraction on which reasoning can be performed efficiently and with a good balance of the ability to find a solution and effectiveness of excluding non-realizable solutions which would lead to unsuccessful attempts. In other words, we are investigating a balance point between quantitative and qualitative simulation for reliable prediction and diagnoses. Our work is motivated by the hypothesis that the ability to simulate a physical scene may still be regarded to be a characteristic of scene understanding [Battaglia *et al.*, 2013].

One ultimate goal is to create robots that only need high-level instructions to formulate plans [Latombe, 1991] and can then act for themselves. As robots capable of performing everyday object manipulation are still not available, investigations related to complex planning may often be conducted in a computer game scenario more effectively. Physics-based logic puzzle games are a good playground for developing and testing algorithms to solve basic physics puzzles. According

to Grace et al. [2018], Angry Birds is one candidate where researchers are hopeful of solving it in the near future. Angry Birds developed by Rovio Entertainment[1] is a game in which the player has a specific number of projectiles (birds) that come in a predefined order. These birds are launched using a slingshot towards structures of objects and targets (pigs). Shot effects are determined by the game using a physical simulation. The player controls three variables: launch angle, launch velocity, and, for some birds, tap time which initiates a secondary action (e.g., splitting into three projectiles). The AIBirds competition [Renz, 2015] is concerned with finding strategies to solve unknown levels with possibly uncertain visual information and no information about the underlying physics engine (gravity, friction, object mass, etc.). There has been steady progress in the complexity of levels that can be solved, but in the Man vs. Machine Challenge[2] where humans compete with the agents it becomes clear that there is still a long way to go. In the last competition at IJCAI 2019, only one of the agents could even complete one of the 4 given levels.

In this work we try to answer the question of how to find possible initial angles to hit pigs that are placed behind corners such that they are only hittable after one or multiple rebounds. In particular we investigate whether the qualitative planning technique proposed by Ge et al. [2016] can be extended to consider gravity which occurs in Angry Birds. Using Qualitative Reasoning is promising way to solve this problem, since it only requires minimal information about the physical properties of the system as well as less computational effort than sampling based approaches.

This paper is structured as follows. We summarize the approach of Ge et al. [2016] in Section 2, before we present our adaption in Section 3. We present an evaluation of our approach in Section 4 and discuss the results in Section 5. We conclude the work with a brief outlook in Section 6.

2 Related Work

The problem we want to solve in this paper is very similar to the minigolf problem considered in Ge et al. [2016] where a ball is shot to a hole with a single shot, i.e., the aim is to identify a trick shot exploiting obstacles and using multiple rebounds. Ge et al. propose the Hole-In-One physical action selection problem and present an approach where the free space is triangulated to construct a discrete search space, in which qualitative simulation is performed. Three movement types *FLY*, *BOUNCE* and *SLIDE* are considered for ball movements. A state is defined by a range of ball movement directions, the triangle visited by the ball, its movement type and the edge where the ball entered the cell. The action space is continuous and given by the starting angle of the ball. In the approach transition rules for entering a new state are formulated. These rules state how movement types change (e.g., bouncing only occurs at an obstacle boundary) and how the range of movement directions is affected by a state transition. Constraints on movement directions are formulated that relate incoming and outgoing directions, for example, the direction change that occurs at a rebound. By search, paths from target to goal are identified which respect state transition rules and direction constraints. Generated paths are grouped and ranked, and a physical simulation is performed for the most promising path groups to filter impossible paths. Applying this approach to Angry Birds comes with one key challenge that was not yet discussed in detail: gravity. In Hole-In-One, effects of gravity are minor.

By contrast, motions trajectories in Angry Birds must be treated as parabolas. In combination with a significant loss of energy during rebounds, constraints on directions are less tight. Moreover, no physical simulation can be performed to single out the winning shot as physical parameters are unknown.

There have already been other approaches to solving AngryBirds with qualitative reasoning [Zhang and Renz, 2014; Walega *et al.*, 2016], however they only consider the implications of hitting a destroyable or movable block onto other blocks or pigs, and not interactions of the bird with static targets and the implications on the bird. Extending these approaches to this use case is not trivial, since they don't do any temporal reasoning or progression of states. Most of the agents participating at the AI Birds Competitions [Renz, 2015] have used approaches similar to the mentioned works, physical simulation or machine learning, but none of these was yet able to solve levels with necessary rebounds.

Approaches to similar physical games like throwing objects at a specific point in three dimensional space [Wolter and Kirsch, 2015], playing pool [Archibald *et al.*, 2010] or soccer [Zickler and Veloso, 2009] have mostly used sampling based techniques.

3 Problem Statement and Approach

Behind-the-Corner is an instance of the Hole-In-One physical action selection problem presented by Ge et al. [2016]. The tuple $\langle E, O, P, T, B, G\rangle$ representing the problem is instantiated as follows: E is the restricted plane in Euclidean coordinates, O are all objects in the scene given by their outline, P are the physics parameters that hold in the environment (specifically gravity), T are the object types and their physical properties, $B \in O$ is the bird on the slingshot and $G \in O$ is a goal or target object. The problem is to identify launch angle (and launch velocity) of a shot originating at B that will reach target G.

3.1 Approach

We aim to identify angle intervals $I = [\Theta_1, \ldots, \Theta_n]$ that approximate shots that arrive at G, assuming a fixed launch velocity. As we cannot further filter I by means of physical simulation but have to try the shot in the game (which wastes precious time in the competition if a shot is unsuccessful), we aim to find the optimal balance between precision (not missing a shot option) and recall (avoiding infeasible shot candidates). The approach to find I is structured into four parts. First, the plane E is partitioned into square-shaped cells of equal size, called zones. Information about objects O inside these zones is stored in a scene representation $\mathbf{R}$. Second, states and possible state transitions are defined. Third, physical equations are used to calculate the angle interval Θ for

[1] https://www.rovio.com/

[2] http://aibirds.org/man-vs-machine-challenge.html

reaching a new state given the current state. Fourth, to traverse through the available space and arrive at the goal with less computational effort, heuristic search is used. The semi-qualitative simulation with search is then executed using the state transitions and physical equations.

3.2 Scene representation

For the simulation, scene representation $\mathbf{R}$ must provide answers to two questions:

1. Is there an object and, if so, what type of object?
2. Is there an edge and, if so, what is its normal vector?

$\mathbf{R}$ is composed out of zones which are stored in a three-channel matrix, as $\mathbf{R}_{x,y} = [T, \vec{n}_x, \vec{n}_y]$ with x, y the index of the zone Z, T the object type or background at (x, y) and $\vec{n}$ the normal vector of the edge. If there is no edge, $\vec{n} = \vec{0}$.

3.3 Qualitative States

For simulation, each step is saved in a fixed state S, which is a tuple $\langle Z, T, E, M, \Theta, V \rangle$ with

Z: zone $z \in \mathbf{R}$ which gives the location of the state,

T: the last state where a rebound happened (turn point) or the initial state,

E: the edge from where the simulation entered into Z,

M: type of motion $\in (FLY, BOUNCE, SLIDE)$,

Θ: the closed interval of angles at T to enter Z,

V: the velocity at T.

Storing angles and velocity not in the current state but the last turn point reduces the number of calculations that need to be performed in the simulation and removes the necessity to check for the consistency with previous states since that is implicitly given. The reference to the last turn point is required for performing the physical calculations since it defines the origin of the parabola. The simulation is controlled by a set of transition rules as introduced by Ge et al. [2016]. In the rules shown in Fig. 1, notation $S.X$ is used to refer to a specific element of a state, e.g., $S.Z$ stands for the zone of state S. Function $\text{free}(Z)$ determines if there is no object inside zone Z. The state transitions from S to S_{new} in Fig. 1 are an extension of the rules by Ge et al., given E as the connecting edge, $\vec{n}$ as the normal vector of the edge at Z or the zone below Z if $S.M = SLIDE$. Θ_{new} represents the resulting range of directions as determined by Algorithm 1 explained further below.

Simply put, the transitions describe the following behaviour: If the current state of the object is FLY, the next state will be FLY again if there is no obstacle in the way, otherwise it will be in the $BOUNCE$ state. The $BOUNCE$ state describes the moment when the object touches the obstacle. From that point there are three options. Either it will leave the zone and FLY on, there is again an obstacle in the next zone so it will $BOUNCE$ again or the impact angle is close to the obstacles angle, so it will start to $SLIDE$ on the obstacle. If it is then in the $SLIDE$ state, it may either continue $SLIDING$, start to FLY if the obstacles edge ends, or

- From $S.M = FLY$
  - $\rightarrow FLY$
    If $\text{free}(Z_{new})$ continue with $S_{new}.\Theta \leftarrow \Theta_{new}$, $S_{new}.E \leftarrow E$ and $S_{new}.Z \leftarrow Z_{new}$
  - $\rightarrow BOUNCE$
    If $\neg\text{free}(Z_{new})$ recalculate Θ_{new} and velocity for bounce and set $S_{new}.Z \leftarrow S.Z$ and $S_{new}.E \leftarrow E$
- From $S.M = BOUNCE$
  $S_{new}.T \leftarrow S$
  - $\rightarrow FLY$
    If $\text{free}(Z_{new})$ equivalent to $FLY \rightarrow FLY$
  - $\rightarrow BOUNCE$
    If $\neg\text{free}(Z_{new})$ equivalent to $FLY \rightarrow BOUNCE$
  - $\rightarrow SLIDE$
    If $\neg\text{free}(Z_{new})$ and $S.\Theta$ is close to be parallel to $\vec{n}^{-1}$ or the velocity is not high enough for bouncing
- From $S.M = SLIDE$
  - $\rightarrow SLIDE$
    If $S.\Theta$ is close to parallel to $\vec{n}^{-1}$. The current velocity needs to be calculated according to rolling/sliding motion
  - $\rightarrow FLY$
    If there is no wall beneath $S.Z$ or $\vec{n}^{-1}$ is not close and lower than $S.\Theta$, $S_{new}.T \leftarrow S$
  - $\rightarrow BOUNCE$
    If $\vec{n}^{-1}$ is not close to and is higher than $S.\Theta$. The current angles are calculated the same way as in $FLY \rightarrow BOUNCE$

Figure 1: State transition rules for shot prediction.

Algorithm 1 Calculate range of angles from $S.E$ to E_{new} under consideration of gravity

```
procedure GRAVITY(S, E_new)
    Θ_pos ← possible angle interval for transition S.E → E_new
    P_rel ← vertices of E_new relative to S.T
    Θ ← ∅
    for high and low parabolas do
        Θ_new ← {p ∈ P_rel | θ_0(p_x, p_y, S.V)}      ▷ angle interval to hit edge
        Θ_new ← Θ_new ∩ S.Θ
        Θ_cur ← {p ∈ P_rel, θ_0 ∈ Θ_new | θ(p_x, θ_0, S.V)}
    ▷ current angle interval at P_rel
        Θ_combined ← Θ_cur ∩ Θ_pos
        if Θ_cur ≠ Θ_combined then
            Θ_new ← {p ∈ P_rel, θ ∈ Θ_combined | θ_0(p_x, θ, S.V)}
                                  ▷ starting angles with the angles Θ_combined at P_rel
        Θ ← Θ ∪ Θ_new
    return Θ
```

it will $BOUNCE$ at an obstacle with a different angle than the current obstacle.

The state expansion is executed for each edge in $S.Z$ except for $S.E$. The only option where multiple states can be formed from the conditions are in $S.M = BOUNCE$. $\Theta_{new} = \emptyset \wedge S.M \neq SLIDE$ means that there is no transition possible from $S.Z$ to Z_{new} over E. The states are defined to never be inside of zones where an object is present.

The presented transitions are very similar to the ones by Ge et al. [2016], but here a state can have the movement type $BOUNCE$, which converts transitions $FLY \rightarrow BOUNCE \rightarrow [FLY|SLIDE]$ to transitions $FLY \rightarrow BOUNCE$ and then $BOUNCE \rightarrow [FLY|SLIDE]$. This change is required for saving a specific state as the turn point of S. Slide movement and its transitions are not further specified because they lie out of the scope of this work.

3.4 Physical Algorithms

The following physical algorithms implement a simplified 2D physics engine, that is required for calculating the new angles after a transition from one state to the next. The goal is to reduce these calculations to simpler rules, but this could not yet be achieved with a still reasonable amount of choices. Using qualitative representations for velocity here allow for the change of direction at each new state, and would in consequence create too many possibilities that would later need to be simulated with physical parameters anyway, as it was done in the cleanup process by Ge et al. [2016].

Algorithm 1 for applying gravity works as follows: First, the possible angle interval for the transition between the current and the new edge is calculated. Next, the vertices of the new edge are calculated as relative points to $S.T$. Both vertices are needed for computing maximal and minimal transition angles. For simplicity, the center of $S.T.E$ is used for computing rebounds. The next part needs to be done twice with high and low parabolas. This is because if $S.V > v_{min}(x, y)$ with v_{min} the minimum velocity to reach the point $(x|y)$, there are always two possible angles, one higher and one lower than the angle for v_{min}. Now the angle interval to hit the new edge needs to be calculated using the relative vertices as target points and the velocity in equation 1. To make sure that the edge is relevant, the calculated angle interval is intersected with the angle interval of the current state $S.\Theta$. The current angles at the vertices are calculated using (3) and then intersected with the possible angle interval. If the intersection changed something, the starting angle interval needs to be recalculated to resemble the actual angles using (2). Finally, the possibly two intervals are united without regarding space between the intervals.

$$\theta_0(x, y, v_0) = \tan^{-1} \frac{v_0^2 \pm \sqrt{v_0^4 - g(gx^2 + 2yv_0^2)}}{gx} \quad (1)$$

$$\theta_0(x, \theta_x, v_0) = \tan^{-1} \frac{v_0^2 \pm \sqrt{v_0^4 - 4gx(gx + v_0^2 \tan\theta_x)}}{2gx} \quad (2)$$

$$\theta(x, \theta_0, v_0) = \tan^{-1}(\tan(\theta_0) - \frac{g \cdot x}{v_0^2 \cdot \cos^2\theta_0}) \quad (3)$$

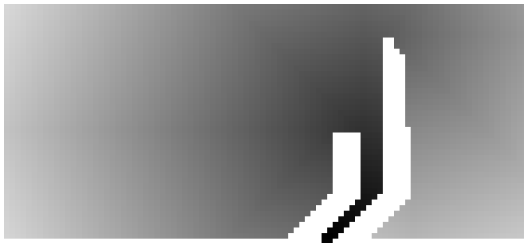


Figure 2: Dijkstra search for the shortest path from every point to the Goal in Scene 4a. A darker color represents a smaller distance to the target, the white regions are obstacles.

The second physical calculation is for the angle and velocity after a bounce. A bounce from an immovable object is characterized by two parameters, the coefficient of restitution (COR), which represents the elasticity of an object, and the coefficient of friction (COF), which depends on the materials of the moving and the stationary object [Halliday *et al.*, 2013]. Other variables required for the calculation are the current velocity components at impact and the normal of the immovable object. The latter is saved in the representation (see Section 3.2) and the former can be calculated with the starting angle, starting velocity, and the x coordinate of the impact. For both limits of Θ, the new angle after the bounce is calculated by splitting the current velocity vector into two vectors, one vector $\vec{u}$ perpendicular and one vector $\vec{w}$ parallel to the wall. Calculating the velocity vector after the bounce is done by applying COF to $\vec{w}$ and the COR to $\vec{u}$ and then adding them together. The COR is always negative, so the direction of $\vec{u}$ gets flipped. The magnitude of the vectors is the new velocity, and the limits of Θ_{new} are the angles of the vectors.

3.5 Search for relevant paths

The search is split into two parts:

Step 1: Only considering adjacency of zones, the shortest paths from G to all zones $z \in Z$ are computed and saved. These paths are used as a heuristic for the actual simulation.

Step 2: The actual qualitative simulation is performed using an initial State S_0 with the parameters B as starting zone, E as starting edge, Θ_0 as the starting angle interval and v_0 as the starting velocity. To reduce the number of state expansions, a Best-First-Search is performed using the matrix from step 1 as heuristic for the distance to the target. Using shortest paths as heuristics makes the search prefer paths with fewer rebounds.

A state that reached the goal H is indicated with $S.M \leftarrow GOAL$ and defined with the following criteria. For calculating, if there is a wall between $S.Z$ and G, the Bresenham's line algorithm [Bresenham, 1965] is used to generate all zones.

1. $d(S.Z, H) \leq \mathbf{R}_{el} + \text{radius}(H)$
   A goal state needs to be in the range of the goal. The edge length is required since $S.Z$ may contain the goal but the coordinate of $S.Z$ is not inside or on the edge of the goal object.
2. $\mathbf{R}_{S.Z.x, S.Z.y, 0} = T(H)$

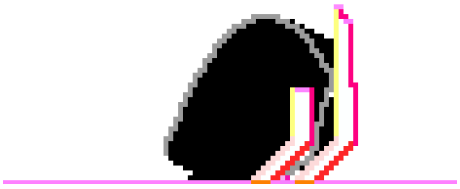


Figure 3: First result for simulation with Figure 2 as heuristic. Black points are visited zones, grey points describe the path of the found solution. $\mathbf{R}_{el} = 9$

The value of the zone in $\mathbf{R}$ is required to contain the object type of the goal. For $\mathbf{R}_{el} > 1$ any value in the range of the edge length in $\mathbf{R}$ needs to equal the $T(H)$.

3. $\{p \in \text{bresenham}(S.Z, H) \mid \mathbf{R}_{p_x,p_y,0} = T(wall)\} = \emptyset$ There is no wall in between the current state S and the goal H

Finally, all paths found are returned. If Θ at the first turn point T of a state S with $S.M = GOAL$ is smaller than a specified threshold[3], it can be assumed that all paths lead to the target. $T.\Theta$ can then be selected as a possible solution because it contains the angle before the bounce. Otherwise, the starting angle interval needs to be calculated recursively from the other turn points until it is lower than the threshold. To provide multiple solutions and not just the first, all states that lie in the previously calculated starting angle interval of S can be removed from the set of not expanded states. At some point, there are no possible goal states left, but since the search algorithm has no way of knowing that as long as it has not covered the full search space, a maximum number of expansions is needed in a practical implementation.

4 Experiments

We have implemented our method and in the following we present a case study on 10 levels (see Figure 4) with exactly one target that can only be hit after bouncing from another object. **BTC-1** and **BTC-2** are the levels 23 and 24 from Stephenson and Renz [2018]. **BTC-3** is level 12 from Stephenson and Renz [2018] with the modification that a red bird instead of a black one is used since with a red bird only the rebound option is present and no explosion (secondary action) near the target. **BTC-4** to **BTC-6** are levels from the AIBirds Competitions in 2016 and 2019 [Renz, 2015]. **BTC-7** is an official level from the Angry Birds Chrome Halloween Pack. **BTC-8**–**BTC-10** were created for this study. The levels cover the different aspects of one (**BTC-4**) or multiple (**BTC-7**) rebounds necessary, one (**BTC-9**) or multiple (**BTC-1**) options for reaching the target, possible angles close (**BTC-2**) or far (**BTC-10**) from the shortest path, and sliding (**BTC-6**). They existing ones were used since they were the only levels that were found under the official and the competition levels with the required traits. The additional levels were created to enrich the variety of problems. However they do not yet describe a full set of features, this and similar algorithms should be able to solve, due to time constraints.

In our evaluation, we performed qualitative simulation with a maximum of 50000 state expansions and for edge lengths $e \in [1, 15]$. One state expansion takes 1–2 ms on a standard computer with an implementation of the algorithm in Python 3.8. The portion of simulations with at least one possible solution is shown in Figure 7a and the correlation with the edge length in Figure 6. Precision and recall in respect to the possible angle intervals that were identified by hand for each level are shown in Figures 7b and 7c. Precision is defined here as the portion of degrees returned by the algorithm that lie inside the possible intervals, recall as the portion of degrees in the possible intervals that were returned by the algorithm.

The edge length is one of the most impactful variables of the algorithm right now. When increasing it, the number of required calculations reduces, however it also makes some narrower tunnels impassable. In Figure 6 the impact of this tradeoff on the possibility of finding at least one possible solution is shown.

Since the algorithm is designed to find multiple solutions, it is necessary to know after how many iterations the algorithm has found sufficiently many possible solutions. Figure 5 shows this in respect to a specific edge length.

In comparison, the Naive Agent provided by the organizers was not able find solutions for any of these scenes. A recent version of the BamBirds Agent with a simple ray-casting approach to rebounds was able to solve the scenes **BTC-1**, **BTC-6** and **BTC-10**. It also destroyed the target in **BTC-7** through a happy side effect and the target in **BTC-8** by using random variations of the naive agents result.

5 Discussion

As can be seen in the figures, the performance of the method depends on the discretization. First, Fig. 5 clearly shows that a smaller edge length increases the expansion necessary to find solutions with a growth more than linear as is to be expected. Figure 6 indicates that in the range 6-10 pixels for cell edge lengths the best balance between the limiting factors of efficiency and over-generalization errors can be achieved. However, precision and recall shown in Fig. fig:eval-level do yield a satisfactory performance with precision and recall both close to 1. The algorithm is very good for levels where the actual path along the parabola is very close to the shortest path to the target, such as in levels 2, 5, 6, and 8. Still good results, but with a longer runtime, are found in levels with a larger difference to the path such as in 3, 8, and 10. This shows the general utility of the qualitative simulation using transition rules. However, the algorithm performs badly in levels 1, 4, and 7. The reason for the problems in level 1 is that the representation with zones with larger edge lengths block the way through smaller tunnels, but a smaller edge length makes search less efficient and leads to generalization errors from individual zone transitions summing up considerably. In level 4 the reason of failure is mostly that the physical parameters are not accurate enough to handle rebounds from the ground. In level 7 the target is far away and it also re-

[3] In most cases the angle interval at the first turn point is very small with less than 0.1 degrees. That is already lower than the precision for shots in the AIBirds competition.

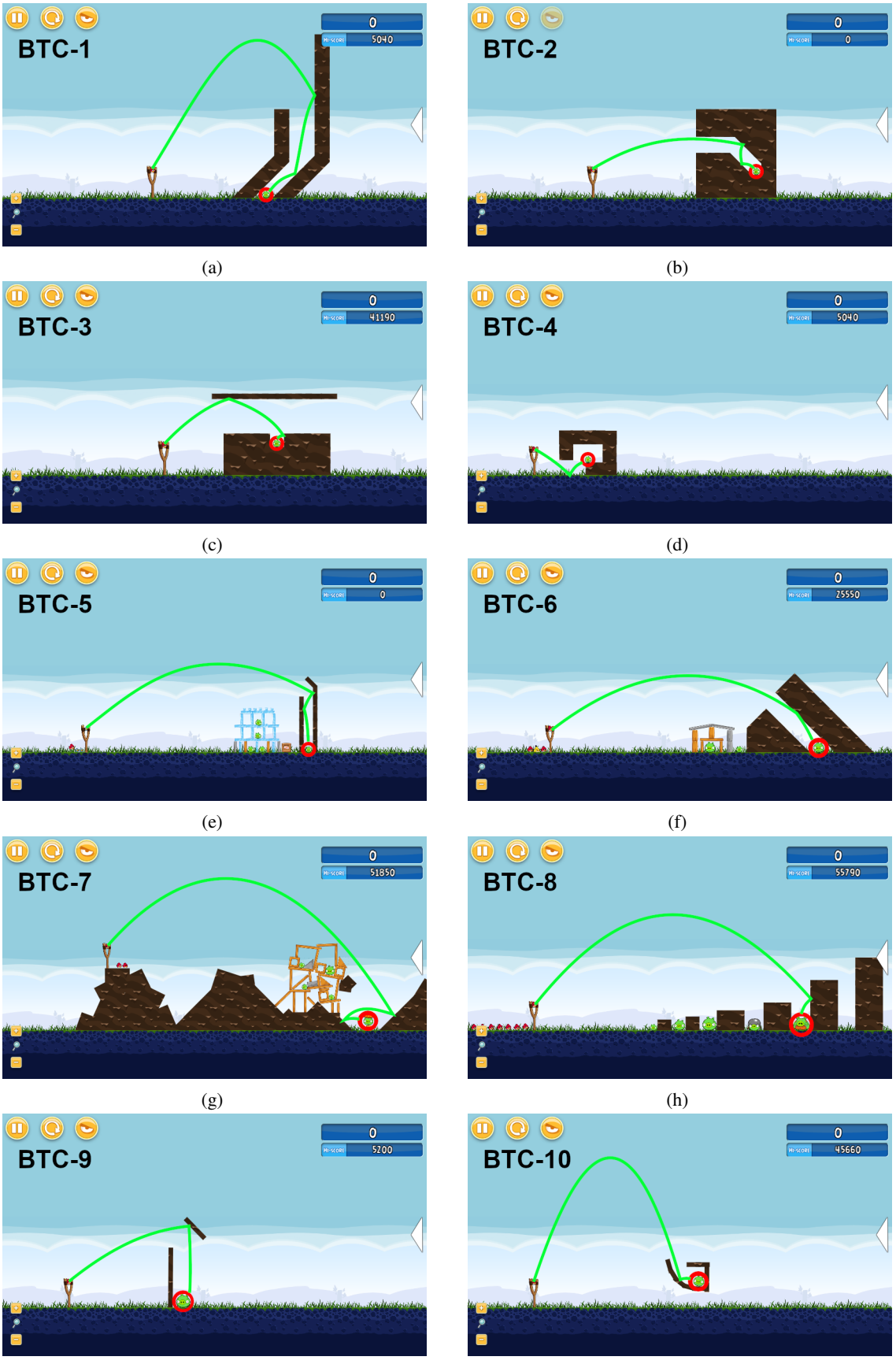


Figure 4: Scenes used for evaluation, the object with the red circle is the target, the green line is one possible solution

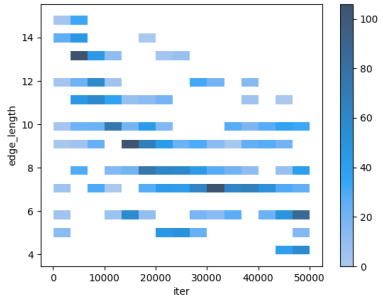


Figure 5: Histogram of the number of possible solutions at specific iterations for each edge length

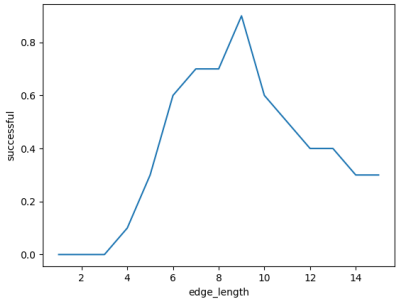


Figure 6: Percentage of simulations with at least one possible solution for different edge lengths

quires very good physical accuracy, which challenges search and avoiding build-up of generalization errors. In addition, the algorithm itself requires for this level with 9–40 seconds in its python implementation too much time for practical application.

In summary, the problems we observe arise from two factors. First, the chosen representation needs to be improved in order to achieve better precision and recall. The square-shaped zones were chosen for their simplicity and uniformity with respect to propagating direction changes. An adaptive cell decomposition, possibly using large cells (e.g., Delaunay triangulations as used by Ge et al. [2016]) would possibly be advantageous as the total amount of propagations is reduced. However, larger cells require a more detailed quantitative propagation of directions, steering the whole method closer to standard physical simulation.

Second, search needs to be improved. While breadth-first search achieves good efficiency, levels are not solved satisfactory if shots significantly differ from the heuristic given by the shortest path. Reducing the search space will reduce the time spent overall and can be achieved by a better representation as mentioned above, or by only searching in the required areas. For example, it is relatively easy to find all hittable object edges in the scene, so it would be a possibility to search only from these edges towards the target. Moreover, state-of-the art randomized algorithms such as Monte-Carlo Tree Search are applicable, similar to sampling-based motion planning in robotics [Elbanhawi and Simic, 2014].

Another limitation of the approach presented here is that it requires a sufficiently accurate knowledge of all physical properties, i.e. location of the objects and estimated rebound characteristics. To loosen this prerequisite, the dimension of the state space could be increased, incorporating more physical parameters similar to the approach of Forbus et al. [1991]. This would mean that zone transitions would not only propagate constraints regarding projectile direction, but also constraints regarding all other physical parameters get propagated. Of course, this would further challenge the search task.

The algorithm is expected to be generalizable to other similar problems with the requirement that object locations and sizes are known. The biggest problem are however as previously described are the physical parameters. In the current version of the algorithm they need to be accurate or else the algorithm will not produce reasonable results. Adjustment through observation and trial and error is an option to solve this, however that will often have undesirable side effects in real world scenarios. Compared to the results of the naive agent this still provides a major improvement. The raycasting approach in the recent BamBirds agent was also able to solve some of these levels, notably also ones that are harder for this algorithm (level 1 and 10), but also performed worse.

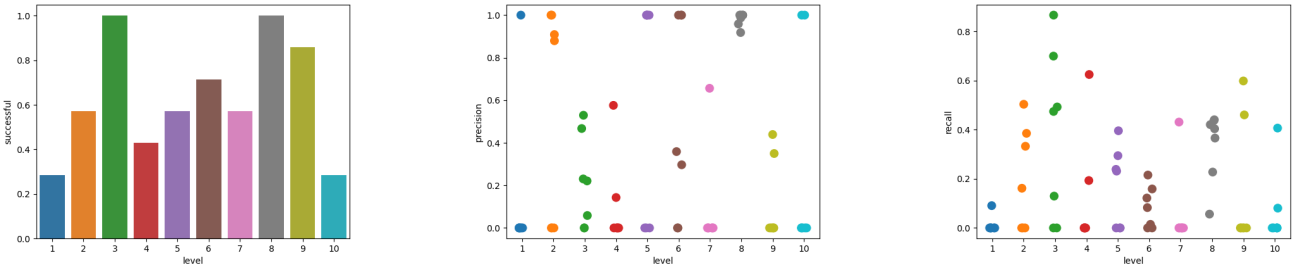
6 Conclusion

The presented approach addresses the challenge of adding gravity to the qualitative simulation by Ge et al. [2016], which complicates the quantitative propagation of directions as direction and velocity are inter-dependent. A uniform cell decomposition into square-shaped zones has been performed to avoid over-generalization errors arising from the non-linear dependence of direction and velocity in projectile motion. Our approach achieves solutions for a variety of example levels in Angry Birds. In particular, it can find accurate solutions for two of the hardest levels presented by Stephenson and Renz [2018] and other similar and previously unsolvable levels in under 50000 state expansions (which require around 40 seconds on an average desktop computer used for evaluation). However, the approach is far from being efficient or even close to human-level performance.

In future work we will investigate re-formulations of the search problem and alternative cell decompositions. We are interested to see whether a coarse planning with near-perfect recall (but low precision) can be realized that serves well as a heuristic for more fine-grained planning. We will also investigate the precision required by a qualitative simulation in order to make a sampling-based approach like Wolter and Kirsch [2015] effective for in-game trial and error by comparing missed shots against a qualitative plan in order to adjust shot parameters.

References

[Archibald *et al.*, 2010] Christopher Archibald, Alon Altman, Michael Greenspan, and Yoav Shoham. Computational pool: A new challenge for game theory pragmatics. *AI Magazine*, 31(4):33–41, 2010.



(a) Portion of simulations with at least one possible solution

(b) Precision of the solutions

(c) Recall of the solutions

Figure 7: Simulation results for each level, edge lengths between 6 and 12 are considered. Precision is here the amount of solutions that actually work, recall how many of the working solutions were found.

[Battaglia *et al.*, 2013] Peter W Battaglia, Jessica B Hamrick, and Joshua B Tenenbaum. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110(45):18327–18332, 2013.

[Berleant and Kuipers, 1997] Daniel Berleant and Benjamin J Kuipers. Qualitative and quantitative simulation: bridging the gap. *Artificial Intelligence*, 95(2):215–255, 1997.

[Bobrow, 1984] Daniel G Bobrow. Qualitative reasoning about physical systems: an introduction. *Artificial intelligence*, 24(1-3):1–5, 1984.

[Bresenham, 1965] Jack E Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems journal*, 4(1):25–30, 1965.

[Elbanhawi and Simic, 2014] Mohamed Elbanhawi and Milan Simic. Sampling-based robot motion planning: A review. *IEEE Access*, 2:56–77, 2014.

[Forbus *et al.*, 1991] Kenneth D Forbus, Paul Nielsen, and Boi Faltings. Qualitative spatial reasoning: The clock project. *Artificial Intelligence*, 51(1-3):417–471, 1991.

[Forbus, 1984] Kenneth D Forbus. Qualitative process theory. *Artificial intelligence*, 24(1-3):85–168, 1984.

[Forbus, 1997] Kenneth D Forbus. Qualitative reasoning, 1997.

[Ge *et al.*, 2016] Xiaoyu Ge, Jae Hee Lee, Jochen Renz, and Peng Zhang. Hole in one: Using qualitative reasoning for solving hard physical puzzle problems. In *Proceedings of the Twenty-second European Conference on Artificial Intelligence*, pages 1762–1763. IOS Press, 2016.

[Grace *et al.*, 2018] Katja Grace, John Salvatier, Allan Dafoe, Baobao Zhang, and Owain Evans. When will ai exceed human performance? evidence from ai experts. *Journal of Artificial Intelligence Research*, 62:729–754, 2018.

[Halliday *et al.*, 2013] David Halliday, Robert Resnick, and Jearl Walker. *Fundamentals of physics*. John Wiley & Sons, 2013.

[Kuipers, 1986] Benjamin Kuipers. Qualitative simulation. *Artificial intelligence*, 29(3):289–338, 1986.

[Latombe, 1991] Jean-Claude Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[Renz, 2015] Jochen Renz. AIBIRDS: the angry birds artificial intelligence competition. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[Stephenson and Renz, 2018] Matthew Stephenson and Jochen Renz. Deceptive angry birds: Towards smarter game-playing agents. In *Proceedings of the 13th International Conference on the Foundations of Digital Games*, FDG '18, New York, NY, USA, 2018. Association for Computing Machinery.

[Venkatasubramanian *et al.*, 2003] Venkat Venkatasubramanian, Raghunathan Rengaswamy, and Surya N Kavuri. A review of process fault detection and diagnosis: Part II: Qualitative models and search strategies. *Computers & chemical engineering*, 27(3):313–326, 2003.

[Walega *et al.*, 2016] P. A. Walega, M. Zawidzki, and T. Lechowski. Qualitative physics in angry birds. *IEEE Transactions on Computational Intelligence and AI in Games*, 8(02):152–165, apr 2016.

[Wolter and Kirsch, 2015] Diedrich Wolter and Alexandra Kirsch. Leveraging qualitative reasoning to learning manipulation tasks. *Robotics*, 4(3):253–283, 2015.

[Zhang and Renz, 2014] Peng Zhang and Jochen Renz. Qualitative spatial representation and reasoning in angry birds: the extended rectangle algebra. In *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR'14)*, 2014.

[Zickler and Veloso, 2009] Stefan Zickler and Manuela Veloso. Efficient physics-based planning: sampling search via non-deterministic tactics and skills. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 27–33. International Foundation for Autonomous Agents and Multiagent Systems, 2009.