Exploring the Design Space of Social Physics Engines in Games

Shi Johnson-Bey[1], Mark J. Nelson[2], and Michael Mateas[1]

[1] University of California Santa Cruz, Santa Cruz CA 95060, USA
[2] American University, Washington D.C. 20016, USA

Abstract. Social simulation in video games approximates believable social behavior between characters. Game franchises like *Crusader Kings*, *The Sims*, and *Dwarf Fortress* became famous for using social simulation for emergent storytelling. Despite the success of using social simulation as a core aspect of gameplay, there is a seeming lack of publicly available tools for helping game developers create these types of experiences. To help encourage the development of open-source social simulation tools, we further explore the concept of social physics engines — self-contained solutions for modeling dynamic social relationships between non-player characters and players. We propose a design space for constructing social physics systems. It is inspired by rigid-body physics engines and is informed by a design space analysis using commercial and academic social simulation games and systems.

Keywords: social physics · social simulation · emergent narrative.

1 Introduction

Character-based social simulation (CBSS) games model dynamic social states comprised of multiple non-player characters (NPCs), players, and their relationships with one another. The simulation's job is to approximate believable social behavior through character interactions that evolve the social state. Social simulations situate characters in social contexts of inter-character and player-character relationships that may include power imbalances, familial ties, romances, friendships, and rivalries. Social simulation games provide players with a playable model of social interaction that affords local and global autonomy [16], as players may plan and observe the immediate and long-term effects of various social interactions.

Social simulation games create opportunities for emergent narratives by relying on the unscripted actions of individual characters to drive the story instead of scripted content. Game franchises like *Crusader Kings*, *The Sims*, and *Dwarf Fortress* became famous for using the emergent nature of CBSS for emergent storytelling. These games have active communities of players who eagerly recount their gameplay experiences online[3].[4][5]. Gameplay *retellings* are one of the ways

[3] *Sims 4* subreddit: `https://www.reddit.com/r/Sims4/`
[4] Crusader Kings subreddit: `https://www.reddit.com/r/CrusaderKings/`
[5] *Dwarf Fortress* subreddit: `https://www.reddit.com/r/dwarffortress/`

players engage in the narrative generation process [10]. Also, Their retellings are a valuable tool for evaluating the game's AI. They capture more subjective aspects of the player experience that pure playtrace evaluation methods might not capture [14].

Despite CBSS' success in producing interesting and emergent gameplay, there are few publicly available tools for integrating social simulation systems into games. For instance, within Unity's Asset Store of more than 60,000 packages, only four (ranging in price from $25 to $175 per license) offer game makers capabilities to do any social modeling or create socially intelligent characters. This observation is interesting, given that many social simulation systems have been proposed in the academic literature to help game developers. This disconnect could be addressed by developing better nomenclature for the various aspects of social simulation systems and designing custom systems that target popular platforms.

This paper discusses the design space of *social physics engines*, tools for developing social simulation games. The term "social physics" was first proposed by [19] when discussing their work on *Comme il Faut (CiF)* the social simulation engine that powered the game *Prom Week* [18]. In *Prom Week*, the player must choose a series of social interactions to achieve a given character's social goals (become friends with a particular character, be mean to 3 people, start dating someone). The authors likened their social puzzles to physics-based puzzles in games like *Angry Birds*[25]. There is no single method for solving these puzzles; players can use their knowledge of the game rules to devise new solutions.

The purpose of performing a design space analysis is to identify design questions, provide options that address those questions, and provide criteria for assessing and comparing design options [15]. One of our aims here is to more fully develop this framing of interactive social simulation systems as *social physics engines*, drawing an analogy to the traditional physics engines found in many game engines. This analogy is a deliberately opinionated take on the more general concept of computational social simulation, which we explain in more detail in Section 3. Here we describe our design space and use it to make clear the significant design considerations when designing social physics for games. Our design space should allow us to better reason about the designs of social simulation systems and discuss their affordances for various end-user experiences and storytelling.

2 Related work

CBSS has a foothold in both digital entertainment and computational social science. We build on previous efforts to understand social simulation and make it a more accessible tool to game makers and storytellers. This effort is divided between system design, taxonomy building, and tool evaluation. Simulating autonomous characters for interactive and emergent storytelling has been a topic of interest since the late 1990s when Aylett described the challenges of using autonomous agents (characters) for generating narratives [2]. Since then, there

has been much work on developing systems for socially intelligent characters in games. We cover a tiny subset of these systems as part of our design-space analysis.

Social physics engines as a concept began with *CiF* [19] and *Prom Week*'s [18] social physics gameplay. Players need to understand the game's social context (how characters feel, their current/past relationships, and their goals) to find strategies for actions to take. *CiF* was succeeded by the *Ensemble Engine* [31], an attempt to take the lessons learned from working with *CiF* and repackage its rule-based AI architecture as a general-purpose tool for social simulation. It was released as a desktop application that helped users create social worlds by defining the current cast of characters, actions, relationship values, and social rules. We discuss other academic social simulation tools throughout the rest of this paper. Please see Table 1 for a list of the systems.

[4] provides a taxonomy of social simulation in games and a shared vocabulary for comparative analysis between different social simulation systems. They are an ideal reference for thinking about design options when designing a social simulation system. Their taxonomy surfaced the following themes: *communication*, *flow of knowledge*, *relationships* and *emotions*. Our work complements theirs as we add additional layers to enable people to compare social simulation systems. From a user perspective, many social simulation games can feel like they are doing the same thing, but each approach expresses concepts differently from a developer's perspective.

Finally, recent efforts have looked into challenges encountered while teaching people outside of computational media how to leverage social physics to author custom experiences [8]. As we will discuss, deriving the math and processes that define the social dynamics of a game world is not a trivial task. Much more work must be done to properly teach people the procedural literacy needed to successfully develop experiences that leverage social physics. Evaluating the usability of a given social physics system from the perspective of a particular target audience of designers may raise additional interesting system-design questions.

3 Drawing Inspiration from Physics Engines

We draw inspiration for our social physics engine design space from rigid-body physics engines used in games. We found them helpful when developing design questions for social physics engines because they perform a similar role. Physics engines approximate believable physical interactions between objects in virtual space. They are an almost essential part of all games today and come prepackaged with most game engines. Physics engines alleviate the growing complexity of creating believable physics by supplying reusable libraries that handle common physics concepts such as forces, movement, collisions, and constraints like joints and springs. Physics engines are not concerned with graphical presentation. They only track where objects are and how they interact with other objects.

Popular examples of physics engines are *Bullet*[6], *Box2D*[7], *PhysX*[8], and *Havok*[9]. These physics engines pull from a shared ontology of fundamental physics concepts like velocity, acceleration, force, mass, and friction. They then apply these concepts to 2D or 3D domains and allow users to model *rigid-body dynamics*[10], *soft-body dynamics* [11], and *fluid-dynamics* [12].

Rigid-body physics engines represent the world using three core software abstractions, *bodies*, *constraints*, *forces*, and *dynamics functions* that proceduralize physics mechanics of forces and motion. *Bodies* are the objects that exist in the world. They can have various attributes like position, mass, velocity, acceleration, gravity effects, and collision bounds. *Constraints* represent interconnections between bodies, such as fixed distances, joints, and spring connections. *Forces* like gravity act on bodies to produce movement. Finally, dynamics equations govern how bodies, constraints, and forces interact. With this toolbox, users can represent many physical scenarios. For example, a scene with two cars colliding can be modeled as two rigid bodies moving on intersecting paths with masses and collision bounds. Users can also simulate soft objects, like cloth or plants, using a mesh of bodies interconnected with constraints. Lastly, they can simulate fluid movement, such as flowing water, by treating it as a large collection of tiny bodies. One does not need to be an expert in physics to take advantage of physical behavior. Ideally, all one needs is a general understanding of the underlying concepts.

An interesting aspect of video game physics is its focus on believability over realism. Most games do not have the computational resources to perform high-precision physics simulations involving many parameters. Their goal is to produce plausible physical behavior given the player's knowledge of the game world. Sometimes this involves relaxing the rules of physics to achieve desired gameplay. For instance, it is not realistic for a 5-foot tall, 300-lb plumber to have a vertical jump that is twice their height, but this makes sense in the context of the *Super Mario* series[13]. Game physics is a computational caricature [34] as it exaggerates the most salient components of real-life physics. The rocket-boosted soccer-playing cars of *Rocket League*[23] are not running *NASA*-level rocket simulations, nor are they running hyper-realistic driving simulations like those found in *Forza Horizon 5*[36] or *Gran Turismo 7*[9]. It is running a bespoke kart physics model that relaxes the constraints of realism to support an intuitive yet surreal play experience[6].

[6] `https://pybullet.org/wordpress/`
[7] `https://box2d.org/`
[8] `https://developer.nvidia.com/physx-sdk`
[9] `https://www.havok.com/`
[10] The simulation of bodies under the influence of external forces with the assumption that all bodies are solid/non-deformable
[11] The simulation of realistic deformable objects like cloth, hair, and plants
[12] The realistic simulation of fluids like water and smoke
[13] Mario stats taken from fan wiki: `https://characterprofile.fandom.com/wiki/Mario`

What we want to borrow from physics engines is the clean abstraction of physics concepts into bodies, relationships between those bodies, and dynamics that define how bodies interact. Using a familiar architecture to design and describe social physics engines should help people better conceptualize their role in game development. In the following section, we describe our design layers based on concepts we borrow from physics engines.

4 Sampling Social Simulation Projects

We selected a subset of commercial social simulation games, experimental games, and academic systems to form our design space. We found academic projects by searching *Google Scholar* and various conference venues, and commercial games were chosen based on the complexity of their social mechanics. We chose to focus on games that require social reasoning and feature non-linear, systemic behavior resulting from autonomous characters acting independently of the player. Games like *Animal Crossing: New Horizon* [21], *Stardew Valley* [5], and dating sims were excluded because our understanding is that their relationship models are linear progression systems driven by players presenting gifts to NPCs. This interaction style does not require any social reasoning beyond determining the best gift to give each character. We also excluded commercial games with very little information online to provide insight into their social systems and character AI. We excluded *Crusader Kings III* from direct analysis due to there not being enough information online specifically describing *CKIII's* systems. Instead, we use system descriptions for *Crusader Kings II*, which seems to share much of the same underlying social systems.

We then gained information on underlying social systems using hands-on experience, research publications, *Game Developers Conference* presentations, game architecture deep-dive videos, crowd-sources wikis, and games journalism articles. Some academic projects do not have fully-fledged game experiences and were included for their potential to create interesting gameplay.

We chose the following projects/games as the focus of this design space analysis: *Talk of the Town/Bad News* [27, 29, 32], *Versu/Blood and Laurels* [12, 11], *PsychSim* [24], *Comme il Faut/Prom Week* [19, 18, 30], *City of Gangsters* [35, 40, 26], *The Sims 4* [17], *Lyra* [3], *Crusader Kings II* [22, 13, 38], *Shadow of War*[20], *Kismet* [37], *Dwarf Fortress* [1],and *Gossip* [7]. A short description of each project is given in Table 1.

5 The Design Space

A design space analysis (DSA) places an artifact in the space of possibilities and seeks to explain it was particularly chosen [15]. We borrow the QOC methodology from [15], which has the following parts:

- **Questions**: Identifying key design questions
- **Options**: Providing options to answer design questions

– **Criteria**: Methods to evaluate various options

We apply this DSA method to the various layers for social physics engines (bodies, constraints, dynamics, forces, and collisions) that we adapted from rigid-body physics engines: *character and relationships*, *social dynamics*, and *social interaction* (see Fig 1). For each layer, we ask, "how would this be represented?" then, we supply design options based on our selected sample of social simulation games and systems. We do not apply any criteria for evaluating which options are better than others. These criteria are best left to the authors of the final experience and what their social physics engine implementation needs to support. We recommend that designers choose the simplest approach that achieves their goals.

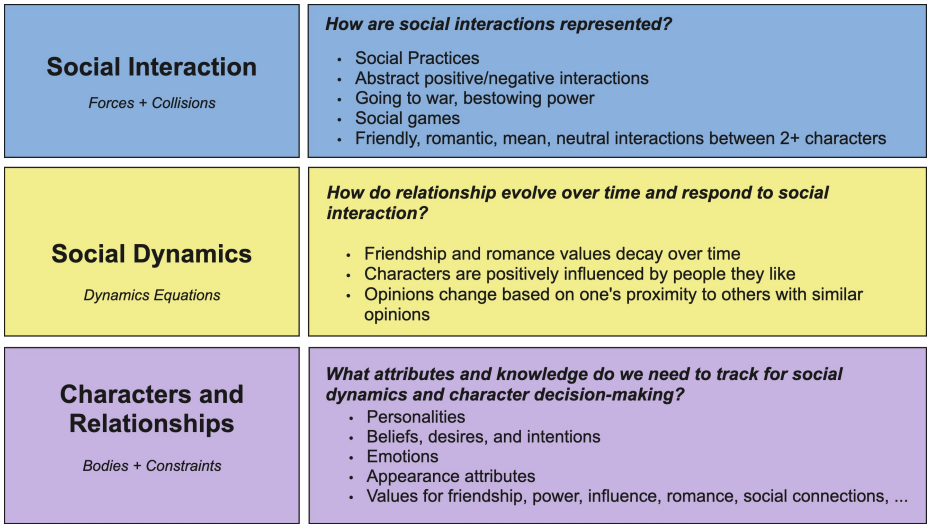

Fig. 1. Our layered design space for social physics engines is based on traditional physics engines in games.

Our design space is divided into three layers. The first layer is characters and relationships. It is concerned with managing what information characters need to know for action selection. The social dynamics layer is concerned with the mathematics and processes operationalizing how character relationships evolve. It is based on the concept of physics dynamics, the study of the movement of physical bodies over time. Finally, the social interaction layer encompasses character decision-making, action definitions, and how actions use social dynamics to evolve the current social state.

There is no particular order to follow when designing a social physics engine. It is OK to start with a guiding social concept like, "scratch my back, and I will

scratch yours," then proceed to determine the exact attributes, dynamics, and interactions needed to realize it.

6 Characters and Relationships

All social simulation systems contain characters and relationships. They are essential to modeling social structures and are analogous to bodies and constraints in rigid-body physics. Here we explain what kinds of data are modeled within characters and relationships and provide examples from our selection of projects.

6.1 Characters

Characters can represent individuals (players and non-players), groups, institutions, and concepts. They are similar to the physical bodies modeled in physics engines. Characters have self-contained collections of attributes that represent knowledge characters have about themselves and other characters. Attributes can be simple values like name, age, money, gender, and ethnicity. Character models may also contain more complex structures like personality, emotion, belief, and goal models. NPCs leverage their internal models to decide what social actions to take and how to respond to actions initiated by other characters. We discuss mechanisms for NPC decision-making within the *Social Interaction* section.

Traits and Statuses Two common abstractions for more complex attribute representations are *tags* and *statuses*. Tags are generally used like static/long-term runtime type information and can modify other aspects of the character's state. For instance, in *Prom Week*, traits represent a character's permanent personality traits like bravery or intelligence that affect the social exchanges in which a character engages. *Crusader Kings II* also uses an extensive set of traits to calculate how characters feel towards one another in the opinion system (covered in the relationship section). Their traits may also evolve into other traits over time. For example, a character with a *playful* trait could develop *deceitful*, *gregarious*, or *lunatic* traits later in life. Statuses serve the same purpose but are temporary modifications to the character's state such as *moodlets* [14] in *The Sims 4* and emotions in *Prom Week* [18].

Personalities There is no set way to represent a character's personality. Personality is meant to be one of the drivers of character behavior, so it is tightly coupled with the action-selection strategy. *Dwarf Fortress*, models personalities as a collection of character goals (fall in love, rule the world, master a skill, ...), beliefs (power, loyalty, harmony, ...), and relationship facets (greed, anxiety,

[14] `https://sims.fandom.com/wiki/List\_of\_Moodlets\_(The\_Sims\_4)/Life\_Status`

humor, ...) [15]. *Talk of the Town* relies on the Big 5 Personality model (openness, conscientiousness, extraversion, agreeableness, and neuroticism) for character decision-making and social dynamics. Relationships are then calculated as a function of the compatibility between two characters' personality traits. Finally, *Versu* represents character personality as world states that characters wish to make true. Not only do these describe who the characters are, but they also serve as goal states for ranking the utility of potential actions.

Theory of Mind Theory of mind models track information about what characters think other characters believe. *PsychSim* uses this to enable characters to make decisions based on recursive beliefs of how the other characters will change their beliefs of the first character.

Shared Knowledge Shared knowledge encompasses all data that is not explicitly part of any single character's model. These are global resources that any character can update and leverage for decision-making. Shared knowledge is where a designer may choose to encode generic social norms. For example, *Prom Week* used a shared global knowledge base that encoded rules for "cultural knowledge" on what is considered cool or lame. A typical example is homework and cafeteria food being considered lame, while skateboards and cell phones are deemed cool [19].

6.2 Relationships

Relationships track objective facts, subjective feelings, and shared understanding of characters' various social connections. Azad and Martens give an extensive explanation of relationships in social simulations [4]. Here we give a brief overview and suggestions for abstractions to use when representing character relationships.

Relationships are the other core component of social simulation, after characters. They are analogous to constraints in physics engines because they connect one character to another. Relationships are best represented as a graph data structure where the characters are the nodes and relationships are the edges that connect characters. Relationship graphs may be undirected or directed. Undirected relationships have information that is mutual between characters. For example, if two characters work for the same company, they both consider the other a coworker and thus can share a single connection in the relationship graph. However, sometimes characters may have differing opinions of each other. Undirected graphs are suitable for representing power imbalances or cases where perhaps one character feels a strong romantic attraction to another, but their love is unrequited.

Like attributes on characters, relationships can have an arbitrary amount of complexity depending on the associated data. Modeling a character family tree

[15] `https://dwarffortresswiki.org/index.php/DF2014:Personality\_trait`

like in *Talk of the Town*, *The Sims 4*, and *Dwarf Fortress* can be accomplished by creating a directed graph where each relationship has a tag indicating the type of connection (parent, sibling, uncle, grandparent, ancestor, in-law). Relationships can also hold valence values representing abstract feelings like friendship, romance, opinion, and salience. This convention is most common among the projects we selected. *Talk of the Town* and *The Sims 4* represent relationships as a combination of two scalar values that track friendship and romantic affinity.

Moreover, *Crusader Kings II* was praised for its Opinion System [38] that leverages character traits and past social exchanges between characters to calculate a single value that represents how much a character respects another. The opinion system is at the core of how characters make decisions regarding the player. Characters with a higher opinion of the player are less likely to betray them and align with another kingdom. Opinions are not mutual values, so one character can respect another that does not respect them.

Event histories Designers may choose to utilize recorded events that have transpired between characters. *City of Gangsters* uses event histories directly in their relationship model. NPC attitudes are a sum of the valence buffs for each event associated with a relationship. NPCs factor in their interactions with the player and interactions the player has had with other NPCs into their relationship's score. So, if an NPC learns that the player took a transgressive action against one of the NPCs family members, they will be less likely to trade goods with them or offer favors.

In *Shadow of War*, Orcs can recall specific events such as the player retreating from a battle, killing the player, or the player humiliating the orc following a defeat. They use these events for cueing voice clips to get an emotional rise out of the player. We are unsure if these events factor directly into the *Nemesis System*.

7 Social dynamics

Social dynamics are the math and processes that operationalize changes to the social state. The critical design question for this section is, "how do characters calculate how they feel about each other?". Developing the social dynamics for a system requires designers to express abstract sociological theories in code. This concept is the most challenging part to explain because most games do not publish their exact algorithms for character relationships.

We start with an example from *Gossip*[7], by Chris Crawford (see Figure 2). In it, characters (including the player) call one another and display facial expressions to communicate their opinion of someone else. Characters then change their internal opinions based on whom they are talking to and whom they are talking about. They are influenced positively by people they like and negatively by people they do not like. By the end of the game, the player's goal is to have the most characters like them. Figure 3 shows two equations for how character attitudes evolve based on gossip that a character hears from another. These

Fig. 2. Screenshots of *Gossip*. (Left) The main game screen showing two characters talking on the phone. Dan is telling the player that Liz likes them. (Right) The pause menu that displays the player's knowledge of how characters feel about each other and the player. Question marks indicate unknown knowledge and are replaces with the appropriate facial expression when players interact with characters.

$$\Delta x_{l,s} = \frac{x_{l,o} x\prime_{s,o}}{k_1} \tag{1}$$

$$\Delta x_{l,o} = \frac{x_{l,s} x\prime_{s,o}}{k_2} \tag{2}$$

Fig. 3. Equations that govern the change in opinion of a character when listening to another character's opinion in *Gossip*[7]. $x_{a,b}$ is a's actual opinion of b. $x\prime_{a,b}$ is a's declared opinion of b. l is the listener, s is the speaker, and o is the object (character) being gossiped about.

equations define the social dynamics of the world. Relationships (character attitudes toward each other) are a function of how much they like the person speaking, the opinion given by the speaker, and their opinion of the subject of conversation.

Calculating character relationships can also be as simple as summing the effects of various relationship factors. *City of Gangsters* and Crusader Kings II NPCs calculate their opinions as a linear combination of relationship modifiers (traits, events, succession law, and their relative positions of power). This approach keeps the math simple and instead relies on the data structure of the modifiers to provide expressivity and non-linearity to relationship development.

Lyra [3] calculates characters' opinions on topics of discussion based on their internal bias, private/public opinions, and uncertainty on the topic. Characters are then influenced by their proximity to the clusters of the opinions of all simulated characters.

Talk of the Town calculates character relationships over time as a function of the personality compatibility between characters, elapsed time since their last interaction, and each character's personality. This feature results in asymmetric

relationships that evolve organically and reflect differences in character personalities [28].

8 Social interaction

The last major component in the social simulation toolbox is the model of social interaction. As it implies, social interactions are how NPCs and players interact. Social interaction is usually defined by actions, each with specified preconditions and effects. Executing actions should change the current social state in accordance with the social dynamics (discussed in the previous section). The chosen social interaction model can be inspired by sociological theory or a designer's vision of players' core interaction loop with NPCs.

8.1 Models from sociology & social psychology

Versu uses the concept of social practices derived from Schankian Scripts [33] to define what actions are available to characters in a given situation and how other characters should react to those actions. Social practices are triggered automatically when certain preconditions are met, and players and NPCs are free to choose actions from all the actively engaged practices. *Prom Week* does something similar but uses the concept of social games, multi-character social interactions intended to modify the social state. Social games are based on Goffman's dramaturgical analysis and encode patterns of behavior based on how characters present themselves and how they want others to perceive them[18]. In the case of *Versu*, social practices were used to encode the rules of etiquette for historical Rome for the game *Blood and Laurels*. In comparison, *Prom Week* encodes all the many social conventions one would see in an American high school.

8.2 Designer-constructed models of desired experience

Commercial games aim to provide players with a specific experience rather than focusing on operationalizing a particular theory. In their 2018 Game Developer Conference talk, *Crusader Kings II* lead designer, Henrik Fåhraeus, states that *Crusader Kings II* revolves around the concept of "dubious morality". Characters choose actions based on their agendas for power. In their discussion of social modeling in *City of Gangsters*, Robison et al. outline the core mentality for playing the game as "You gotta know a guy". That translated to players needing positive interactions with characters to increase their relationship score and gain favor points [26]. These favor points could then get exchanged to increase the player's relationship score with someone in the NPC's immediate social network. The entire experience involves leveraging the NPCs in the player's social network to expand their territory and gain new allies. *Middle Earth: Shadow of War* lets players build revenge relationships with orcs as they kill each other over multiple encounters. While each example does not directly call on sociological or social

psychological theories, as we explained with *Gossip*, they can be derived upon closer inspection.

8.3 Action Selection

Action selection is part of the social interaction design and defines how NPCs decide what social actions to take. Actions selection blurs the line between social modeling and traditional autonomous AI techniques. Since readers can find more comprehensive explanations of the following action selection mechanisms, we will not go into much detail.

Relationship Threshold-based Relationship thresholds are probably the easiest to implement as characters make decisions based on the current valence value of the relationship. *Crusader Kings II* and *City of Gangsters* employ this approach when determining when to cooperate or oppose the player. Decisions can be made on a simple threshold of "if friendship is greater than X do Y", or they can connect the relationship value to a probability distribution for how likely a character is to take a specific action.

Rule and Utility-Based methods Utility methods score potential actions based on how well they benefit the character. They tend to consider multiple factors about the character's state, resulting in more believable behavior than strictly greedy action selection methods. *The Sims 4* uses utility AI to make characters seem more life-like. Sims make decisions based on basic needs like comfort, social, hunger, hygiene, and energy. When AIs select actions based on utility, they score each action, then choose the highest or from among the highest scoring. *Prom Week* uses a rule-based system and scores different social games based on rules that define how much a character wants to perform the action [18].

Limited Look-ahead Limited lookahead simulates the side-effects of taking actions to determine their utility. *Versu* and *PsychSim* use a form of lookahead. Characters in *Versu* simulate each action on the ground-truth state of the simulation, one step into the future, and weigh the action based on how many of the character's wants are satisfied. For instance, if a character wants to annoy another, and an action is available to achieve that goal, they are more likely to pick it. On the other hand, Characters in *PsychSim* simulate the effects of their actions using their internal theory of mind models of how characters might react.

Weighted Random with Preconditions This method assigns preconditions that characters need to meet before executing an action. Perhaps, they need to be an unemployed adult living with their parents before they may execute the *move out of parents* action [29]. In addition to these preconditions, a random

probability also determines if an action takes place. *Talk of the Town* and *Kismet* feature this form of action selection. At times it does border on utility-based. For example, *Kismet* uses character personalities and traits to determine which actions a character is most likely to take.

9 Discussion

9.1 Limitations of the "physics engine" analogy

The analogy works well for data modeling and mathematical processes (characters, relationships, social dynamics functions) but becomes stretched as we consider character autonomy. The analogy was initially chosen to reference physics games like *Angry Birds* and *Cut the Rope* in which the *player* takes actions, and the game responds, via a physics engine, in a systematic but non-prescripted way. Similarly, social simulation games provide this combination of systematicity and emergent dynamics. In this paper, we are building on this previously employed metaphor to make it more rigorous and highlight where it becomes problematic.

9.2 Accounting for time and space

Time and space were two domains that were not mentioned in the core DSA. However, they can be essential factors for social dynamics and character knowledge. Our definition of CBSS states that character relationships evolve over time. Time within a social simulation system can be used to track chronicles of events, decay relationship strength over periods of inactivity, or gate social rules based on the time period of the simulation.

Spatial representations offer context to characters' actions, determine the availability of actions, define socially acceptable actions, delimit what actions are observable to third parties, and limit with whom characters may interact. Space can be represented continuously in 2D/3D or as discrete spaces where the distance between entities is not represented. In *Crusader Kings II* and *City of Gangsters*, space is a limited resource, and characters battle for control over territory. *Kismet* and *Talk of the Town* divide the world into discrete locations where characters may be present. Each location lets characters know who is available to interact with, their role, and their available actions. *Versu* uses a similar representation to represent multiple rooms rather than different buildings within a town. *Versu* and *Kismet* allow designers to make interactions between characters visible to proximal, uninvolved third parties. These actions, of course, cannot be observed by characters in other places. *The Sims 4* also used a type of region system to determine how characters join ongoing social activities and interact with objects and other sims.

9.3 Comparing social physics engines

As one motivation for developing a taxonomy of social simulations, Azad and Martens state that currently, there are no standard evaluation methods for social

simulation systems [4]. They acknowledge the challenge of comparing systems that afford the same social interactions to players but model those interactions differently behind the scenes. For example, betraying someone in *Prom Week* and betraying someone in *Crusader Kings II* are handled differently by each system. *Prom Week* does not have the same type of Opinion system that drives the NPCs of *Crusader Kings*.

We believe that our design space shows why system comparisons are difficult. Translating theoretical social dynamics into code is an exercise in interpretation. There are no objectively-right answers, and multiple competing theories explain why people act the way they do. Traditional rigid-body physics engines share an agreed-upon set of dynamics equations and have collision detection and resolution as two standard bases for comparison. Social simulations, however, do not have a globally agreed-upon set of social dynamics for how humans interact. Also, the model of social interaction depends on the type of experience players should have and the emergent scenarios that designers want to create. Do characters enter into nemesis-style relationships over the course of multiple encounters? Do they need to reason about diplomatic relations with surrounding countries? Do they need to leverage a series of social games to make other characters see them as popular?

So what can we do? Our model provides multiple levels for evaluating social simulations. We could also evaluate systems based on their expressivity. How well can someone encode new social rules within the constraints of the abstractions? Finally, how believable are the social behaviors exhibited by the characters? Yes, this method depends on a final end-user media experience, but it is important to evaluate how legible the social rules are to the player. Making informed predictions about the effects of one's actions on the game is core to a player's feeling of agency and ability to perform social reasoning.

9.4 Feasibility and challenges of implementing systems in practical settings

There are very few social simulation tools for commercial game engines. It is unclear if this results from the tools being challenging to design or social simulation being an afterthought during game design. A way to investigate this problem is employing *Research through Design* [39] methods. Future work should aim to develop tools, get them in the hands of developers (or direct stakeholders), observe how they use them, and take note of what they like/dislike. We are currently working on a lightweight tool inspired by *Crusader Kings*' opinion system to test this hypothesis.

10 Conclusion

In this paper, we propose a design space for social physics engines in games based on the design of traditional rigid-body physics engines. Our model has

three layers: Character/Relationship Modeling, Social Dynamics, and Social Interaction. Social physics engines have many concepts that they could choose to represent, such as persuasion, self-concept, social mobility, group dynamics, and more. Their job is to provide game makers with a toolbox to leverage social intelligence in their game worlds. Ideally, they would empower people to make games that explore how sociocultural norms can affect people differently in society or how they can lead to population-level phenomena. Our proposed design space gets us closer to a reality where social physics engines are off-the-shelf solutions that game developers can add to their games like any other plugin.

References

1. Adams, T., Adams, Z.: Dwarf fortress. [Linux, macOS, Microsoft Windows, Macintosh operating systems, Classic Mac OS] (2009)
2. Aylett, R.: Narrative in virtual environments-towards emergent narrative. In: Proceedings of the AAAI Fall Symposium on Narrative Intelligence. pp. 83–86 (1999)
3. Azad, S., Martens, C.: Lyra: Simulating believable opinionated virtual characters. In: Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment. pp. 108–115 (2019)
4. Azad, S., Martens, C.: Little computer people: A survey and taxonomy of simulated models of social interaction. Proceedings of the ACM on Human-Computer Interaction **CHI PLAY** (2021)
5. Barone, E.: Stardew valley. [Nintendo Switch, Android, PlayStation 4, macOS, iOS] (2016)
6. Cone, J.: It is rocket science, `https://www.gdcvault.com/play/1024972/It-IS-Rocket-Science-The`
7. Crawford, C.: Gossip. [Unpublished Atari 8-bit] (1983)
8. DeKerlegand, D., Samuel, B., Treanor, M.: Pedagogical challenges in social physics authoring. In: Proceedings of the International Conference on Interactive Digital Storytelling. pp. 34–47. Springer (2021)
9. Digital, P.: Gran turismo 7. [PlayStation 4, PlayStation 5] (2022)
10. Eladhari, M.P.: Re-tellings: the fourth layer of narrative as an instrument for critique. In: Proceedings of the International Conference on Interactive Digital Storytelling. pp. 65–78. Springer (2018)
11. Emily Short: Blood and laurels. [iPad OS] (2014)
12. Evans, R., Short, E.: Versu—a simulationist storytelling system. IEEE Transactions on Computational Intelligence and AI in Games **6**(2), 113–130 (2013)
13. Kaiser, R.: The surprising design of Crusader Kings II. gamedeveloper.com (January 2013), `https://www.gamedeveloper.com/design/the-surprising-design-of-i-crusader-kings-ii-i-`
14. Kreminski, M., Samuel, B., Melcer, E., Wardrip-Fruin, N.: Evaluating AI-based games through retellings. In: Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment. pp. 45–51 (2019)
15. MacLean, A., Young, R.M., Bellotti, V.M., Moran, T.P.: Questions, options, and criteria: Elements of design space analysis. In: Design Rationale, pp. 53–105. CRC Press (2020)
16. Mateas, M., Stern, A.: Procedural authorship: A case-study of the interactive drama façade. Digital Arts and Culture (DAC) **61** (2005)

17. Maxis: The Sims 4. [PlayStation 4, Xbox One, macOS, Microsoft Windows, Macintosh operating systems, Classic Mac OS] (2014)
18. McCoy, J., Treanor, M., Samuel, B., Reed, A.A., Wardrip-Fruin, N., Mateas, M.: Prom Week. In: Proceedings of the International Conference on the Foundations of Digital Games. pp. 235–237 (2012)
19. McCoy, J., Treanor, M., Samuel, B., Tearse, B., Mateas, M., Wardrip-Fruin, N.: Authoring game-based interactive narrative using social games and Comme il Faut. In: Proceedings of the International Conference & Festival of the Electronic Literature Organization (2010)
20. Monolith Productions: Middle Earth: Shadow of War. [PlayStation 4, Xbox One, iOS, Microsoft Windows] (2017)
21. Nintendo: Animal crossing: New horizons. [Nintendo Switch] (2020)
22. Paradox Interactive: Crusader Kings II. [macOS, Microsoft Windows, Linux, Classic Mac OS] (2012)
23. Psyonix: Rocket league. [PlayStation 4, Nintendo Switch, Xbox One, Microsoft Windows, Macintosh operating systems, Linux] (2015)
24. Pynadath, D.V., Marsella, S.C.: Psychsim: Modeling theory of mind with decision-theoretic agents. In: Proceedings of the International Joint Conference on Artificial Intelligence. pp. 1181–1186 (2005)
25. Ravio Entertainment: Angry Birds. [Android, iOS] (2009)
26. Robison, E., Viglione, M., Zubek, R., Horswill, I.: AI design lessons for social modeling at scale. In: Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment. pp. 213–219 (2021)
27. Ryan, J.: Curating Simulated Storyworlds. Ph.D. thesis, University of California, Santa Cruz (2018)
28. Ryan, J., Mateas, M., Wardrip-Fruin, N.: A simple method for evolving large character social networks. Proceedings of the Social Believability in Games Workshop (2016)
29. Ryan, J., Summerville, A., Mateas, M., Wardrip-Fruin, N.: Toward characters who observe, tell, misremember, and lie. In: Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment. pp. 56–62 (2015)
30. Samuel, B., Lederle-Ensign, D., Treanor, M., Wardrip-Fruin, N., McCoy, J., Reed, A., Mateas, M.: Playing the worlds of prom week. In: Narrative Theory, Literature, and New Media, pp. 87–105. Routledge (2015)
31. Samuel, B., Reed, A.A., Maddaloni, P., Mateas, M., Wardrip-Fruin, N.: The Ensemble Engine: Next-generation social physics. In: Proceedings of the International Conference on the Foundations of Digital Games. pp. 22–25 (2015)
32. Samuel, B., Ryan, J., Summerville, A.J., Mateas, M., Wardrip-Fruin, N.: Bad News: An experiment in computationally assisted performance. In: International Conference on Interactive Digital Storytelling. pp. 108–120. Springer (2016)
33. Schank, R.C., Abelson, R.P.: Scripts, plans, goals, and understanding: An inquiry into human knowledge structures. Psychology Press (2013)
34. Smith, A.M., Mateas, M.: Computational caricatures: Probing the game design process with AI. In: Proceedings of the Artificial Intelligence in the Game Design Process Workshop at the 2011 AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (2011)
35. SomaSim: City of Gangsters. [Microsoft Windows (2021)
36. Studios, T..: Forza Horizon 5. [Xbox One, Xbox Series X/S, and Windows 10] (2021)

37. Summerville, A., Samuel, B.: Kismet: A small social simulation language. In: Proceedings of the Casual Creator Workshop at the 2020 International Conference on Computational Creativity (2020)
38. Wiltshire, A.: How Crusader Kings 2 makes people out of opinions (November 2016), https://www.rockpapershotgun.com/crusader-kings-2-characters
39. Zimmerman, J., Forlizzi, J., Evenson, S.: Research through design as a method for interaction design research in hci. In: Proceedings of the SIGCHI conference on Human factors in computing systems. pp. 493–502 (2007)
40. Zubek, R., Horswill, I., Robison, E., Viglione, M.: Social modeling via logic programming in City of Gangsters. In: Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment. vol. 17, pp. 220–226 (2021)

Table 1: Chosen Social Simulation Samples

Title	Type	Description
`Crusader Kings II`	Commercial	Set in the middle-ages, *Crusader Kings II* has players fight for power against hundreds of NPCs as they lead their dynasty on a conquest for power and influence. Players need to leverage their own character attributes and the attributes of others to sway opinions an make their dynasty last.
`The Sims 4`	Commercial	The classic virtual dollhouse game, where players create and guide their cast of Sims through their virtual lives, making friends, having families, and achieving life goals.
`Shadow of War`	Commercial	Set in the world of Lord of the Rings, players engage in revenge-loops with casts of procedurally generated Orcs. The Nemesis System simulates an Orc military hierarchy, with specific Orcs gaining and losing power based on their battles against the player.
`Gossip`	Commercial	Made for the Atari, players are tasked with trying to become the most popular among a cast of characters by exchanging opinions about characters.
`Talk of the Town/Bad News`	Academic	Talk of the Town (TotT) simulates the 140-year history of a procedurally generated small American town town. Character live their lives on routines, making friends, lovers, families, rivals and more. Bad News is live-acting game that leveraged TotT for generating characters. Players play as a morticians assistant tasked with finding the next of kin of a recently deceased town resident.
`Dwarf Fortress`	Commercial	Players manage a fortress of Dwarves in a procedurally generated fantasy world. Dwarves interact with other Dwarves, having families, and pursuing life goals.
`Comme il Faut (CiF)/Prom Week`	Academic	CiF is a social physics engine that models characters engaging in social games. Prom Week is *CiF*'s accompanying game where players have to help a character achieve their relationship goals before the Prom by manipulating a social landscape of cool/lame and various personal relationships.

`Versu/Blood and Laurels`	Commercial	Blood and Laurels was an interactive fiction game on the iPad that let players observe or engage in an unfolding story generated by a cast of autonomous characters interacting based on personal goals. Versu was the engine that powered Blood and Laurels.
`Lyra`	Academic	Simulates opinion change between characters by modeling group beliefs, personal beliefs, and the influence of groups on individual identity.
`PsychSim`	Academic	Models *theory of mind* taking into consideration their internal beliefs about other characters beliefs when selecting actions.
`City of Gangsters`	Commercial	Set in prohibition America, players leverage their relationships with characters to gain favors that grow their business and territory.
`Kismet`	Academic	Kismet is a small language for authoring social simulations that can be used like modules of content in tabletop role-playing games. Characters are cast into roles and their roles determine the actions they can take and how they interact with others.