Simulating and visualising infection spread using real-world movement data

Simulation und Visualisierung der Ausbreitung eines Virus anhand realer Bewegungsdaten

M. Jerome Stephan

Universitätsbachelorarbeit
zur Erlangung des akademischen Grades

Bachelor of Science
(B. Sc.)

im Studiengang
IT Systems Engineering

eingereicht am 31. Juli 2022 am
Fachgebiet Data Analytics and Computational Statistics
der Digital-Engineering-Fakultät der Universität Potsdam

Gutachter	Prof. Dr. Bernhard Renard
Betreuer	Ferdous Nasri

Abstract

While many models have simulated viral infections and their effects, especially concerning the SARS-CoV-2 pandemic which has changed the way of life all around the world, visualizations of the spread and its effects on the mobility of a population are often unintuitive and behind a layer of abstraction. Using real mobility data from the co-author NET CHECK as well as official freely available infection data, this work puts forward a simultaneous individual-based simulation and visualization of the effect of SARS-CoV-2 on the German population. Each particle decides, based on daily information of the county it is in, which county to travel to, whether it is infected and if so, how long that is the case. Applying probabilities calculated from the decisions of the particles on the day before brings forth predictions of travels and infections.

Utilizing the node- and Python-based programming environment of the visual effects software Houdini, it is possible to create a procedural setup workflow to show the processing of this project's data both in a visual way by displaying particles representing the population on the map of - in this case - Germany besides evaluating it on a spreadsheet at the same time.

This not only helps to intuitively present the effects of the pandemic and improve the simulation more efficiently by the ability of visual debugging but also to easily adjust it if more detailed data, for example more accurate location data, is available or to up the resolution, i.e. the number of simulated particles, in an instant.

Zusammenfassung

Während viele Modelle entwickelt wurden, um Virusinfektionen und ihre Auswirkungen zu simulieren - insbesondere während der SARS-CoV-2-Pandemie, die die Lebensweise auf der ganzen Welt verändert hat - sind Visualisierungen der Ausbreitung und ihrer Auswirkungen auf die Bewegungsmuster einer Bevölkerung oft unintuitiv. Mittels realer Mobilitätsdaten des Co-Autors NET CHECK sowie offizieller, frei verfügbaren Infektionsdaten wird in dieser Arbeit eine individuenbasierte, gleichzeitige Simulation und Visualisierung der Auswirkungen von SARS-CoV-2 auf die deutsche Bevölkerung vorgestellt. Jedes Individuum entscheidet auf der Basis der täglichen Daten über den Landkreis, in welchem es sich befindet, in welchen Landkreis es reist, ob es sich infiziert und falls ja, wie lange dies der Fall ist. Nutzt man die Infektions- und Reisewahrscheinlichkeiten, die aus den Entscheidungen der simulierten Personen am Vortag berechnet werden, kann man Vorhersagen von Reisen und Infektionen durchführen.

Mithilfe der node- und pythonbasierten Programmierumgebung der Visual-Effects-Software Houdini ist es möglich, einen prozeduralen und automatisierten Workflow zu erstellen, der die Verarbeitung der Daten dieses Projekts sowohl visuell durch die Darstellung von Partikeln, die die Population auf der Karte von - in diesem Fall - Deutschland repräsentieren, als auch durch die gleichzeitige Auswertung auf einem Spreadsheet darstellt.

Dies hilft nicht nur, die Auswirkungen der Pandemie intuitiv darzustellen und die Simulation durch die Möglichkeit visuellen Debuggings effizienter zu gestalten, sondern auch, sie leicht anzupassen, wenn detailliertere Daten, z. B. genauere Standortdaten, verfügbar sind, oder die Auflösung, d. h. die Anzahl der simulierten Partikel, zu erhöhen.

Contents

Abstract iii

Zusammenfassung v

Contents vii

1 Introduction 1
- 1.1 Motivation 1
- 1.2 Related Work 2
- 1.3 Background 3
  - 1.3.1 COVID-19 infection data 3
  - 1.3.2 Population movement data 3
  - 1.3.3 Technical keywords 4
- 1.4 Proposition 5
  - 1.4.1 Concept 5

2 Method 7
- 2.1 Data retrieval and pre-processing 7
- 2.2 Implementation 8

3 Results 15
- 3.1 Data & Prediction 15
  - 3.1.1 Infections 15
  - 3.1.2 Travels 17
- 3.2 Visualisation 19
- 3.3 Optimizations 22

4 Discussion 23

5 Conclusions & Outlook 25
- 5.1 Conclusion 25
- 5.2 Outlook 25

Bibliography 27

Declaration of Authorship 31

1 Introduction

1.1 Motivation

There is no denying that the COVID-19 pandemic has changed the way of life around the world, especially concerning governmental restrictions and the push to work from home, both leading to fewer commutes and travels [De 20]. Though there have been many evaluations of this behavioral change, neither an individual-based visualization of daily travels between counties nor the spread of the SARS-CoV-2 virus exist for Germany, which would help to present the matter intuitively. While diagrams and graphs have an advantage in the face of evaluating the topic scientifically, a simulation and visualization on the individual level are more relatable to the viewer and serve the purpose of exposing the pandemic and its effect in an engaging manner [GMB16; ZK09].

One of the aforementioned effects is government-issued lockdowns resulting in fewer people moving about, or the stark inverse following the lifting of social restrictions timed with the beginning of summer [Tor+21]. Another effect to be shown are local virus outbreaks, displayed by a substantial increase in the number of infected simulated people in specific counties. Adding onto this list, a further behavioral change to be observed is the ratio of local travels compared to extended trips spanning the size of Germany evolving along the timeline of the pandemic.

As each simulated person is deciding for themselves where they are traveling to, whether they get infected, and if so, for how long, on any given day, it is possible to extend the visualization beyond given data and thus simulate and predict movements and infections. This enables, again, an intuitive display of what to expect in the coming days.

Lastly, this thesis strives to show the versatility of the visual effects software Houdini as a programming environment, shining light on its features to make it more attractive in a scientific context. Not only is it possible to utilize it in a traditional way with a Python terminal, but also to set up a network of python scripts and have the processed data visible in both a spreadsheet and a 3D viewport. The latter enables us to do visual debugging, making it easier to see errors, zoom in on specific details, and generally learn more about what is happening. On the other hand, the network of scripts permits us to work on particular parts of the program more easily besides being able to up the resolution of the simulation and plugging in more detailed data in an instant.

1.2 Related Work

The initial approach of this work was based on detailed cellphone data, i.e. specific locations of the population at given times, simulating transmission based on close contact with infected people, akin to the individual-based model for a small sample size on a simulated population of Daegu, Korea, or the Full-Scale Agent-Based Model considering Lombardy in Italy [Gia21; ST20]. However, as has been evaluated in a review of agent-based models concerning COVID-19, there are quite a few shortcomings that prevent them from being used as support for decision-making processes in the face of governmental restrictions [LJD21].

Unfortunately, none of these seem to consider the educational purposes for the standard population. While the results of all these approaches show the potential of agent-based models, they do not present their findings intuitively. If they did, the general population could better understand the effects of different restrictions or the pandemic.

1.3 Background

1.3.1 COVID-19 infection data

This section provides an explanation of the relevant and available data on the COVID-19 pandemic in general and in Germany, as this version of the model runs on data relating to the latter.

Robert Koch Institut The primary resource for data on diseases, in our case SARS-CoV-2, in Germany is the Robert Koch Institut (RKI). The RKI is the German national institute for health research within the German Ministry of Health, responsible for researching diseases and public health risks, and combating infectious diseases [Ins].

Infection Numbers Germany is divided into 16 states, which are further divided into 400 counties. Each district's public health department forwards its collected case numbers to the RKI. They collect and publish the data daily on a GitHub repository as a CSV file, detailing when the infections were reported, which county it was reported from, the age group and the gender of the infected [Ins22].

Infectiousness Time The United Kingdom Health Security Agency (UKHSA) published a paper justifying the reducing of the self-isolation time to 7 days [Age22]. In it, they showed a distribution of how many people were still infected after a certain number of days since their infection, resembling a normal distribution around 4 days.

1.3.2 Population movement data

A key feature of the model is that it uses real-world movement data. These paragraphs explain its origin and definition.

NET CHECK GmbH NET CHECK GmbH is a consulting company focused on cellular providers, which conducts the largest annual independent cellular network test spanning all of Germany [Wol]. They gather the data needed by partnering with mobile app developers to incorporate NET CHECK's API collecting cellular reception strength, location data and more [Gmb].

Travel Data Each location ping from NET CHECK's data can be assigned to a county. Further, a *travel* is defined as a device being in one county on one day, and in another on the following day.

1.3.3 Technical keywords

GeoJSON A GeoJSON file encodes polygon objects and other geographic data structures, their properties, and their spatial extents using a geographic coordinate reference system [But+16].

Houdini SideFX's Houdini is a 3D animation software application most commonly used for the creation of visual effects in film and games. In it, every action is stored in a *node*. These nodes are then “wired” into networks, which define a “recipe” that can be tweaked to refine the outcome. The ability for nodes to be saved and to pass information, in the form of *attributes*, down the chain is what gives Houdini its procedural nature. [Sofa]

LRU Cache Caching improves performance by keeping recent or often-used data items in memory, saving time when repeatedly accessing the same data. An LRU (least recently used) cache discards the least recently used items first. Pythons *functools* package makes it easy to implement such a cache [Pyt].

1.4 Proposition

This work proposes a simultaneous visualization and simulation model based on individuals. It aims to set up an easily modifiable framework that strives to present the spread of the pandemic intuitively, and the effect governmental restrictions and the virus itself might have on the population's behavior, visible in differing traveling patterns and stark differences in the number of infections.

1.4.1 Concept

The approach we are looking for is highly modular, such that the data used to drive the simulation calculations can easily be swapped out. This is useful if we want to run the model for other countries or plug in more detailed data, e.g., county travel information taking more people into account.

Given that we are looking to develop a model acting on the individual level, each person must act as an individual, i.e., other individuals' decisions do not actively affect them. However, if many individuals choose to be infected in a specific county, the chance of another individual being infected increases. We consider the population of Germany to drive the model as an example.

The probability of an individual becoming infected is defined by the incidence data of the county it currently resides in. The daily travel data between counties determine the probability of traveling to another county.

A basic setup for the proposed approach can be seen in figure 1.1, showing the workflow of the model.

Lastly, we need to define a way to represent an individual in the software we are using. Houdini works best on points defined in 3D space, so we will use a simple 1:1 relationship between individuals and geometry points in Houdini with all the attributes needed for the simulation and visualization to work.

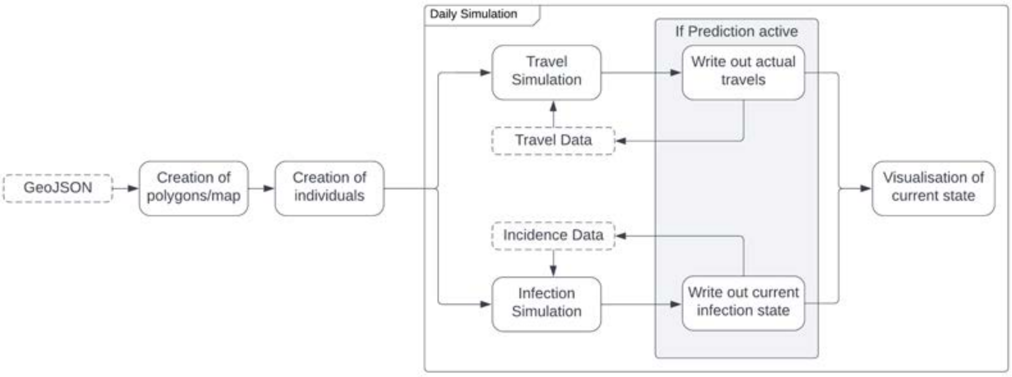


Figure 1.1: Conceptual process of the model. Data sources are marked in dashed boxes and algorithms are written in solid boxes. Arrows show the flow of the program. The big frame titled *Daily Simulation* is the process repeated for each day of the pandemic, the gray frame inside of it marks the two algorithms which are only enabled when the prediction part of the model is active.

2 Method

2.1 Data retrieval and pre-processing

In the previous chapter we mentioned the modularity of our approach, such that the data listed here is rather exemplary. Even so, there are still three fundamental types of data that are needed, which are as follows.:

- Daily incidence data: We used freely available COVID-19 incidence data on a county level from the Robert Koch Institut, the official German health institution [Ins22].
- Daily travel data: From the data NET CHECK provided, we as a team calculated a 3-dimensional travel matrix in which, for each day and each county in Germany, it lists how many people traveled to each other county.
- A GeoJSON file listing the counties as polygons with an attribute to map the polygons to the counties: We used such a GeoJSON file for Germany available from the ESRI COVID-19 dashboard, which also divides Berlin into districts, to get a more accurate simulation and visualization [ESR22].

We worked with a 7-day rolling average applied to the daily incidence data to iron out differences created by reporting dates and inaccuracies. This mitigates the peaks on reporting dates and lows on weekends. As NET CHECK's travel data works in absolute numbers as well, we estimated that only 5% of the German population is represented by comparing the number of devices tracked with the population number. The given numbers were multiplied by 20 to remedy this. In general, we always worked with percentages relative to the county's population to scale the simulation's resolution more efficiently, which we calculated by using freely available census data listing the population number for each German county [Bun].

The simulation was slowed down by repeatedly searching for the matching county data for each simulated person. By splitting up the incidence data into a new file for each day, instead of having one vast file, as well as further dividing the travel data into not only each day but also each county, accessing times could be reduced. section 3.3 reports these optimizations in detail.

2.2 Implementation

To create a simulation and a visualization, we need a population on which to run them. For this, we have modified *vvzen's Houdini GeoJSON Tools* Python package such that it not only imports GeoJSON polygons but also reads in attributes of polygons [vvz21]. Each county is represented by a polygon group, whose members are the imported polygon shapes representing it in 3D space. This now enables us to read in a spreadsheet listing each county's population and use a standard Houdini scatter node to distribute points accordingly on the surfaces of the polygons.

Once distributed, we need to define the attributes each simulated individual requires. Since we are looking to store infection data, we need an *infected* attribute, coupled with an *infection time* attribute, to know how long this individual will be infected. The following important part is the traveling; to store the information needed, we create a *current county* attribute as well as a *next county* attribute. The former is essential to calculate the latter, but also to calculate the infection based on the incidence data of the county it is currently in. On top of that, we will also save the *point number* of the point to use as part of the seed for the randomness calculation in both the infection and traveling portion.

Now we can start the daily loop of simulating and visualizing the infections and travels. We separated the infection and the traveling portion for each day into different python scripts run one after the other to be able to isolate them and work on them separately. First, we will take a look at the infection calculation. Based on the incidence data we have of the specified day, each simulated point has the following chance of being infected, being in a specified county:

$$\text{infection chance} = \frac{\text{number of infections in the county}}{\text{population of the county}} * \text{infection multiplier}$$

where the *infection multiplier* is an arbitrary value we can easily change with a slider to show a more drastic infection effect, by default it is set to 1.0.

Once an individual is marked as infected, we need to define the number of days that individual will stay infected. To achieve this, the following probabilities have been implemented, based on an estimate of the research the *UK Health Security Agency* conducted concerning the isolation period and the number of people still being infectious [Age22]:

infected days	1	2	3	4	5	6	7	8	9
probability	5%	5%	15%	20%	25%	10%	5%	5%	5%

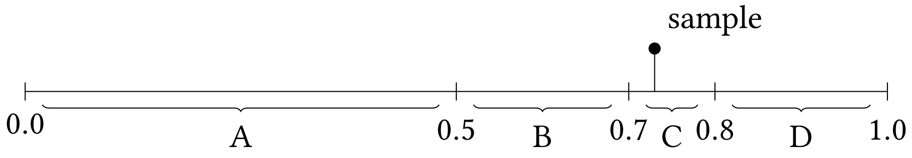
The next step in our model is to calculate the traveling of individuals. Assume the individual we are looking at is situated in county X. Based on the travel data we are given on the specified day for county X, we will look at how many people traveled to each other county and calculate our probability of traveling to county N in the following way:
Suppose the probability of an individual traveling from county X to county N on the given day is

$$\text{probability}_{XN} = \frac{\text{number of people traveling from } X \text{ to } N}{\text{population of county } X}$$

and, as an example, we will use the following relative distribution of travelers from county X:

traveling to county	A	B	C	D
probability	50%	20%	10%	20%

then we can take a random sample from an even distribution between 0.0 and 1.0 and get the county this individual is traveling to by partitioning the range according to the relative portions of travelers:



In the sample taken above, the individual would travel to county C.

Once the individual has decided which county to travel to, we need to find a position in that county the point representing the individual can travel to. We could use the standard Houdini scatter node to achieve this, and while accurate, it is incredibly slow. We have written a custom scattering alternative which tends to be inaccurate when working with oddly-shaped polygons, e.g., when the centroid isn't located inside the polygon. Still, this method of distributing points is 95% faster than the standard scatter node. The way it works is as follows:

From the polygon group representing the county, we choose a random polygon. Then we sample a random point *a* from the outer edge of the chosen polygon and, lastly, a random position on the line from *a* to the polygon's centroid. This could further be improved by checking if the position gathered by this method is actually on the polygon; this would, however, slow the process down again, which is why we decided against it, especially considering that this happens only in relatively rare cases.

To further speed up the calculation steps, we implemented a *LRU cache* when accessing the input data so that the duplicate files do not need to be read repeatedly for every individual in the same county.

We would not be able to see individuals moving to other counties if each day was just one frame, which is why each day is represented as one second on the timeline of the visualization. To show movements, if the individual traveled, we use *LERP* - linear interpolation of vectors - between the position in the old county and the new position in the county the point is traveling to.

As previously stated, there is also the possibility of using the model in a predictive way. By using the methods discussed beforehand, because of the nature of randomness, it is likely that the result of infection and traveling is not the same as in the data that is fed in. Suppose we save and write out these choices that each individual made on that day. In that case, we can use them as inputs for the next round of calculation, thus engaging in a prediction akin to the naive forecast. Though, as movement patterns differ from weekdays to weekends, we save the choices based on the day of the week to still be able to differentiate between these patterns.

We have always used relative numbers when calculating infections or travels. Not working with absolute numbers enables us to easily adjust the number of simulated individuals, with just one change of a slider of distributed points at the very beginning, while retaining the correct factor of infections and people traveling to other counties. This is especially useful when the need to reduce calculation time arises to see changes made in the scripts quickly.

An overview of the Houdini network of the scripts described above is pictured for the initial setup of the simulated population as well as the daily calculation in the two figures 2.1 and 2.2 respectively.

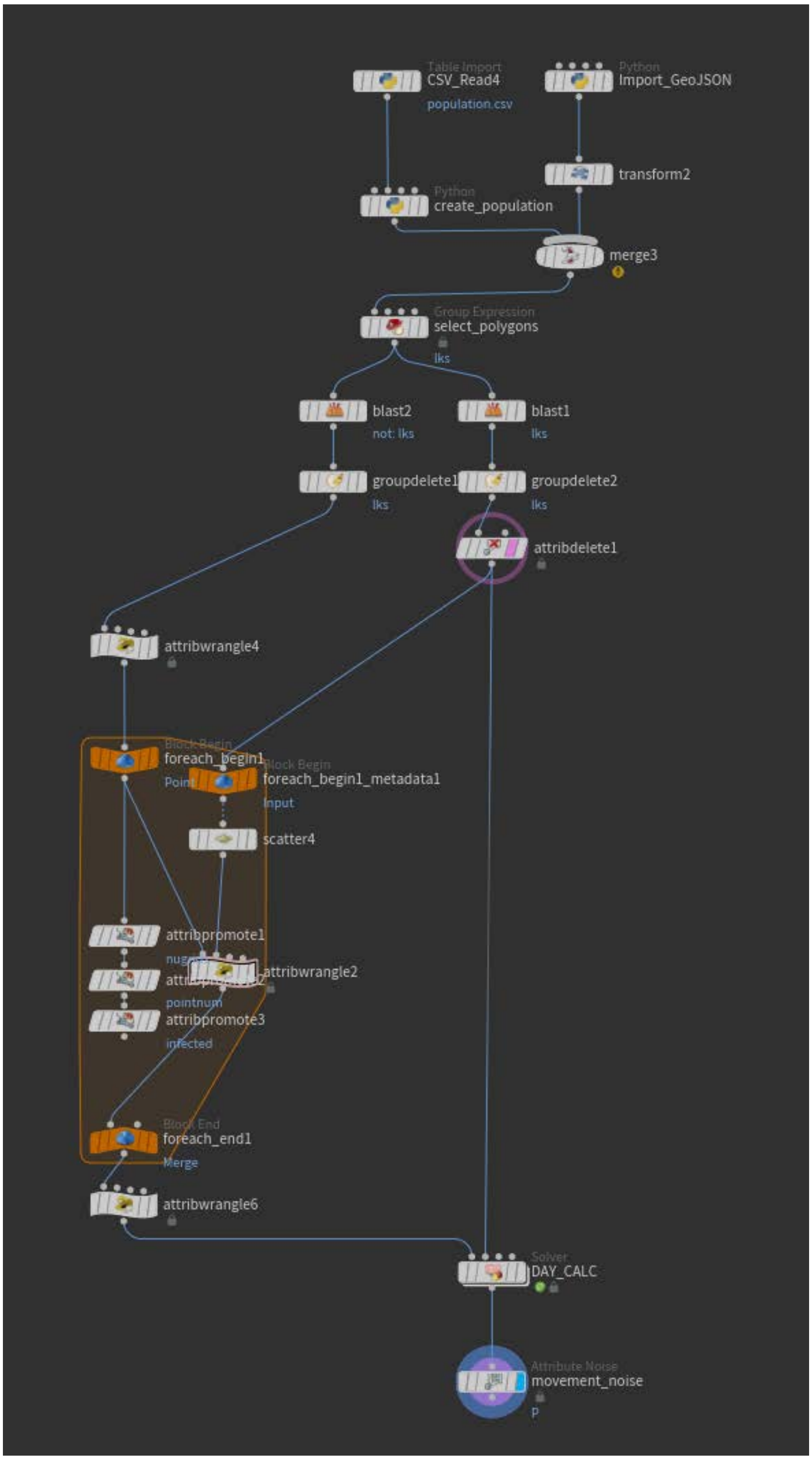


Figure 2.1: Houdini network of the initial creation of the map and the population distribution. The program works its way downward from the top through each of the nodes, modifying the data in each step.

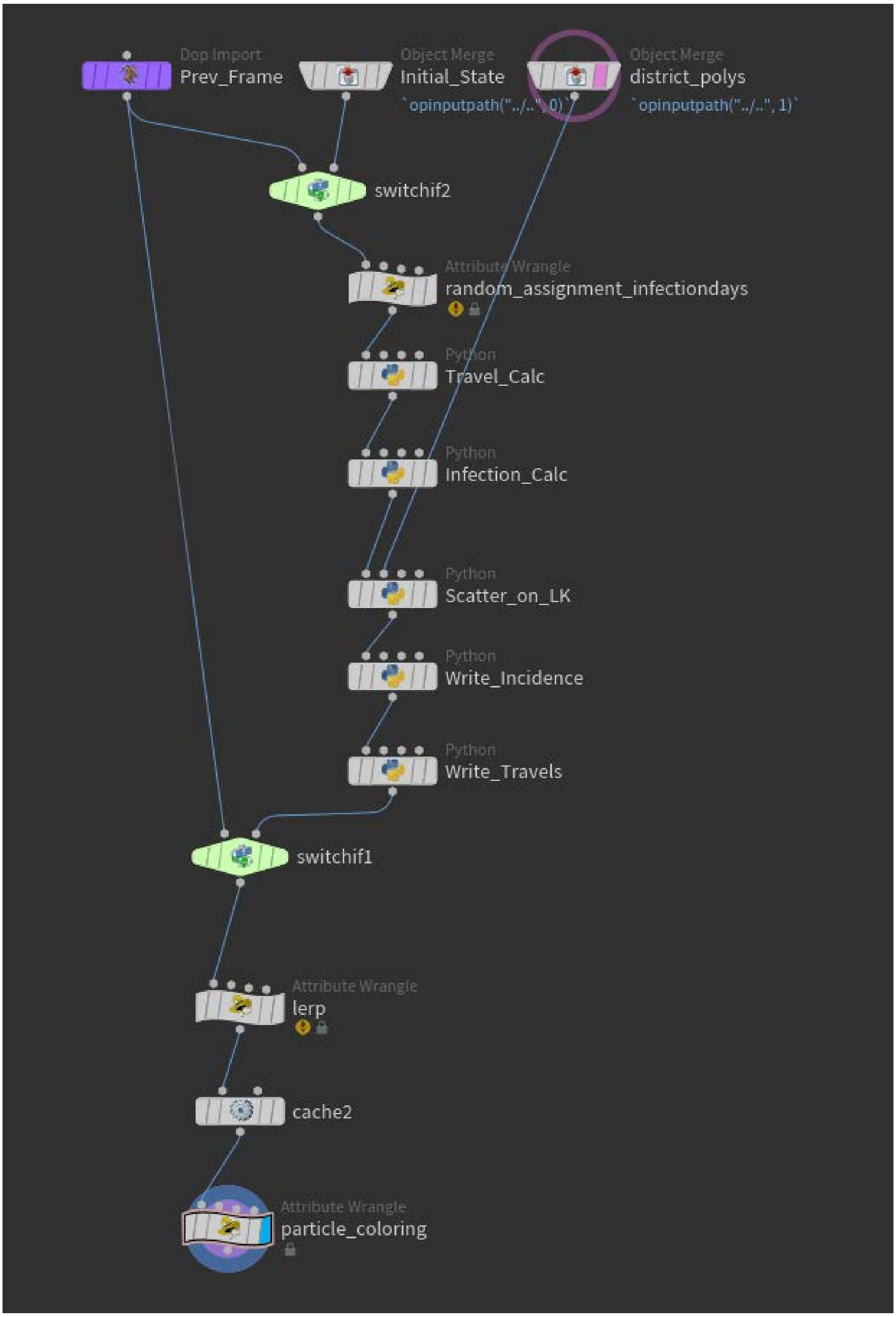


Figure 2.2: Houdini network of the daily calculation repeated for each day of the pandemic. Again, the program works its way downward from the top through each of the nodes, modifying the data in each step.

3 Results

3.1 Data & Prediction

3.1.1 Infections

After taking a detailed look at how the model works, it is time to evaluate it. We will start by comparing the ground truth data fed into the model with the summed-up choices every individual made concerning infections and use January to February 2022 as a timeline. As stated in the previous section on data, we applied a 7-day rolling average to iron out inaccuracies stemming from the days infections are reported. To follow the plot visually, we have decided to average the incidence numbers of all counties in both the input data and the simulated choices in figure 3.1.

Of course, when zooming into specific counties, we can observe an increased rate of difference from the actual data due to fewer individuals being able to average the choices out overall. Still, it follows the same trend. This can be seen when we look at Munich as an exemplary county in figure 3.2.

We can see that the model stays close to the ground truth, even when only simulating every hundredth individual in Germany (820.000 points).

In figure 3.3, we will be doing the same as in figure 3.1, except that we will be switching to the predictive model on the 32nd day, the 1st of February 2022. As expected, the simulated data stays close to the real data for a few days before differing. Hence, this prediction, akin to the naive forecast, is rather suited for predictions in the near future; further predictions and suggestions to improve this model will be discussed in the upcoming *Outlook* chapter 5.2.

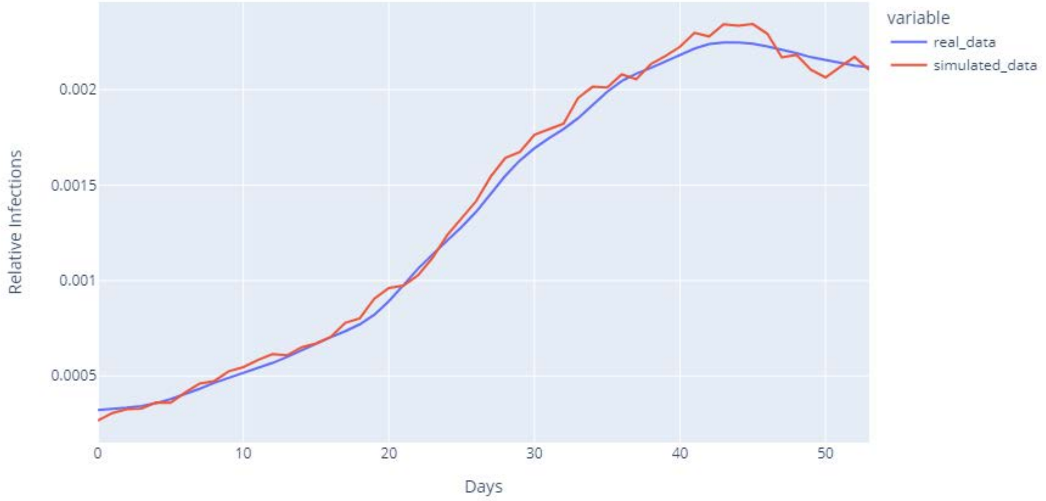


Figure 3.1: Comparison of real and simulated infection data (infections divided by population) in Germany, January and February 2022

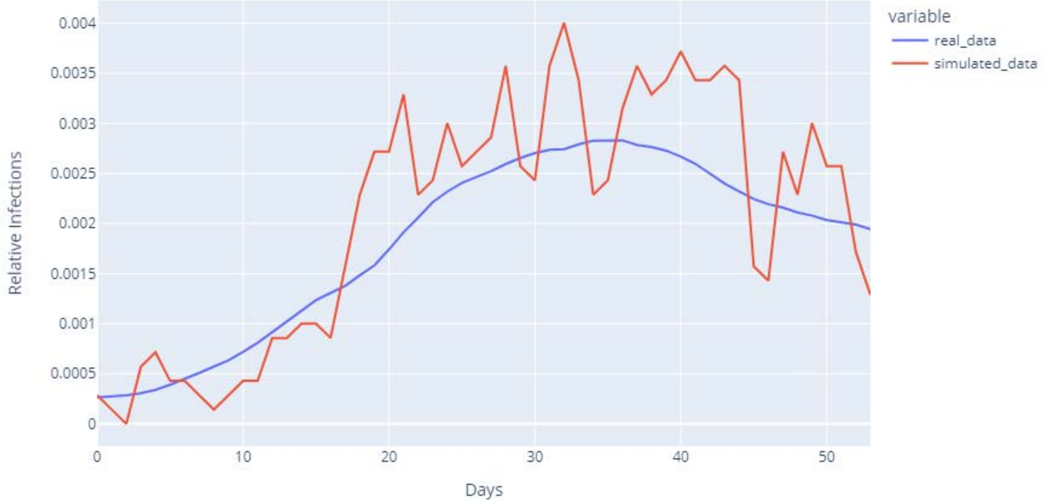


Figure 3.2: Comparison of real and simulated infection data (infections divided by population) in Munich, January and February 2022

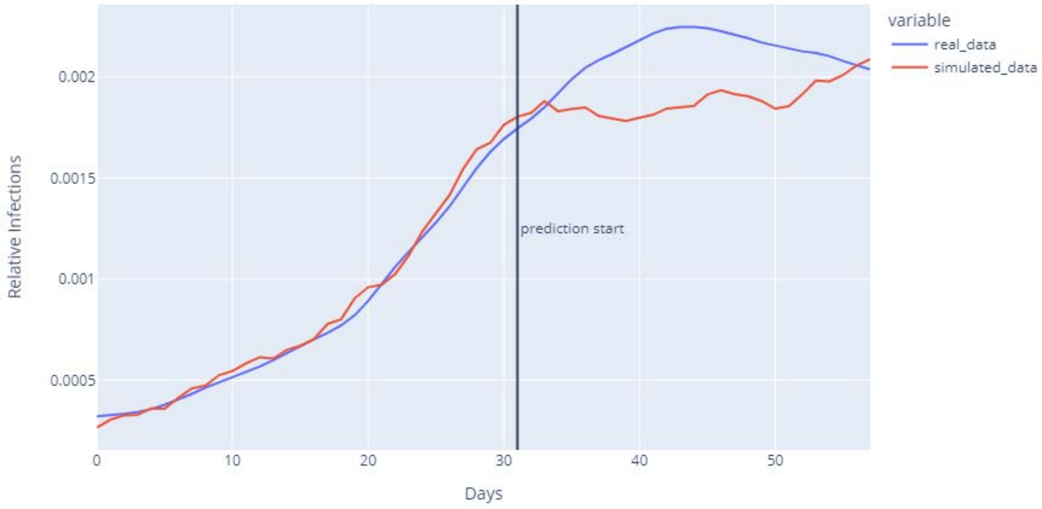


Figure 3.3: Comparison of real and predicted infection data (infections divided by population) in Germany, January and February 2022. Prediction starting on day 32

3.1.2 Travels

The next step is to inspect how the model behaves concerning travel. To evaluate them, we will assess *close* and *far* travels by calculating the mean relative travels between Munich and the adjacent county Ebersberg as examples for close travels. We calculate the sum of relative travels between Munich, Stuttgart, Frankfurt a.M., Berlin (Center), Cologne, and Hamburg as examples of far travel. As can be seen in figure 3.4, we can discern the waves stemming from weekday travels compared to weekend movements, though this pattern is harder to see in the curves representing farther travels. Again, the simulated choices stay relatively close to the ground truth data for both close and far travels.

Figure 3.5 compares the ground truth travel data to the predictive simulation. Similar to the predicted infection data, the prediction differs more than if the actual data drove it. However, because the curve presents a strong weekly wave pattern which we incorporate by implementing a prediction with respect to the simulated weekday, the predicted data stays considerably closer to the visible trend.

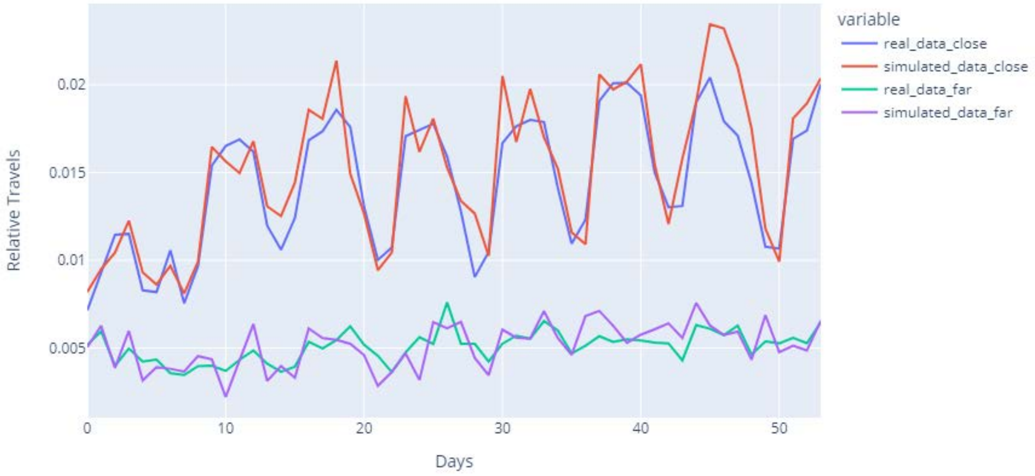


Figure 3.4: Comparison of real and simulated travel data (travels divided by population) for close and far travels, January and February 2022

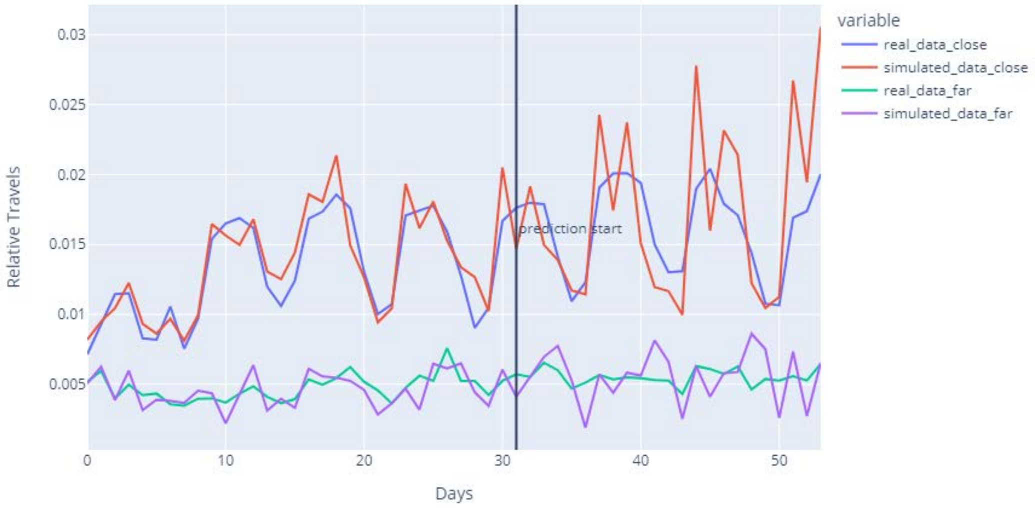


Figure 3.5: Comparison of real and predicted travel data (travels divided by population) for close and far travels, January and February 2022

3.2 Visualisation

Arguably the most crucial part of this work is its intuitive visualization of infections and travels. Figure 3.6 communicates clearly where infected individuals reside by coloring them red and enlarging them. What has to be kept in mind is the resolution of the simulation, which in this case is 100 real people per simulated particle. As a sample, we have chosen the 10th of January 2022.

The chosen resolution affects the size of the shown particles inversely, as being able to see more points leads to more clutter in clusters - areas with a high population density such as North Rhine-Westphalia and Berlin. Suppose the size of the particles did not decrease with an increase in the number of simulated individuals. In that case, we could no longer discern single particles in these mentioned clusters.

Zooming into Czechia enables us to discern movement, even in a static image. Individuals' positions are interpolated linearly, which means a traveler from Bavaria to Berlin will move through what would be Czechia; that makes it easy for us to see them against a blank background, as seen in figure 3.7. On top of that, the motion blur applied to the moving particles creates a small "trail" of the movement, which helps to perceive the existing movement even though the image shown in this thesis is still.

For interested readers, a video of the simulation and visualization displaying January and February 2022 can be accessed on this project's GitLab repository [Ste].

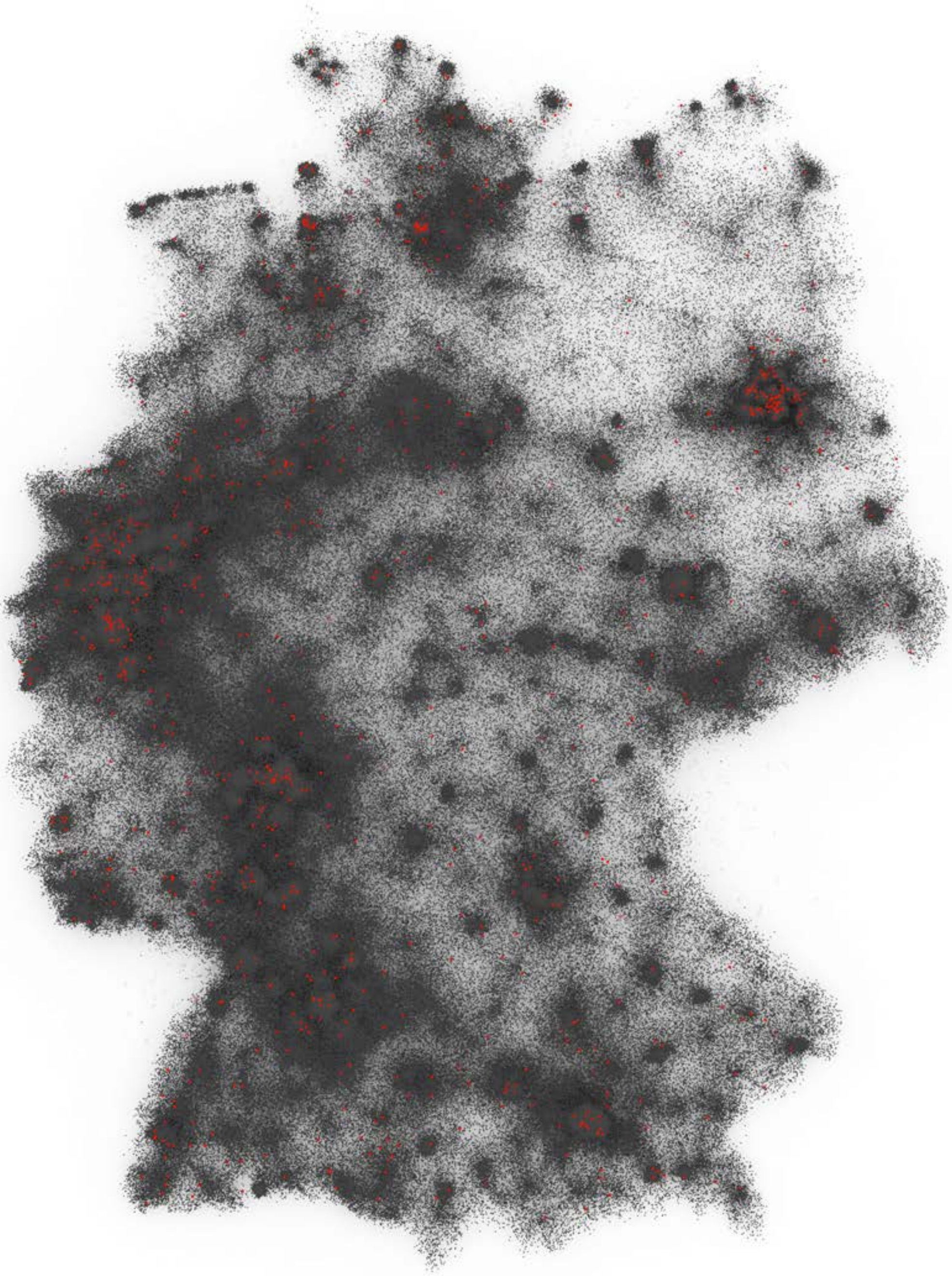


Figure 3.6: Visualization of infections and travels in Germany (January 12th, 2022). Each point represents a simulated individual, which in this case is each 100th person in Germany. Infected individuals are shown in red.

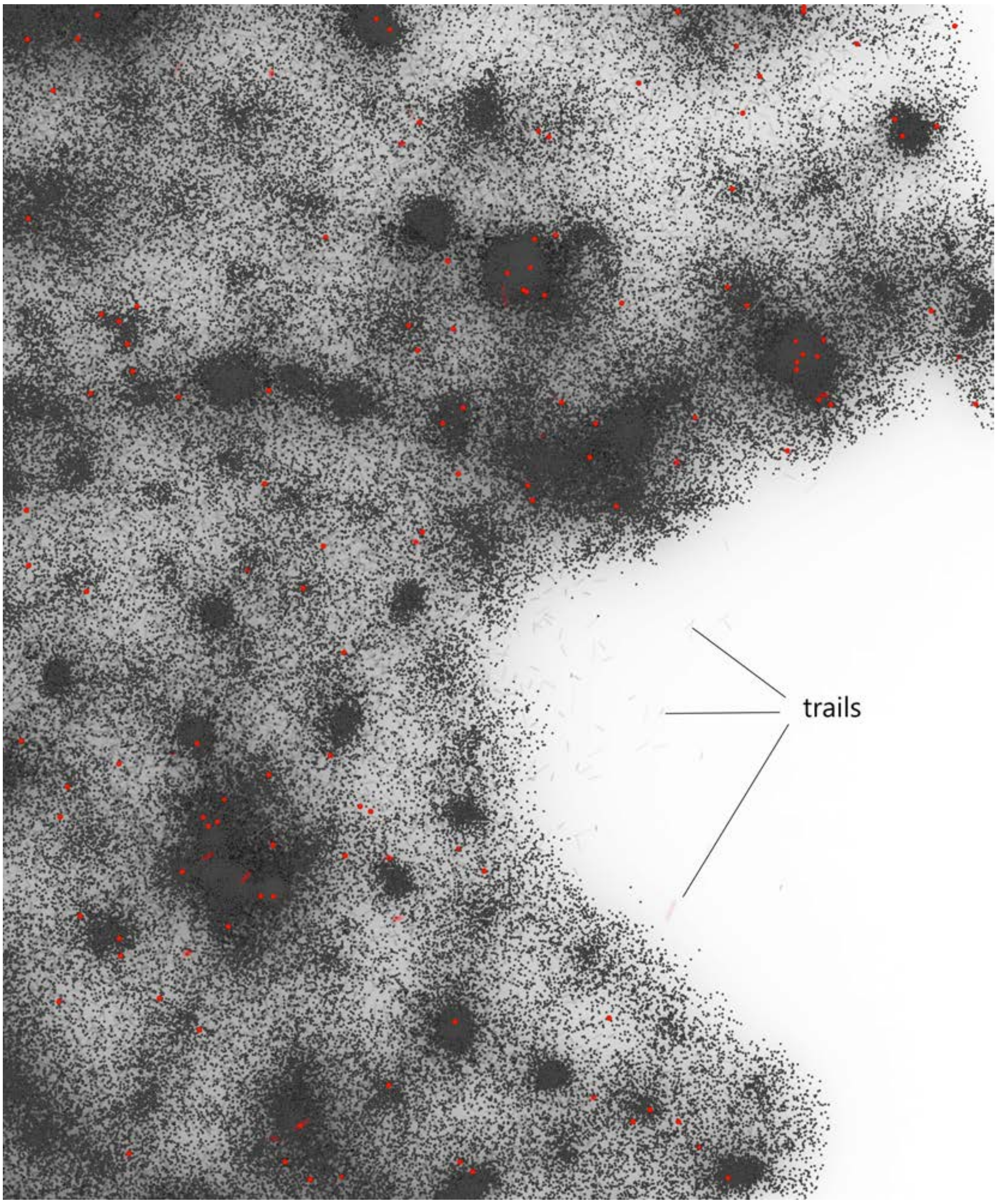


Figure 3.7: Visualization of infections and travels in Germany (January 12th, 2022; closeup). Again, each point represents a simulated individual, which in this case is each 100th person in Germany. Infected individuals are shown in red, moving particles due to them traveling to other counties can be discerned by their trail.

3.3 Optimizations

In chapter 2, we mentioned various runtime optimizations concerning data preprocessing as well as using caches for specific functions to facilitate data retrieval.

The simulation was slowed down by repeatedly searching for the matching county data for each simulated person. By splitting up the incidence data into a new file for each day, instead of having one vast file accessing times could be reduced by 30%, equivalating to 20 minutes of runtime per simulated day. Similarly, by further dividing the travel data into not only each day but also each county, accessing times in the simulation were reduced by 90%: a reduction of 30 minutes of runtime per simulated day on our machine.

A further improvement has been made using LRU caches for both functions reading in data - infections and travels - as accessing the data from scratch for every individual in a county on a specific day is unnecessary. We kept the number of entries in the cache high enough to store each county's data for a given day as RAM size was not an issue on our machine. These caches have led to a reduced runtime of 1 minute per simulated day, equivalent to a reduction of 50% at that point of optimization.

Overall, with these optimizations we could reduce the runtime from 10 minutes per simulated day for 80.000 individuals to 1 minute of calculation per simulated day for 800.000 individuals.

4 Discussion

Though Houdini is, maybe unexpectedly as it is a VFX software, a great tool in a scientific context enabling new ways to create innovative methods, simulations and visualizations, there were also a few limitations we encountered while working with it.

First, conforming to the framework Houdini gives you is necessary. When working with polygons, for example, we need to use the polygon objects given by Houdini's Python package, as there is no easy way to convert from other ways of representing them. The way Houdini represents objects in its Python environment also needs some getting used to, though the online documentation is very in-depth and could resolute questions quickly [Sofb].

A significant advantage highlighted in the chapters beforehand is the interactive, instant visual feedback one can receive inside Houdini's viewport. What you win in that regard, you lose in interactivity. While it is possible to cancel calculations between different nodes and Houdini-given custom nodes, it is impossible to interrupt the process if we run into a problem like an infinite loop inside a python node. In that case, the whole of Houdini needs to be restarted.

On top of the issue mentioned in the paragraph above, there is another downside of using Houdini as a programming environment: it is a rather expensive program. Even with remote access to a computation cluster, we are not able to simply transfer the files and run them there. It would need an extensive installation process coupled with a costly subscription to be able to use it. Once it is installed, though, due to the modular setup for this project, it is possible to run the model on that computer without the need to edit anything.

Our model is based on permanent travels, such that when an individual travels to another county, they become part of its population and diffuse into it. This might falsify the simulation, as on this scale, we would rather see commuters travel between counties and return home at the end of their workday. To include these daily commutes, the model could be adapted to simulate not only each day but two points in time per day so that commuters would return home on the same day. However, to adapt the model to this fact, we would also need more detailed travel data.

Assuredly, the topic of computational power is one often brought up in this kind of chapter, and alas, it resurfaces here as well. Python's lack of proper multiprocessing implementation means that we are limited by the core speed of the processor running the program. A newer, faster processor (and, in general, newer computational hardware) would have enabled me to run the model in a higher resolution; a 1:1 representation of simulated individual to real individual in Germany is an exciting approach for which we lacked the time or power needed.

5 Conclusions & Outlook

5.1 Conclusion

We have created a working, easily modifiable model inside of Houdini, with which we can intuitively watch the infection itself spread as well as discern movement patterns of the population. The model is correctly driven by real-world data fed into it, which can be changed instantly. Furthermore, Houdini has proven itself as a competent, feature-rich, and innovative programming environment, albeit its usual use cases of being a pure VFX software may differ from the context used in this thesis. Still, the downsides of the price and being confined to a single program, in general, should be kept in mind.

Both the project file and a rendered video displaying a simulation and visualization of January and February 2022 can be accessed on this project's GitLab repository [Ste].

5.2 Outlook

With an organized network of scripts in place which drive our model, there are a few directions in which to expand. For instance, putting a stronger emphasis on the prediction by including more data sources like census data and developing the individual-based model to include age ranges for the individuals to act accordingly. For example, children go to school on weekdays, where they have a much higher chance of infection.

The initial idea of this project was to use actual location data of people living in Germany to drive a contact-based infection simulation. This method would have enabled us to transfer the infection states across individuals based on a contact radius - for example, a few meters - and evaluate an intuitive correlation between infection transmission and contacts. Unfortunately, access to that kind of data was denied at short notice. If that data were made available again, it would help to create a convincing model with a better way to simulate and predict infections.

Ultimately, with census data as well as map data from *OpenStreetMap*, which marks residential areas, schools, offices, and other buildings [Ope], it would be possible to create an even more accurate simulation for the population of Germany. Simulated individuals could go to places with higher infection rates like schools and work, back to their home location to possibly infect the people living in proximity, which then again go to their offices. Hence, we could intuitively follow a chain reaction of infection. As one might guess from this explanation already, this would require a lot of time and effort to set up in exchange for such a detailed model, though we suppose the results will very well be worth it.

Bibliography

[Age22] UK Health Security Agency. *COVID-19 self-isolation changes: scientific summary*. 2022. URL: https://ukhsalibrary.koha-ptfs.co.uk/wp-content/uploads/sites/40/2022/01/20220110_Self-isolation_Scientific-Summary_Final-clean.pdf (see pages 3, 9).

[Bun] Statistisches Bundesamt. *Gemeindeverzeichnis-Informationssystem*. URL: https://www.destatis.de/DE/Themen/Laender-Regionen/Regionales/Gemeindeverzeichnis/_inhalt.html. (accessed: 06.07.2022) (see page 7).

[But+16] H. Butler, M. Daly, A. Doyle, Sean Gillies, T. Schaub, and Stefan Hagen. *The GeoJSON Format*. RFC 7946. Aug. 2016. DOI: 10.17487/RFC7946. URL: https://www.rfc-editor.org/info/rfc7946 (see page 4).

[De 20] Jonas De Vos. **The effect of COVID-19 and subsequent social distancing on travel behavior**. *Transportation Research Interdisciplinary Perspectives* 5 (2020), 100121. ISSN: 2590-1982. DOI: https://doi.org/10.1016/j.trip.2020.100121. URL: https://www.sciencedirect.com/science/article/pii/S2590198220300324 (see page 1).

[ESR22] ESRI. *NPGEO COVID-19 Datenhub*. 2022. URL: https://npgeo-corona-npgeo-de.hub.arcgis.com/datasets/917fc37a709542548cc3be077a786c17_0/explore (see page 7).

[Gia21] Giuseppe Giacopelli. **A Full-Scale Agent-Based Model to Hypothetically Explore the Impact of Lockdown, Social Distancing, and Vaccination During the COVID-19 Pandemic in Lombardy, Italy: Model Development**. *JMIRx Med* 2:3 (Sept. 2021). ISSN: 2563-6316. DOI: 10.2196/24630. URL: http://www.ncbi.nlm.nih.gov/pubmed/34606524 (see page 2).

[Gmb] FOCUS Online Group GmbH. *Datenschutzerklärung für die FOCUS Online - Nachrichten App*. URL: https://www.focus.de/intern/datenschutzerklaerung-datenschutzerklaerung-fuer-die-app-focus-online-nachrichten_id_7288898.html. (accessed: 06.07.2022) (see page 4).

[GMB16] Sam Grainger, Feng Mao, and Wouter Buytaert. **Environmental data visualisation for non-scientific contexts: Literature review and design framework**. *Environmental Modelling & Software* 85 (2016), 299–318. ISSN:

1364-8152. DOI: https://doi.org/10.1016/j.envsoft.2016.09.004. URL: https://www.sciencedirect.com/science/article/pii/S1364815216305990 (see page 1).

[Ins] Robert Koch Institut. *RKI Leitbild.* URL: https://www.rki.de/DE/Content/Institut/Leitbild/Leitbild_node.html. (accessed: 06.07.2022) (see page 3).

[Ins22] Robert Koch Institut. *SARS-CoV-2 Infektionen in Deutschland: Datendokumentation.* 2022. URL: https://github.com/robert-koch-institut/SARS-CoV-2_Infektionen_in_Deutschland (see pages 3, 7).

[LJD21] Fabian Lorig, Emil Johansson, and Paul Davidsson. **Agent-based Social Simulation of the Covid-19 Pandemic : A Systematic Review**. *JASSS: Journal of Artificial Societies and Social Simulation* 24:3, 5 (2021). DOI: 10.18564/jasss.4601 (see page 2).

[Ope] OpenStreetMap. *openstreetmap.org database files.* URL: https://planet.osm.org/. (accessed: 06.07.2022) (see page 26).

[Pyt] Python. *Python fuctools package documentation.* URL: https://docs.python.org/3/library/functools.html. (accessed: 06.07.2022) (see page 4).

[Sofa] SideFX Software. *Houdini product page.* URL: https://www.sidefx.com/products/houdini/. (accessed: 06.07.2022) (see page 4).

[Sofb] SideFX Software. *Houdini Python: hou package documentation.* URL: https://www.sidefx.com/docs/houdini/hom/hou/index.html. (accessed: 06.07.2022) (see page 23).

[ST20] Woo-Sik Son and RISEWIDs Team. **Individual-based simulation model for COVID-19 transmission in Daegu, Korea** (2020). DOI: 10.4178/epih.e2020042. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7644934/ (see page 2).

[Ste] Marvin Jerome Stephan. *Visualising infection spread using real-world data: project repository.* URL: https://gitlab.com/HerzogVonWiesel/visualising-infection-spread-using-real-world-data. (accessed: 06.07.2022) (see pages 19, 25).

[Tor+21] Vincenza Torrisi, Tiziana Campisi, Giuseppe Inturri, Matteo Ignaccolo, and Giovanni Tesoriere. **Continue to share? An overview on italian travel behavior before and after the COVID-19 lockdown**. *AIP Conference Proceedings* 2343:1 (2021), 090010. DOI: 10.1063/5.0048512. URL: https://aip.scitation.org/doi/abs/10.1063/5.0048512 (see page 1).

[vvz21] vvzen. *Houdini Geospatial Tools.* 2021. URL: https://github.com/vvzen/Houdini-Geospatial-Tools. (accessed: 06.07.2022) (see page 8).

[Wol] Markus Mandau und Wolfgang Pauler. *Deutschlands härtester Netztest.* URL: https://www.chip.de/artikel/Deutschlands-haertester-Netztest-So-testen-wir-die-Netzqualitaet_183946645.html. (accessed: 06.07.2022) (see page 4).

[ZK09] Caroline Ziemkiewicz and Robert Kosara, 307–326. In: *Advances in Information and Intelligent Systems*. Ed. by Zbigniew W. Ras and William Ribarsky. Springer Berlin Heidelberg, 2009. ISBN: 978-3-642-04141-9. DOI: 10.1007/978-3-642-04141-9_15. URL: https://doi.org/10.1007/978-3-642-04141-9_15 (see page 1).

Declaration of Authorship

I hereby declare that this thesis is my own unaided work. All direct or indirect sources used are acknowledged as references.

Potsdam, July 28, 2022

M. Jerome Stephan