

Final Year Project

B.Sc. CS&IT

**Simulation and Visualisation of Student Movements**

*Author: Supervisor:*

Alan Devane Dr. Owen Molloy

April 3rd, 2019

**Contents**

[Table of Figures 3](#_Toc5181706)

[Declaration 4](#_Toc5181707)

[Acknowledgements 4](#_Toc5181708)

[1.Introduction 5](#_Toc5181709)

[1.1 Project Brief 5](#_Toc5181710)

[1.2 Motivations 5](#_Toc5181711)

[1.3 Initial Questions 6](#_Toc5181712)

[1.4 Aims & Goals 6](#_Toc5181713)

[1.5 Deliverables 6](#_Toc5181714)

[2. Related Work and Background 7](#_Toc5181715)

[2.1 Monte Carlo Method 7](#_Toc5181716)

[2.2 System Dynamics Simulation 7](#_Toc5181717)

[2.3 Discrete Event Simulation 8](#_Toc5181718)

[2.4 Agent-Based Simulation 9](#_Toc5181719)

[3.Design 10](#_Toc5181720)

[3.1 Choosing Area to Model 10](#_Toc5181721)

[3.2 Simulation Modelling Software 12](#_Toc5181722)

[3.2.1 Desmo-J 12](#_Toc5181723)

[3.3.2 Anylogic 13](#_Toc5181724)

[3.3.3 MassMotion 14](#_Toc5181725)

[3.4 Software Package Decision 15](#_Toc5181726)

[3.3 Design of chosen Area 16](#_Toc5181727)

[3.3.1 Designing the 2D Model 16](#_Toc5181728)

[3.3.2 Designing the 3D Model 17](#_Toc5181729)

[4 Implementation 18](#_Toc5181730)

[4.1 Technologies 18](#_Toc5181731)

[4.1.1 Java 18](#_Toc5181732)

[4.1.2 AnyLogic 19](#_Toc5181733)

[4.2 Designing the Simulation 20](#_Toc5181734)

[4.2.1 Implementation 22](#_Toc5181735)

[4.2.2 Logic Explained 24](#_Toc5181736)

[4.3 Features/Code Snippets 26](#_Toc5181737)

[5 Testing and Results 29](#_Toc5181738)

[5.1 Iteration 1 – Evacuation Model 29](#_Toc5181739)

[5.2 Iteration 2 – Walkways/Paths version 31](#_Toc5181740)

[5.3 Analysis 33](#_Toc5181741)

[5.3.1 Congestion Analysis 33](#_Toc5181742)

[5.3.2 Evacuation Analysis 34](#_Toc5181743)

[6 Conclusion 35](#_Toc5181744)

[6.1 Evaluation 35](#_Toc5181745)

[6.2 Future Work 35](#_Toc5181746)

[6.3 Lessons Learned 36](#_Toc5181747)

[Bibliography 36](#_Toc5181748)

[Source Repository 37](#_Toc5181749)

[AnyLogic Cloud: Link to Run Model 37](#_Toc5181750)

## **Table of Figures**

[Figure 1.1 Stock and Flow Model 8](#_Toc5182031)

[Figure 2 Collection of I.T. Building images 10](#_Toc5182032)

[Figure 3 iPhone pictures taken of pedestrian bridge 11](#_Toc5182033)

[Figure 4 Sample Desmo-J Project 12](#_Toc5182034)

[Figure 5 Desmo-J Histogram Data 12](#_Toc5182035)

[Figure 6 AnyLogic Statistics Demo 13](#_Toc5182036)

[Figure 7 Airport 3D Model – AnyLogic 13](#_Toc5182037)

[Figure 8 MassMotion 3D Train Station 14](#_Toc5182038)

[Figure 9 MassMotion Statistics Output 14](#_Toc5182039)

[Figure 10 Software Package Comparison 15](#_Toc5182040)

[Figure 11 AutoCAD Drawing of building 16](#_Toc5182041)

[Figure 12 Adding Walls to 2D model 17](#_Toc5182042)

[Figure 13 3D Model - Initial Build 17](#_Toc5182043)

[Figure 14 Overcrowding in first simulations 22](#_Toc5182044)

[Figure 15 Agent Speed 23](#_Toc5182045)

[Figure 16 From Concourse Logic 24](#_Toc5182046)

[Figure 17 From Outside Logic 25](#_Toc5182047)

[Figure 18 Lecture Ending Logic 25](#_Toc5182048)

[Figure 19 Iteration 1 - 2D View 29](#_Toc5182049)

[Figure 20 Iteration 1 - 3D View 29](#_Toc5182050)

[Figure 21 Statistics View 30](#_Toc5182051)

[Figure 22 Menu View 31](#_Toc5182052)

[Figure 23 Adding Pathway's 31](#_Toc5182053)

[Figure 24 Iteration 2 - With Pathways 32](#_Toc5182054)

[Figure 25 Iteration 1 Congestion 33](#_Toc5182055)

[Figure 26 Iteration 2 Congestion 33](#_Toc5182056)

[Figure 27 Evacuation Graphs 34](#_Toc5182057)

**Declaration**

I declare that this project contains my own work except where otherwise stated.

Signed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Alan Devane

## **Acknowledgements**

I’d like to thank my supervisor Dr. Owen Molloy for his guidance and help during this project. His attention to detail and expertise greatly benefitted me in the completion of this project.

I’d also like to thank my classmates who helped test the project and advised me throughout. Finally, I would like to thank my family for encouraging and motivating me during this project.

# **1.Introduction**

## **1.1 Project Brief**

The university's ever-expanding student population explodes onto the corridors every hour at the same time and must vie for space as they beat a path to the next lecture. This project will use a simulation system such as DESMO-J or AnyLogic to simulate and visualise these movements. This system will allow us to experiment with ideas such as traffic lanes, variable lecture times and lecture locations. It may involve the use of specialise libraries for visualisation of the simulation outputs

## **1.2 Motivations**

Simulation systems act as a substitute for physical experimentation, where computers calculate the results/outcomes of some physical phenomenon. Using simulation software, one can create a mathematical model which contains an approximation of the parameters of the physical model, representing the physical model in virtual form. The conditions one would like to experiment are applied to the model and simulation begins. Simulations help people visualise experiments that would be too costly to replicate in real-world situations. Simulations can often be even more realistic than traditional experiments. They also allow for faster speeds than real time. They can help reduce costs, increase the quality of systems and products, while documenting lessons learned along the way.

As stated in the project brief, an ever-expanding student and world population, mass gatherings are becoming more and more frequent. Everyday life in cities is becoming increasingly crowded with people. With more cost-effective transportation, creation of better solutions to alleviate crowded areas and make them safer and more efficient for pedestrians is a huge area of interest right now. Simulation software can help visualize, plan and optimize these flows to prevent such gatherings. Examples of areas that they are routinely used in are, for planning of major events, and also for optimizing transport systems, assessing building evacuations, optimizing the organization of airports and train stations.

Being a travel/sport fanatic, I can see from my experiences the need for more active and accurate planning for events, train stations, concerts etc. Crowd safety and control are areas that require detailed planning well in advance. Stampedes and squashes are becoming increasingly evident and a change must occur soon to prevent more injuries/loss of life. “Safety must be paramount issue in such events not the profitability. The safety can be ensured by managing crowd properly*”* (Bolia, 2015)

## **1.3 Initial Questions**

On completion, this project should be able to answer:

• How can a simulation/visualisation help an organisation monitor and improve pedestrian traffic?

• Are there any models/previous simulations to learn from?

• Where are the problem areas where this hourly congestion is seen?

• What effect would varying lecture times have on the traffic volumes?

• Would dedicated traffic lanes decrease congestion?

## **1.4 Aims & Goals**

My target outcome for this project is to make significant advancements in improving pedestrian traffic flow and congestion in the college. I hope to run numerous scenarios and collect the results of these. Based on these results I hope to see what changes can be made in the college to help optimize the best lecture times, lecture locations and possible walk on certain side lanes.

Reducing the mass gatherings, congestion areas and delay times are at the top of my list of aims for this project. If I can improve these three areas, I will be satisfied that I have met the main requirements of this project.

Visualising the abovementioned areas in a clean, understandable form is also top priority. I would for this project to make total sense to anyone who views it, even without prior knowledge of the building, area or congestion issues that are currently an issue,

## **1.5 Deliverables**

As a result of the work on this project I hope to have designed and created the following deliverables:

* A working model of student movements in the college
* A running simulation of students(agents) moving throughout the model area
* Statistical analysis of movements (Time/Speed/Number of Students)
* Visualisations of evacuations and walk lanes

# **2. Related Work & Background**

## **2.1 Monte Carlo Method**

Monte Carlo methods refer to a group of computational algorithms that used repeated random sampling to obtain statistical results. The idea is to use randomness to solve problems that might be deterministic in principle. Monte Carlo methods are mainly used in three problem classes: [optimization](https://en.wikipedia.org/wiki/Optimization), [numerical integration](https://en.wikipedia.org/wiki/Numerical_integration), and generating draws from a [probability distribution](https://en.wikipedia.org/wiki/Probability_distribution). (Botev, 2014)

Monte Carlo algorithms tend to be simple, flexible, and scalable. When applied to physical systems, Monte Carlo techniques can reduce complex models to a set of basic events and interactions. Monte Carlo–based predictions of failure, [cost overruns](https://en.wikipedia.org/wiki/Cost_overrun) and schedule overruns are routinely better than human intuition or alternative "soft" methods. (Hubbard, 2009)

Monte Carlo simulation performs risk analysis by building models of possible results by substituting a range of values—a probability distribution—for any factor that has inherent uncertainty. It then calculates results over and over, each time using a different set of random values from the probability functions.

By combining the distributions and randomly selecting values from them, it recalculates the simulated model many times and brings out the probability of the output. Because of the statistics and data that Monte Carlo simulation creates, it can be quite easy to build graphs of different outcomes and their chances of occurring. Results show not only what could happen, but how likely each outcome is.

## 2**.2 System Dynamics Simulation**

System dynamics is an approach to understanding the [nonlinear](https://en.wikipedia.org/wiki/Nonlinearity) behaviour of [complex systems](https://en.wikipedia.org/wiki/Complex_system) over time. System dynamics is a highly abstract method of modelling. It isn’t concerned with the fine details of a system, such as the individual properties of people, products, or events, and produces a general representation of a complex system. These abstract simulation models may be used for long-term, strategic modelling and simulation.

Stock and flow diagrams along with feedback loop structures are two examples of system dynamics modelling. In the model below, potential adopters and adopters are the stocks and the flows are all the things that can affect the stocks. For example, as “New adaptors” goes up, the number of “Adopters” goes up.

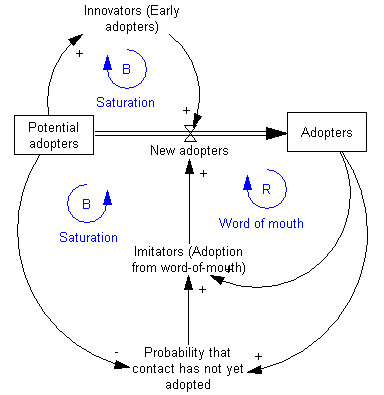


Figure 1.1 Stock and Flow Model

## **2.3 Discrete Event Simulation**

In discrete event simulation the model is discrete, and the simulated clock always jumps from one event time to the most imminent event time. At each event time the corresponding action (state change) is performed, and simulated time is advanced to the next time when some action is to occur. Thus, discrete event simulation assumes that nothing happens between successive state changes.

There are basically three approaches that can be used for discrete event simulation: the event-based, the activity-based and the process-based approach (Kreutzer, 1980)

*(1) The event-based approach*

Event based is the simplest implementation style of discrete event simulation because it can be implemented in any programming language. In this process the model contains a collection of events. Each event represents a state change and in turn is responsible for scheduling other events that were dependent on that event.

Each event has associated an event time and some actions to be executed when the event occurs.

*(2) The activity-based approach*

This type of model consists of a collection of activities. Each activity models some time-consuming action performed by an entity. A starting condition, actions to be executed once started, activity duration and executions on finish, are the conditions associated with each activity. Although it is a quiet straight forward approach, compared to the event-based approach, it normally suffers from poor efficiency.

*(3) The process-based approach*

In the process-based approach the model consists of a collection of processes. Each process models the life cycle of an entity and is a sequence of logically related activities ordered in time. Since processes resemble objects in the real world, process-based simulation is often easy to understand. Implementation, however, is not easy and execution efficiency may be poor if the implementation is not done properly

## **2.4 Agent-Based Simulation**

Agent-based simulation, unlike system dynamics and discrete event modelling, focusing directly on individual objects, their behaviour and their interaction. It’s model is a set of interacting objects that reflect relationships in the real world. The result generated from agent-based simulation models are a natural advancement in the understanding and analysing the complexity of today’s business and social systems.

Agents are said to have behaviours, or pre-set rules, and interactions with other agents, which then influences their behaviour and how they react to different events/surroundings. Agents are modelled individually allowing us to observe the true randomness and diversity of their actions. Patterns, behaviours and structures emerge that were not explicitly programmed into the models but take place due to the interactions with other agents in the population.

Common agent interaction protocols include contention for space and collision avoidance; agent recognition; communication and information exchange; influence; and other domain-or application-specific mechanisms*.* (M.Macal, 2009)

# **3.Design**

## **3.1 Choosing Area to Model**

When I first met with my supervisor, we both agreed that the I.T building was an area that we saw daily congestion and delays occurring. These delays were not just on random days, or during once-off events/occasions. These mass gatherings, queues and delays are evident everyday at this location, particularly at ten to the hour as this is when lectures end. Currently, we see approximately 300 students empty out, in a 5/10-minute period, from lecture halls, while also seeing the same amount trying to enter the lecture halls before the hour. This ten-minute period causes chaos daily between the times of 09:50 am to 16:50.

We both agreed that we had experienced this hinderance on regular occasions and that if traffic congestion is reduced in any way, that it would improve both our daily trips to and from the I.T building each day. We settled on the I.T building as the area to model and focused on the pedestrian bridge as the target are for the simualtion.

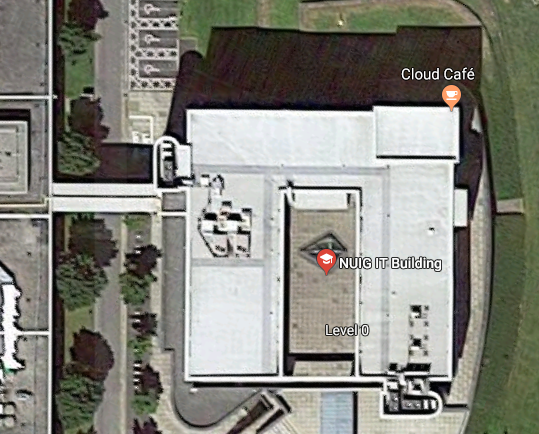
 

Figure 2 Collection of I.T. Building images

These images were all sourced from Google Maps and Google Street View (Maps, 2019)

The two photos on the left(above) display a top-down map and plan of the building to be modelled. The two photos on the right(above) of show the location of the pedestrian bridge. This bridge is the link between the busy concourse and physics labs to the I.T building itself. The main area where these mass gatherings/delays occur is just as one enters the I.T building from the Concourse via this bridge.

Figure 3 iPhone pictures taken of pedestrian bridge

There are several reasons for these delays:

* Students stopping to talk to a friend can create a blockage
* Approximately 300 students enter/exit the building each hour
* No walk on one side lanes occur
* If one door is closed (See Figure 3) even further delays
* The pedestrian bridge realistically only allows for three individuals to walk side by side at a time.

Given these facts, both my supervisor and I agreed on this location as the final model area. This are would have to be designed intricately to show accurate results and visualisations. We spoke of the possibility of making a 3D model, but a 2D model was our main priority to begin with.

## **3.2 Simulation Modelling Software**

## **3.2.1 Desmo-J**

DESMO-J is an object-oriented framework targeted at programmers developing simulation models. The acronym "DESMO-J" stands for "Discrete-Event Simulation and Modelling in Java".

It features abstract classes that can be adapted to model-specific behaviour, like model, entity, event, and simulation process. It comes with ready-to-use simulation infrastructure comprising scheduler, event list, and simulation time clock. It also provides tools for the generation of reports and traces of a simulation run (See Figure 3.8)

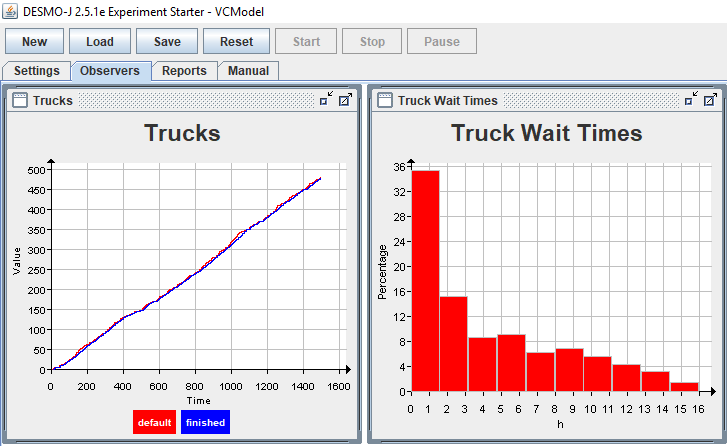


Figure 4 Sample Desmo-J Project

Desmo-J uses an outdated GUI and has not been updated since early 2015. It does not allow for 3D simulation and can be hard to visualise runs/simulations. Desmo-J, however, allows for excellent reports and result visualisations.

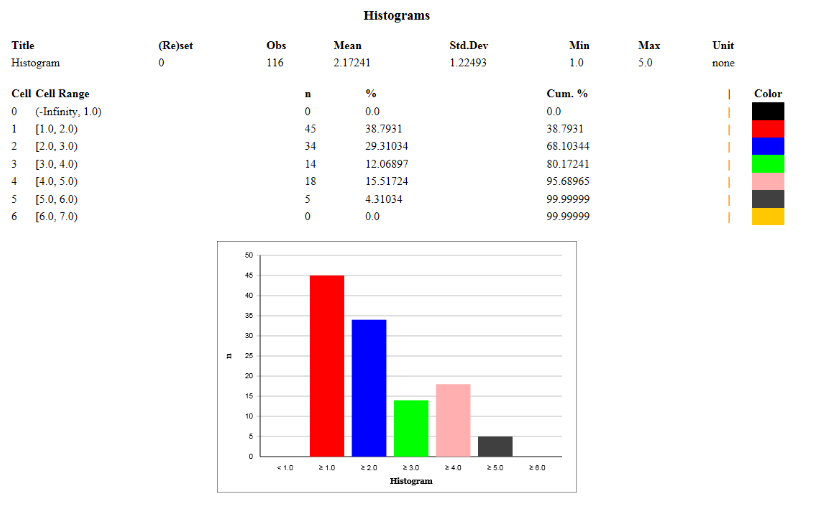


Figure 5 Desmo-J Histogram Data

## **3.3.2 Anylogic**

AnyLogic simulation software is multimethod simulation model tool that allows both 2D and 3D models. It has support for agent based, discrete event and system dynamics simulation methodologies. It is a very popular tool worldwide and is used to model areas such as manufacturing, healthcare business processes and pedestrian dynamics.



Figure 6 AnyLogic Statistics Demo

The above is an example of what a 2D model looks like during a simulation run in AnyLogic’s software, where a simple model of a supply chain consisting of a retailer, a wholesaler, and a factory is taking place. Live statistics and graphics are shown underneath.

AnyLogic also allows for 3D simulation and supports the importing of CAD drawing making animations more realistic and life-like. The image below shows a 3D model of an Airport check-in area.

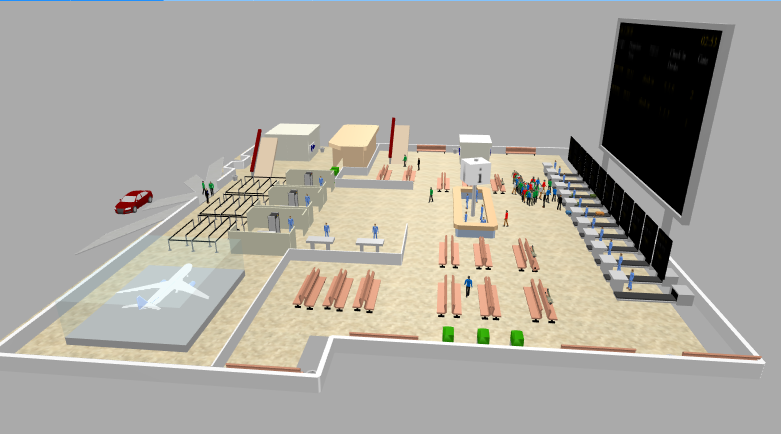
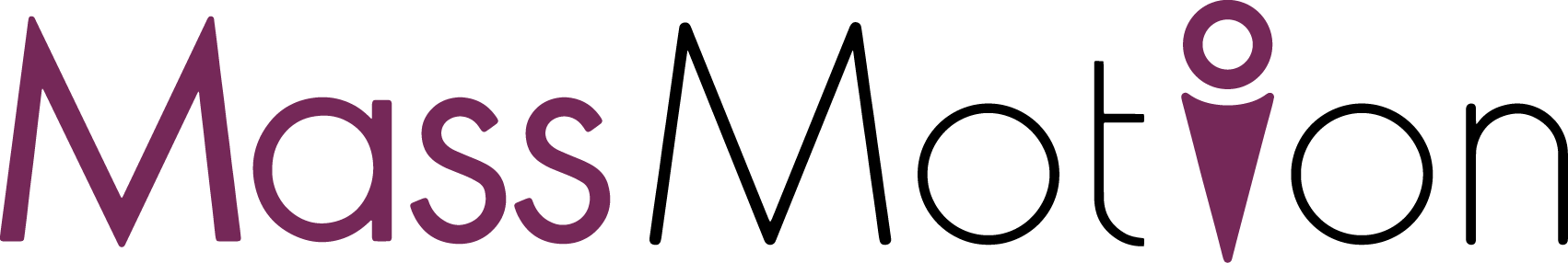


Figure 7 Airport 3D Model – AnyLogic

## **3.3.3 MassMotion**



Mass Motion’s ability to model emergent crowd behaviour makes it invaluable for exploring solutions. The software also has a rich feature set to enable its users to simulate environments that are defined using BIM or other 3D modelling tools.  It is used for simulating events with huge numbers of attendees such as concerts and sporting events.

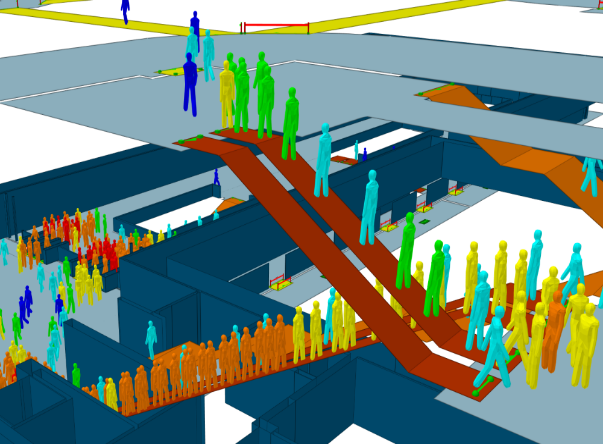
 

Figure 8 MassMotion 3D Train Station

MassMotion allows you to report in a variety of ways with visually engaging graphics, tabular and data-driven outputs, all presented as easy-to-understand graphics and visuals. Its stand out feature is how quickly it is able to run a large number of models with 100,000’s of agents.

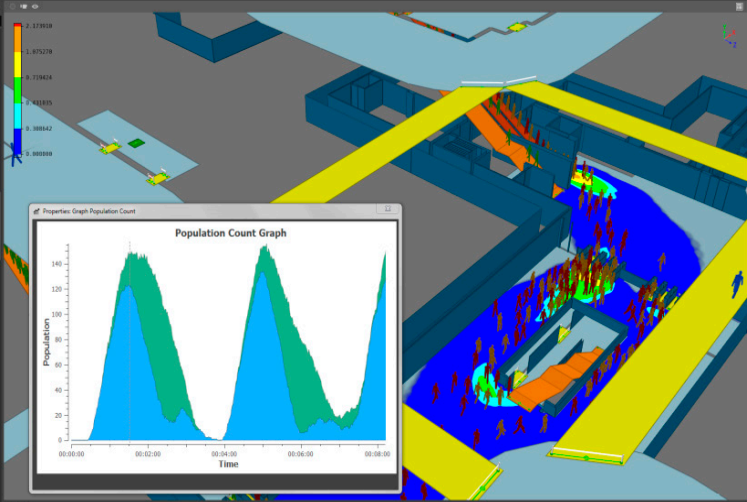


Figure 9 MassMotion Statistics Output

## **3.4 Software Package Decision**

The software package I have chosen for this project is AnyLogic. I have compared the three packages above and conclude that AnyLogic will work the best with the given brief. The main reason was its use of “agent simulation” which is the best environment to simulate pedestrian traffic and visualizations.

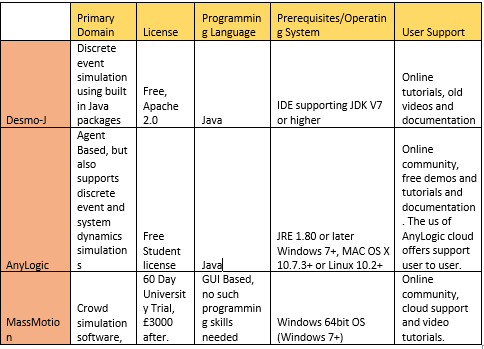


Figure 10 Software Package Comparison

I chose AnyLogic as I had access to a free student license which gave me the ability to use almost all of the features, which was sufficient enough for implementing the design I wanted for the model.

Although MassMotion, graphically, has better features and outputs, it costs a hefty £3000 to purchase which would not be feasible for a project like this. However, it is a software package that I would love to try out in the future as its features are unrivalled when it comes to large scale simulations.

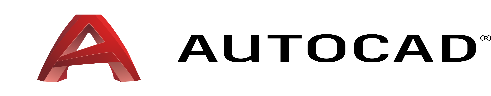
Desmo-J was severely outdated and did not offer any of the features that I would have liked for presenting this project in the way I wanted.

I was happy with my choice of package and was happy to start using the software to create the model and to see the simulations take place.

## **3.3 Design of chosen Area**

Now that the model area was chosen, I had to carefully plan how I would go about the design process for this. I gathered all plans and maps I could find for the I.T building and concluded that a trace of the map features above would be enough for a 2D version of the building. The 3D model would be more complicated, and this would have to be created using a 3D building tool.

## **3.3.1 Designing the 2D Model**



I used AutoCAD’s free student edition on the college PC’s to create the 2D model. Using the map used above for the top down view of the building I was able to pull it into the software for accurate tracing. I attached the raster image using the IMAGEATTACH command (Ribbon-> Insert -> Reference pane -> Attach) and then traced it.

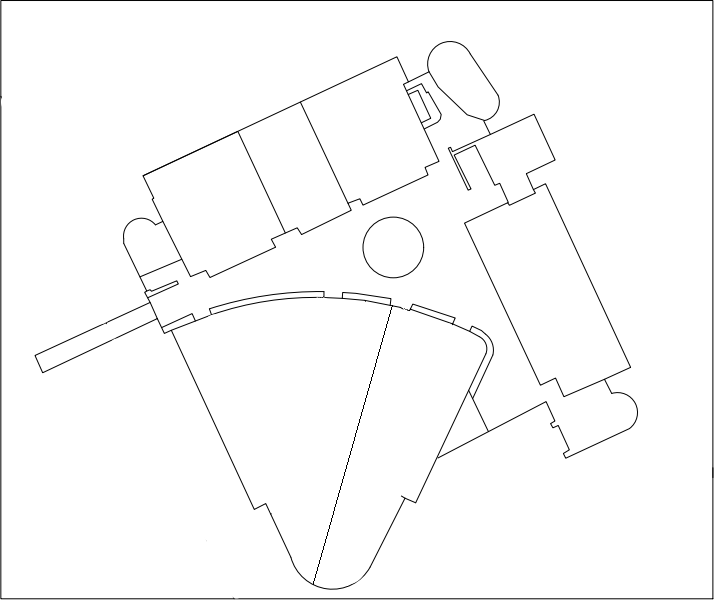


Figure 11 AutoCAD Drawing of building

I used the “Line” command to draw the straight lines and the “Arc” one, to draw circular arcs/lines. I feel it came out very well and gives an accurate two-dimensional representation of the building. I feel it will suit the 2D simulation because of the white space will show off the agents well.

## **3.3.2 Designing the 3D Model**

To design the 3D model, I used Any Logic’s built in Pedestrian Library space mark-up tool. When double clicked, walls and boundaries can be were added to the 2D image created in the previous step. The images below show the walls being added to the image, on the left, and they are added on the image on the right. The image on the right gets rid of the 2D image and what is left is the outline of the walls.

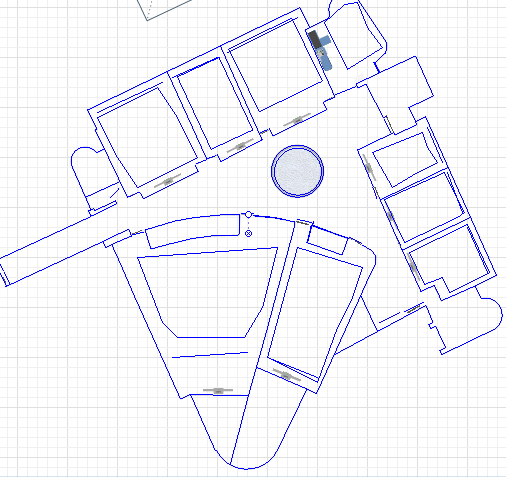
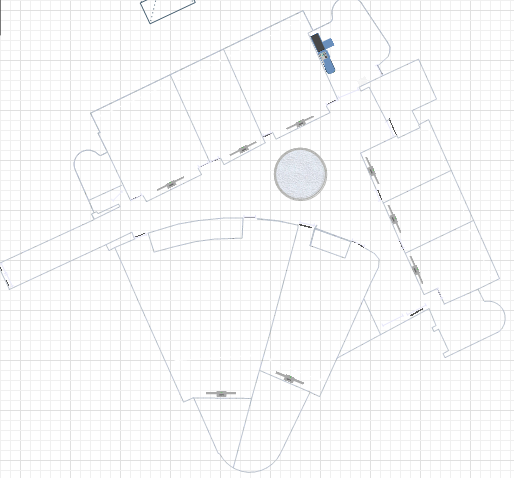
 

Figure 12 Adding Walls to 2D model

Next , to visualize these walls and boundaries in 3D, a 3D window is added to the view (view3D). The heights, thickness and colour of these walls are all customizabale through the “Position and size” option in the wall properties.

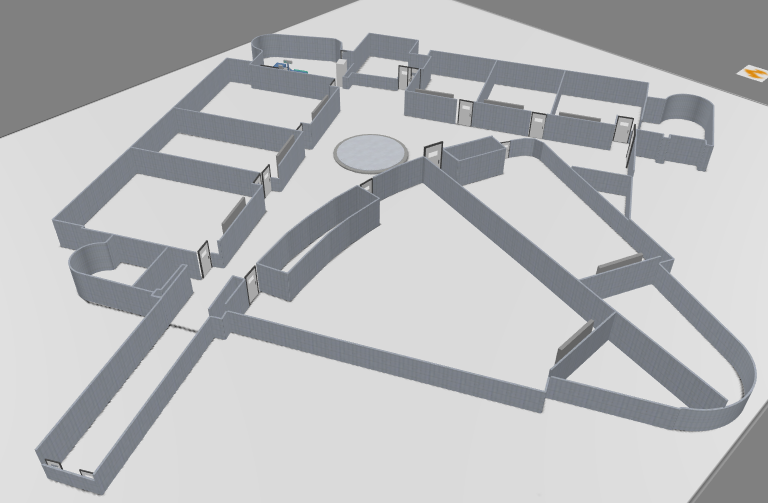


Figure 13 3D Model - Initial Build

# **4 Implementation**

## **4.1 Technologies**

## **4.1.1 Java**



Java is a general-purpose computer programming language that is [concurrent](https://howtodoinjava.com/java-concurrency-tutorial/), class-based and specifically designed to have as few implementation dependencies as possible. It follows an object- oriented programming paradigm. Java is free to access and can be run on all platforms. Java is a concurrent whereby you can run/execute multiple statements instead of sequentially executing it. Compiled code can run on all platforms that support JAVA.

One of the most significant advantages of Java is its ability to move easily from one computer system to another. The ability to run the same program on many different systems is crucial to World Wide Web software, and Java succeeds at this by being platform-independent at both the source and binary levels.  It is normally compiled to the bytecode instruction set and binary format defined in the Java Virtual Machine Specification.

Java remains popular with software managers due to its ease of learning and ubiquity. Seasoned developers take advantage of extensive Java online resources and communities to streamline application development and simplify troubleshooting, reducing time to market and increasing quality. For both business and technical reasons, Java will continue to be an essential technology in software groups developing enterprise applications.

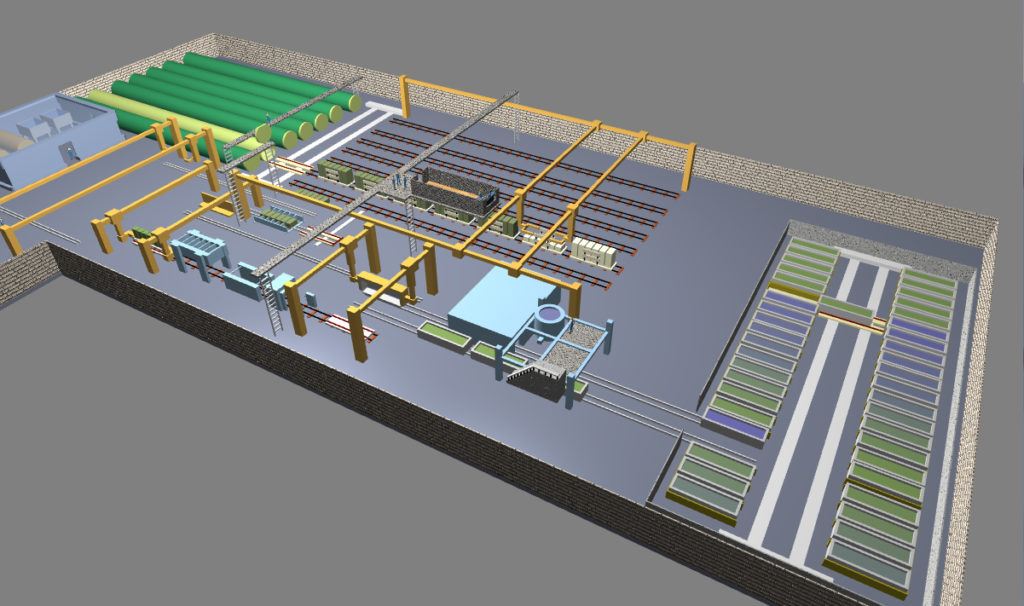
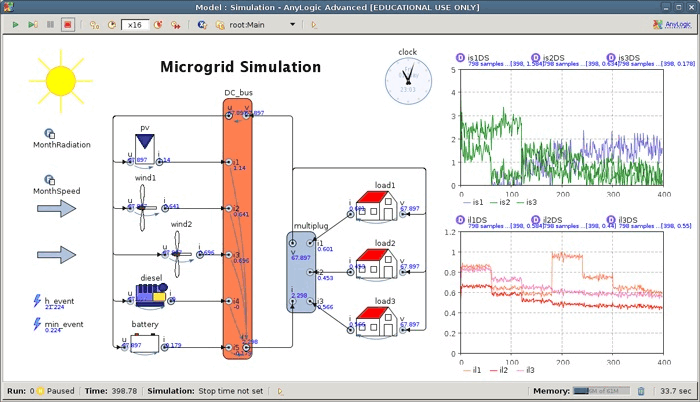
JAVA will be the main programming language used in this project, primarily because AnyLogic models are coded in JAVA. AnyLogic could be used without JAVA, but however to create a complex pedestrian model my JAVA skills will be tested so a good JAVA understanding will be needed. JAVA ties in with AnyLogic software, as any model developed in AnyLogic is fully mapped to into Java code. These models and projects become completely independent standalone Java applications. AnyLogic models can run on any Java-enabled environment.

Model simple, think complicated (PIDD, 1996)

## **4.1.2 AnyLogic**

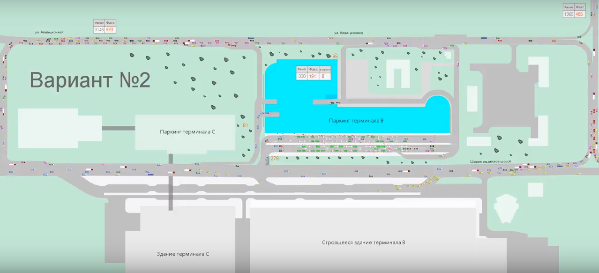


AnyLogic is a flexible, multi-method modelling tool capable of producing a variety of simulations. From supply chain and logistics, to manufacturing and market analysis, to disease and social norm dispersion, AnyLogic allows businesses, researchers, and public policy makers to evaluate decisions before they're implemented in practice.

It’s aesthetically pleasing visual outputs make it a market leader in simulation software and it has been used to model major airports and subways tations worldwide.

In a simualtion created by Sheremetyevo Airport to help visualize traffic congestion in their newly created terminal B, the team were able to see what traffic plan worked best using AnyLogic.

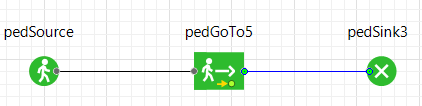


“With the obtained results from the model experiments, the customer could select the optimum plan of traffic planning at the airport terminal square, which would later be implemented in the construction project.” (AnyLogic)

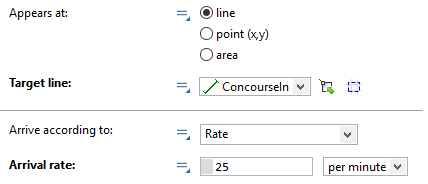
I think AnyLogic will be a great fit for this project as it has an excellent pedestrian library that will help assess the ability of the I.T. Building at NUIG to cope with planned loading and comply with safety requirements

## **4.2 Designing the Simulation**

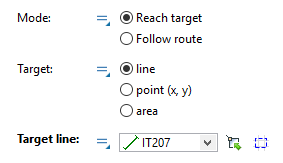
Now that the model has been designed, it was time to add agents to the simulation. For this I used AnyLogic’s pedestrian library in the Pallete section of the GUI. It allows you to chose multiple “blocks” and add them to your model. Below is the first set of blocks I added to my model.



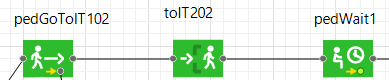
This set of blocks creates a source of pedestrians, then sends them to a set location and finally removing them from the model. The “pedSource” requires a starting location, which can be an area, line or a point(x,y) , and an arrival rate or schedule to add the agents to the model. I wanted these agents to start at the concourse side of the building and to stream in at a rate of 25 agents per minute.



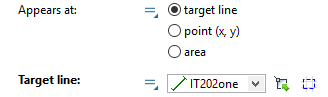
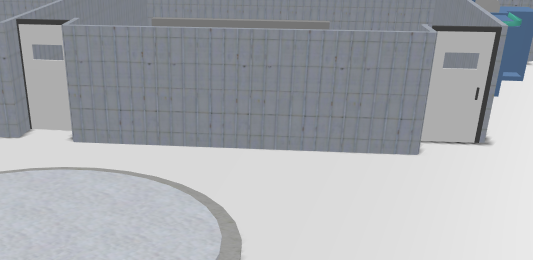
The “PedGoTo5” requires an end location which again can be an area, line or a point(x,y), but also requires how it is supposed to get there. “Reach target” will get to the desired target using the shortest path, while “Follow route” will use a user created “pathway” and try follow that route as close as possible. In this case I wanted the agents to got to the IT207 room, which is defines by the line “IT207”.



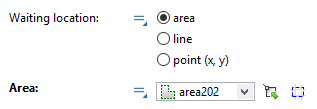
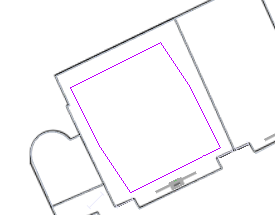
The next step was to try and get the agents to wait at a location for a certain time (i.e. 50-minute lectures). This was achieved by using the same “pedGoTo” from the previous step along with a “pedEnter” and a “pedWait” from the pedestrian library.



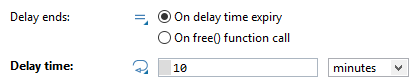
The “pedEnter” combines with the “pedGoTo2 as once it reaches the target line, it enters through that line, as if it is going through a door. The only parameter required is the name of the line it is entering. I added doors at these points to make the visualization more realistic.

The “pedWait” specifies at a wait time at a specific location. This location can be an area or point again, and combine attracttors inside an area to show agents where in that area to wait. The location is designed on the top 2D level of the model. A capacity can be added to the location also

Agents can be set free from the location after a certain time(i.e. lecute end) or using the free() function call. The free function call can be triggered during an evacuation or similar event.



## **4.2.1 Implementation**

As with the development of any project, one can run into many errors/issues. The main problem I had for the first couple of months was overcrowding and speed of agents. This was mainly evident in the first iteration of the project where all the agent’s routes were hardcoded into the model. This meant there was no randomness and agents would repeat the same actions repeatedly during simulation runs.

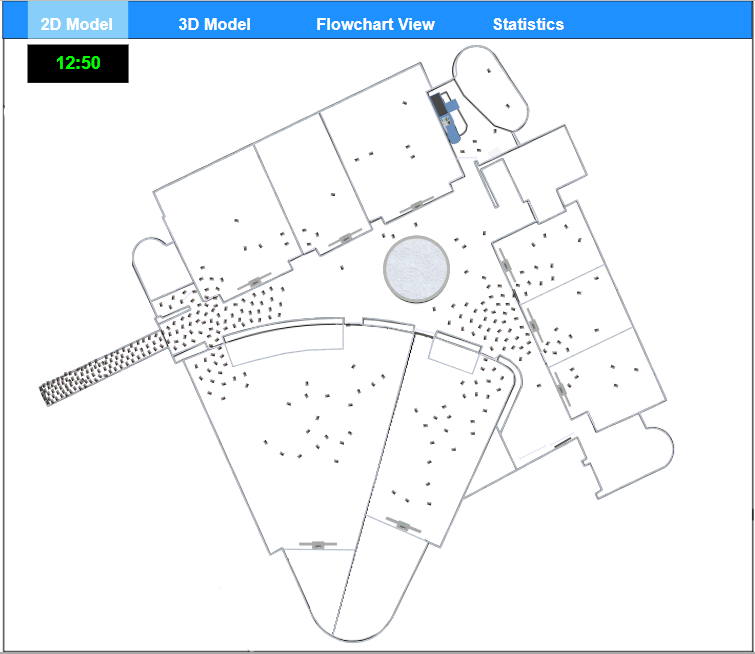
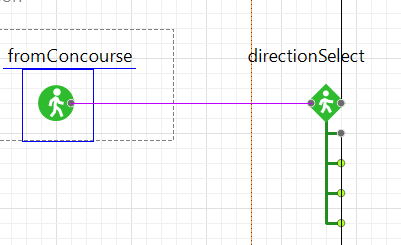
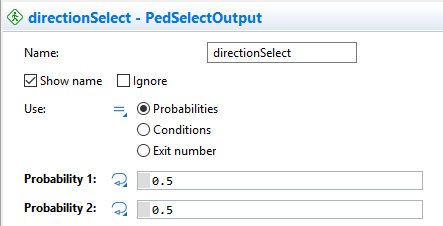


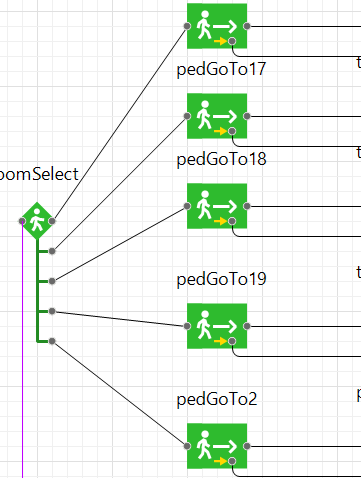
Figure 14 Overcrowding in first simulations

A way to overcome this was to introduce a probability distribution when routing agents. I separated the two entrances into two logic areas, and also added a select direction two each entrance. I concluded that on entry, each student has a 50/50 chance of going left or right.





Once the pedestrian then choses either left or right they have 5 options based on the side of the building they are on. Here another probability distribution was used with 5 options instead of two, with each option representing 1 of 5 room/exit options.



*Agent speed*

In order to introduce some randomness of user speed I decided to use a uniform distribution. Using a distribution like this enabled the agents to have a random speed between a stated maximum and minimum. I set the average walking speed between 0.3 and 0.7 feet per second, as the average walking speed for a human is approximately 4.6 feet per second. (Baker, 2006)



Figure 15 Agent Speed

## **4.2.2 Logic Explained**

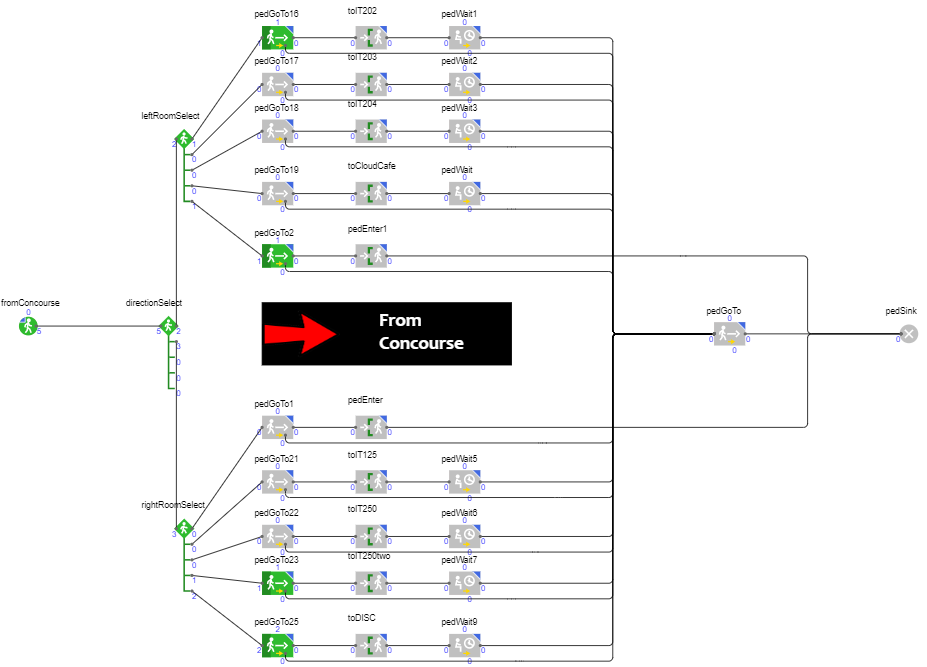


Figure 16 From Concourse Logic

Above is the final logic view of one of the flowcharts, for students entering from the concourse. The students enter at a rate if 20 students per minute from the concourse target line “ConcourseIn”. Next, they get to a direction select where they either go left or right, with each direction having a .5 probability. If they go left, they have an option of going to 5 locations, IT202, IT203. IT204, Cloud Café or Downstairs. These all have equal probability of 0.2. The first four go to the target line of that room, then enter via the “pedEnter” and wait the area inside that room until lecture is over or building is evacuated, where a free function call cancels their wait, and directs them to the nearest exit. The students who go downstairs are removed from the model via the “pedSink”.

The same logic applies when a student goes right on entering the building, except its choices are Downstairs, IT125, IT250, IT250 second entrance and DISC. Also, almost an exact replica of this logic applies to the students entering from outside. They too have 5 options once they pick their direction of choice.

I the alarm sounds, and the building is to be evacuated, all students in a “pedWait” zone will be set free using and sent to nearest exit, any student who is on route to a room or exiting the building, they too will have their task cancelled and directed to the nearest exit.

The next logic flowchart created was the students who start the simulation in the building. These are the students that are in lectures and when the simulation starts at 11:50am, they are released from the lecture halls at over a 10-minute period (i.e. before the next lecture starts and new students arrive.

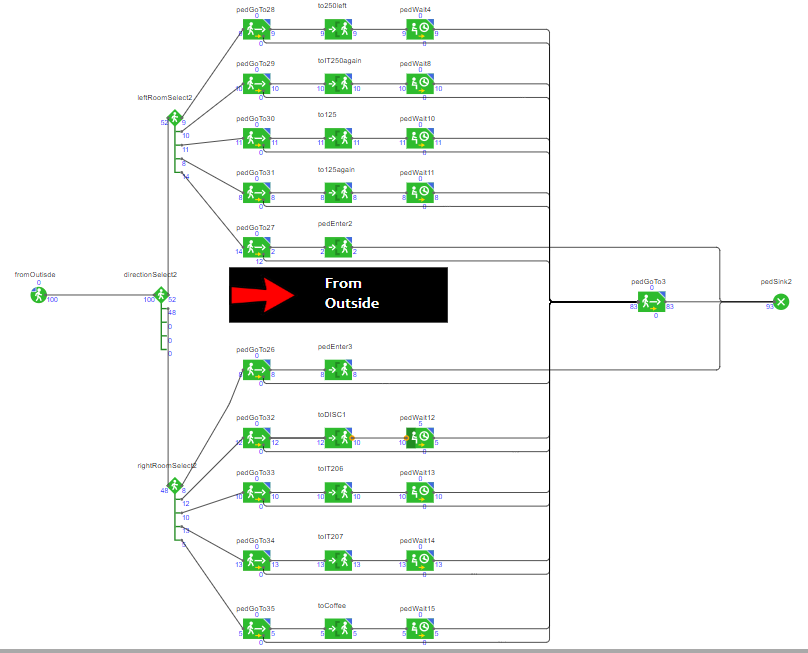


Figure 17 From Outside Logic

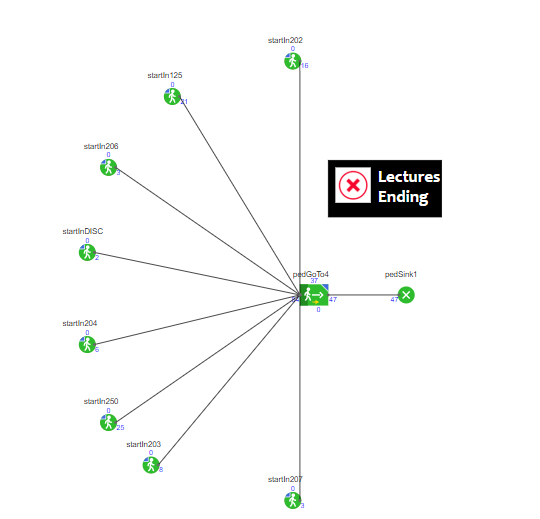
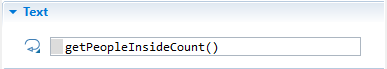
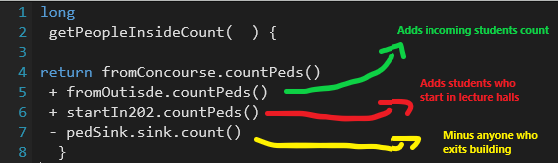
The counters on the multiple “pedSource” indicators shown above, give a live view of how many students started there and have now passed through that’s state and onto the next one. Once they have vacated the room, the use a “pedGoTo” to find the nearest exit, from either of the “ConocurseOut” or “OutsideOut” exits.

Figure 18 Lecture Ending Logic

## **4.3 Features/Code Snippets**

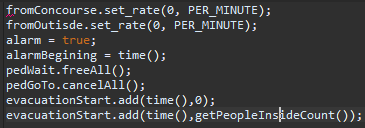
***A real-time student’s in the building count***





Works by adding on the total number of students that enters the building from certain entrances (Concourse and Outside), adding that to the total that begin the simulation in the lecture halls (at given areas) and then taking away the total of students that exit the building at the two exits (ped. sinks).

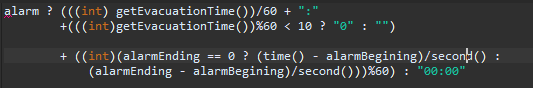
***An alarm to signal a building evacuation***



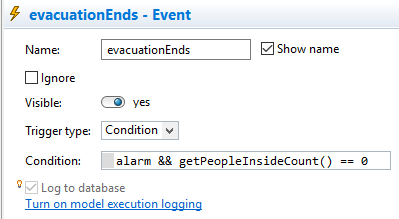


Above is a button from the model that signals an alarm in the building. Its on-click properties are shown above. Once activated, the rate of inflow coming into the building immediately gets set to 0, the evacuation timer begins, every student in each lecture hall represented by its pedWait, gets set to free, which sends them to the nearest exit. Every student who is in the building, but not in a lecture hall, their goTo method gets cancelled, and they two get sent to the nearest exit.

***An Evacuation Timer***

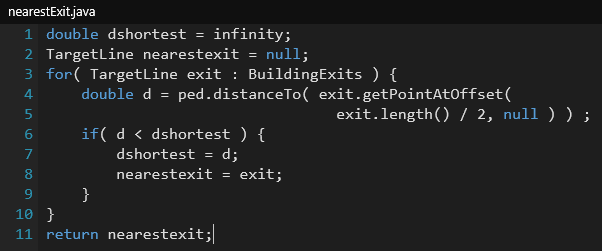


Once the alarm is activated from the button click, a timer will will be activated and the clock will start. It will end once the student count in the building is equal to zero. It works alongside the alarmEnding variable (pictured below).



***Nearest Exit Function***

The nearestExit function return the nearest exit for the agents when either the building is getting evacuated or the lecture ends (pedWait comes to an end).



It works by calling a collection of exits called “BuildingExits” which holds the two emergency exits of the building which are “ConcourseOut” and “OutsideOut” and based on the location it calculates the shortest distance to one of those exits and returns it.

***A menu bar for switching between views***

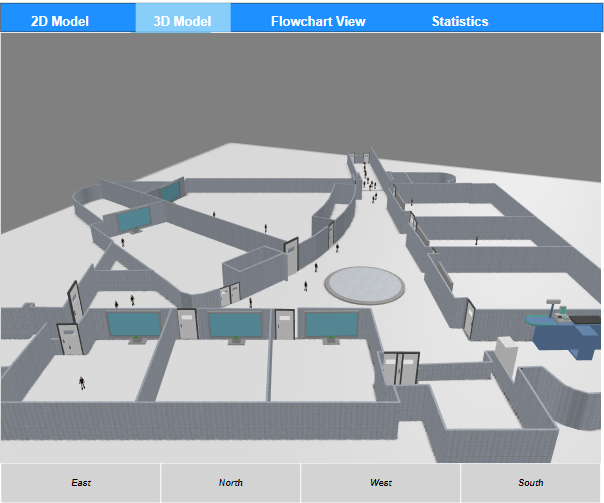
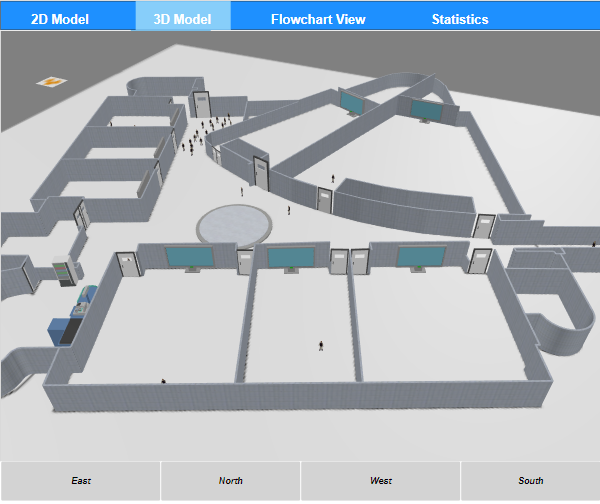


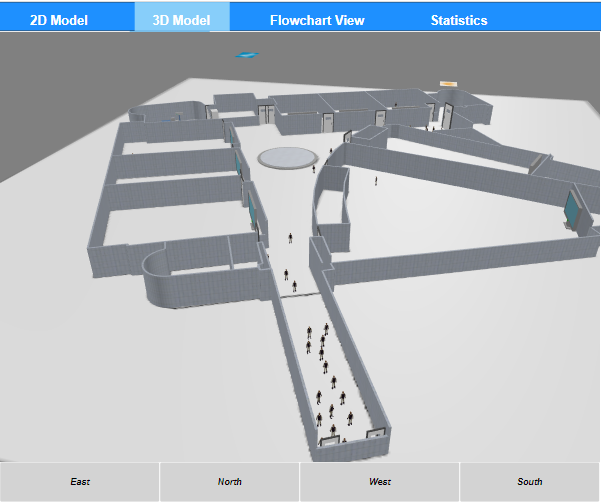
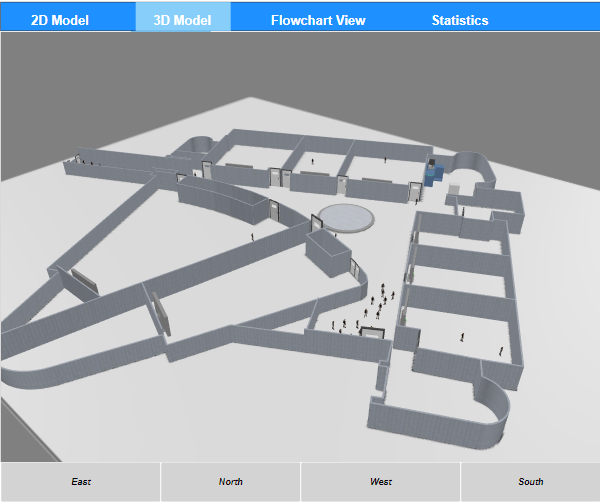
Once the simulation is launched, this bar will display at the top of the view. Each text box has on click properties to navigate to the desired view. An example is shown below where clicking on “Flowchart View” will move from the 3D Model view to the Flowchart view.



***A 3D view menu***

When the simulation starts its default view is a 2D view of the model. Once the “3D Model” is clicked in the menu, the 3D model can be seen. By default, it starts at a view from the east of the building.

The four buttons on the bottom navigation bar make switcching between the different 3D views/angles of the model. They are controlled by their action properties, which sets the choses the desired camera location( Figure X.X)

# **5 Testing and Results**

## **5.1 Iteration 1 – Evacuation Model**

Iteration 1 consists of the following:

* A 2D Model view where agents enter from two entrances, from the concourse and from outside. Agents chose the room they wish to enter using a probability distribution based on the direction they pick when entering the building. Agents are unable to leave the walls of the building and upon exiting they are removed from the model space.

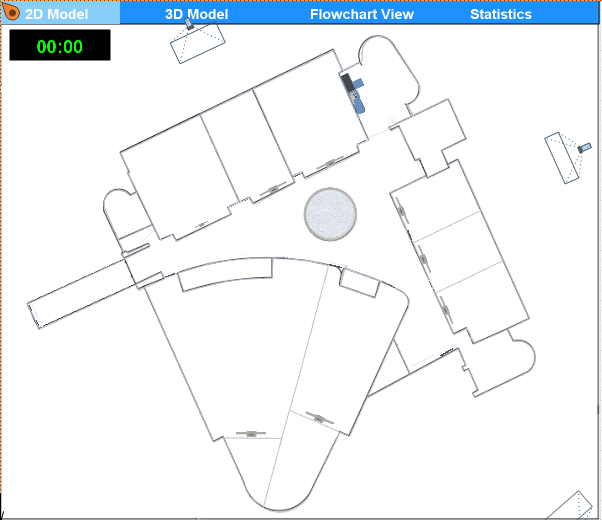


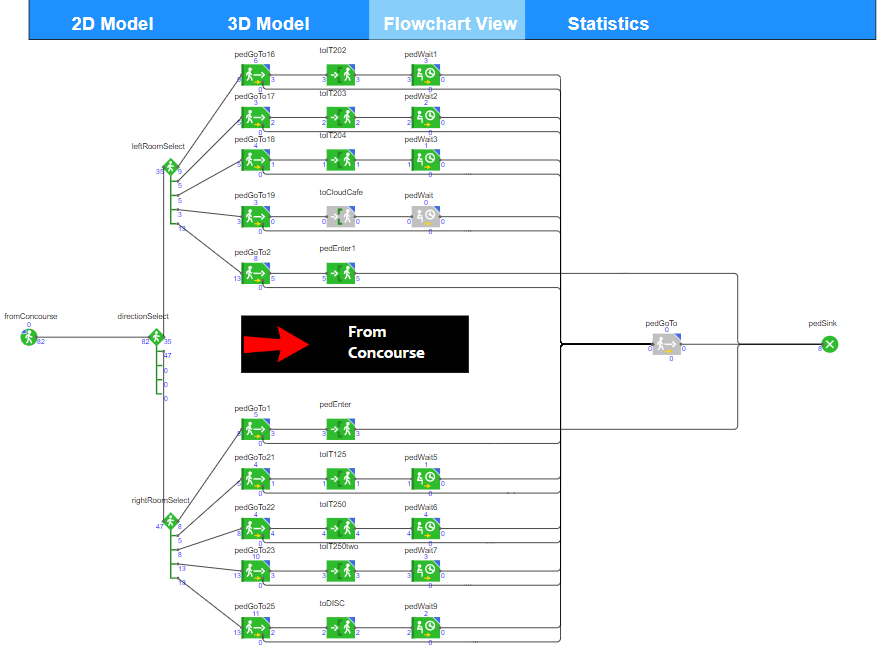
Figure 19 Iteration 1 - 2D View

* A separate model view where agents and objects (walls, projectors, doors) can be seen in 3D. Students can be seen here colliding with objects, interacting with each other and entering/exiting the model. Different angles of the 3D view can easily be switched between here using the navigation bar at the bottom.



Figure 20 Iteration 1 - 3D View

* A flowchart/logic view where the state charts of each logic set can be viewed. The process can be easily followed as live updating figures show the number of students that are in each room/part of the building, or that have passed through that point.



* A statistics tab where live updating graphs of different student’s data are modelled.



Figure 21 Statistics View

* A side menu option, where a brief description of the project, live student counter, evacuate building button, evacuation timer and two sliders to control attendance (room capacity) and rates of entry (students/min).

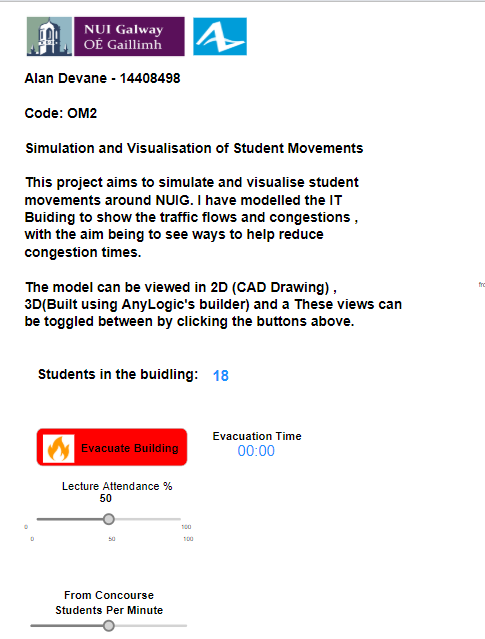


Figure 22 Menu View

## **5.2 Iteration 2 – Walkways/Paths version**

Iteration 2 is pretty similar to the previous one, except it introduces the use of a walk-on-one-side system. This process works by using AnyLogic’s pedestrian library to incorporate the “Pathways” tool.

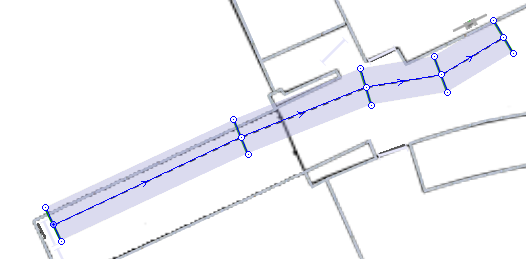
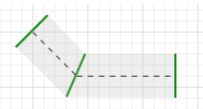
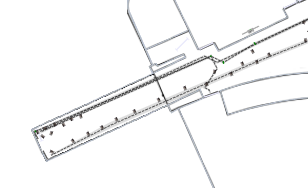
 

Figure 23 Adding Pathway's

Agents still us the probability distribution to chose room/direction of travel, however they follow a strict pathway from start of journey until the end. When travelling along the pathway, the students aim to stay within the pathway boundaries. However, if the pathway is overcrowded, students can easily cross over the borders and travel closely to the path. Pathway borders do not act as walls.

These pathways are designed with a “Walk on the left “principle. Students try to stick to the left side of the direction they are travelling. The reason for this is to try and reduce congestion, especially coming through the pedestrian bridge.

Both students entering and exiting the building on the concourse side are using the pathways to walk on the left, resembling a queue system.

Although we can’t guarantee that each student will stick to this, it significantly decreases the mass gatherings that can be seen on iteration 1

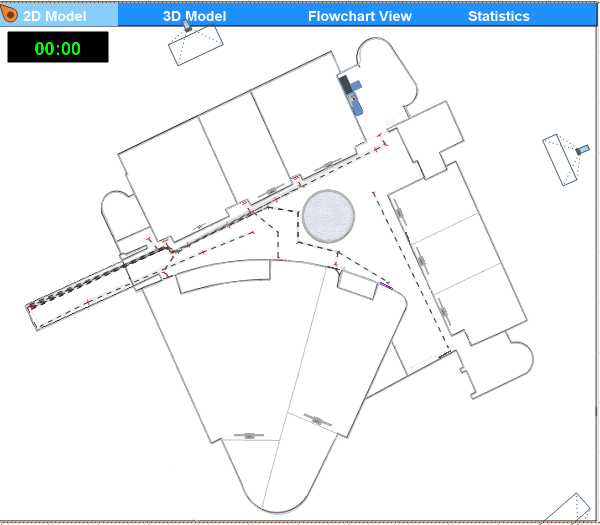
Below is an example of what the 2D model looks like with the pathways drawn on. These were eventually set to invisible on the simulation’s 2D and 3D layers to show clearer image of traffic congestion using these paths.

Figure 24 Iteration 2 - With Pathways

## **5.3 Analysis**

## **5.3.1 Congestion Analysis**

For this section I ran the model for both iterations to take note of the congestion levels at the entrance to the I.T Building using the pedestrian bridge entrance. The following are examples of student congestion and gatherings during the simulation of iteration 1.

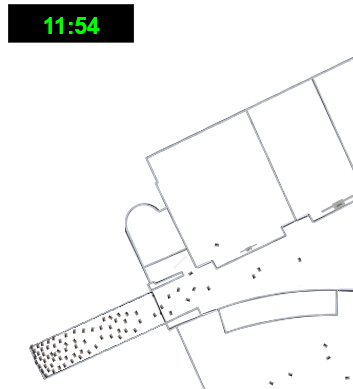
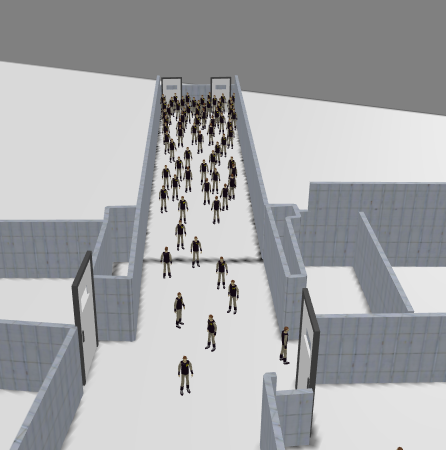
 

Figure 25 Iteration 1 Congestion

Here are some screenshots of iteration 2 in action, this is where the pathways are introduced.

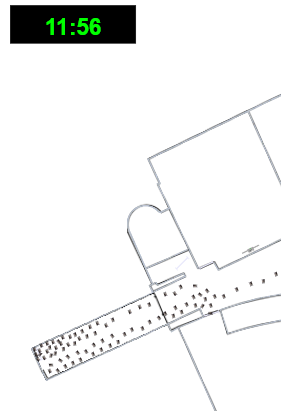
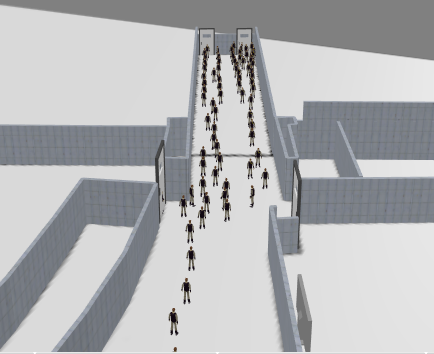
 

Figure 26 Iteration 2 Congestion

Iteration 1 (Figure 25) shows many students gathering at the pedestrian bridge. The traffic is coming both ways, from students going to lectures and another group of students exiting the lecture halls. The clock shows 11:56 am which is between the peak times of 11:50am and 12pm. These gatherings mirror the ones experienced on a day-to-day basis in the building.

Iteration 2 (Figure 26) introduces the use of pathways (walk on the left lanes) to the model. The congestion levels experienced in iteration 1 are not as apparent in this simulation. The images clearly show the improvement in the flow of students as they are almost moving in a queue like formation.

To conclude, the use of walk lanes would improve pedestrian flows in the building with less delays, reduced overcrowding and improved lecture punctuation coming as a result.

## 5.3.2 Evacuation Analysis

Figure 27 Evacuation Graphs

Average evacuation time = 4.11 minutes Average evacuation time = 3.281 minutes

The above graphs compare the evacuation times of both iterations, where iteration is the model with pathways introduced. Both models were ran 10 times and the average time for each is displayed underneath each graph. Iteration 2 performed much better than iteration1, with almost a full minute between the two averages. The agents seem to get stuck for space in the first iteration as they are all vying for the quickest path to the exit, while in iteration 2, the agents conform to the path to the nearest exit, and if overcrowded they move marginally to the left and right of the drawn.

To conclude, the college could use this data to implement a safety evacuation policy using these pathways. A full minute is a significant chunk of time when it comes to an emergency evacuation.

# **6 Conclusion**

## **6.1 Evaluation**

Now that I have completed the project, I have had time to reflect on how it went:

I am happy with how the model looks for the I.T. Building. It shows the model in real scale and the congestion occurs in the same areas as it does in a real-world environment. My friends and classmates who have experienced these delays and gatherings at the model location, have also agreed that my model portrays similar issues to the ones faced in the building each day.

I feel the simulation is quite life-like as the agents don’t just walk passed each other, some can stop for a second or move to the side to let others pass. While this is an advantage of AnyLogic’s intelligent software, it also was a bit of a nuisance for my project. I experienced many agents “jittering” and causing a simulation to slow down significantly. The agents rotate towards a given location, but when an area such as the pedestrian bridge becomes crowded with 2 or more agents in close proximity, this jitter can be seen and can move many agents within a certain perimeter well off track.

I am delighted with how the evacuation simulation worked out. Its’s on-click properties were easy to program, and it worked seamlessly with my model. I feel it resembled a real-world simulation as agents/students are seen to be pushing/jostling for space to exit the building as fast as possible. Their increase in speed also replicates how a student would react in such a situation.

The walk-lanes mentioned in the project brief also worked pretty well. With each student walking on the left congestions were less apparent, leading to more students being on time for lectures and increased safety within the building

Finally, I feel I could have improved on the statistics and analysis side of the project. The live student’s in the building counter and the evacuation timer worked reasonably well, while the sliders were somewhat glitchy and often inaccurate.

## **6.2 Future Work**

This project could be expanded to model other areas of the college or the college as a whole to improve congestion and student flows around the campus. It could also be adapted to model the traffic, parking and bus systems in the college. Also, many large scale events in NUIG experience pedestrian traffic and congestion, for example during graduations and events in the college bar. These events could benefit from simulation to prevent injury and delays.

## **6.3 Lessons Learned**

If I was to start-over I would to the following things differently:

* Retrieved exact measurements of the I.T Building before recreating the model in AutoCAD. This would have given an even more accurate simulation of the model.
* In my project definition report, I stated I was going to use a Trello board system for planning and project management. I ended up not using it and looking back now, it would have benefited me throughout the development process
* Used a uniform distribution for agent widths to give a more realistic feel to the simulations, as right now all agents are the same size.

# **Bibliography**

AnyLogic. (n.d.). Retrieved from https://www.anylogic.com/pedestrian-simulation-and-road-traffic-modeling-at-sheremetyevo-airport/

Baker, E. (2006). *Effects of obesity and sex on the energetic cost and preferred speed of walking.* Retrieved from http://jap.physiology.org/content/100/2/390.full

Bolia, L. T. (2015). *Risk management strategies to avoid stampede at Mass gatherings.* Retrieved from https://www.researchgate.net/publication/327716728\_Risk\_management\_strategies\_to\_avoid\_stampede\_at\_Mass\_gatherings

Botev, D. P. (2014). *Why the Monte Carlo method is so important today.* Retrieved from Wiley.com: https://onlinelibrary.wiley.com/doi/full/10.1002/wics.1314

Hubbard, D. (2009). *Modelling Without Measurements.* Retrieved from zmags.com: http://viewer.zmags.com/publication/357348e6#/357348e6/28

Kreutzer, W. (1980). *System Simulation Programming Styles and Languages.* Retrieved from Researchgate.net: https://www.researchgate.net/publication/234805907\_System\_Simulation\_Programming\_Styles\_and\_Languages

M.Macal, C. (2009). *Agent-based modeling and simulation.* Retrieved from https://ieeexplore.ieee.org/abstract/document/5429318

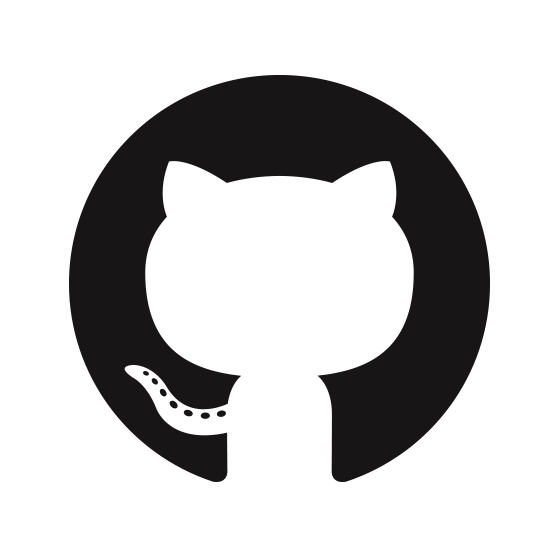
Maps, G. (2019). Retrieved from https://www.google.com/maps/@53.280227,-9.0583774,20.5z

Mr. Muhammad Ali, U. o. (n.d.). *A Cost Effective Methodology for Pedestrian Road Crossing for.* Retrieved from http://www.asee.org/file\_server/papers/attachment/file/0003/3939/ASEE\_Paper\_Ali\_031613\_fianl\_submission.pdf

PIDD, M. (1996). Tools for thinking. In M. PIDD. Great Britain: John Wiley & Sons Ltd.

Studies, A. C. (n.d.). Retrieved from https://www.anylogic.com/pedestrian-simulation-and-road-traffic-modeling-at-sheremetyevo-airport/

## **Source Repository**

<https://github.com/AlanDevaneGit/Final-Year-Project>

## **AnyLogic Cloud: Link to Run Model**

<https://cloud.anylogic.com/model/e847510c-d12c-40dd-b51b-e81e6899ff7b?mode=SETTINGS&tab=GENERAL>