

Team Control Number
22875633

Problem Chosen

D

2022

IMMC

Summary Sheet

Building a virtual Hong Kong in metaverse

Team#22875633

January 24, 2022

Contents

1	Introduction	1
1.1	Background	1
1.2	Problem Restatement	1
1.3	General Assumptions	2
2	Model A	2
2.1	Model Overview	2
2.2	Notation	3
2.3	Creating the Blocks	3
3	Calculating the Fitting Degree	5
3.1	Problem Overview	5
3.2	Notation	6
3.3	Simulating the Shape	6
3.4	Combining the Degree of Each Block	8
4	Model B	8
4.1	Model Overview	8
4.2	Assumptions	8
4.3	Obtaining Data	10
4.4	Notation	11
4.5	Calculating Local Speed	11
4.5.1	Observing From the Ground	12
4.5.2	Observing From the Air	13
4.5.3	Observing From Planes	14
4.5.4	Observing From Roads	15
4.6	Calculating Speed(# undetermined)	16
5	Strengths and Weaknesses	16
References		18

1 Introduction

1.1 Background

Nowadays, *metaverse* has become a heated topic and a symbol of development in virtual technology. It refers to a three-dimensional virtual world, especially in online role-playing games such as *Second Life*, *Minecraft* and *Roblox*. In this system, people can build virtual items according to their own needs and complete basic communication with others such as buying and selling items from others as well as sharing information of their creations. In other words, this is a society which only depends on the instructions of game players. Using this technology, we can simulate the society by uploading the data given and create the items which exist in daily life to predict the future development of this system.

In the three games mentioned above, players use limited basic blocks and their own imagination to build their own virtual world. It can be seen that the type of basic blocks is very important. They must meet the needs of construction to the greatest extent, and at the same time, the number of types and size should be appropriate so that players can easily understand the function of each blocks and make full use of them.

Now Hong Kong is also confronted with the problem of how to improve its urban construction without imposing policies on the real city. On this occasion, creating a virtual Hong Kong by using metaverse is key to this challenge. We can create simple elements in this virtual world which fit the reality so that government can observe the effect of their policies by giving basic instructions in this virtual city. In this way, there will be no loss in the real world due to strategic mistakes because all of the mistakes actually happen in real life. City construction in Hong Kong can be easier and more efficient.

1.2 Problem Restatement

To build a virtual city and make sure that it can be used in daily thing, we need to figure out its basic elements and find out the computing and storage resources needed to maintain a virtual Hong Kong. So our work is divided into 2 parts:

- Build the basic blocks that satisfy the needs of citizens and can be seen in a real city. We will explain the functions of each block and how it can be planted in the virtual city.
- Calculate the fitting degree of basic blocks to make sure our virtual city can simulate Hong Kong.
- Describe the computing and storage resources required in our design by calculating the speed at which it update the state of moving objects(such as pedestrians and cars) and how much bytes of storage it needs to maintain a virtual city.

1.3 General Assumptions

- **The ground of Hong Kong can be regarded as a plain.**

Though we all know the earth is an ellipsoid, Hong Kong is so small that we can regard its ground as a plain. Also, the buildings in Hong Kong are generally not high. So we can neglect factors of ground inclination.

- **The data will be updated non automatically when serious natural disasters such as typhoons and hurricanes destroy buildings.**

We divide the shifts in location and shape of blocks into two types: small-scale shifts such as car driving and large-scale shifts such as decay of buildings. The latter happen rarely, so the data can be uploaded non automatically.

2 Model A

2.1 Model Overview

In this model, we will mainly discuss the basic block sets in this society. We divide the process into two parts: how to describe the location and features of the blocks and how to make sure our blocks fit the constructions in real Hong Kong. we will find that what we need to build is divided into two parts: blocks with static attributes and blocks mobile entities which we call *entities*. Later we will discuss the relationship between them.

When we take everyday life into consideration, we find the common blocks in a city. The first two types are entities and the last one belongs to static attributes.

- **Transportation vehicles**

In Hong Kong, there are four types of transport: rail transit, air transport, transport on sea and road transport.

- **Pedestrians**

It's common to see pedestrians walking in Hong Kong.

- **Buildings**

These buildings consists of building blocks and are seen as static objects.

In this way, we can build a basic structure of our virtual city.

2.2 Notation

$f(\text{block})$	properties of the block
V	volume of the block
ρ_1	continuity of lines in patterns
ρ_2	density of lines in patterns
L	dimension of lines in patterns
α_{radius}	a certain proportion which determines L
r	radius of the dots in patterns
θ_1	angle with O_x
θ_2	angle with O_y
k	a special property of each block
a	common parameter of the geometry
b	common parameter of the geometry
h	common parameter of the geometry
α	common parameter of the geometry
β	common parameter of the geometry

2.3 Creating the Blocks

In our model, we use a coordinate system to describe the absolute location of each blocks which can enable us to quickly track the route of each moving block. Also, we use a matrix which is made up of the properties.

We use a shape set to describe the geometric features of blocks. There are several kinds of geometrical configurations.

Type 0 is hexahedron, which is shown in the picture below. $\alpha, \beta, \gamma, a, b, c, \theta_1, \theta_2$ can be introduced to describe its shape. Also, k is introduced to describe its dimension. It means that, when $k=0$, it's a pyramid while when $k=1$, it's a parallelepiped.

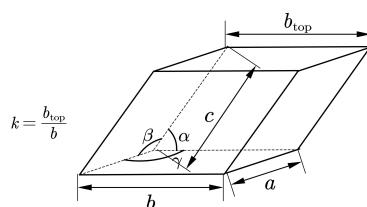


Fig.3 An example of parallelepiped

Type 1 is ellipsoid. a, b, c is introduced to determine its shape.

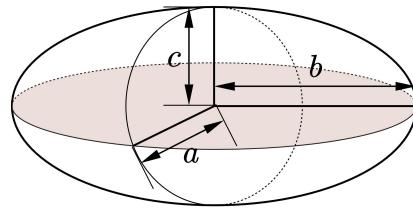


Fig.4 An example of ellipsoid

Type 2 is elliptic column platform. This time, $a, b, h, \alpha, \beta, k$ is introduced to describe its shape. When $k = 1$, the block is a elliptical italic and when $k = 0$, the block is a elliptical colic. α_{radius} should be at a moderate size to prohibit regional distortion of the image.

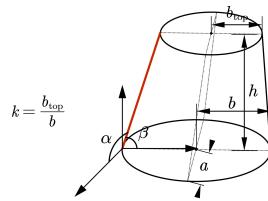


Fig. 5 An example of elliptic column platform

Secondly, we need to describe the color and patterns on the blocks. We use three primary colors(red, blue and yellow) to describe the color of blocks. For the patterns of the blocks, we adapt two indicators ρ_1 and ρ_2 to respectively describe the features of these patterns; fro the coloring factor, we use the international standard of (R,G,B) color base.

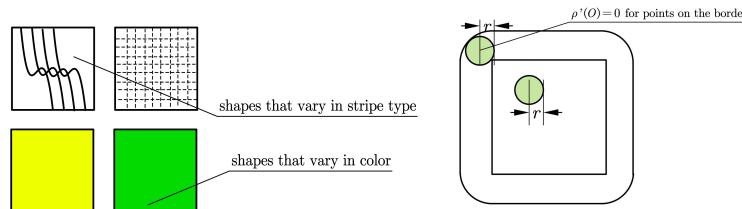


Fig.1&2 Patterns on the blocks

According to the picture, we can know the correct definition of r and L . We use α_{radius} to determine the proportion of this dimension.

$$r \stackrel{\text{def}}{=} L \times \alpha_{radius} \quad (2.1)$$

Therefore, the proportion of stripes in the drawn circle can be calculated as:

$$\rho'(o) = \frac{\text{Line Area}}{\pi r^2} \quad (2.2)$$

Now we can define the two indicators introduced as before. Continuity, which in other words, can be described as measured as following. Whereas density can be approximately seen as the average $\rho'(0)$ after being integrated on a two-dimensional scale.

$$\rho_1 \stackrel{def}{=} \frac{\max_{\text{Surface}} \rho'(o) - \min_{\text{Surface}} \rho'(o)}{L} \quad (2.3)$$

$$\rho_2 \stackrel{def}{=} \frac{\iint_{\text{Surface}} \rho'(x, y) d\sigma}{S(\text{Surface})} \quad (2.4)$$

Thirdly, we will take its volume into consideration. Its volume is easy to be calculated according to its different geometric features.

So, when we combine all the properties together, we can get the matrix:

$$f(\text{block}) = \begin{pmatrix} V & (\rho_1, \rho_2) \\ & (R, G, B) \\ & \text{vectors in the shape set} \end{pmatrix}$$

(In this part, vectors refer to their determining factors of the block's shape as mentioned at the beginning of section 2.3.)

3 Calculating the Fitting Degree

3.1 Problem Overview

In this part, we will discuss the fitting degree of our blocks. Now that we have established a characteristic indicator (shown in a matrix) to represent each block, we can now consider the process of fitting them into a complete architectural structure. Suppose there are numerous "blueprints" automatically devised by our algorithm (which will be introduced later), we need to evaluate its "excellence" for a specific building.

Therefore, to quantify this "how good" factor, we set up a "fitting degree", which can be implemented to get the best "blueprint" and do the required calculations. Most of the indicators introduced below are related to $f(\text{block})$ itself, which makes the calculations more accurate.

3.2 Notation

F_1	the fitting degree in volume
F_2	the fitting degree in lines
F_3	the fitting degree in color
F_4	the fitting degree in shape
ϕ	the fitting degree of the whole block
$Overlap_{Cluster}$	the degree in which blocks and items overlap
$Excessive_{Cluster}$	the degree in which blocks vary with items
$\Delta\rho_1$	change in continuity
$\Delta\rho_2$	change in density
Δ_e	the maximum color difference distinguishable by human eyes
Δ_{color}	color difference of blocks
I	definition of shape
Δ_{shape}	shape difference of blocks

3.3 Simulating the Shape

Firstly, we need to calculate the fitting degree of the block's shape. A brick too small or large in size certainly does no good to our construction. Consider the overlapped and the outside parts: $Overlap_{Cluster}$ and $Excessive_{Cluster}$. So we can the fitting degree shape can be described as:

$$F_1 = \frac{Overlap_{Cluster} - Excessive_{Cluster}}{V} \quad (3.1)$$

Secondly, we will discuss the fitting degree in patterns on blocks. It will be divided into lines, colors and shape, which will be easy to calculate with $f(block)$ as introduced before. $\Delta\rho_1$ and $\Delta\rho_2$ is introduced to explain the relationship between blocks and items. So we can get the following formulae:

$$\Delta\rho_1 = |\rho_1(surface(Block)) - \rho_1(surface(Target))| \quad (3.2)$$

$$\Delta\rho_2 = |\rho_2(surface(Block)) - \rho_2(surface(Target))| \quad (3.3)$$

$$F_2 = e^{-(k_1\Delta\rho_1 + k_2\Delta\rho_2)} \quad (3.4)$$

Under the circumstance where k_1 and k_2 are constant weights, the indicator declines at exponential speed as $\Delta\rho_1$ and $\Delta\rho_2$ increases, meaning that there is little tolerance of the disparity in these two factors between the target and the actual block.

When we discuss the fitting degree in color, we identify Δ_e as the maximum color difference distinguishable by human eyes. So when we compare $(R_{Block}, G_{Block}, B_{Block})$ with $(R_{Target}, G_{Target}, B_{Target})$, similar to the pattern

part,, we define:

$$\Delta_{color} \stackrel{\text{def}}{=} \sqrt{\sum_{R,G,B} \frac{(R_{Block} - R_{Target})^2}{3}} \quad (3.5)$$

$$F_3 = e^{1 - \frac{\Delta_{color}}{\Delta_e}} \quad (3.6)$$

(When $\Delta_{color} = \Delta_e$, the fitting degree is 100%.)

The last factor is shape. We assume that in calculation, the type the block belongs to must be consistent with the building's shape. It means that if the shape of block is consistent with the type in calculation, $I=1$. Otherwise $I=0$. So we can find that:

$$\Delta_{shape} = \sqrt{\frac{\sum_{x \in \text{attribute}_{\text{set}}(\text{block})} |x_{\text{Block}} - x_{\text{Target}}|}{\text{attribute}_{\text{set}}}} \quad (3.7)$$

$$F_4 = I \cdot e^{-\Delta_{shape}} \quad (3.8)$$

With all the previous factors combined, we get a "vector" in the same 4×4 "format" as $f(\text{block})$, for easy calculation. Multiply the pre-factor matrix by $f(\text{block})$.

$$\phi = (F_1, F_2, F_3, F_4) \stackrel{\text{def}}{=} (V_v, V_{\text{line}}, V_c, V_{\text{sh}}) \quad (3.9)$$

$$\begin{aligned} \begin{pmatrix} V_v \\ V_{st} \\ V_c \\ V_{sh} \end{pmatrix} \times f(\text{Block}) &= \begin{pmatrix} V_v \\ V_{st} \\ V_c \\ V_{sh} \end{pmatrix} \times \begin{pmatrix} V \stackrel{\text{def}}{=} J_1 \\ (\rho_1, \rho_2) \stackrel{\text{def}}{=} J_2 \\ (R, G, B) \stackrel{\text{def}}{=} J_3 \\ (\alpha_1, \dots, \alpha_k) \stackrel{\text{def}}{=} J_4 \end{pmatrix} \\ &= \begin{pmatrix} V_v J_1 \\ V_{st} J_2 \\ V_c J_3 \\ V_{sh} J_4 \end{pmatrix} = \begin{pmatrix} V_v \\ (V_{st} \rho_1, V_{st} \rho_2) \\ (V_c R, V_c G, V_c B) \\ (V_{sh} \alpha_1, \dots, V_{sh} \alpha_k) \end{pmatrix} \\ &\stackrel{\text{def}}{=} f_{\text{same}}(\text{Block}, \text{Cluster}) \end{aligned}$$

The result of the calculation is a sort of "compromise"——the balance point of the current set of blocks and the actual building. To illuminate this concept, we introduce the definition of the "same" blocks of a building don't necessarily exist, they can be used to equally quantify the consequent calculations.

Finally, we can get the fitting degree:

$$\alpha_{\text{fit}} = \frac{\sum_{\text{Block}} \frac{f_{\text{same}}(\text{Block}, \text{Cluster})}{f_{\text{same}}(\text{Block}, \text{Building})}}{\text{Properties of blocks}} (0\% \text{ } 100\%) \quad (3.10)$$

3.4 Combining the Degree of Each Block

After we come up with the 'same block' notion, we can actually calculate how much kinds of blocks it takes to construct the virtual world. Firstly, we need to try this on a smaller scale. Consider a single building. We've already got the 'same block' of each block of the building. However, the disparities of some blocks may only vary so little on the numerical scale that the difference is not detectable by the human eye. We define the difference between 'same block's as

$$\Delta_{\text{same}}(\text{Block1}, \text{Block2}, \text{Cluster}) = \sqrt{\frac{\sum (\text{difference of four factors})^2}{4}} \quad (3.11)$$

and suppose the least detectable value of this by naked eye equals $\delta_{mathrm{eye}}$, which can be calculated with enough information or experiments.

Now all we need to do is to calculate the ideal number of blocks. Therefore, we use the K-means algorithm, where a cluster with a 'distance' of points less than δ_{eye} can be aggregated as one block. Running the following algorithm can help us get the number of blocks of a specific building. Repeat this on all buildings, and again use this algorithm, deriving the 'same same' blocks, the amount of which is what we want.

From section 3.3 we get the fitting degree of a specific building plan. So next we need an algorithm to search the optimal one among the plans. In order to achieve this without repeated or excessive calculations, we use Q-Learning technology to help us realize the whole process. This can make calculations in the second model easier. The steps are shown below.

Algorithm 1 Q-Learning algorithm

```

divide the whole area into several cubic parts
for part (i, j, k): # simulate the single part do
    initialize Q table # the potential benefits of the plan, a matrix
    initialize S # the set of actions to simulate the city with geometric forms, a list
    initialize  $\alpha$  and  $\gamma$  # is the rate of learning while  $\gamma$  is the rate of declining. Both  $\alpha$  and  $\gamma$  are in the range of (0, 1).
    choose action A from S due to a policy from Q # A is a plan of fitting the blocks with geometric forms, choose the plan with the highest simulation level
    while not done: do
        take action A, observe R, S'
         $Q(S,A) \leftarrow Q(S,A) + [R + \gamma \max_a Q(S', a) - Q(S,A)]$  # update the Q table and the status S
         $\leftarrow S'$ 
    end while
end for
for the borders of parts: do
    small adjustments to simulate the area better
end for

```

4 Model B

4.1 Model Overview

Now that we've figured out all the features of basic blocks and how to calculate its fitting degree, we need to calculate the running speed of our virtual society so that it can be put into real calculation. In this model, we will focus on the byte required in different moving circumstances and different transportation devices. In addition, we will calculate its lagging index according to its updating speed.

4.2 Assumptions

- All the straps and lines have width.
- All the entities can be seen as a particle.

To fasten the calculating speed, we assume that all the entities' volume can be neglected. In other words, they can be seen as particles.

- There is a "middle" area between blocks and entities.

There are places that people can enter but cars cannot and certain areas that only special transportation vehicles can enter. To make it easier for calculation, a "middle" area is set between blocks and entities. **If the distance from one entity to a block is constant from any perspective, the measure of the area it has observed is constant.**

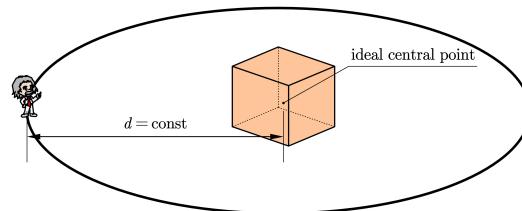


Fig. 6 Explanation of constant measure

- Entities don't keep cars out.

After observing the traffic condition in Hong Kong, we find out that there aren't much cars in Hong Kong. So we assume that the entities will not affect cars.

- Only flights recorded in HKG is considered in this model.

We find that most of the flights passing Hong Kong arrive or take off at HKG while there are no planes observed on Hong Kong's territory. Therefore, we assume that only flights recorded in HKG has an impact.

- **All the data are collected after the pandemic.**

To make sure that our calculation matches the latest information, all the data will be collected after the pandemic(2020).

- **Observation of planes can be identified as a two dimensional picture.**

Considering the assumptions above, there will be no planes flying over the HKG in this model, which means if a person want to observe a plane, its flying route must be flush with line of sight.

- **We take the standard horizontal plane as the datum.**

4.3 Obtaining Data

As we need to build the virtual Hong Kong as real as possible, we need some data provided by the government to make the virtual Hong Kong look like Hong Kong in real life.

Firstly, we need the information of the buildings in Hong Kong, as we need to place the blocks in the virtual Hong Kong. Meanwhile, we need the data about Hong Kong's population and transportation to estimate how much calculation the virtual Hong Kong needs. This kind of data include data of Hong Kong's population, total amount of cars, total traffic volume and total area. These data can be gotten from the government. In addition, we need the data of human eye resolution to confirm an adequate resolution for the virtual Hong Kong. Fortunately, he data can be found from the Internet.

4.4 Notation

α_{lagging}	lagging index
α_c	clarity index
P_{reload}	pixels needed for reload
P_{target}	pixels targeted for use
x	a variable related to reloading speed
y	a variable related to clarity degree
V_{reload}	reloading speed
V_{eye}	eye's reacting speed
C	clarity degree of blocks
C_{eye}	clarity degree of human eye
V_E	normal moving speed of an entity
L_E	dimension of an entity
S_E	measure of an entity
d	distance to the observer
N_E	number of pixels an entity require
ρ_E	density per unit area of an entity
δ_E	proportion of entities in observable area
r_g	observation range of man's eye
r_{street}	observation range of man's eye on streets

4.5 Calculating Local Speed

We need to calculate the reloading speed(V_{reload}). The process is divided into several parts.

First, we need to calculate the lagging index. The formula is shown below.(In this formula, V_{eye} refers to human eyes' reacting speed, which is about 16 Hz.)

$$x = \frac{V_{\text{reload}}}{V_{\text{eye}}} \quad (4.1)$$

$$\alpha_{\text{lagging}} = \begin{cases} 1 & , x \in (0, \infty) \\ 2 - e^{1-x} & , x \in \left[\ln \frac{e}{2}, 1 \right] \\ 0 & , x \in \left(0, \ln \frac{e}{2} \right) \end{cases} \quad (4.2)$$

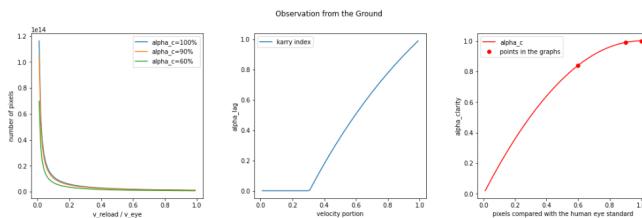


Fig. 7 Pattern of x

Secondly, we need to calculate the clarity index. In this part, we identify C as the clarity degree now and C_{eye} as the clarity degree human eyes can distinguish. According to research data, the clarity degree human eyes can distinguish is about 40960000. The index can be calculated by the formula below:

$$y = \frac{C}{C_{\text{eye}}} \quad (4.3)$$

$$\alpha_c = \begin{cases} 1 & , y \in (1, \infty) \\ -\left(\frac{4096}{4095}\right)^2 (y - 1)^2 = 1 & , y \in \left[\frac{1}{4096}, 1\right] \\ 0 & , y \in \left(0, \frac{1}{4096}\right) \end{cases} \quad (4.4)$$

(The bigger α_{clarity} is, the more comfortable people feel when looking at the virtual city.)

In addition, we need to find out the pixels required when a block move when it's uploaded. First we calculate how much pixels will be uploaded when the city is reloaded (for example, the observer walks a step).

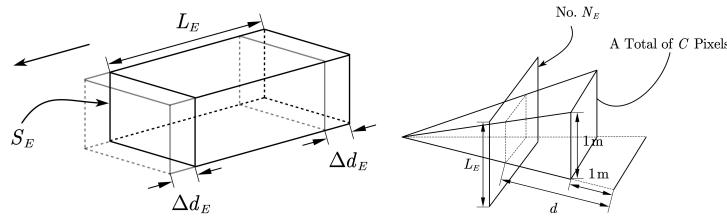


Fig. 9&10 State of observer when he moves

4.5.1 Observing From the Ground

When man observe from the ground, the range that man can observe is about 2 kilometers. So we can find that:

$$\Delta V_E = S_E \times \Delta d_E \quad (4.5)$$

$$N_E = \left(\frac{r - d}{r - 1}\right)^2 C \quad (4.6)$$

$$\frac{2\Delta d_E}{L_E} = 2\Delta d_E L_E C \left(\frac{r - d}{r - 1}\right)^2 \quad (4.7)$$

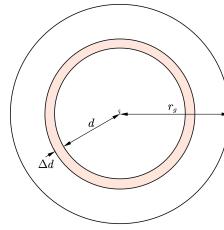


Fig. 8 An example of observing from the ground

Let us assume that there are $N_E = \pi r_g^2 \rho_E$ entities in this area. So in an area of which the radius is d ,

$$N_g = \pi d^2 \rho_E \quad (4.8)$$

$$\delta_E = \frac{N_g(d)}{N_g} = \left(\frac{r_g}{d}\right)^2 \propto d^2 \quad (4.9)$$

$$\frac{1}{r_g^2} [(d + \Delta_d)^2 - d^2] \xrightarrow{\Delta_d \rightarrow 0} \frac{2}{r_g^2} \Delta_d \quad (4.10)$$

$$\frac{2\Delta_d}{r_g} N_g \times P_{\text{Target}} = 2\Delta_d L_E C \frac{r - d}{r - 1} \quad (4.11)$$

$$(4.12)$$

So, when we sum all the pixels required together, we can get the following definition formula:

$$\begin{aligned} P_{\text{reload}} &= \lim_{\Delta d \rightarrow 0} \sum_{i=1}^{\lfloor \frac{r_g}{\Delta d} \rfloor} \left[\frac{2\Delta d}{r_g} \cdot N_g \cdot 2\Delta d L_E C \cdot \left(\frac{r_g - i\Delta d}{r_g - 1} \right)^2 \right] \\ &\stackrel{\text{def}}{=} A \cdot \lim_{\Delta d \rightarrow 0} \sum_{i=1}^{\lfloor \frac{r_g}{\Delta d} \rfloor} \Delta d \cdot \left(\frac{r_g - i\Delta d}{r_g - 1} \right)^2 \\ &= A \cdot \int_0^{r_g} \left(\frac{r_g - x}{r_g - 1} \right)^2 dx \\ &= \frac{4\pi r_g^4 \rho_E \Delta d L_E C}{3(r_g - 1)^2} \\ (A &= \frac{4N_g \Delta d L_E C}{r_g} = 4\pi r_g \rho_E \Delta d L_E C) \end{aligned}$$

4.5.2 Observing From the Air

When man observe from the sky, the average observing range r_a is about 1300 kilometer.

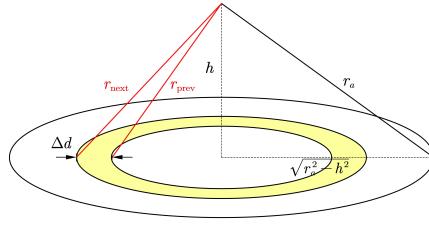


Fig. 11 An example of observing from the air

For entities in the marked part, for $\Delta_d \rightarrow 0$ ($\Delta_d \ll h$), we take r_{next} as the distance from the observer to all the entities in the ring. So we can get the formula:

Number of pixels within a Δd range

$$\begin{aligned}
 &= \frac{2\Delta d}{\sqrt{r_a^2 - h^2}} \cdot N_g \left(\sqrt{r_a^2 - h^2} \right) \cdot 2\Delta d E L_E C \cdot \left(\frac{r_a - r_{\text{next}}}{r_a - 1} \right)^2 \\
 &= \frac{4\pi\sqrt{r_a^2 - h^2}\rho_E\Delta d E L_E C}{(r_a - 1)^2} \cdot \Delta d (r_a - r_{\text{next}})^2, \text{ where } r_{\text{next}} = \sqrt{h^2 + (i\Delta d)^2} \\
 &\stackrel{\text{def}}{=} B \cdot \Delta d (r_a - r_{\text{next}})^2 \\
 &(B = \frac{4\pi\sqrt{r_a^2 - h^2}\rho_E\Delta d E L_E C}{(r_a - 1)^2})
 \end{aligned}$$

$$r_{\text{next}} = \sqrt{h^2 + (i\Delta d)} \quad (4.13)$$

In conclusion, we can get a summarized formula:

$$\begin{aligned}
 P_{\text{reload}} &= \lim_{\Delta d \rightarrow 0} \sum_{i=1}^{\lfloor \frac{\sqrt{r_a^2 - h^2}}{\Delta d} \rfloor} B \Delta d \cdot \left(r_a - \sqrt{h^2 + (i\Delta d)^2} \right) \\
 &= \int_0^{\sqrt{r_a^2 - h^2}} B \cdot \left(r_a - \sqrt{h^2 + x^2} \right) dx \\
 &= B \left[x \left(h^2 + r_a^2 \right) + r_a \sqrt{h^2 + x^2} \cdot \left(-\frac{h \cdot \sinh^{-1}(\frac{x}{h})}{\sqrt{(\frac{x}{h})^2 + 1}} - x \right) + \frac{x^3}{3} \right] \Big|_0^{\sqrt{r_a^2 - h^2}} \\
 &= B \left[\sqrt{r_a^2 - h^2} \left(h^2 + r_a^2 \right) - r_a^2 \cdot \left(\frac{h \cdot \sinh^{-1}(\frac{\sqrt{r_a^2 - h^2}}{h})}{\frac{r_a}{h}} + \sqrt{r_a^2 - h^2} \right) + \frac{(r_a^2 - h^2)^{\frac{3}{2}}}{3} + r_a h^2 \sinh^{-1}(0) \right] \\
 &= \frac{4\pi\sqrt{r_a^2 - h^2}\rho_E\Delta d E L_E C}{(r_a - 1)^2} \cdot \left(\sqrt{r_a^2 - h^2} \cdot h^2 - h^2 \cdot r_a \sinh^{-1} \left(\frac{\sqrt{r_a^2 - h^2}}{h} \right) + \frac{(r_a^2 - h^2)^{\frac{3}{2}}}{3} \right)
 \end{aligned}$$

4.5.3 Observing From Planes

The third condition is observing from planes, we abstract it as a regional calculation on the ground. The average observation range of human height is far less than r_a under

this circumstance. We find out the following formula:

$$P_{\text{Reload}} = \frac{4\pi L_{\text{airport}}^4 \Delta d_E L_E C}{3 (L_{\text{airport}} - 1)^2} \quad (4.14)$$

According to the data online, we find that L_{airport} is about 3452.5979 meter. So we find out that :

$$V_{\text{plane}} \approx 900 \text{m/s} \quad (4.15)$$

$$L_{\text{plane}} \sqrt[3]{58.82 \times 12.61 \times 17.39} \approx 23.452 \text{m} \quad (4.16)$$

$$\rho_{\text{plane}} = 0.163 \times 10^{-6} \text{m}^{-2} \quad (4.17)$$

$$(r_g \approx r_a >> L_{\text{airport}}) \quad (4.18)$$

4.5.4 Observing From Roads

Finally, we will consider the condition where observers walk on streets. To facilitate our calculation, we see the streets as straight lines which have width shown in the figure below.

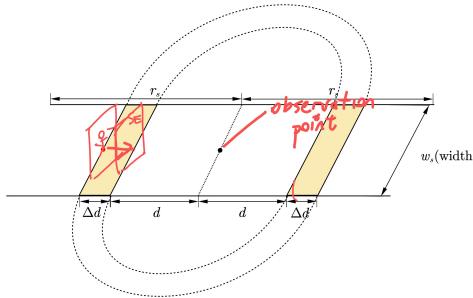


Fig.12 An example of observing from roads

From this picture, we can figure out the relationship of P_{reload} with other determining factors in the following formula:

$$\begin{aligned} P_{\text{reload}} &= \lim_{\Delta d \rightarrow 0} \sum_{i=1}^{\lfloor \frac{r_s}{\Delta d} \rfloor} \rho_E \Delta d \cdot W_s \cdot C \cdot \left(\frac{r_s - i \Delta d}{r_s - 1} \right)^2 \cdot \frac{S_E}{L_E} \cdot 2 \Delta d_E \\ &= \frac{2 \rho_E W_s C S_E \Delta d_E}{(r_s - 1)^2 L_E} \int_0^{r_s} x (r_s - x)^2 dx \\ &= \frac{r_s^4 \rho_E W_s C S_E \Delta d_E}{6(r_s - 1)^2 L_E} \end{aligned}$$

4.6 Calculating Speed(# undetermined)

When we calculate the byte required to store this virtual city, we also need to find out how many times it calculates in one second.

beiyong

$$P_{\text{calc_cloud}} = V_{\text{reload}} \cdot u_E \cdot (2L_E^2 \cdot \Delta_{d_E}) \quad (4.19)$$

$$P_{\text{calc_local}} = V_{\text{reload}} \cdot \sum_{E \in \text{entity}} \left(\frac{r-d}{r-1} \right)^2 L_E C \quad (4.20)$$

Then we need to calculate how much storage space it requires to maintain this virtual city ($S(\text{storage})$), it is divided into S_{cloud} and S_{local} . We can find the following formula:

$$S_{\text{cloud}} = \sum_{\text{number of entity types}} \text{number of parameters} \cdot \text{number of entities} \quad (4.21)$$

$$S_{\text{local}} = \sum_{E \in \text{entity}} N_E = k \sum_{E \in \text{entity}} \left(\frac{r-d}{r-1} \right)^2 L_E C \quad (4.22)$$

To further evaluate the storage level of virtual city, we introduced a comprehensive index to describe it. It's related to α_c , α_{lagging} , α_{fit} , α_{reload} . Their weight are shown below.

$$K \stackrel{\text{def}}{=} \begin{cases} \alpha_c & + (0, 1) W_1 \\ \alpha_{\text{lagging}} & + (0, 1) W_2 \\ \alpha_{\text{fit}} & + (0, 1) W_3 \\ \alpha_{\text{reload}} & + (0, 1) W_4 \end{cases} \quad (4.23)$$

We have to note that $\sum W_i = 1$, "+" refers to it has a positive impact on the virtual city.

5 Strengths and Weaknesses

Strengths

- **Authenticity**

All the data mentioned in our model is provided by authority in Hong Kong, so it can simulate the real Hong Kong well.

- **Simplicity**

All the variables in our model is simple, so there isn't much calculation needed and is easy to handle.

Weaknesses

- **Inaccuracy**

Due to our assumptions, we can not describe the shape of blocks accurately. This may lead to inaccuracy of the whole virtual city.

- **Non-real Time**

There are many kinds of observation angles in our model, so we can not realize the real time upload in our model.

References

- [1] EPSG:2326 Hong Kong 1980 Grid System, MapTiler Team
<https://epsg.io/2326>