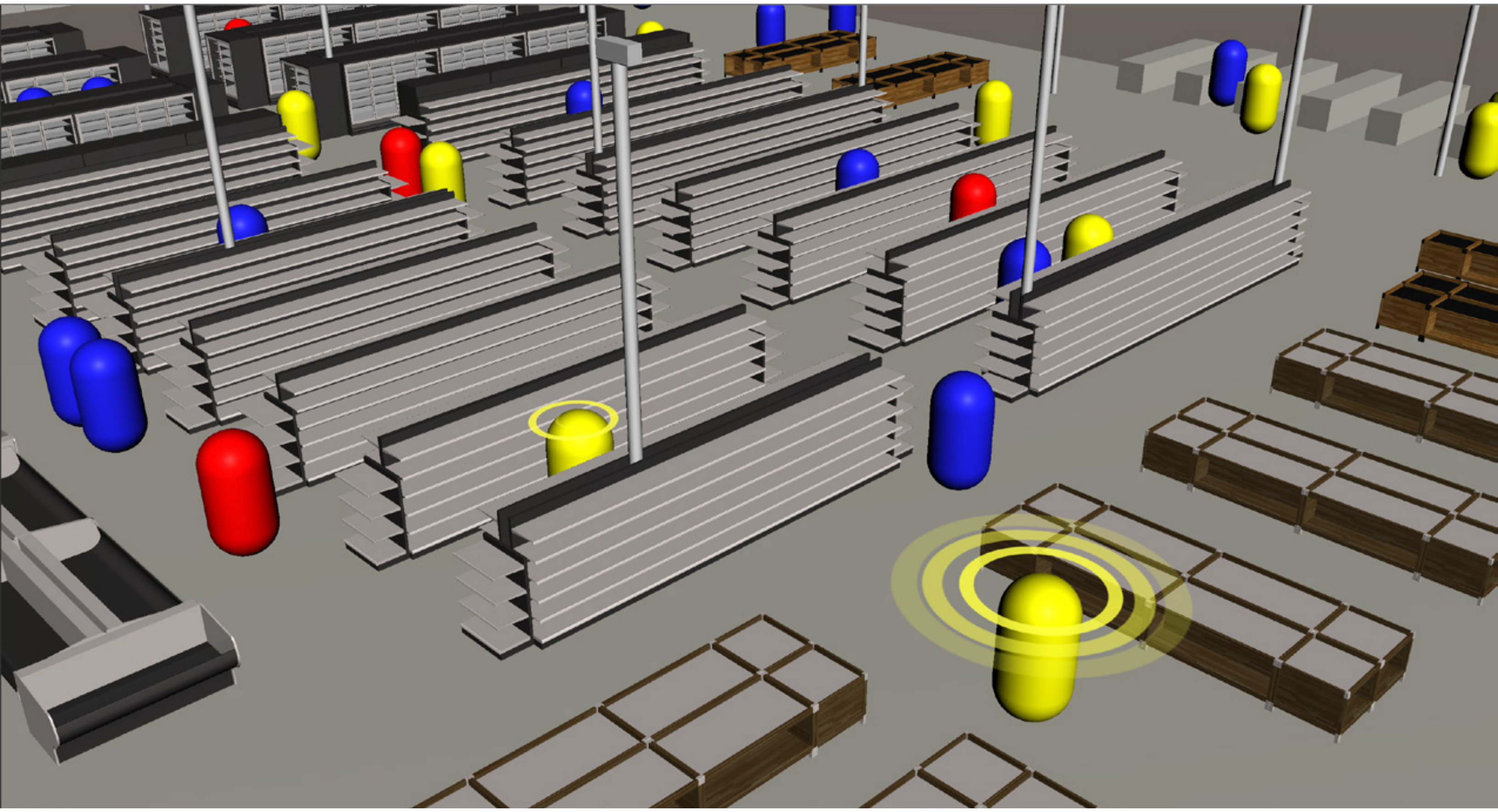
Research study

# Simulation + coronavirus

Unity Simulation for avoiding exposure to coronavirus:
A demonstration of concept to inform policy-making



**Authors**

James Fort, Adam Crespi, Chris Elion, Rambod Kermanizadeh, Priyesh Wani, Danny Lange

**Disclaimer**



Without limiting the foregoing, we want to be absolutely clear: We are not epidemiologists or doctors. Nothing in this project can or should be taken as medical or other guidance or advice of any kind. This is a conceptual model based on simplified rules. It is not scientifically validated. For guidance or advice concerning the novel coronavirus, how it is spread, what steps you might take to avoid infection, or anything else health-related, consult a competent health professional.

# Contents

# Simulation and coronavirus

Computer simulation has been used for decades by researchers, engineers, problem solvers, and policy makers in many fields. It is a central solution to the problem of predicting what might happen under complex and uncertain conditions. Simulation plays a large role in research for infectious disease and has been applied to diseases ranging from the bubonic plague to HIV/AIDS. Epidemiologists use simulation to comprehend the transmission dynamics of pathogens and to inform strategies for prevention and control. For example, researchers use predictive simulation to better understand viral evolution, which can directly inform the development of seasonal flu vaccines[1] or to help search for a cure for HIV.[2]

Researchers start this process by first collecting relevant data from the field. They develop theoretical and heuristically informed mathematical models using the data. The mathematical models are then built into a simulation tool that calculates the outcomes of interest, such as viral spread, from various input parameters, such as population density exposure factors. Finally, the simulation allows big-picture observations and what-if studies, ultimately leading to a conceptual understanding of real-world situations that experts can use to inform challenging policy decisions.

As the novel coronavirus has spread throughout the world, governments have taken drastic measures to curb the severity of the pandemic. Meanwhile, leading research institutes are developing rigorous scientific simulations to help inform government policies. An article in *The New York Times* describes some of the projections that The Institute for Disease Modeling (IDM), a private institution, is developing to help inform policy.[3] Several conceptually oriented simulations have recently been designed to help the public visualize and understand how the novel coronavirus progresses and how our actions can help prevent its spread. Here are some interesting examples we've come across:

- A conceptual simulation published by *The Washington Post* demonstrating the impact of "social distancing"[4]
- A collaborative study by the Katholieke Universiteit Leuven and Eindhoven University of Technology using detailed fluid simulation to show the potential for spread during biking and running situations[5]
- A study from Finland using detailed fluid simulation to show how a cough can spread in a grocery store[6]

# Simulating spatial environments in Unity

Researchers and businesses use Unity to efficiently simulate spatial environments. Unity enables real-time simulation in interactive mode and can simulate frames even faster than real-time in offline mode. By developing projects in Unity that represent real-world environments, parameterizing those projects for simulation, and running many simulated scenarios in the cloud, they can holistically understand complex systems to inform key policy or design decisions.

Let’s define a few of these terms:

- **An environment** in Unity is a 3D spatial representation of a defined space and all of the objects it contains. Those objects can be fixed in position, or they can move and change according to programmed rules, physics models, or machine-learning algorithms. Objects in an environment can be simplified or highly visually detailed.
- **A parameter** is an identified aspect of the Unity environment that can change from one simulation to the next. Because Unity is a highly extensible framework based on C#, parameters can be very simple, such as a change in the speed of a single object’s motion, or they can be more complex, such as the logic handling crowd behavior.
- **A scenario** is a combination of parameter settings to be run in an environment over the course of a simulation. Each simulation typically corresponds to a single scenario. A simulation batch job in the cloud can consist of many scenarios.

Let's explore these definitions through a relatable use case: the spread of the novel coronavirus within a system of shoppers in a grocery store.

The environment in a simulation like this would be the spatial representation of the store, the healthy and infected shoppers it contains, and the rules for virus transmission. The parameters might include the store's rules, such as the allowable number of shoppers or the number of open cash registers. Each combination of these parameter values would comprise a separate scenario.

As each scenario is simulated, the emergent behaviors of the dynamical system play out, and it becomes clear how a store's rules might lead to more or fewer individuals becoming exposed to the virus. A simulation like this could provide helpful insight into the variables that affect the exposure rate so the store's management can develop policies and procedures to potentially reduce the spread of disease in those environments.

Our team built a simplified demonstration project to illustrate what this could look like (Figure 1) and to inspire the Unity user community and disease modeling community to think about new ways to model coronavirus spread. IDM has guided our team over the course of this project, referring us to several existing scientific models for transmission of the coronavirus that helped inform our vision.

Unity works well for systems modeling because it can efficiently simulate systems over longer timescales than many other types of simulators. This means that instead of simulating a short virus-spreading cough event over a single second using many hours of computation in a fluid dynamics simulator, Unity can simulate a group of 50 individuals moving in an environment over the course of 10 minutes using simplified spread dynamics. By simulating a longer time period, larger-scale trends of spread become evident. Leveraging batch cloud simulation capabilities, the simulation can be sped up to compute faster than real-time and run tens or hundreds of thousands of scenarios in a batch in a matter of hours.

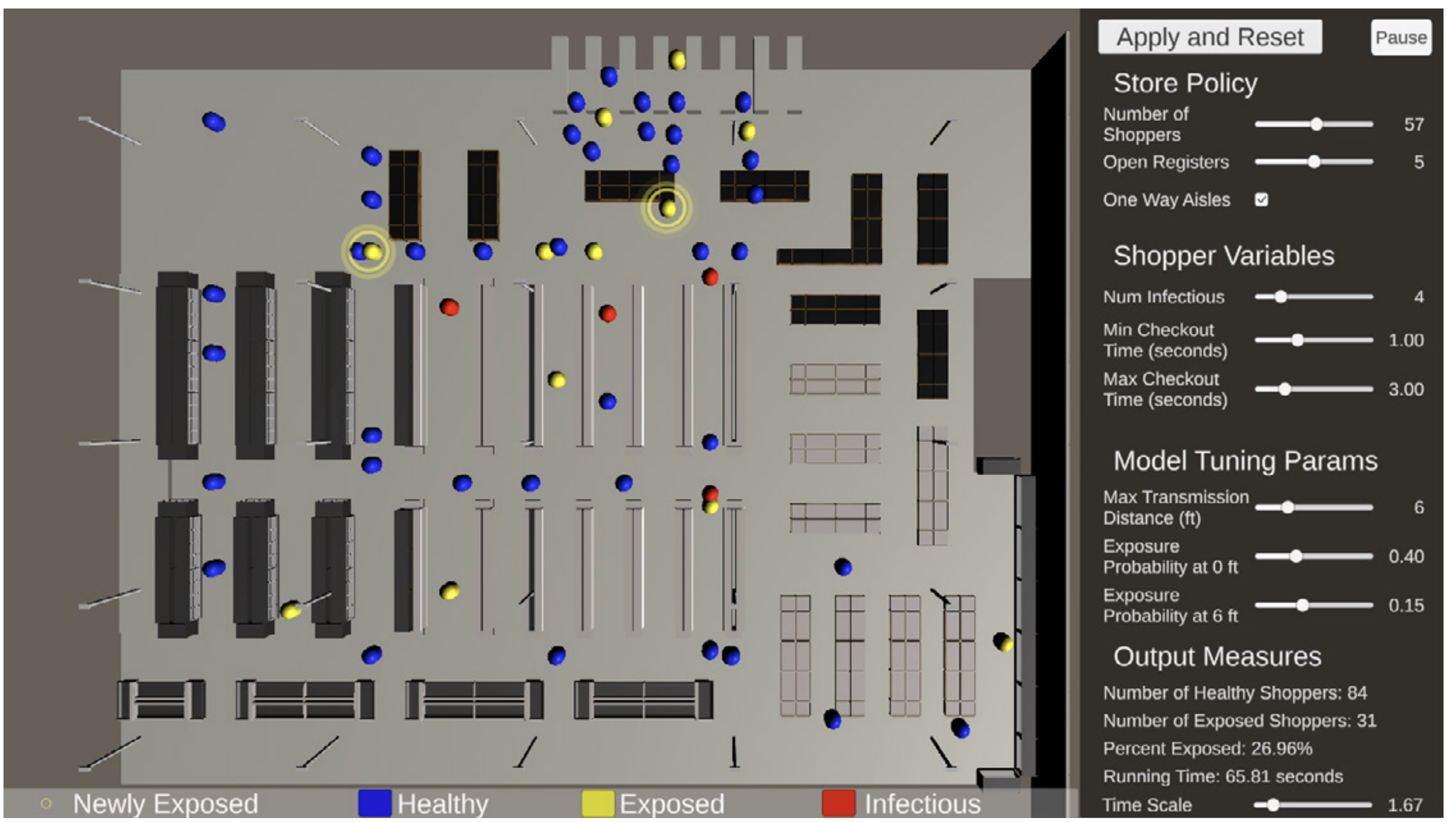



**Figure 1** A screenshot of our simulation demo web application

# Simulation workflow steps

Creating and executing a simulation in Unity involves the following high-level steps:

1. **Project creation:** Create a project with a representative spatial environment and its dynamic elements that change inside the simulation over time. Here, we created a sample grocery store with shoppers that can transmit infection.

2. **Parameterization:** Parameterize the project to identify the ways that it will change from one simulation instance to the next. Almost any aspect of a Unity project can be parameterized through C# scripting. For this example, we identified a range of tuning parameters for the mathematical virus transmission model and parameters for the store's policies, such as the number of allowable shoppers in the store and the allowable traffic direction in aisles.

3. **Output definition:** Specify the output quantities to be measured in the simulation. We looked for the total number of virus-exposed shoppers and the number of virus-exposed shoppers to total shoppers.

4. **Job execution:** Define a job specifying the ranges and combinations of parameter values to simulate. We ran these jobs in a cloud environment to leverage parallelism and complete the task faster.

5. **Results analysis:** Review the summarized output data from the simulation job, perform data analysis, and draw conclusions. We could use the insight from our simulation to make an informed policy decision around how many checkout counters to have open in the store to minimize virus exposure, for example.

# 1. Project overview

We created a simple grocery store environment (Figure 2) and populated it with shoppers that move according to possible navigation paths and some special logic for handing queuing at checkout (Figure 3). The shoppers enter the store, traverse the aisles for a given time, and line up to check out and leave the store.

**Figure 2** Virtual store

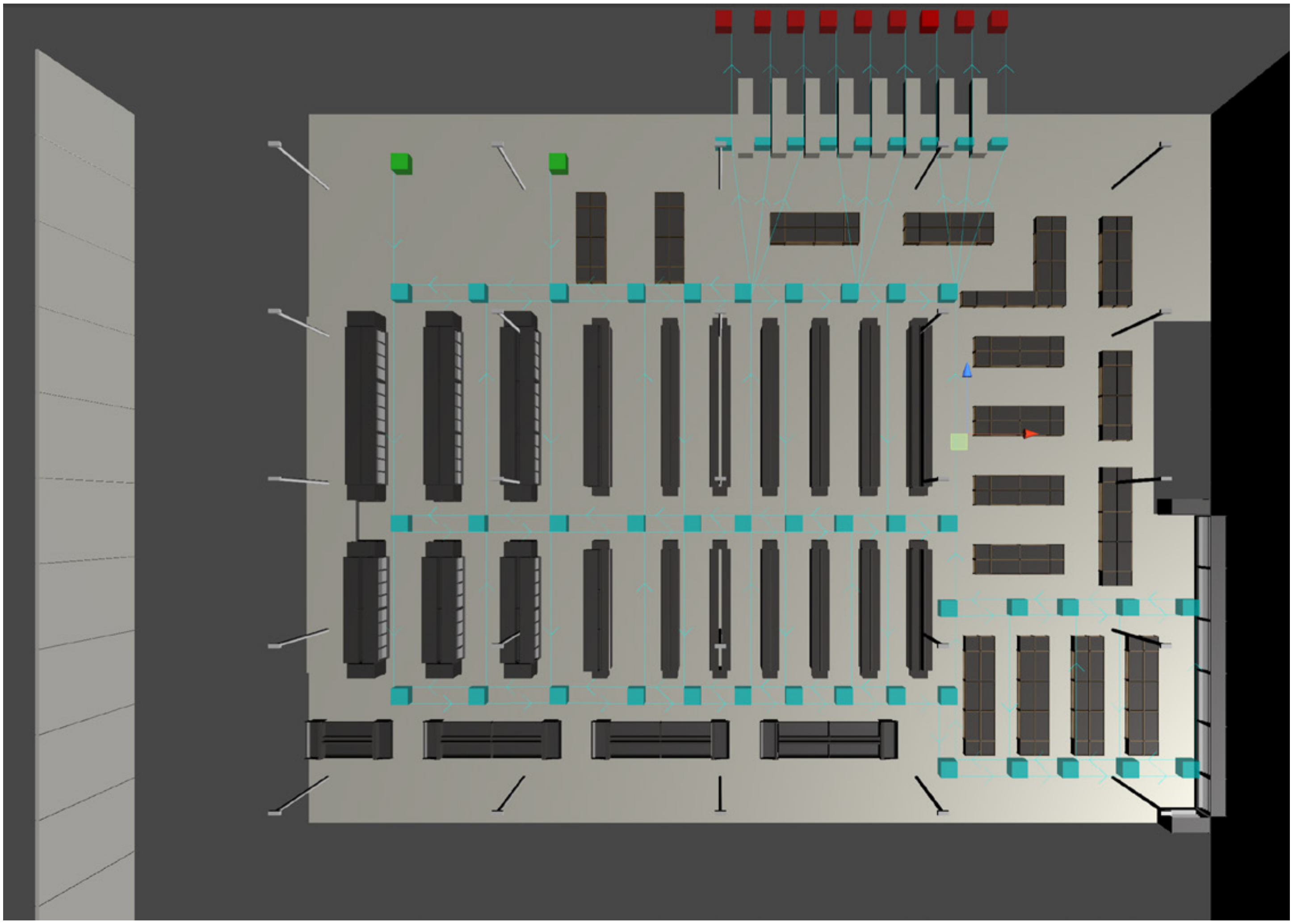


**Figure 3** Allowable motion paths for shoppers

Shoppers can assume three possible states: healthy (represented as blue), infectious (represented as red), or exposed (represented as yellow). A certain fraction of shoppers enter the store in the infectious state. Infectious shoppers can transmit to healthy individuals according to a distance-based probability function that works as follows: When an infectious and a healthy shopper come within a certain distance of each other, they enter an exposure zone. When two individuals cross that threshold, there is a probability of infection. The infection probability increases linearly as they approach each other, until the probability reaches a maximum when the shoppers physically touch. A "healthy" shopper may or may not become "exposed" depending on the calculations related to the model's tuning parameters. As the project runs, you see healthy shoppers gradually become exposed.

# 2. Parameterization

The rate at which shoppers become exposed is controlled by a number of adjustable parameters in the simulation. We categorized these parameters as (a) tunable parameters associated with the mathematics of the transmission model, (b) parameters associated with the store's policy for handling shoppers, and (c) the number of shoppers with an infection who enter the store. Let's take a look at each of these categories in detail.

**a. Model tuning parameters**
Our transmission model includes three tunable parameters: (i) the maximum possible distance for exposure to occur, (ii) the probability of exposure at that maximum distance, and (iii) the probability of exposure upon physical contact between two people. The transmission model is central to the accuracy of the simulation. Note that researchers typically develop and validate transmission models over long periods of time using real-world data. We want to emphasize that the model we employed here is a simplification for demonstration purposes.

Later, to demonstrate the importance of getting the model right, we'll explore how sensitive the output is to the choice of parameter values. A logical next step to develop this model would be to tune it to match real data from scientific studies that demonstrate exposure dynamics. (We share some data sources that might be useful for this purpose at the end of this paper.)

**b. Store policy parameters**
Recent news reports show how stores are trying to manage the risk of spread through various policies. We modeled a few of these in our simulation:

- The total number of customers allowed in the store at one time[7]
- The number of counters open at checkout[8]
- The enforcement of one-way movement in aisles[9]

**c. Shopper infection parameter**
A store has no control over the percentage of infectious shoppers who enter the store. We modeled this factor as a variable in this simulation, and we'll examine how much it impacts the total exposure rate.

## 3. Simulation output variables

Running a simulation is only useful if it outputs informative data. In this simulation, we see shoppers becoming exposed in real-time, but more importantly, we can directly measure the percentage of shoppers who become exposed over the course of the simulation. You can try out an embedded version of our simulator that lets you experiment with the input parameters and see their impact on the output variables in real-time. After several minutes of running, the rate of exposed shoppers over total shoppers reaches a steady-state value, indicating that the simulation has essentially converged.

## 4. Cloud simulation of the whole parameter space

As a next step, we sampled each parameter at multiple values and ran a batch simulation job on the cloud using 10,000 simulation instances to test all of the combinations of our sampled parameter values in just three hours. Each combination of parameter values was simulated five times to account for randomness and the output averaged. This batch job produced a results file containing the number of healthy shoppers and number of exposed shoppers from each run. We easily calculated the exposure rate as the number of exposed shoppers divided by the number of total shoppers by post-processing the data in a spreadsheet.

# 5. Data review and analysis

Using the simulation output data from all of our runs, we performed a sensitivity analysis to see the impact that some of our input parameters have on the output. One useful tool for performing sensitivity analysis is a tornado diagram, which shows how each input parameter causes variation in the output when set to its extreme values while all other input parameters are held constant (Figure 4). The larger the spread of the bar, the more influential the parameter is at its sampled values. If the left-most part of the bar is blue, the variable is directly correlated with the output, whereas yellow to the left means the variable is inversely correlated with the output.

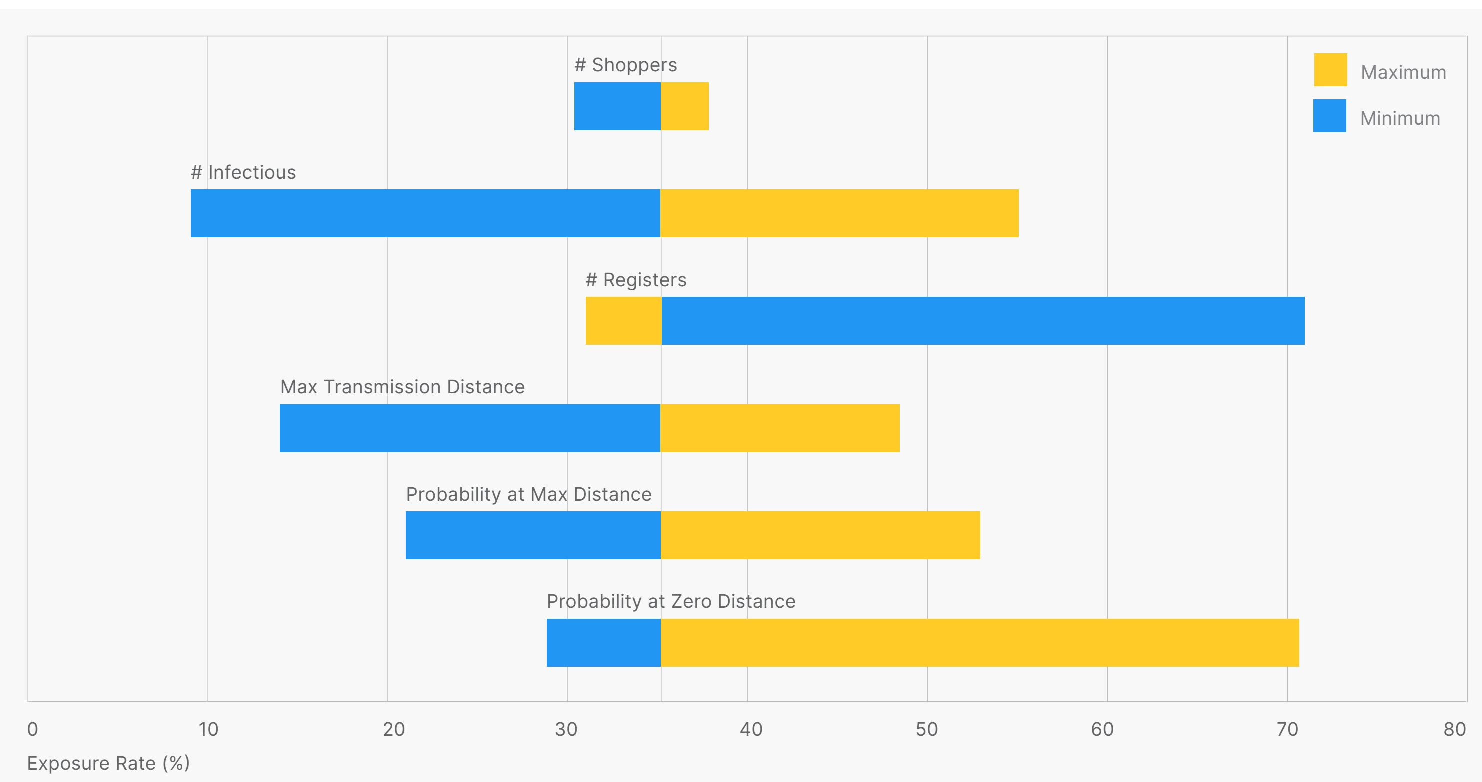



**Figure 4** Sensitivity analysis of input parameters to the exposure rate

From this view of the data, we can conclude:

- The exposure rate is most sensitive to the number of infectious shoppers in the store. This is intuitively expected. As the number of infectious shoppers increases, more infectious shoppers will interact with healthy shoppers.
- The only variable that shows inverse correlation with the output is the number of open registers. Increasing throughput significantly reduces the chance for exposure. However, the variable's sensitivity is asymmetrical, indicating that going from just one to several registers makes a large difference but the impact trails off upon further increase.
- The three model tuning parameters (maximum transmission distance, probability of exposure at zero distance, and probability of exposure at maximum distance) are nearly as influential as the store and shopper parameters. It is critical to tune these parameters in the simulation.

Now we have a general understanding of the sensitivities, but let's look at some other trends that the tornado diagram does not reveal, since it cannot take into account how multiple parameters interact. We'll focus on the impact that store policy parameters have on simulated exposure rates and keep the tuning parameters fixed at some default values. For normalization purposes, these diagrams were all generated while holding the rate of infectious shoppers to total shoppers constant.

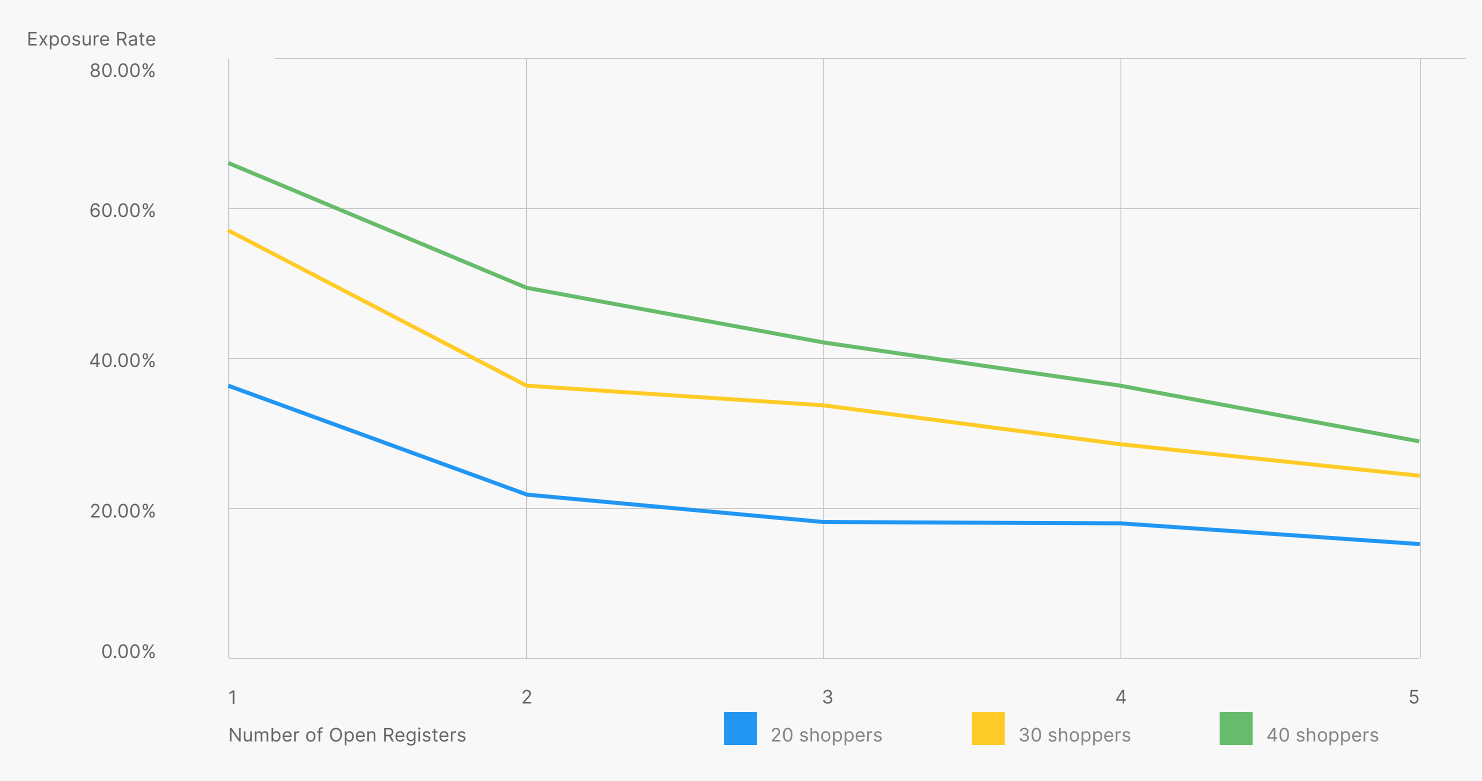



**Figure 5** Exposure rate as a function of the number of open registers

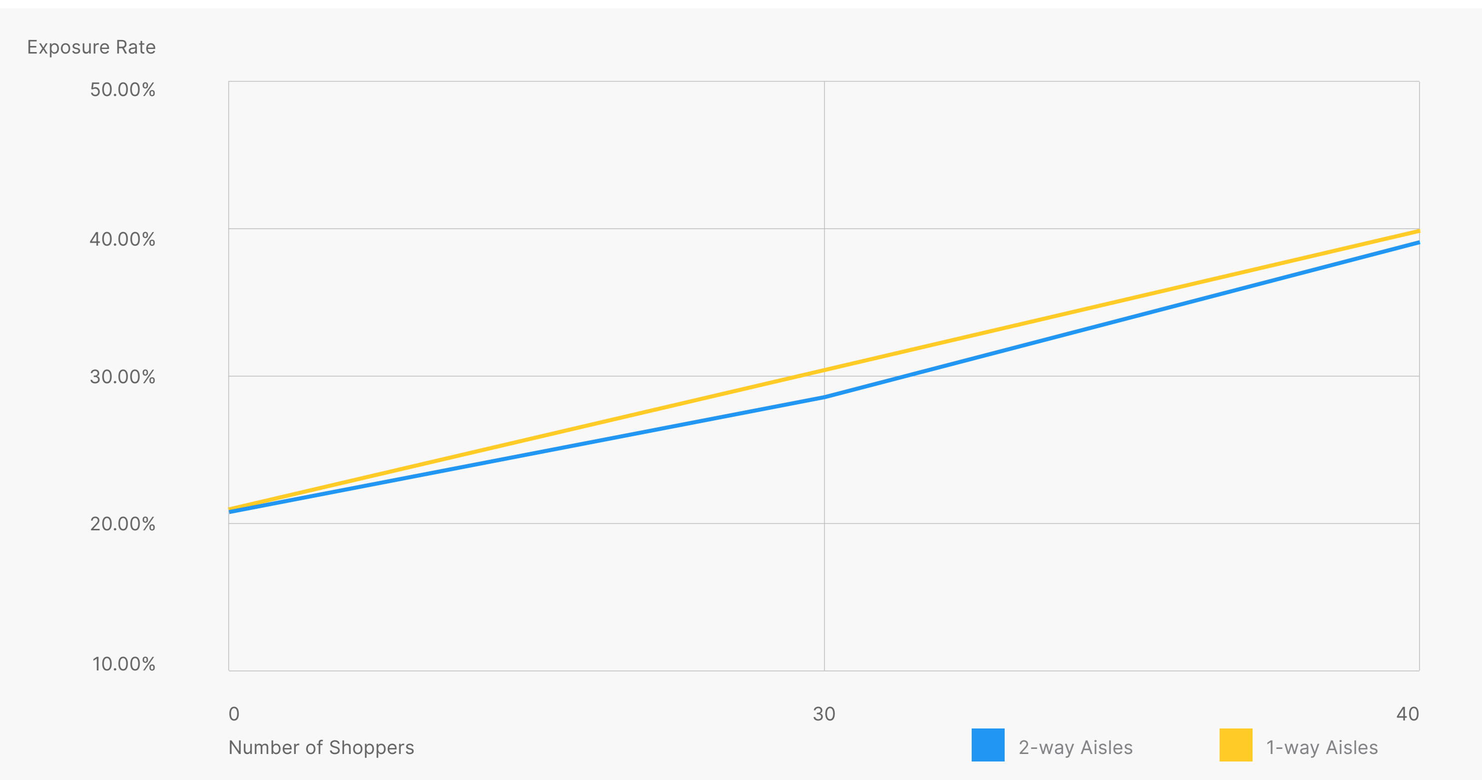



**Figure 6** Exposure rate as a function of the number of shoppers for one-way and two-way aisles

Figure 5 shows the relationship between the number of open registers for different numbers of allowable shoppers. When only one register is open, the exposure rates are very high, especially when the store is crowded with 30 or 40 shoppers, due to backups at the checkout area. Opening up a second register has the largest impact on reducing the exposure rate. The benefits level off after that point when the store has 20 shoppers, but opening up to five registers is helpful for when the store has 30 or 40 shoppers.

Figure 6 reveals how one-way versus two-way allowable movement in the aisles affected the simulation. The results show negligible differences between one-way and two-way aisle movement. Note, however, that two-way movement was restricted only in the vertical aisles of the store, and many infections occur in the horizontal aisles and the checkout location. Additionally, there are currently no rules in the simulation to prevent two shoppers from getting very close to one another, even when moving in the same direction. At the end of this paper, we suggest additional features that we believe could be added to the simulation to make this effect more realistic and potentially impact exposure rates.

# Conclusions and takeaways

Our team built a simplified demonstration project to illustrate new ways to model coronavirus spread. The results of our simplified model show that the policies in our fictitious store indeed have an impact on the spread of the virus as we've modeled it.

The impact was evident by running the simulation in real-time and adjusting parameter values individually, but we needed to run the simulation over many parameter combinations to determine what parameters mattered the most and at what values they made the biggest difference in the results. This example demonstrates that to alleviate exposure rates, the store should open two registers when 20 shoppers are in the store and up to five registers when 30 or more shoppers are in the store. Most importantly, we note that the parameters we use in the mathematical model for transmission have a large impact on the results and need to be rigorously developed using real data.

# Extend this project on your own

This simulation is just a starting point. We're making this project, including its full source code, available via GitHub for anyone to experiment with. You can run the demo we've described here using the free tier of the Unity Game Simulation beta, which leverages Unity Simulation and Unity Remote Config under the hood. We suggest a number of ways to expand this project:

**Virus transmission model:** Our model is conceptually very simple and not scientifically validated. As a starting point, the model could be better formulated or calibrated with scientific data. IDM suggests the following resources for potential data:

- The IDM InfoHub, which has some tables on typical parameter configurations
- IDM's own research reports, such as those concerning estimates of burden in specific regions
- Several curated literature hubs, including the NIH COVID-19 Portfolio, LitCovid, and the COVID-19 Preprint Index

**Shopper behavior model:** The shoppers in our simulation are simple symbolic representations of shoppers, and their movement patterns are completely rectilinear. The check-out system is limited in length according to a predefined path. In reality, shoppers move and line up in much more complex ways, especially when they're afraid of being infected by other shoppers. How might one develop a more realistic movement and interaction model for these shoppers? We imagine creating distributions of behavior to reflecting that some shoppers keep more physical distance than others. and using Unity's NavMesh functionality might be used here to define more organic allowable zones for movement that is more organic. Perhaps the simulated shoppers could be treated as agents and trained using an exposure-avoidance objective function. The Unity ML-Agents toolkit could help here.

**Individual-to-surface transmission:** In reality, transmission is not limited to individual-to-individual contact. The virus can also spread from an individual to a surface, it can linger on surfaces according to a certain decay function, and then can transmit to an individual that comes into contact with the surface. For example, IDM indicated a letter published in *The New England Journal of Medicine* that gives data for how coronavirus decays over time on various surfaces.[10] How might this be modeled in the project?

**Other types of environments:** A grocery store isn't the only place that coronavirus transmission can happen. What about modeling transmission in a park, in an airport, or at a school? Perhaps you can create your own environment to test this out or have a look at some of the assets on the Unity Asset Store. How would you model the policy rules in those other environments and the behaviors of the individuals?

# Stay connected

We encourage interested readers to explore and extend this project, and if you do, share the results on social media using the hashtag **#UnitySimulation**. You are also welcome to contact us directly about any project ideas you have at simulation@unity3d.com.

# References

[1] www.bedford.io/pdfs/papers/morris-predictive-modeling.pdf
[2] www.genetics.org/content/190/3/1087
[3] www.nytimes.com/2020/03/29/us/seattle-washington-state-coronavirus-transmission-rate.html
[4] www.washingtonpost.com/graphics/2020/world/corona-simulator/
[5] www.medium.com/@jurgenthoelen/belgian-dutch-study-why-in-times-of-covid-19-you-can-not-walk-run-bike-close-to-each-other-a5df19c77d08
[6] www.hpcwire.com/2020/04/08/supercomputer-modeling-tests-how-covid-19-spreads-in-grocery-stores/
[7] www.cnn.com/2020/04/04/business/walmart-limits-customers-covid-19/index.html
[8] www.seattletimes.com/business/local-business/seattle-qfc-store-managers-bulletin-to-workers-says-social-distancing-will-not-always-work-amid-coronavirus/
[9] www.abc7news.com/social-distancing-grocery-stores-one-way-aisles-coronavirus-safeway-covid-19-outbreak/6074817/
[10] www.nejm.org/doi/full/10.1056/NEJMc2004973

## Related reading

Dynamics Analysis and Simulation of a Modified HIV Infection Model with a Saturated Infection Rate. Computational and Mathematical Methods in Medicine. 2014.

Computer Simulations Help Explain Why HIV Cure Remains Elusive. Infection Control Today. 2012.

# Suggested resources

COVID-19 Portfolio, National Institutes of Health (NIH)
COVID-19 Preprint Index
Institute for Disease Modeling
- InfoHub
- 2019-nCoV: preliminary estimates of the confirmed-case-fatality-ratio and infection-fatality-ratio, and initial pandemic risk assessment
- Working paper – model-based estimates of COVID-19 burden in King and Snohomish counties through April 7, 2020

LitCovid, National Library of Medicine
Unity Game Simulation
Unity Simulation
Unity Machine Learning
Unity Coronavirus Simulation Source Code

unity

unity.com