Predictive Physics Simulation in Game Mechanics

Perttu Hämäläinen, Xiaoxiao Ma, Jari Takatalo
Aalto University
firstname.lastname@aalto.fi

Julian Togelius
NYU Tandon School of Engineering
julian@togelius.com

ABSTRACT

Computers can now simulate simple game physics systems hundreds of times faster than real-time, which enables real-time prediction and visualization of the effects of player actions. Predictive simulation is traditionally used as part of planning and game AI algorithms; we argue that it presents untapped potential for game mechanics and interfaces. We explore this notion through 1) deriving a four-quadrant design space model based on game design and human motor control literature, and 2) developing and evaluating six novel prototypes that demonstrate the potential and challenges of each quadrant. Our work highlights opportunities in enabling direct control of complex simulated characters, and in transforming real-time action into turn-based puzzles. Based on our results, adding predictive simulation to existing game mechanics is less promising, as it may feel alienating or make a game too easy. However, the approach may still be useful for game designers, for example, as it allows one to test designs beyond one's playing skill.

Author Keywords
Game design; visualization; physics simulation;

ACM Classification Keywords
H.5.2. User interfaces: Interaction styles

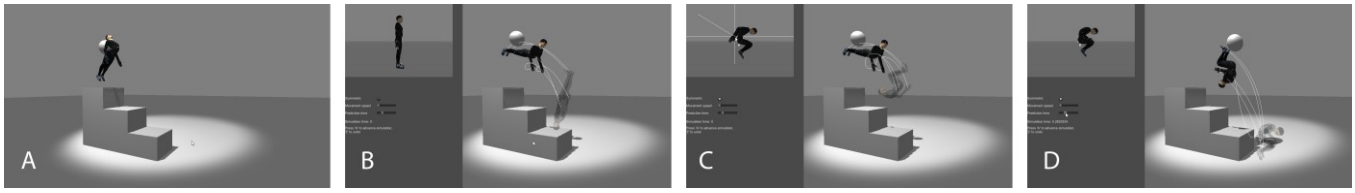

Figure 1. Our "animation as a game" prototype, where the player's goal is to save the falling character. A) Initial situation where the character is hit by a projectile. B) Game interface appears on the left, and the main view shows predictive movement trajectories and future pose; the character will hit its head unless the player acts. C) Using the posing controls (top-left), the player adjusts the future pose. Tucking increases rotation speed, which is predicted and visualized in real-time. D) Once satisfied with the prediction, the player advances the simulation. See the supplemental video at 03:00 for live gameplay.

INTRODUCTION

Simulation of dynamic physical systems utilizes a set of rules that are unambiguous and mathematically well-defined [5], but can at the same time lead to emergent, rich behavior. This has inspired many types of physically based games and game mechanics, with popular examples such as The Incredible Machine [4], Angry Birds [25], Cut the Rope [36], Where's My Water [3], QWOP [2], and Portal [34]. Gradually, game physics has evolved towards more complex systems, i.e., from 2D to 3D, and from rigid bodies (e.g., stacked objects) to soft bodies (e.g., ropes and cloth) and fluids. During recent years, no fundamentally new physics simulation types have appeared, but on the other hand, personal computing devices can now simulate simple systems hundreds or even thousands times faster than real-time. This paper focuses on capitalizing this excess computing power for novel game mechanics and interfaces.

In this paper, we argue that *real-time predictive physics simulation* provides an underexplored tool for game design. Most games simulate physical systems at a fixed rate such as 60 simulation steps per second, and only use simulations in a *reactive* manner to compute the interactions of objects based on player input. In contrast, we denote by predictive simulation a process of:

1) Saving the current simulation state
2) Simulating the physics forward for multiple steps, up to a prediction horizon,
3) Using the simulation results to preview and evaluate the results of player decisions
4) Restoring simulation state back to the saved state, so that the game may continue without discontinuities.

CHI PLAY '17, October 15–18, 2017, Amsterdam, Netherlands

ACM ISBN 978-1-4503-4898-0/17/10.
DOI: https://doi.org/10.1145/3116595.3116617

As a simple example, while the player aims a projectile, one can provide immediate visualization of the predicted trajectories of impacted objects. Figure 1 presents a more complex example of controlling a humanoid model; the same principle of providing immediate feedback through predictive simulation trajectories still applies.

As summarized in Table 1, predictive physics simulation is not a new concept, and it has been increasingly utilized by recent artificial intelligence (AI) and animation research (e.g., [11,32]). However, research has focused on non-player characters or enabling simple high-level control of a simulated avatar, e.g., making a bipedal character walk in a desired direction. This has obvious applications in action games such as God of War [27] or Uncharted [20], which presently need vast quantities of animation data to enable such high-level control. However, there are also various games such as Angry Birds [25] and QWOP [2], where the player directly controls simulations on a low level. In such games, predictive simulation seems less explored. This observation prompts our primary research question: *What novel possibilities and challenges emerge from combining predictive simulation with low-level direct control of simulated characters?*

We contribute to answering the research question through an exploratory research process of 1) defining a four-quadrant design space model based on human motor control theory, and 2) developing and evaluating six novel prototypes that demonstrate the potential and challenges of each quadrant. Thus, using the contribution types of Wobbrock and Kientz [35], we make both a theoretical and an artifact contribution. In the following, we first start with the design space model, and then discuss the relevant background, our prototypes, their evaluation with game designers, and finally, key design insights and lessons learned.

	Reactive simulation	**Predictive simulation**
High-level (abstract) control	Animation-driven action games, e.g., God of War, Uncharted.	Recent AI and animation research.
Low-level (direct) control	Direct physics control games (QWOP, Angry Birds)	***Underexplored.***

Table 1. A categorization of physics simulation uses. Predictive simulation is increasingly used in AI and animation research, which is however focused on high-level control of simulated characters.

	Simple actions	**Complex actions**
Turn-based control (no time pressure)	2D ball shooting (**Figure 2**)	Character posing, turn-based QWOP, timeline QWOP (**Figures 1,3,4**).
Real-time control (time pressure)	3D car (**Figure 5**)	3D bike (**Figure 6**)

Table 2. A four-quadrant model of the underexplored design space of Table 1, with our prototypes placed in the quadrants.

DESIGN SPACE MODEL

In game and animation AI, a primary function of predictive simulation is to help an intelligent controller evaluate different options when making decisions. Such controllers typically use a feedback control model, i.e., decisions are evaluated through comparing simulation results to some goals, and the comparison results then inform further decisions. A key observation behind our design space model is that much of human motor control and decision making can also be modeled using a similar feedback or closed-loop model [29]. Further, control difficulty is heavily affected by the following three factors:

- *Time pressure*, which depends on the control task. Adjusting actions such as steering a car based on sensory feedback from previous actions always takes some time [29].
- *Complexity of actions*, which depends on both the control task and control interface. Complexity also interacts with time pressure, as complex decisions take more time; for example, reaction time grows as a function of the number of stimulus-response mappings [29].
- *Possibility of anticipation*, which depends on information provided by the task and interface. Although our reaction time is limited, we can still execute sequences of quick actions if we can anticipate and plan ahead of time [29]. An example of this is provided by various music games which display timelines of upcoming notes to play, providing more anticipation and reaction time.

A key opportunity of predictive simulation is in providing anticipation and enabling rapid evaluation of action results before action is taken, which should help in control tasks with more time pressure and/or increased action complexity. Time pressure and action complexity are up to the game designer to decide, and the decision is crucial since designing appropriately difficult challenges for the player is a central part of game design [26,28]. Hence, we propose the 2D design space model of Table 2, with time pressure and action complexity as the coordinate axes. We have divided the model into four quadrants, and following this division, we have developed and evaluated at least one prototype to explore each quadrant.

To reiterate, the model in Table 2 is based on both game design literature (importance of appropriate difficulty) and human motor control research (factors affecting difficulty). In low-level control of simulated characters, action complexity increases with the number of controlled simulation parameters and the need to plan for multiple actions instead of just a single action. With respect to time pressure, simulations can be controlled in real-time or turn-based fashion. Overall difficulty grows towards the bottom

right corner of our model. We do not claim that our model captures all dimensions of the design space; our purpose is to highlight important dimensions that should provide sufficient focus for the rest of this paper. Of the three factors affecting difficulty listed above, anticipation is not explicitly included in the model in Table 2, as improved anticipation is automatically implied by using predictive simulation and visualization. In each quadrant, the degree of anticipation can be fine-tuned by selecting what to visualize and adjusting the prediction horizon.

BACKGROUND AND RELATED WORK

Before describing our prototypes in detail, we now make a brief detour into reviewing related work.

Reactive physics simulation in games

Reactive physics simulation in digital games dates back to at least the gravity simulations of Space War, Lunar Lander and Artillery [18]. Advances in simulation technology and computing power have inspired new game mechanics and types such as the water manipulation in Where's my Water [3]. Initially, simulations were only computed in two dimensions and with rigid bodies, but recently, even 3D large scale water simulation has found use in gameplay [13].

While our work focuses on simple mouse and keyboard interfaces, our Table 1 is inspired by Liu and Zordan [17], who similarly divide game interfaces into literal (low-level) and symbolic (high-level) in the context of gestural interaction. Both interface types have their strengths and weaknesses. Low-level control of simulation parameters such as a bipedal character's joint motor speeds can give rise to emergent and interesting movement, as in the interfaces of Laszlo et al. [15] and later in the games QWOP [2] and Toribash [19]; both games are however extremely difficult. Although the difficulty can give rise to physical comedy [31], many games instead simplify and abstract the interface to a level where the player simply presses a button to trigger complex predefined character animations. Creating and finetuning the animations is however costly, and some degree of movement emergence is lost, although physics simulation can still be used for reactions of the game world.

Predictive physics simulation in animation and AI

Predictive physics simulation is increasingly used in procedural animation and AI research. The basic idea is that since the outcomes of multiple alternative actions can be simulated and evaluated, an optimization algorithm can choose which action to take and which actions to evaluate next. This has a long history in animation synthesis and simulated robot control (e.g., [1,9,16,21]), with recent methods capable of real-time control of 3D humanoid bipeds [11,32]. Predictive physics simulation has also been used, especially together with stochastic search algorithms such as Monte Carlo Tree Search, for controlling non-player characters in video games [7,12], and for providing real-time feedback to level designers regarding the playability and possibilities of level designs [30].

In close relation to our work, Twigg and James [33] investigated predictive trajectory visualizations for adjusting the starting conditions of simulations such as trebuchet shooting; our ball shooting prototype in Figure 2 extends this with the interactive heatmap. We are also inspired by Laszlo et al. [14] who introduced predictive simulation and visualization for animation interfaces close to our prototype in Figure 1. We extend their work with full-body humanoid characters, visualization of trajectories in addition to poses, and the timeline interface of the prototype in Figure 4.

Predictive physics simulation in game interfaces

Although many games such as Angry Birds do visualize predictions of object movement trajectories, it appears that such predictions are mostly computed using more computationally efficient but also more limited methods than predictive simulation. For example, basic ballistic equations allow computing projectile trajectories but not the subsequent complex interactions of impacted objects. Toribash [19] and Peggle [22] provide rare counterexamples. In Toribash, the player controls a 3D simulated humanoid, and the game proceeds in a turn-based fashion. While the player is adjusting simulation controls, the game simulates and renders what will happen in the next few seconds. Interestingly, Toribash frames the complexity of Laszlo et al. [14] style physically based animation as a game challenge. However, predictions are only computed one rendered frame at a time, which adds significant delay in contrast to our immediate visualizations. Peggle employs immediate projectile trajectory visualization akin to Twigg and James [33]. It is noteworthy that although such visualization might make a game too easy, Peggle cleverly uses it as a limited resource, allowing longer predictions as a power-up.

PROTOTYPES

The following lists our prototypes, structured following the four-quadrant model of Table 2, and the monikers used when referring to them in the rest of the paper:

1. *Turn-based control of simple actions*: the "ball shooting" prototype of Figure 2 (supplemental video at 00:20).
2. *Turn-based control of complex actions:* the "character posing" prototype of Figure 1 (video at 03:00), "turn-based QWOP" of Figure 3 (video at 02:02), and "timeline QWOP" of Figure 4 (video at 02:28).
3. *Real-time control of simple actions:* the "car" prototype of Figure 5 (video at 01:05).
4. *Real-time control of complex actions:* the "bike" prototype of Figure 6 (video at 01:28)

We start our discussion from the most straightforward case of simple actions and turn-based control, and then extend the ideas to real-time control and more complex actions.

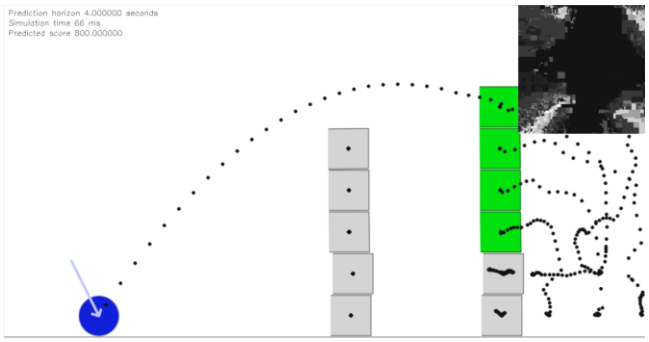

Figure 2. A ball shooting prototype inspired by Angry Birds. The player defines launch speed and angle of the blue ball, with the goal of only moving the green cubes. One can also shoot using a heatmap (top-right), where bright pixels correspond to good speed and angle combinations.

Figure 3. A turn-based version of the game QWOP, where the effect of keypresses is predicted and visualized, and time only advances when the player holds down the spacebar.

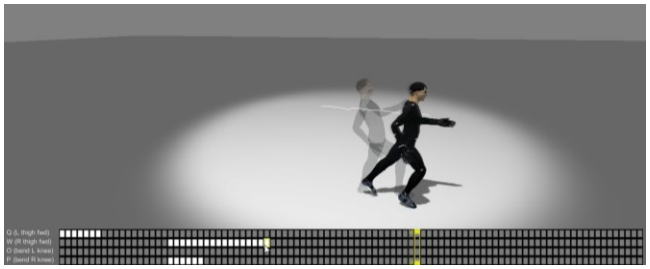

Figure 4. Timeline-based version of the game QWOP. In this prototype, the player paints the keystrokes on an animation timeline with tracks for the Q,W,O,P keys. Animation automatically plays in a loop (yellow frame indicator, solid character). The ghost character shows state at mouse position.

A note on the design process: The work towards this paper originates from a serendipitous discovery in context of intelligent animation control research, as the first author developed an initial implementation of the character posing prototype. We considered the prototype interesting, which motivated the construction of the design space model and developing prototypes for the other quadrants. The development of the QWOP versions then followed, as the emergent and rich movements of a simulated humanoid appeared more interesting than the ball, car, or bike.

Turn-based control of simple actions

In turn-based control, feedback is often delayed and the player must try and iterate multiple times to get an action right. Here, the key potential of predictive simulation is in providing feedback already while the player searches for a suitable action. We demonstrate this in the ball shooting prototype of Figure 2. The player controls the launch speed and angle similar to Angry Birds [25], with the goal of only moving the green boxes. We use predictive simulation to provide immediate visualization of the trajectories of objects and a prediction of the score (200 points for every green box that moves more than a threshold, minus 100 points for every gray box that moves).

In the ball shooting case, the action space is only two-dimensional (a shot is fully defined by speed and angle). In such cases, actions can be explored automatically to produce a heat map that gives a visual overview of possible player actions (the top-right grayscale rectangle in Figure 2). Four high-scoring strategies show up as bright clusters, the higher ones corresponding to bouncing the ball off the ground, and rightmost ones corresponding to bouncing the ball off the left wall. The user can interactively browse the heatmap with mouse to view the shot trajectories.

Brute force heatmap computation can be slow; at about 5-10ms per simulation and 256x256 pixel heatmap, the computation takes several minutes. However, as we demonstrate on the supplemental video, a non-negative scoring function can be treated as an unnormalized probability density function, which allows for quick computing of an approximate heatmap using importance-sampling. We use kD-tree Sequential Monte Carlo Sampling [10], which adaptively increases resolution where score is high.

Turn-based control of complex actions

Our prototypes in Figure 1, Figure 3, and Figure 4 extend turn-based control to the more complex case of a simulated humanoid. In our designs, we recombine ideas from three games: QWOP, Toribash, and Backflip Madness [2,8,19]. All three games implement an interface for controlling a physically based simulated humanoid character, but the games differ in how they manage the control complexity. QWOP only uses a 2D character, which prevents falling sideways, and simplifies the controls into discrete keystrokes; Q and W drive the left and right thigh forward, and O and P bend the left and right knee. Toribash uses a full 3D humanoid, but gives the player unlimited time to think – the player can explicitly control when simulation is advanced. Backflip Madness reduces complexity by allowing the player to only control the timing of transitions between predefined poses such as crouching, extended, and tucking.

We consider two main sources of complexity. The character posing prototype of Figure 1 focuses on the high dimensionality of action space, and the QWOP-inspired prototypes of Figure 3 and Figure 4 utilize simplified actions (only a set of 4 actions corresponding to the Q,W,O,P keys) that however require precise timing and coordination.

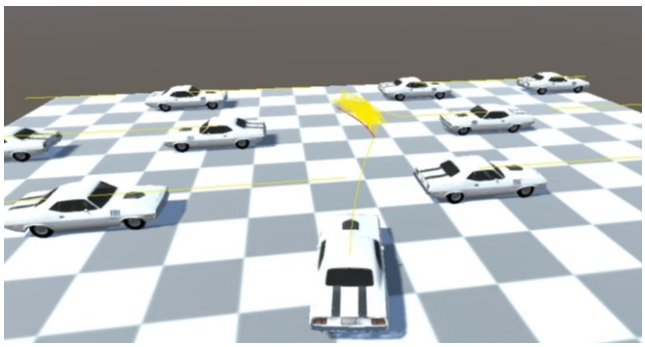

Figure 5. Prototype where the player tries to drive through the traffic at full speed. Holding down ctrl key shows predicted trajectories of all cars. In this image, the end of the player's trajectory is red, which signals predicted future collision.

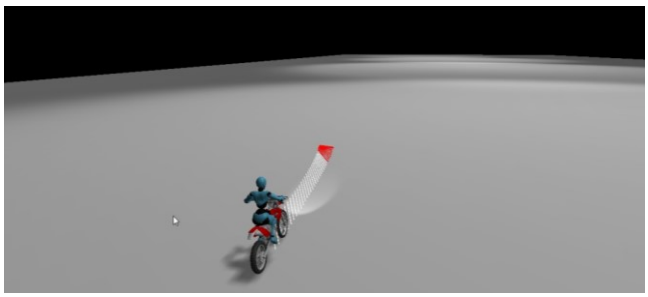

Figure 6. Prototype where horizontal mouse movement controls the steering of a 3D bike without any artificial balancing. The bike's predicted movement and balance is predicted. Red color signals falling.

In our character posing prototype, the user controls when simulation is advanced; compared to Toribash, we provide faster feedback and more precise control. The game interface features an animation rig that the player can manipulate using a mouse. This defines a target pose that the simulated character is driven towards. The predicted trajectories and the resulting pose at the prediction horizon are also visualized. In Figure 1 the player is making the character tuck to rotate faster and avoid hitting its head. The player can also adjust the prediction horizon with a slider. The supplemental video shows how the character can be landed safely by first tucking, and then extending the body near the ground.

Figure 3 shows a turn-based version of QWOP, where the player is given a fifth key – simulation time advances only when the player holds down spacebar. The effects of other keys are predicted and visualized, allowing the player to explore the best combination of keys.

In initial testing, the turn-based QWOP was found quite difficult despite the unlimited time for deciding on the keypresses. This is why we developed the concept further, resulting in the prototype in Figure 4. Instead of pressing the keys, the player paints the keypresses on a timeline with a track for each key. The painted keypresses can also be freely erased, in essence transforming an originally real-time action game into a puzzle. The interface provides immediate visualization of character state at the mouse position (the semitransparent character) and the character's movement trajectory over the whole timeline.

Real-time control of simple actions

Our car and bike prototypes in Figure 5 and Figure 6 examine how the ideas presented above can be extended to real-time control. The car driving represents the case of simple actions. Horizontal mouse movement is mapped to steering, and the player has also keys for accelerating, braking, and turning on the predictive simulation and visualization. The player's goal is to drive through the traffic at full speed without hitting other cars. The predictive trajectory's color indicates whether the car will collide with others using the present steering direction.

Real-time control of complex actions

As the more complex real-time case we use the 3D bike steering and balancing prototype of Figure 6. We are inspired by the bikes of Trials HD [23], which however are only simulated in 2D. A 3D bike is more complex to control than a 2D bike or a car, as balance is coupled to steering through centripetal force; when steering right, one has to actually first countersteer left to make the bike start falling right. To prevent the bike from falling all the way, one must then steer right. Complexity arises from the needed precise temporal coordination of steering movements. As shown on the supplemental video, balancing is highly difficult because it is nearly impossible to estimate the correct steering magnitude, and both over- and understeering quickly lead to lost balance.

Similar to the car prototype, horizontal movement of the mouse is directly mapped to bike steering. Left mouse button controls throttle.

Technology: simulation cost and determinism

In modern game development, physics simulation is rarely implemented from scratch. Instead, games often use established libraries and toolkits such as the Open Dynamics Engine (ODE), Bullet, or Box2D. Our ball shooting prototype uses Box2D v2.3.0 for simulation and OpenCV for visualization. The other prototypes use Open Dynamics Engine (ODE) integrated with Unity 3D game engine as a native plugin. The predictive simulations take around 5-40ms on a 2.6GHz Intel i7-6700HQ processor, depending on simulation complexity and prediction horizon. The car prototype is the most computationally heavy due to multiple simulated cars and 3D simulation being slower than 2D.

In the implementation, we had to solve the problem of neither ODE or Box2D simulations being *fully deterministic and reproducible*. The simulators do not by default provide functionality for simulating forward from a given state and then restoring the starting state in a way that would ensure exactly same results in further simulations. This was also the reason for us not being able to use Unity3D's built-in physics, which has the additional complication that

simulation time cannot be advanced independent of graphics rendering time.

In principle, saving and restoring movement state (position, rotation, velocity, and angular velocity) of all objects should be enough, but subsequent simulations can still diverge, due to the following main reasons:

- Simulators use random numbers, e.g., to process solver constraints in random order for better simulation stability.
- Simulators use additional internal state for speeding up the simulation. For example, ODE caches some variables inside the joint classes, and Box2D builds a dynamic spatial subdivision data structure for faster collision handling.

For deterministic simulation, all internal state, including random number generator state, must be saved and loaded together with movement state. We have implemented this in both ODE and Box2D and the source code is available at *https://bitbucket.org/perttuhamalainen/predictivesimulation*

EVALUATION DESIGN

Participants

We evaluated the prototypes qualitatively with 6 game design M.A. students and one game designer/design teacher (all male, age m=30, sd=5). All participants had experience from designing several games (m=9, sd=4).

Procedure

Each participant played all prototypes for 5-10 minutes while thinking aloud, and then answered a set of final questions. The order of prototypes used Latin Square counterbalancing. The total experiment duration was about 1 hour per participant.

For the prototypes that could also be played "regularly" without the predictive simulation (ball shooting, car, bike), the participant first played without prediction and then again with prediction turned on. We then asked which game version was better and why. The QWOP game versions were played with order altered between participants, and we again asked which version was better and why.

After all prototypes had been tested, we asked the following questions, and also probed further with why-questions:

1. Which of the prototypes were the most interesting?
2. Which of the prototypes were the least interesting?
3. What benefits and challenges does predictive physics simulation present from a game design perspective?
4. Which quadrant of Table 2 would the participant be interested in exploring further or utilizing in a game?

Data analysis

A thematic analysis of participant answers and comments was performed, with the goal of informing future design and research, and to identify the primary benefits and challenges of predictive physics simulation.

EVALUATION RESULTS

In the following, the participants are referred to as P1…P7 and their quotes are in italics.

Most interesting prototypes

The prototypes that were found most interesting were the character posing prototype of Figure 1 (all participants), the QWOP versions (5 participants mentioned one or both of the two, and there was no clear consensus of which version was better), and the bike (two participants). All of these except the bike also frequently resulted in comical blunders that made the participants laugh.

A central theme was that participants found it intriguing to understand and learn about the dynamics of movement. "*You never really pay attention to what parts of the body need to move to execute some actions. You have to think about individual actions that we otherwise are not aware of. Also the motorbike one was interesting because the countersteering is so counterintuitive – learned about how movement and physics work.*" (P2), *"I felt like having a superpower, that I understood better how people and the world work, and I got the character to make an insane acrobatic move"* (P6 on the character posing), *"Something like Quicksilver in X-Men movies"* (P4 on the character posing), *"Somehow makes me want to try again and again"* (P1 on the bike), *"Prediction is better. It seems to teach me how the bike works. I cannot otherwise understand why I fail."* (P6 on the bike), "*A fun variation of the original, which I have tried to learn for a long time. I now succeeded on first try"* (P6 on the turn-based QWOP).

One participant also selected the ball shooting prototype as the most interesting when considering the heatmap as a tool for level design. As a player, he too preferred the character posing prototype.

Least interesting prototypes

The prototypes that were found least interesting were the car (3 participants), the ball shooting (3 participants), the bike (2 participants), and the QWOP versions (1 mention for each).

Most of the mentioned prototypes are games that one could also play without predictive simulation. Although some participants preferred the versions with prediction because they were easier, a recurring concern was that prediction make the gameplay different and even alienating, as one's focus shifts from the controlled character to the trajectory visualizations. *"Prediction system was confusing, I was looking at the line, not the guy."* (P4 on the bike), *"Prediction makes it different, and removes the core game"* (P5 on the car), *"Prediction requires re-learning with the bike and car, because you pay attention to completely different things. I was not looking at the car at all."* (P7), *"It feels like it removes something"* (P6 on driving games), *"No feeling of being an awesome motorcyclist. Hard to be excited when the game's challenge is not to do cool tricks but instead just to follow a line and try to keep it white instead of red"*

(P3), *"I seem to stop using my intuition, I just think of the trajectories"* (P6 on the ball shooting).

An additional reason for mentioning the bike was too hard difficulty. None of the participants could keep the bike balanced without the predictive visualization. Four participants could keep the bike balanced with the predictive visualization, but balancing required so much effort and concentration that they could not steer the bike where they wanted.

Too hard difficulty was also the reason for mentioning the QWOP versions.

Benefits and challenges of predictive simulation

To summarize the pros and cons discussed during testing and in response to our questions 3 and 4, we identified the following main themes:

1. Positive: predictive simulation allows players to tackle new challenges and discover novel solutions and movements. "*An interesting puzzle. I've sometimes tried to play Toribash, but I got nowhere."* (P6 on the prototype of Figure 1), *"Discovery was the most interesting thing about the prediction"* (P2), *"Enables completely new levels and impossible scenarios that have a solution that the player otherwise wouldn't find except through insane trial and error"* (P7), *"Novel game mechanics in the style of the posing prototype."* (P6), *"It was interesting to see that there are four solutions when I assumed there's only one"* (P6)
2. Negative: predictive simulation can make a game too easy. *"Prediction clearly helped, but made the game less interesting due to reduced challenge"* (P7 on the car), *"Prediction is too easy."* (P6 on the car), *"It's hard to balance the challenge – easily either too hard or too easy and feels like cheating"* (P5), *"It would maybe be more fun to use the prediction for searching for a strategy if the puzzle was more difficult"* (P5), *"Prediction makes me lose interest in prototypes that otherwise feel challenging"* (P7 on the car and bike)
3. Mixed: even if always-on predictive simulation can be detrimental in some games, it may be useful in level design and testing or as a limited resource or power-up. *"Prediction is more useful for level designer, if you want to design and test a really hard level"* (P1 on the car prototype), *"Would be useful as a limited resource or power-up. As a player I would not use all the time, if it is not necessary for the game mechanics"* (P1), *"Seeing the prediction ruins the game, but would work for a tutorial"* (P2), *"Useful as a design tool in a car game if I want to test whether something is possible* <for a highly skilled player>*"* (P7), *"The heatmap is useful as it shows distinct solutions and rules out things you might otherwise try in testing"* (P2), *"Prediction would work well for game design and testing"* (P4)

Most interesting quadrants

All participants found turn-based control of complex actions as the most interesting quadrant, although P2 commented: *"As a game designer, turn-based control of complex actions, because I can learn and analyze. As a player, real-time control of complex actions, as I don't want to analyze so much during gameplay*".

DISCUSSION

In this section, we present some additional observations and revisit our design space dimensions in light of the results.

Turn-based vs. real-time control

Considering all the prototypes and the evaluation results, turn-based control based on predictive simulations seems more promising than real-time control.

In our interpretation, the key difference between turn-based and real-time control is that optimizing the level of challenge is much more difficult in the latter. In real-time gameplay, it can be difficult to process visual information fast enough; predictive simulation can require the player's full attention as in our bike prototype, while still not making the control task easy enough to be enjoyable.

In real-time gameplay, predictive visualization also easily moves the focus of attention away from the controlled object or character, which many participants considered distracting. In our turn-based prototypes that were found most interesting (character posing and QWOP versions), one's focus of attention is not completely on the control interface or predictive visualization; instead, focus shifts back to the character when one advances the simulation or evaluates the animation resulting from keypresses painted on the timeline. Turn-based control also has less restrictions on how much visual information can be processed and utilized.

Simple vs. complex actions

Our prototypes with complex simulations appear more successful, in particular the ones with humanoid characters. In such simulations, predictive visualization can enable novel control interfaces and discovery of interesting emergent movement.

For simple actions, it appears that predictive simulation is not as useful as it can make a game too easy. The movement discovery capabilities provided by predictive simulation might be more useful as a game design and testing tool.

Designing novel mechanics and interfaces

We agree with P4 and P5 who commented that rather than adding predictive simulation to existing games and interfaces, one should probably design novel game mechanics and interfaces around the predictive simulation. Our best performing prototypes (the posed character and QWOP versions) were of the latter type. Although our QWOP versions are based on an existing game, they do not simply reduce the level of challenge; they play completely differently than the original. A key lesson learned is that *predictive simulation and visualization allows transforming real-time action games into turn-based puzzle*s. Our

prototypes demonstrate that achieving the intended movement is still challenging and can provide surprising outcomes, as slight differences in input can result in large deviations of realized movement. This is characteristic to all physics simulations with contacts; shooting a ball at slightly different angles can result in completely different trajectories depending on whether or not the ball hits an obstacle.

Interestingly, P5 explained that the original QWOP feels more comical because of the faster try and fail cycle, but P6 on the other hand commented that the turn-based QWOP is better because *"I see the possible futures and they are comical in the same way as in the original, but I don't have to fail for real."*

LIMITATIONS AND FUTURE WORK

Due to our qualitative and exploratory research approach, our evaluation data does not provide quantitative evidence in favor or against hypotheses such as "visualization A is better than B in task C". Readers should also use their own judgement in assigning weight to the user comments – they represent the views and experiences of individual players rather than the collective experience of a larger player population.

We opted for this approach as we wanted to gain a broad understanding of the design space and generate a wide variety of examples that could provide ideas for future games and ideas or hypotheses for future research. As an example of such a hypothesis, our results suggest that predictive visualization helps in the complex case of steering and balancing a 3D bike; this could be validated in a quantitative follow-up study, possibly comparing different visualizations in terms of task performance and player enjoyment.

Based on our ball shooting prototype (Figure 2), we also see future potential in characterizing level designs based on multiple simulations – for example, we hypothesize that good levels have multiple bright heatmap areas (i.e., multiple alternative strategies), and each area is large enough (no need for pixel-perfect control). Naturally, with more complex actions, such heatmaps and summary visualizations are likely to need advanced multidimensional data visualization techniques.

Although we have limited ourselves to low-level simulation control as opposed to utilizing predictive simulation for high-level AI-based control, we intend to investigate hybrids of the two approaches in future work; this might provide better tools for adjusting control difficulty and control abstraction/indirection. For example, extrapolating the concept of Draw Race [24] and Flight Control [6] (draw the desired path of a car or a plane to control it), a humanoid parkour character could be controlled indirectly by indicating the desired sequence of foot placements in advance, while getting predictive feedback of how the AI will succeed in obeying the instructions.

CONCLUSION

In this paper, we have proposed a four-quadrant model of the design space of low-level game or simulation control with predictive physics simulation and visualization, based on the key design dimensions of action complexity and time pressure (Table 2). Furthermore, we have presented the design and evaluation of six prototypes that demonstrate novel possibilities and challenges of each quadrant. Drawing on our experiences, observations, and the user comments, we now conclude with a summary of design insights and lessons learned:

1. Predictive physics simulation appears a promising tool for designing novel game mechanics and interfaces, in particular considering turn-based control of complex actions, such as our character posing prototype (Figure 1). Previewing results of player actions before the actions are executed is a useful design pattern, provided that it does not make a game too easy.
2. As demonstrated by our QWOP variations, predictive physics simulation allows creating novel games through transforming a real-time action game into a turn-based puzzle.
3. One must be cautious in simply adding predictive simulation and visualization to existing games and mechanics. For example, it was found detrimental to our car prototype, as it made the game too easy and shifted the player's focus from the car to the trajectory visualizations.
4. Even if predictive simulation does not improve the player's experience, it may still be useful as a limited power-up, in tutorial use, or for the game designer. For example, it can be used to explore strategies that the designer might not otherwise think of, or to test difficult levels beyond the designer's playing ability.
5. Considering technological aspects, it appears that most physics simulators are not designed for predictive simulations. We hope our work informs future game and physics engine development of the need to provide full simulation determinism, reproducibility, and an interface for loading and saving both movement state and all internal state.

ACKNOWLEDGEMENTS

We thank our participants and the anonymous reviewers for their insightful comments. Takatalo is supported by Academy of Finland grant 299358.

REFERENCES

1. Christopher G Atkeson. 2007. Randomly sampling actions in dynamic programming. In *2007 IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning*, 185–192.
2. Bennet Foddy. 2008. *QWOP*. http://foddy.net, Game [browser].
3. Creature Feep. 2011. *Where's my Water*. Game [iOS].
4. Dynamix. 1993. *The Incredible Machine*. Game [PC].

5. David H Eberly. 2010. *Game physics*. CRC Press.
6. Firemint. 2009. *Flight Control*. Game [iOS].
7. Jacob Fischer, Nikolaj Falsted, Mathias Vielwerth, Julian Togelius, and Sebastian Risi. 2015. Monte Carlo Tree Search for Simulated Car Racing. In *Proc. FDG'15*.
8. Gamesoul Studio. 2013. *Backflip Madness*. Game [iOS].
9. T. Geijtenbeek and N. Pronost. 2012. Interactive Character Animation Using Simulated Physics: A State-of-the-Art Review. *Computer Graphics Forum* 31, 8: 2492–2515. https://doi.org/10.1111/j.1467-8659.2012.03189.x
10. Perttu Hämäläinen, Sebastian Eriksson, Esa Tanskanen, Ville Kyrki, and Jaakko Lehtinen. 2014. Online Motion Synthesis Using Sequential Monte Carlo. *ACM Trans. Graph.* 33, 4: 51:1–51:12. https://doi.org/10.1145/2601097.2601218
11. Perttu Hämäläinen, Joose Rajamäki, and C. Karen Liu. 2015. Online Control of Simulated Humanoids Using Particle Belief Propagation. *ACM Trans. Graph.* 34, 4: 81:1–81:13. https://doi.org/10.1145/2767002
12. Emil Juul Jacobsen, Rasmus Greve, and Julian Togelius. 2014. Monte Mario: Platforming with MCTS. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation* (GECCO '14), 293–300. https://doi.org/10.1145/2576768.2598392
13. Timo Kellomäki. 2015. *Large-Scale Water Simulation in Games*. Tampere University of Technology.
14. Joe Laszlo, Michael Neff, and Karan Singh. 2005. Predictive Feedback for Interactive Control of Physics-based Characters. *Computer Graphics Forum* 24, 3: 257–265. https://doi.org/10.1111/j.1467-8659.2005.00850.x
15. Joseph Laszlo, Michiel van de Panne, and Eugene Fiume. 2000. Interactive Control for Physically-based Animation. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (SIGGRAPH '00), 201–208. https://doi.org/10.1145/344779.344876
16. Stephen R. Lindemann and Steven M. LaValle. 2005. Current Issues in Sampling-Based Motion Planning. In *Robotics Research. The Eleventh International Symposium*, Paolo Dario and Raja Chatila (eds.). Springer Berlin Heidelberg, 36–54. https://doi.org/10.1007/11008941_5
17. C Karen Liu and Victor B Zordan. 2011. Natural user interface for physics-based character animation. In *Proc. International Conference on Motion in Games*, 1–14.
18. Henry Lowood and Raiford Guins. 2016. *Debugging Game History: A Critical Lexicon*. MIT Press.
19. Nabi Studios. 2006. *Toribash*. Game [PC].
20. Naughty Dog. 2007. *Uncharted: Drake's Fortune*. Game [PlayStation 3].
21. J. Thomas Ngo and Joe Marks. 1993. Spacetime Constraints Revisited. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (SIGGRAPH '93), 343–350. https://doi.org/10.1145/166117.166160
22. PopCap Games. 2007. *Peggle*. Game [PC].
23. RedLynx. 2009. *Trials HD*. Game [Xbox 360].
24. RedLynx. 2009. *Draw Race*. Game [iOS].
25. Rovio Entertainment. 2009. *Angry Birds*. Game [iOS].
26. Katie Salen and Eric Zimmerman. 2003. *Rules of Play: Game Design Fundamentals*. The MIT Press.
27. SCE Santa Monica Studio. 2005. *God of War*. Game [PlayStation 2].
28. Jesse Schell. 2008. *The art of game design a book of lenses*. Elsevier/Morgan Kaufmann, Amsterdam; Boston.
29. R. A. Schmidt and C. A. Wrisberg. 2008. *Motor Learning and Performance*. Human Kinetics.
30. Noor Shaker, Mohammad Shaker, and Julian Togelius. 2013. Ropossum: An Authoring Tool for Designing, Optimizing and Solving Cut the Rope Levels. In *Proc. AIIDE 2013*.
31. Jaroslav Švelch. 2014. Comedy of Contingency: Making Physical Humor in Video Game Spaces. *International Journal of Communication* 8, 0: 23.
32. Yuval Tassa, Nicholas Mansard, and Emo Todorov. 2014. Control-Limited Differential Dynamic Programming. In *Proc. IEEE International Conference on Robotics and Automation* (ICRA'14).
33. Christopher D. Twigg and Doug L. James. 2007. Many-worlds Browsing for Control of Multibody Dynamics. In *ACM SIGGRAPH 2007 Papers* (SIGGRAPH '07). https://doi.org/10.1145/1275808.1276395
34. Valve Corporation. 2007. *Portal*. Game [PC].
35. Jacob O Wobbrock and Julie A Kientz. 2016. Research contributions in human-computer interaction. *interactions* 23, 3: 38–44.
36. ZeptoLab. 2010. *Cut the Rope*. Game [iOS].