# Making Games to Teach Physics and Mechanics

**Dr. David I. Schwartz, Rochester Institute of Technology (GCCIS)**

David I. Schwartz, Ph.D. has been working in the academic field of game design and development since 2001 when he founded the Game Design Initiative at Cornell University. In 2007, Schwartz moved to the Rochester Institute of Technology. He was part of the founding department in 2009, which became the School of Interactive Games and Media in 2011. After receiving tenure in 2011, he became IGM's Director in 2015. His research focus is on cyber security games, geogames, and physically-based animation.

# Making Games to Teach Physics and Mechanics

David I. Schwartz, PhD
School of Interactive Games and Media
Rochester Institute of Technology
igm.rit.edu

Abstract

This survey paper introduces engineering educators to a subfield of computer graphics called physically-based animation (PBA) to advocate for collaboration in creating courses to improve student learning in STEM fields, especially in engineering. Engineering students may not realize the degree to which they can leverage their education to enter the entertainment and simulation industries. The central hypothesis of the paper is that introductory physics can be taught via PBA. The paper provides case studies that demonstrate early promise. The paper gives an overview of how “game engineers” leverage theoretical physics and mathematical concepts merged with design aesthetics to portray realistic and fun experiences, manifesting as game physics. As computing power has increased, the convergence of *real* and *fake* physics presents an opportunity to teach physics to non-game students. The field of PBA shows promise for enhancing physics education, but much work remains to determine how it may happen and the place in a college curriculum.

## 1 Call to action

This paper initiates a call-to-action for STEM educators who may be unaware of a kind of *entertainment computing*—game physics and the broader field of physically-based animation, which are defined below. Games are very appealing, and there are entire colleges producing graduates in the entertainment arts and technology. There is an opportunity to “join forces,” so to speak (and forgive the pun), to collaborate to improve learning and retention for all:

**Can students learn physics by making games?**

The paper summarizes current research and findings in physics/game physics education that show the potential of teaching the *real* with the *fake*. Starting with basics of animation, the paper steps the reader through the various fields and references that introduce concepts from outside of typical STEM. Case studies provide animations and interactive examples not easily shown in a paper. To clarify the hypothesis, a brief summary of the author’s game physics course on teaching “fake physics” is presented to help demonstrate the striking connections with “real physics.” The paper concludes with a summary and a roadmap to implement this call-to-action to explore the exciting potential of games physics to teaching physics outside games.

## 2 Introduction

### 2.1 Animation

Even if someone has never played a digital game or seen a movie, used a smart phone or computer, they *have* experienced *game physics* or the broader field of *physically-based animation* (PBA). Before formally defining these terms, consider the following:

- Clicking-and-dragging a window.
- Watching a loading spinner/animation [1].
- Meditating with an animated screen saver.
- Digital accessibility with audio, video, and/or haptic (e.g., vibration) cues.

Each of the above actions or interactions all involve a real, physical aspect of motion. Consider the representation of a virtual object "moving" in which there is a change of pixels lighting up over time. Any motion on a screen (movie or otherwise) is essentially *animation*, which for the purpose of this paper mean representing the visual motion of an object through space over a period of time.

### 2.2 Physically-based animation

There are a multitude of books that discuss animation, artistry, people and creatures, lighting, and so forth. This paper is looking at a "slice" that applies to physically-based animation (PBA). The state of an object/system changes over time [2], and thus, PBA is animation created with physics in mind, as in providing a physical "feel." In a way, one could make the case that PBA is really just representing kinematics in which pixels activate over time to simulate motion. For more about the field of computer graphics and the graphics pipeline, the GPU, graphics APIs (e.g., Vulkan, DirectX, WebGL), and much more, see [3].

In PBA, the algorithms derive from actual physics (often engineering mechanics) *using physics to represent physics*. If that statement seems redundant, think about a loading spinner from [1]. The programming behind something like [4] may seem complicated to someone unfamiliar with JavaScript, but at the core, the code uses parametric equations of circular motion with constant angular velocity with algorithm such as follows:

```
Choose radius(r)
For t ← 0 : 2*PI
   Draw
        x ← r*cos(t)
        y ← r*sin(t)
```

One could theoretically make a spinner more "exciting" by adding angular acceleration, bouncing the spinner around, and much more. Introductory physics classes may even find student engagement by prototyping physical mock-ups of custom spinners adapted from [5].

PBA can be realistic, real, or utterly fake, but the goal is indeed realism in the service of entertainment. Consider another example of a game character about to jump off platform.

Depending on the experience required for a game, the character might follow an arc via the range equation, perform some sort of superpower move (e.g., double jump), or simply float in space. Maybe even the platform has an invisible hole to surprise the player.

To understand games/PBA, consider the notion of *media*, e.g., media film, TV, games, radio, etc. The field of *media studies* investigates the experiences provided to people by/on a variety of platforms upon which we see and engage with media. This paper tends to focus on games, especially because of the real-time demands for interaction, computation, and physical feel for the user/player. However, these concepts extend into media delivery, especially with films, special effects, and animation.

## 2.3 Simulations

*F=ma* may be accurate and incredibly useful, but it is not complete. Dropping an object in real life may dent the object, lose some substance, have heat transfer, and so forth, especially without delving into atomic or even subatomic considerations. Game physics and real physics (e.g., the introductory physics education) simplify or change reality to varying degrees.

Game physics typically has a reputation of falsehood in which fun sacrifices reality—sometimes drastically. Consider the example of a character jumping from a platform. Without giving the specifications, one can't really provide an exact answer, given a whirlwind of parameters: biomechanics, resistance, flexibility of the surfaces, clothing mechanics, and much more, all making the equations rather complex and "murky." Even real-life experiments would yield different results. Section 2.6 explores these concepts in more detail.

Note the repeated use of the term *real*. Throughout this paper, the term is problematic, as the equations and formulas taught to students depart simplify reality. Everything taught in physics simplifies reality as a *simulation*. More formally, a simulation approximates and abstracts a real situation to pose, solve, predict, estimate, analyze, and often, visualize—all to bring greater understanding to a complex object or process. In that lens, much of engineering education is about simulation and (video)game-like aspects. There is even a connection between games and simulations: the quest for accurate representation in a simulation, especially based off of research "wins" by representing reality when the simulation (closely) matches experimental results.

## 2.4 Games

Game researchers have spent years debating an exact definition of *game* despite the seeming simplicity. Using "fun" and "experience" can describe a multitude of things. Is a toy also a game? Trading stocks? The list goes on. For the purposes of this paper, [6] has an excellent summary, which I simplify into as: *a game is an experience in overcoming goals for seeking entertainment*. Relating to PBA also means considering a class of digital experiences and not "table games" (e.g., board games, card games).

In the context of animation, games are real-time, interactive visualizations in which a player does something to cause realistic reactions perceived by the player. However, despite the complexity of algorithms, code, and implementation, the entertainment value must prevail due to the media experience based on the “feel” for the player/viewer given the nature of the product. As such, PBA must often fake reality for the sake of experience, which is set by game design.

## 2.5 Game Design

Engineering design resembles other kinds of design—creating an “experience.” Not everything is a physical object, e.g., software engineering. Although we do not have “film designers” (see any long list of credits after a big-budget movie), the game industry has game designers. In either case, someone ultimately decides on the experience provided to a player.

The game designer is something akin to a cross between a movie’s director and writing staff. *Game design* is the process of determining the rules and experiences for the game. A rule might allow a character being able to do a wall-jump: jump onto a vertical wall, pause, and then jump higher–akin to an insect that can stick momentarily to a wall—a classic example of faking physics for fun. Some games have simple *mechanics* (actions that the player may do) while others are deeply complex (and note the connection to engineering mechanics). Making a game means making a world with its own laws, and interestingly, many game programmers are often called engineers and developers (as in software engineering/development), though not all implement the physics set by the game designers.

## 2.6 Game Physics

The academic and industrial worlds of PBA and game physics generally split into two realms: special effects (movies, tv, …) and games, respectively. *Game physics* [7] thus implies the implementation of physics concepts to provide the game experience intended by game design. Ideally, the reader would see and interact with all kinds of amazing pictures, except that intellectual property constraints of the entertainment industry potentially collides with Fair Use. Instead, the reader should search on YouTube for terms (quotation marks optional), like “SIGGRAPH game physics,” “SIGGRAPH fluid simulation,” and “SIGGRAPH FEM.” In fact, putting “SIGGRAPH” in front of anything on YouTube (clothing, robots, baking, …) will demonstrate rather remarkable videos.

Fake physics is introduced in Section 2.3, but how fake is *fake*? Imagine grading a homework assignment from an engineering class: a particular problem may have a solution, or if it’s a design problem, a range of solutions. When it comes to computer graphics and especially entertainment, no one is really calculating something that needs an exact answer from the perspective of engineering design (e.g., maximum material strength to handle a load).

For game physics, when creating an experience, designers and developers follow a kind of first-law/principle for entertainment computing:

*If it looks and/or feels right, it* **is** *right—regardless if it is theoretically right.*

That principle has some fascinating aesthetic, computational, and mathematical consequences:

- Some animations and interactions happen so quickly a person cannot spot the inaccuracies.

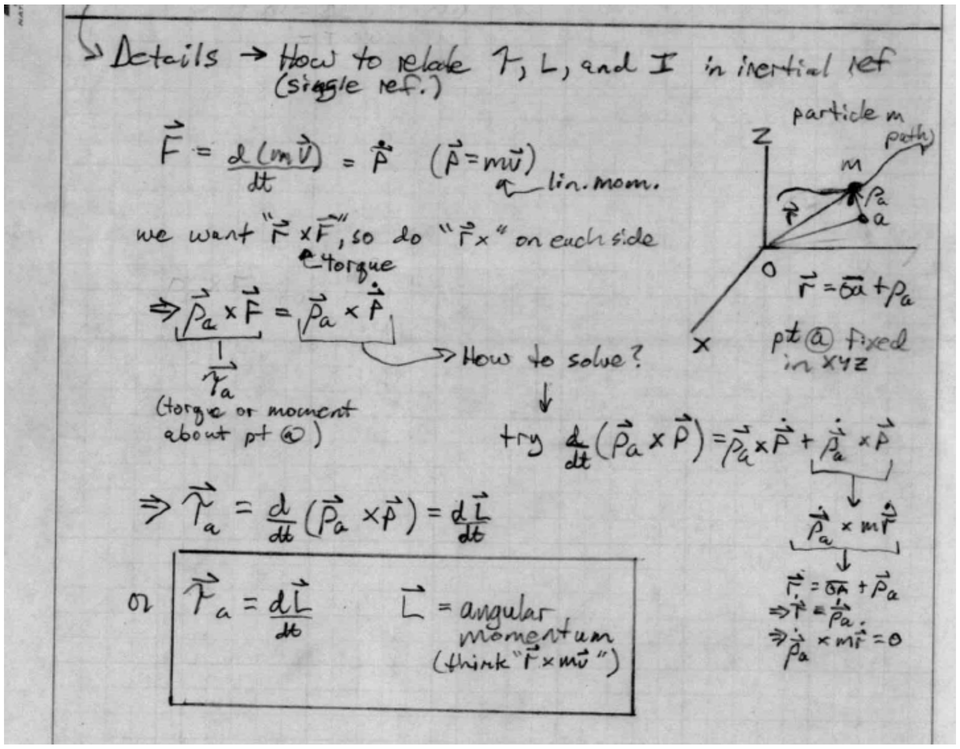
- Limited hardware platforms (e.g., mobile, game consoles) can only process so much.
- Drama/fun is more important than accuracy. Even in movies, there is no soundtrack in space, punches don't really sound "that way," and so forth.

Refer again to the suggested Internet searches at the top of this section. Some of the research has been so accurate that the difference between what one experiences and the underlying math and code is getting blurred. And in some cases, the fake thing *is* the real thing, at least in an ideal sense akin to the formulas used in introductory and advanced physics/engineering education.

## 2.7 Real physics inside fake physics

One may ask why game physics may be interesting from a teaching perspective if, at its core, fun overrides reality. But do another YouTube search and note how strikingly realistic modern animations seem. A quick search of "game physics" and looking at the various tutorials in References will also demonstrate the overlap of game physics and typical physics formulas.

I'm even still using engineering paper and leveraging my old engineering mechanics textbook, as shown below with scanned notes in Figure 1 [8].



Details → How to relate $\tau$, L, and I in inertial ref (single ref.)

$\vec{F} = \frac{d(m\vec{v})}{dt} = \dot{\vec{p}}$ $(\vec{p} = m\vec{v})$ ← lin. mom.

we want "$\vec{r} \times \vec{F}$", so do "$\vec{r}\times$" on each side
← torque

$\Rightarrow \underbrace{\vec{\rho}_a \times \vec{F}}_{\vec{\tau}_a} = \underbrace{\vec{\rho}_a \times \dot{\vec{p}}}$ → How to solve?

(torque or moment about pt ⓐ)

try $\frac{d}{dt}(\vec{\rho}_a \times \vec{p}) = \vec{\rho}_a \times \dot{\vec{p}} + \underbrace{\dot{\vec{\rho}}_a \times \vec{p}}$

$\underbrace{\dot{\vec{\rho}}_a \times m\dot{\vec{r}}}$

$\vec{r} = \vec{0a} + \vec{\rho}_a$

$\Rightarrow \dot{\vec{r}} = \dot{\vec{\rho}}_a$

$\Rightarrow \dot{\vec{\rho}}_a \times m\dot{\vec{r}} = 0$

$\Rightarrow \vec{\tau}_a = \frac{d}{dt}(\vec{\rho}_a \times \vec{p}) = \frac{d\vec{L}}{dt}$

or $\boxed{\vec{\tau}_a = \frac{d\vec{L}}{dt} \qquad \vec{L} = \text{angular momentum (think "}\vec{r}\times m\vec{v}\text{")}}$

Figure 1. Sample game physics course notes on angular momentum

Games physics and PBA references, e.g. [2, 9, 10], will show how critical angular momentum is when considering objects spinning around. It may seem odd, but students in computing majors are learning and applying rigid body motion along with many other topics typically seen in a sophomore engineering mechanics course. Interestingly, game "engineers" need to extend momentum with collision detection and response [11-13]. Introductory physics students might be surprised that after learning about linear momentum, using vectors and geometry adds a fun aspect of collision detection and response through "Pool Hall Lessons" [14].

## 3 Case studies in game physics and PBA

There are a multitude of examples of special effects in films and interactive game physics that help to answer the question posed in Section 1. Two examples provide brevity and connections to engineering, though there are many fantastic examples of past, present, and upcoming games that promote their use of game physics (e.g., recent reviews in [15]) as well as several modern films using cutting-edge PBA research.

### 3.1 Entertainment

Again, we have a collision of sorts when it comes to Fair Use and company IP, e.g., [16]. Moreover, "classic" academic papers do not lend themselves to demonstrating real-time interaction in games. Instead, the paper explains how the reader *can find* this work to experience animations and review some lectures.

I found myself very inspired to explore PBA and game physics by an early implementation of real-time finite element analysis (FEA) in *Star Wars: The Force Unleashed* ™. After waiting for the ads, look for the fractures and explosions in [17]. The material found in these references adapts real-time FEA for the purpose of entertainment.

One of the researchers [18] provides an overview, spending time teaching an audience of game developers about FEA (and I think I see the back of my own head off to the right of the screen) [19]. From [20]:

> *"Finite Element Analysis (FEA) is not new. In fact, it is a main stay of serious engineering, but until Pixelux the idea of it running in anything like the speeds required for gaming was just unheard of – the technique is literally used to model the stresses on nuclear power plants and these guys wanted to use it to bounce Imperial Stormtroopers off the digital walls of the LucasArts cafeteria."*

The continuing development on this particular physics engine essentially stopped a few years ago, but it has left something of a legacy in PBA and game physics [21].

Just as FEA helps with fracture mechanics, classic engineering mechanics has helped animators with Hank, the octopus in *Finding Dory*™ [22]. Interview very briefly mentioned the research into Hank that involved thinking an animator's mouse/pen as a static/constant force on an elastic surface [23]:

> *"…Our formulation is based on the elastic response to localized force distributions associated with common modeling primitives such as grab, scale, twist, and pinch. The resulting brush-like displacements correspond to the regularization of fundamental solutions of linear elasticity in infinite 2D and 3D media. These deformations thus provide the realism and plausibility of*

> *volumetric elasticity, and the interactivity of closed-form analytical solutions....”*

Essentially, Pixar engineers took classical elasticity, something already taught in engineering mechanics, and adapted the formulas into closed-form solutions with animators acting as the “forces.”

### 3.2 Physics education

When teaching introductory physics and engineering mechanics, there are many approaches and hybrid approaches using the following:

- real-life examples
- digital games [24]
- game physics

As with all educational methods, if we knew the perfect approach, we wouldn’t still be studying the matter. When it comes to physics education, it doesn’t appear that the literature reached a resolution. In fact, the introduction of game physics is completely unresolved.

Before the rise of game design and development education in the past two decades, prior academics pondered how videogames could influence physics education. For example, in a 1982 letter in *Physics Today*, researchers debated the effectiveness of games over “real life”/tangible experiments [25]. Studies such as [26] explore how virtual manipulation (e.g., games) engage physics students. [27] finds that a combination of virtual and physical experiments in different situations reinforce each other and suggests ongoing study.

There have been disconnected attempts at the pedagogy of games and physics/engineering, as in the study of design and players/students [28-30] and the exploration and study of games for physics education [31-37].

When it comes to making games, an early exploration had mixed results [38], but given the rapid improvement of game engines, easier/more adaptable engines have appeared [39]. The results indicated a mix of reactions from students from playing the games. However, a recent study involved educators building simulations via the Box2D game engine (if you’ve played/seen *Angry Birds*™, you’ve witnessed Box2D [39]) to provide interactive simulations for students. Although the students did not directly program the simulations, the study showed success. [40] shows similar results, and interestingly, they’re still using Flash, which demonstrates one problem with relying on APIs for developing software—the need for continual development.

## 4 Does game physics teach physics?

The above case studies demonstrate a multitude of examples of game physics, PBA, games for teaching physics, and even preliminary research in the effectiveness of leveraging game physics for teaching physics. There are a few more factors and questions to consider before addressing the central hypothesis of Section 1:

- Physics is often required in game programming jobs [41].
- The math required is non-trivial, and cursory coverage in other courses will not suffice if someone wants to pursue this career path.
- Mathematics helps to reinforce problem-solving skills.
- Games tend to be extremely good for physical actions, e.g., moving through 3-D space and object manipulation. A variety of physics applications can expand a game designer's vocabulary of game mechanics.
- Game physics creates a pathway to more traditional computer science courses and career paths, e.g., physically-based rendering, scientific visualization, animation.

As such, game programmers (and those associated with PBA for special effects) already use and/or learn game physics at least to some degree. These books, techniques, and media experiences are already used regardless of the input and reaction from the physics and engineering fields.

Can someone learn physics from game physics? The evidence presented thus far implies an answer of *yes*. However, there are several subsequent questions to address if game physics were the *only* way one learns physics:

- Handling a lack of incoming programming skills. Will all introductory students for physics have enough programming to warrant a physics-and-calculus-based physics class?
- Choosing a programming language and/or environments. Do a quick search for create a window in Vulkan and see the incredible complexity. Students would need a free visual, highly abstracted programming environment exist that can also run everywhere. And someone would need to keep the environment updated, building in everlasting sustainable software and support. Someone also needs to pay for all the licensing, and/or educators need to rely on constant upkeep through open source.
- Using a physics engine (e.g., Box2d) or build something custom. Should educators "recreate the wheel" instead of leveraging existing technology?
- Drawing the line (so to speak) on when to distinguish *real* from *fake*. It's possible engineering and game physics may need to keep separated due to the need for accuracy over fun.

Refer back to Section 3.1. Pixelux and the FEA physics engine that showed incredible promise…and then *died*. The issue of an evergreen, cheap/free, and little-overhead environment is non-trivial and itself a subject worthy of study.

The remainder of this paper explores some current work and develops a roadmap for future exploration if game physics were to become one of the modalities of teaching physics.

## 5 Physics to teach game physics

The following section summarizes the author's current work aimed to teach game physics to game programmers. The material is included to demonstrated that the flow of a game physics course is rather similar to a "real" physics course and provide further evidence that physics education *may* benefit from a game physics curriculum. Thus, a game physics course topic flow provides an early roadmap in addressing how to construct a game physics course to teach physics. Nevertheless, the material in this course relied on students to do their own programming at a senior or graduate level.

## 5.1 Source Material

Game physics books (e.g. [2, 7]) and tutorials (e.g. [42-47]) demonstrate that PBA and game physics are fields still in development. See also [42], which maintains an extremely helpful collection of research in PBA and other resources. An introductory book on engineering mechanics will actually present a rather comprehensive coverage for mathematical and physics theory. Other faculty have published their game physics courses, which should help the reader to see how the material is taught to game students [47, 48].

Standard game physics references use calculus, linear algebra, and other fundamental theory. Properly tacking fluids and soft-bodies (as PBA calls deformable bodies [2]) then involves continuum mechanics, which in turn uses partial differential equations, tensors, and “3-D math,” which is what the game development community tends to call their version of vectors and linear algebra. The math expectations usually exceed typical computer science requirements, which leaves instructors of game physics limited choices: showing formulas without derivation (a kind of “hand-waving”) teaching abstract concepts using physics APIs, e.g., [40, 50-53], reducing mathematical complexity, or perhaps raising the expectations to graduate level or beyond.

Given that my students already tend to use physics APIs in group projects in other courses, I have concentrated on using smaller examples to develop mastery of the concepts. With regard to physics APIs, I have a section of the course where students use an API, but I also warn the students—why use a sledgehammer when a ballpeen hammer will suffice?

## 5.2 Suggested Flow of Topics

Table 1 has a suggested flow of game physics topics. One course cannot suffice to cover the entirety—and enormity—of game physics/PBA. Even if students have had extensive math, they often need reminders for calculus and linear algebra, likely extending into explanations for differential equations and vector calculus. Part of the inspiration for connecting with engineering educators is that an engineering student will often have more math, physics, and engineering background than the typical computing student. A co-listed course might be worth considering for the future.

Some of these topics seem worthy of entire courses. For example, the author’s current course covers collision detection (and simple collision resolution) in other courses, and so, the course relies on that prior background. But, based on student reactions, they tend to ask for refreshers. The material on soft-body physics and fluid simulation is deeply mathematical. In the current course, fluids are required for a few graduate students who have co-registered.

RIT students and I have been working on an open-source library of examples [55].

## 6 Summary, conclusions, and future work

If the physics taught to students in STEM ultimately simplifies reality, and if the physics taught in entertainment computing uses the same equations/methods, then … have they converged? In a philosophical sense, **the real has always been simplified (is *fake* appropriate?), and the fake has gotten more real**. Thus, the researchers and educators would greatly benefit from more communication and collaboration.

Table 1. Suggested Flow of topics

| | |
|---|---|
| Overview | Explain the "entirety" of Game Physics (Table 2, below). |
| Kinematics | Calculate motion in the absence of forces (acceleration, velocity, and displacement); use Euler Integration. |
| Mass | Particles, points, and aggregates; center of mass to simplify aggregates; inverse mass for "infinite" objects; gravity, weight; buoyancy. |
| Forces | Vectorized form of *F=ma*; D'alembert's Principle; control volumes and free-body diagrams; statics; dynamics. |
| Linear Momentum | Conservation of linear momentum, impulse and momentum, coefficient of restitution; formulas for collisions between two objects. |
| Angular Momentum | Curvilinear motion; rotating reference frames; angular kinematics; conservation of angular momentum, torque; moment of inertia tensors. |
| Collision Detection Revisited | AABB, spheres, separating axis theorem; tunneling and penetration. |
| Integration Revisited | Taylor series expansions; velocity verlet, RK4. |
| Soft Bodies | Multivariable calculus and vector calculus review; energy methods; mass-spring systems; finite differences; finite element method. |
| Fluids | Navier-Stokes; Smoothed Particle Hydrodynamics. |

Imagine if typical introductory physics courses across schools were completely inconsistent from an ABET standpoint. The course presented in Section 5 isn't a real physics course, but it could be with additional work. Game physics provides an in-depth experience for students to reinforce their math, physics, and graphics programming skills. This paper is ultimately a call to action to inspire other educators curious about leveraging the appeal of game physics and PBA.

Preliminary evidence presented in the paper suggests that physics can be taught with game physics, but with two different groups of researchers (physics/engineering and games/computing) tackling the approaches for different goals, much remains to be studied:

- Assessment of game physics in teaching physics to game programmers. The author's current course and related work are part of games and graphics curricula that lack the formal assessment seen in ABET and educational research studies.
- Investigation of programming environments for non-programmers. By requiring programming, a physics course would further restrict a population taking a course and possibly create cognitive overload.
- Developing a coalition for collaboration across STEM domains, including entertainment technology. Part of this paper's goal is drawing awareness to game physics and PBA, as they appear to be relatively unknown.

It is possible that teaching physics from a game physics perspective to "everyone" may involve too many computational, funding, and cognitive hurdles. However, the work may still yield new software and approaches for the traditional physics community, and likewise, the game physics/PBA instructors may greatly benefit from input of that community.

## 7 Acknowledgements

I gratefully appreciate RIT FEAD grant and college funding for supporting this work, and my many students, co-ops, and teaching assistants for helping to make this work happen. Readers can access the entire repository of work [55]. This reviewers' comments on this paper were extremely helpful, especially to clarify my call-to-action (a term which I used from the reviews) to connect researchers from different academic communities with this work.

## References

[1] PlotDB Ltd. "Pure CSS Loaders." https://loading.io/css (accessed March 1, 2021).

[2] D. H. House and J. C. Keyser, *Foundations of Physically Based Modeling & Animation*. CRC Press, 2017

[3] S. Marschner and P. Shirley, *Fundamentals of Computer Graphics: 4th Edition*. A K Peters/CRC Press, 2015.

[4] F. Gnass and T. Brown. "spin.js." http://spin.js.org/spin.js (accessed March 1, 2021).

[5] University of Colorado Boulder Engineering. "Teach Engineering: STEM Curriculum for K-12." https://www.teachengineering.org/activities/view/cub_mechanics_lesson08_activity1 (accessed March 1, 2021).

[6] K. S. Tekinbas and E. Zimmerman, *Rules of Play: Game Design Fundamentals*. The MIT Press, 2003.

[7] D. H. Eberly, *Game Physics, 2nd ed*. CRC Press, 2010.

[8] I. H. Shames, Engineering Mechanics: Statics and Dynamics, 3rd Ed, Prentice-Hall, 1980.

[9] F. Dunn, and I. Parberry, *3D Math Primer for Graphics and Game Development, 2nd ed*. A K Peters/CRC Press, 2011.

[10] G. Szauer, *Game Physics Cookbook*. Packt Publishing, 2017.

[11] T. Schwarzl, *2D Game Collision Detection: An introduction to clashing geometry in games*. CreateSpace Independent Publishing Platform, 2012.

[12] C. Ericson, *Real-Time Collision Detection*. CRC Press, 2004.

[13] G. v. d. Bergen, *Collision Detection in Interactive 3D Environments*. CRC Press, 2003.

[14] J. v. d. Heuvel and M. Jackson. "Pool Hall Lessons: Fast, Accurate Collision Detection Between Circles or Spheres." https://www.gamasutra.com/view/feature/131424/pool_hall_lessons_fast_accurate_.php (accessed March 1, 2021).

[15] E. Schoon. "These 7 Physics Simulator Games Are the Perfect Stress Relief." https://www.reviewgeek.com/41751/these-7-physics-simulator-games-are-the-perfect-stress-relief (accessed March 1, 2021).

[16] The Walt Disney Company. https://disneytermsofuse.com (accessed March 1, 2021).

[17] Fandom. "Star Wars: The Force Unleashed (video game)." https://starwars.fandom.com/wiki/Star_Wars:_The_Force_Unleashed_(video_game) (accessed March 1, 2021).

[18] E. G. Parker and J. F. O'Brien, "Real-time deformation and fracture in a game environment," in *SCA '09: Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, August 2009. Pp. 165–175. https://doi-org.ezproxy.rit.edu/10.1145/1599470.1599492.

[19] E. G. Parker and J. F. O'Brien. "Real-Time Deformation and Fracture." https://www.gdcvault.com/play/1279/Real-Time-Deformation-and-Fracture (accessed March 1, 2021).

[20] M. Seymour. "DMM: FEA for VFX." https://www.fxguide.com/fxfeatured/dmm-fea-for-vfx/ (accessed March 1, 2021).

[21] C. Welsh. "A Matter of Illusion: Interview with James O'Brien." https://www.siggraph.org/a-matter-of-illusion-interview-with-james-obrien (accessed March 1, 2021).

[22] C|Net News. "How Pixar created its most complex character yet for 'Finding Dory.'" https://digg.com/video/hank-squid-pixar-finding-dory-how (accessed March 1, 2021).

[23] F. De Goes and D. L. James, "Regularized kelvinlets: sculpting brushes based on fundamental solutions of elasticity," in *ACM Transactions on Graphics*, 36, 4, Article 40 (July 2017), pp. 40:1-40:11. https://dl.acm.org/doi/10.1145/3072959.3073595.

[24] C. McFadden. "7 Video Games That are Perfect For Teaching Physics." https://interestingengineering.com/7-video-games-that-are-perfect-for-teaching-physics (accessed March 1, 2021).

[25] T. L. Clarke, J. S. Miller, and A. Bork, "Video-game physics?," *Physics Today* 35(12), 15 (1982); https://physicstoday.scitation.org/doi/abs/10.1063/1.2914876., https://physicstoday.scitation.org/toc/pto/35/12?size=all.

[26] Z. C. Zacharia and G. Olympiou, "Physical versus virtual manipulative experimentation in physics learning" in Learning Instruction, 21(3), 2011, pp. 317-331. https://doi.org/10.1016/j.learninstruc.2010.03.001

[27] J.-P. Atanas, "Is Virtual-Physical or Physical-Virtual Manipulatives in Physics Irrelevant within Studio Physics Environment?," in *Athens Journal of Education*, 5(1), 2018, pp. 29-42. https://www.athensjournals.gr/education/2018-5-1-2-Atanas.pdf.

[28] A. J. Stapleton, "Research as Design-Design as Research," in *Proceedings of the 2005 DiGRA International Conference: Changing Views: Worlds in Play*, http://www.digra.org/wp-content/uploads/digital-library/06278.40383.pdf.

[29] S. Slater, A. Bowers, S. Kai, and V. Shute, "A Typology of Players in the Game Physics Playground," in *DiGRA '17 – Proceedings of the 2017 DiGRA International Conference*, 1(14). http://www.digra.org/digital-library/publications/a-typology-of-players-in-the-game-physics-playground.

[30] V. J. Shute and S. Rahimi, "Stealth assessment of creativity in a physics video game," in Computers in Human Behavior, 116, 2020, pp. 1-3. DOI: 10.1016/j.chb.2020.106647.

[31] Hookway, G., Mehdi, Q., Hartley, T., Bassey, N, Learning Physics Through Computer Games. In *Proceedings of The 18th International Conference on Computer Games* (CGAMES2013).

[32] M. Ali, *et al*., "Web Pro-Mc Physics as a support tool for improving physics problem solving skills," in *2015 Game Physics and Mechanics International Conference (GAMEPEC)*, Langkawi, Malaysia, 2015 pp. 26-30. doi: 10.1109/GAMEPEC.2015.7331851 url: https://doi.ieeecomputersociety.org/10.1109/GAMEPEC.2015.7331851

[33] T. A. Philpot, N. Hubing, R. H. Hall, R. E. Flori, D. B. Oglesby, and V. Yellamraju, "Games as Teaching Tools in Engineering Mechanics Courses," in *Proceedings of the 2003 American Society for Engineering Education Annual Conference & Exposition*, 2003, https://peer.asee.org/games-as-teaching-tools-in-engineering-mechanics-courses.pdf.

[34] G. Kortemeyer, "Game development for teaching physics," in *International Conference on Physics Education (ICPE)*, 1512*2020), 2018, doi:10.1088/1742-6596/1512/1/012024.

[35] D. Croxton and G. Kortemeyer, "Informal physics learning from video games: a case study using gameplay videos," in Physics Education, 53(1), 2018, https://iopscience.iop.org/article/10.1088/1361-6552/aa8eb0.

[36] Y. Liao and S. Liu, "MechGames: Teaching and Learning Dynamics through Computer Simluations and Games," in *2020 ASEE North Central Section conference*, 2020, https://strategy.asee.org/mechgames-teaching-and-learning-dynamics-through-computer-simulations-and-games.

[37] L. Buffington and D. Rosengrant, "Making Differentiation Magic in the Classroom with Minecraft," in The Physics Teacher, 58, 564, 2020. https://aapt.scitation.org/doi/abs/10.1119/10.0002378

[38] C. B. Price, "The usability of a commercial game physics engine to develop physics educational materials: An investigation," in Simulation & Gaming, 39(3), 2008, pp. 319-337.

[39] Box2D, https://box2d.org (accessed March 1, 2021).

[40] H. Zeng, S.-N. Zhou, G.-R. Hong, Q.-y. Li, and S.Q. Xu, "Evaluation of Interactive Game-Based Learning in Physics Domain," in *Journal of Baltic Science Education*, 19(32), 2020, pp. 484-498. doi:10.33225/jbse/20.19.484.

[41] indeed. "Game Physics Programmer." https://www.indeed.com/q-Game-Physics-Programmer-jobs.html (accessed March 1, 2021).

[42] C. Batty. "Physics-Based Animation Resources & Courses." http://www.physicsbasedanimation.com/resources-courses. (retrieved March 1, 2021).

[43] E. Neumann. "Physics Simulations." https://www.myphysicslab.com (accessed March 1, 2021).

[44] M. Müller, J. Stam, D. James, and D., N. Thurey. "Real Time Physics Class Notes." https://cgl.ethz.ch/Downloads/Publications/Tutorials/2008/Mue08/coursenotes.pdf (accessed March 1, 2021).

[45] M. Baker. "Euclidean Space." http://www.euclideanspace.com (accessed March 1, 2021).

[46] A. Bargteil and T. Shinar, SIGGRAPH University 2019 Course - An Introduction to Physics-Based Animation. "An Introduction to Physics-Based Animation." https://www.youtube.com/watch?v=b_WJ-HwalwU (accessed March 1, 2021).

[47] L. Kavan. CS6660, Spring 2017 Physics-based Animation." https://www.youtube.com/watch?v=sSKyQIxdhdA&list=PL_a9tY9IhJuPc7e6r-3DMw_PbYbloKoWM (accessed March 1, 2021).

[48] A. Vaxman. "Game Physics." http://www.cs.uu.nl/docs/vakken/mgp/2018-2019 (accessed March 1, 2021).

[49] D. James. "CS 348C: Computer Graphics: Animation and Simulation." http://graphics.stanford.edu/courses/cs348c/ (accessed March 1, 2021).

[50] D. H. Eberly. "Geometric Tools." https://www.geometrictools.com (accessed March 1, 2021).

[51] PyBullet/Bullet Physics, https://pybullet.org/wordpress (accessed March 1, 2021).

[52] PhysX https://developer.nvidia.com/gameworks-physx-overview (accessed March 1, 2021).

[53] Havok Physics https://www.havok.com (accessed March 1, 2021).

[54] S. Niebe and K. Erleben, *Numerical Methods for Linear Complementarity Problems in Physics-Based Animation*, Morgan & Claypool, 2015.

[55] D. I. Schwartz. "ATLAS." http://bit.ly/programgames (accessed March 29, 2021).