***MinecraftEd Report***

This report is split into sections that align with the guidance notes, as reflecting in line with some sort of framework turned out to be way easier than reflecting with no direction to work towards. I hope this is all helpful.

# Some jargon and a weblink

I realise I’m far more acquainted with my work than anyone else is, so the following terms’ contexts are defined, hopefully so that my explanations are more helpful to you.

minecart CaMKII model = first CaMKII model redstone CaMKII model = second CaMKII model DNA synthesis machine

mRNA synthesis machine protein synthesis model translation machine Code for synthesis model:

<https://1drv.ms/w/s!AiOq7JZNG57I8AbS36Kl9VjP8kqC?e=Xp45ey>

## Ease of use

I entered the project as someone used to the mechanics involved in modding and simple redstone usage, as well as in Minecraft’s control scheme and ‘vibe’. Even then, devising methods with which to implement real-world phenomena in the game was difficult. With redstone, this would involve tweaking redstone repeater delays in the CaMKII model via trial and error until I was getting roughly the ‘random’ input that I wanted. I found working with command blocks more fun, as I could see the code changing as I edited and wrote it. I felt I could almost feel it being correct before I ran it within Minecraft. That said, building what was necessary for the redstone and command block models was simple – but obviously could be (and still can be) optimised, e.g. if you compare the circuitry for the 1st and 2nd translation machines. Solving the problems which I came across was enjoyable, though some jobs certainly felt very daunting (e.g. finding a way to represent the autoactivation of CaMKII). In all, difficulties I had were due to the inherent challenge in representing a process using a different platform, than due to limitations I found with Minecraft’s systems.

## Realm helpfulness

I used the Realm for everything I made/did. I found its features helpful and unhelpful in different ways. For example, it was helpful that I could join the Realm on any PC with Minecraft installed and have the world ready to work within. This meant that at home I could flip between using my laptop and my home PC, always able to carry on from where I started. That said, I still needed to transfer the word document with the command block code between my PC and laptop in order to get the most out of my time on the Realm whilst I was computer-hopping.

I think working on my projects in a single-player world would have been helpful for when my internet connection was slow or entirely shut off, as during these times I’d have to either avoid working or become frustrated with how slowly I was able to progress. In addition, I didn’t feel like I wanted to tweak the machines once they were built in the Realm, whereas with my own world I’d have felt more comfortable ripping stuff up to do it again, or copying over large swathes of materials with WorldEdit (or another similar program). Because I couldn’t build on the world’s spawn, I was also forced to deal with non-standardised coordinates (e.g. all tRNAs are found along the x = 1 axis, all the release factors are x=3, etc.), instead using mind-numbing jumbles of 200s, 100s, and 0s.

For teaching, I think it is worth considering how Realms would mean every student had to ‘wait their turn’ to perform certain actions if they were online with other players, whereas offline worlds accessible to each instance of Minecraft used would allow students to fiddle with any Minecraft representations together, not one by one.

## Time

I found 12 hours a week to be enough for most of the work I did, including all my redstone/minecart practice and my CaMKII, DNA synthesis and mRNA synthesis machines.

The flexible nature of the study allowed me to procrastinate in the week, which meant I’d binge on the work, doing it in 4-6 hour shunts as opposed to (the reasonable strategy of) setting aside a few hours a day to work on my projects.

I very much underestimated how long it would take to make the translation machine work, as I’d originally planned to spend around 8 hours on it – whereas the first version of this machine probably took 18-24 hours to finally work (and even then, without support for premature STOP codons). For this reason, at the scheduled end of the project the protein translation model was unfinished – so I made the decision to carry on working, as I wanted to finish and then optimise the challenge that I’d set for myself. *My projects*

My projects were representation of CaMKII (auto/de)activation, and a command block-based representation of protein synthesis, separated into player-directed DNA synthesis and animations of mRNA synthesis and translation. The second project took longer to make and was more difficult to complete, owing to reasons I’ll cover below.

My first project culminated in a redstone model of CaMKII’s activation-autoactivationdeactivation cycle. I made two models, once using minecarts as timers to achieve a distractingly loud but functional model, then again using only redstone circuitry (and some pistons) to replace the minecart timers. I also housed the circuitry for the second model in various colour-coded rooms which could be entered or ignored, so that someone new to the model would be able to look at its inner workings to whatever extent they desired. I used lecterns as a device for relaying information to anyone using the model. In my opinion users, rather than myself, should evaluate whether the lectern-based approach was informative about the nature of the model (or any of my models).

My CaMKII models were highly functional, with no attention at all paid to the structural intricacies of the CaMKII molecule. The models represented CaMKII as one ‘thing’, ignoring the fact that adjacent subunits in real CaMKII molecules can interact with one-another to affect the overall catalytic activity of the complex. I decided to build functional models to begin with as I don’t hold my artistic skills in very high regard. I wanted to prove I could make something that represented concepts in their function before trying to represent concepts visually, combining both structure and function. This project took far less time than the protein synthesis model, even if it did produce two models, taking about 12-14 hours all said and done.

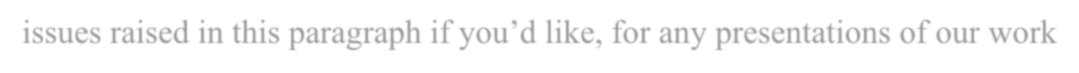
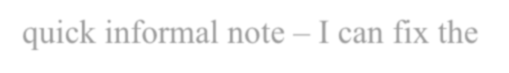
My second model was intended to be more akin to a teaching tool than the CaMKII models.

Hence the ‘bank’ of reference stills that I could aim to recreate in animation and walls of ‘which block represents which biological molecule’. This pre-preparation provided a clear basis from which I could get to making animated representations of protein synthesis. To further achieve this ‘teaching tool’ aim, the designs for the ribosome/tRNAs were based on common textbook representations of the process. This is so that, in theory, the animation would be easier understood by students.

Explaining each project in terms of what it is meant to represent is fairly easy, as all that is required is explaining a biological concept in terms of the Minecraft representations of that concept – though this is simpler for very visual representation of a concept like the protein synthesis model, as opposed to the circuit-based CaMKII model. I believe the harder task would be explaining the principles behind the redstone or command block inputs/outputs and how these are achieved to a non-Minecraft user. I tried to make every bit of my models as little of a ‘black box’ as possible, but I can understand that users new to Minecraft might find what I’d built a bit scary.

I built my concepts piece by piece, testing as I went. For the CaMKII molecule, this meant building my timers first and working to achieve that ‘autoactivation/central circuit’, then trial-erroring until I found a solution which would sometimes deactivate the ‘central circuit’. I should note here that I don’t believe my final model is even a very good representation of CaMKII. This is due to the lack of precision with which I made the timings and the lacklustre output – I never worked out a good way to represent CaMKII’s enzymatic nature within the cell, hence the cop-out “My name’s CaMKII, and I love you!” output. As for the timings, they work because that’s the arrangement in which I found that they work, as opposed to careful planning on my part during building.

For the DNA/translation machines, this approach is even more noticeable, in my opinion. Many improvements could be made with these models. [quick informal note – I can fix the issues raised in this paragraph if you’d like, for any presentations of our work] For DNA synthesis, the machine lags. The lags mean that inputs can overtake earlier inputs, so the bases in your oligonucleotide are of a different order to what you wanted. This could be improved using chain command blocks rather than the current “**command block** → **repeater**



→ **command block** → **repeater**” structure. This would increase the speed with which inputs propagate through the machine. In addition, the command blocks which announce whatever residue has just been added could have a “*/execute if xyz(last phosphate) minecraft:air run etc.*” argument added to prevent it from saying the name of a base once the DNA strand is fully synthesised. The mRNA machine works well, but some extra command blocks could perhaps announce to the player each step of mRNA synthesis as it occurs within the animation.

The translation machine’s first version had a multitude of problems, ranging from ugly animations and animation cycles to bricking up when STOP codons were included in the player-made oligonucleotide. This was due to a linear circuit scheme, which worked well for certain portions of the animation, but really slowed down other parts. For example, players would have to wait for a tRNA to leave the E site before another tRNA would enter the A site). It also made editing the machine difficult, as removing or adding command blocks would sometimes cause the redstone to stop working as it had done before. Luckily the new version of the translation machine is far more tweakable and can handle STOP codons. In theory, I believe this version of the machine could be made to accommodate tens of amino acid codons by some user, if there were time to add them. Each extra amino acid is worth another ~50-55 command blocks (~20 in the tRNA/release factor array, ~20 in the polypeptide description array and ~10-15 in the rest of the machine). Adding extra amino acids would also require other parts of the translation machine to be recoded as well, and the addition of command blocks to the DNA and mRNA synthesis machines (if these were also going to be used by the theoretical user). Another helpful addition to the translation machine would be a set of levers that allow a user to pause the animation at any time, perhaps by removing/adding the repeaters that let the redstone blocks activate command blocks.

I think anyone could build my models, given enough time and some sort of manual which combined the following things: The explanations given in the lecterns, the code used by the command blocks used and top down views of the circuitry. Furthermore, the command blockreliant machines could easily be compacted using multi-storey command block buildings, with each floor containing the circuitry relevant to one portion of the DNA/mRNA/translation machines.

## Teaching Potential

* Lectures:

In lectures, models such as my protein synthesis model could be used to describe simple concepts, such as silent vs. nonsense vs. missense mutations. Videos of the models in action on YouTube, Echo360 or virtual learning environments (VLEs) could also be shown, or provided with online information about the lecture, to provide a moving representation of the diagrams that students are typically presented with.

* Tutorials:

With the amount of content typically covered in a tutorial, I can see some use in using

Minecraft to take students (in groups, or as a large group perhaps) through a set of Minecraft-contained representations of concepts over the course of an hour. Here the students could experience the concepts dynamically, engaging with animated content and maybe editing bits of it themselves to see the results. This might make them respond more positively to concepts they’d usually feel divorced from. The students could then be provided with a link to an unlisted YouTube video that shows the Minecraft representations that they’d been working with. That way, if they wished to go over the material again, they could in video format. Map files could also be provided on a VLE for more Minecraft-savvy students.

A tutor experienced enough to explain what was going on in the Minecraft world would be required for such a tutorial to fully achieve its potential – in addition, in groups where there were no experienced Minecraft/PC players, time would be lost explaining the game’s move-set to unexperienced players. These problems could be alleviated by inserting short ‘tutorial’ sections into the maps used by the students and tutors alike.

* In-course assessments (ICA):

Minecraft could be used to complement ICA using builds that produced predictable outputs from given inputs. A student could be asked to list the outputs of a Minecraft machine from specific inputs, then comment on what these outputs represent biologically. For example, they could apply different ‘drugs’ to various ‘receptors’ and record the output of such changes. This would be helpful where working with another program (that represents the inputs/outputs) was not possible. As another option, students could be asked to build a redstone representation of a protein’s activation cycle.

The biggest caveat for incorporating Minecraft into ICA would be that we inadvertently taught students more about Minecraft’s mechanics than a given biological concept’s characteristics. This could be overcome by keeping students away from deep use of Minecraft’s systems.

* Practicals:

I think engagement with biological concepts in practical classes is best kept to performing lab work. However, before and after the actual practical work takes place, Minecraft could be used to help represent some of the concepts being played with that might be hard to visualise otherwise. Computer-based practical classes could use Minecraft to take students through a concept together en masse – this would be limited to ~20 people by Educational Edition, which could make such sessions very difficult to organise for courses of over 100 or so students. Either educators would have to have many students together in separate groups on different worlds, or they would have to organise lots of sessions to get everyone through the content they wished to deliver via Minecraft.

* Honours Projects:

I don’t believe that I know enough about Honours projects to offer very educated opinions on how well Minecraft could come into such endeavours. To speculate somewhat, students could be offered projects in which they were expected to produce Minecraft-based representations of biological concepts in such a way that was accessible by university or high school students. The maps and code could be hosted on a University of Edinburgh associated webpage designed to provide resources to educators in other institutions. If the Minecraft builds were of high enough quality and took enough time, they could even be commercialised. Issues might arise here if students felt themselves to be ‘above’ Minecraft projects, preferring instead to do lab or library work. In addition, the Honours projects would need to incorporate a research-based element of testing students’ learning with their builds, in order to warrant the award of a degree with Honours.

* Revision:

In my personal experience, revision tools are most helpful for revision when you’ve made them yourself, as the best way to learn is to interact with content enough to be able to teach it yourself. Minecraft resources could be provided via VLEs to students as part of a revision pack, to help out visual learners.

* Outreach:

I believe Minecraft has good potential for use as an outreach tool, with maps/code available online to teachers/interested students or running on PCs at a museum or similar location. Using simplified explanations of the biological concepts represented within the world, young people could engage with scientific principles they don’t typically learn about until later in their educational life.

## Personal Reflection and Conclusions

I told you at the time that I really wanted to see if we could make this work, and I believe Minecraft does have potential in a limited range of applications. especially in letting younger students access biological concepts usually kept away from them due to their complexity. In terms of my friends’ responses to the study, most of them were sceptical at first but said that they believed that Minecraft could have use in teaching after I explained the aims of the study to them. I worry that this immediate response would also present in students, who might take annoyance at being taught with what they perceived as a children’s game. The study was run fine – I didn’t really use Trello as I went along because I was too busy working with Minecraft itself. The meetings were very helpful – I loved being able to share my progress and see what other people were working on. I think the one thing we could have done better is to provide each other with written progress of what we’d been making. In my opinion this would have made us document our progress as we went through, describing how it worked – this would probably have made this writeup a tad easier, too.

In conclusion, I see limited potential in using Minecraft to teach students about biological concepts. That is not to say that Minecraft is useless in this educational sphere, but more that we should see what Minecraft is particularly promising for, teaching-wise, and optimise it for those applications. This would enable us to get the most out of Minecraft where it shines.

Minecraft can’t replace tailor-made protein visualisation software like PyMOL or be a perfect stand-in for static diagrams or molecular animations. However, it can provide easy access to ideas and represent them in a different way to normal, in ways that could engage visual and interactive learners more than their less visually inclined colleagues. If we can use Minecraft to represent concepts which have no simpler, more detailed representations that already exist, we could capitalise on a realm of unexplored ideas – making them accessible to groups of people who might have missed out on their benefit before.