Intelligent Logging System with Enhanced Accuracy

by

Team Grid

Joonha Jung 2011272071
Jihwan Kim 2014147588
Faiz Wong 2015147555

Prof. Won-Suk Lee
TA Ji-ho Kim

Contents

1. Abstract

This research is about developing intelligent logging system with enhanced accuracy. We will be able to obtain refined log data through "grid mapping algorithm" single spot and multiple spots respectively. This pure data acquired from our system will make a big contribution on data science and other various fields due to its accurate nature.

2. Necessity

The original parser analyzing the video has several limitations with regards to detecting all object perfectly. For instance, it considers objects which are same into different objects when passing through certain obstacles. Through our project we hope to generate proper result for convenience, to overcome this inconvenience of having to compare every second of video files.

There are 50,000 of CCTVs in Seoul officially installed by government. If we consider the ones installed privately, the number of it may be over millions and it's obvious that the number has to be increased in the future. Therefore, this kind of object detecting technology will be more important as time goes by.

3. Introduction

The purpose of this research is to enhance the accuracy of CCTV log data just by utilizing the database system to create an intelligent logging system with adaptive grouping. Note that this research will not be attempting to improve the parser's limitations. An assumption will be made throughout the research that the parser's accuracy to detect objects in a video stream will remain constant throughout the logs, and the role of the fore-mentioned intelligent logging system that will be implemented is to make corrections in the logs, thus increasing the accuracy of the CCTV log data. Before moving on to the methodologies, it is important to define what accuracy is with respect to this research.

Data accuracy in its general definition refers to whether the data values stored for an object are the correct values. Since this research is concerned with the accuracy of data provided by a CCTV, it is best to first recognize the kind of data generated by the CCTV. The following is the schema construct of the CCTV log:

Attribute	Data Type
Frame ID	int
Object ID	string
CCTV ID	string
Grid ID	string
X	int
Y	int
Width	Int
Height	Int
Speed	Int
Size	Int
Appear	String
Color	String

Table 1

This research is concerned with the the accuracy of Object ID, whether the Object ID is consistent with the real representation of the object associated in the universe of discourse. It is important to define the problems that might cause an inaccuracy.

Problem

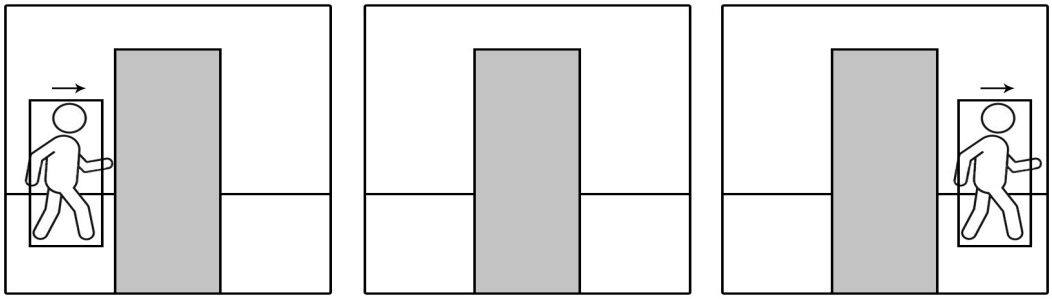


Figure 1

Suppose a CCTV camera is capturing object A moving from left to right as shown in three frames in Figure 1. A move from left to right in a manner that there is an object in between that obstructs the vision of the CCTV camera as shown in frame 2. The parser might label A as Object ID = 1 in frame 1. However, since there is an obstacle in frame 2, the parser has lost track of A since there are no more sightings of A in the CCTV camera's point of view. At this point the parser might have mistakenly assumed that A has already exited the location. However, as shown in frame 3, A has reappeared as any human agent would have predicted, however it is possible that the parser might have interpreted the object in frame 3 as a new object thus labeling it as Object ID = 2.

This research will be attempting to make corrections in the original CCTV logs by altering the values of Object ID to a correct one. This will be done by analyzing multiple CCTV logs provided by multiple cameras installed at multiple locations in a system.

This research will make the following assumptions:

I. A CCTV log is generated by a single CCTV in a system with well documented identification of CCTV installations
II. Every CCTV installed has a unique CCTV ID
III. Every location has a unique Location ID
IV. A single CCTV monitors only one location
V. A location can have multiple CCTV installed for monitoring purposes
VI. All logs have the same time of recording starting from Frame ID = 1 and increase incrementally as well as consistent throughout the whole system

The intelligent logging system and adaptive grouping implementation from this research will be able to track people on a map by calculating their position from original CCTV logs. It is then possible to link many cameras and track a person through an entire building or area. Based on the fact that this logging system is automated and not hard coded, the entire process is repeatable, independent of the CCTV system and can be improved upon frequency of usage. This can be done by multiple methodologies that will later be introduced in this report.

4. Methodologies

The purpose of this research is to The CCTV logs used as an input to create this intelligent logging system could be in a form of an offline data stream or an online data stream. If it is in the case of an offline stream, bulk of CCTV logs will be generated in batches and processed to produce the expected output. This research is concerned with the processing phase of this process. If it is in the case of an online stream, individual addition of a continuously generated logs will be processed to produce the expected output. Individual logs in a form of a tuple will be analyzed by relationships to other data in a data repository. In order to do this, it is necessary for data in individual logs to first be organized and stored in a database. A relational data model will be used throughout this research. The following is the data model used for this purpose:

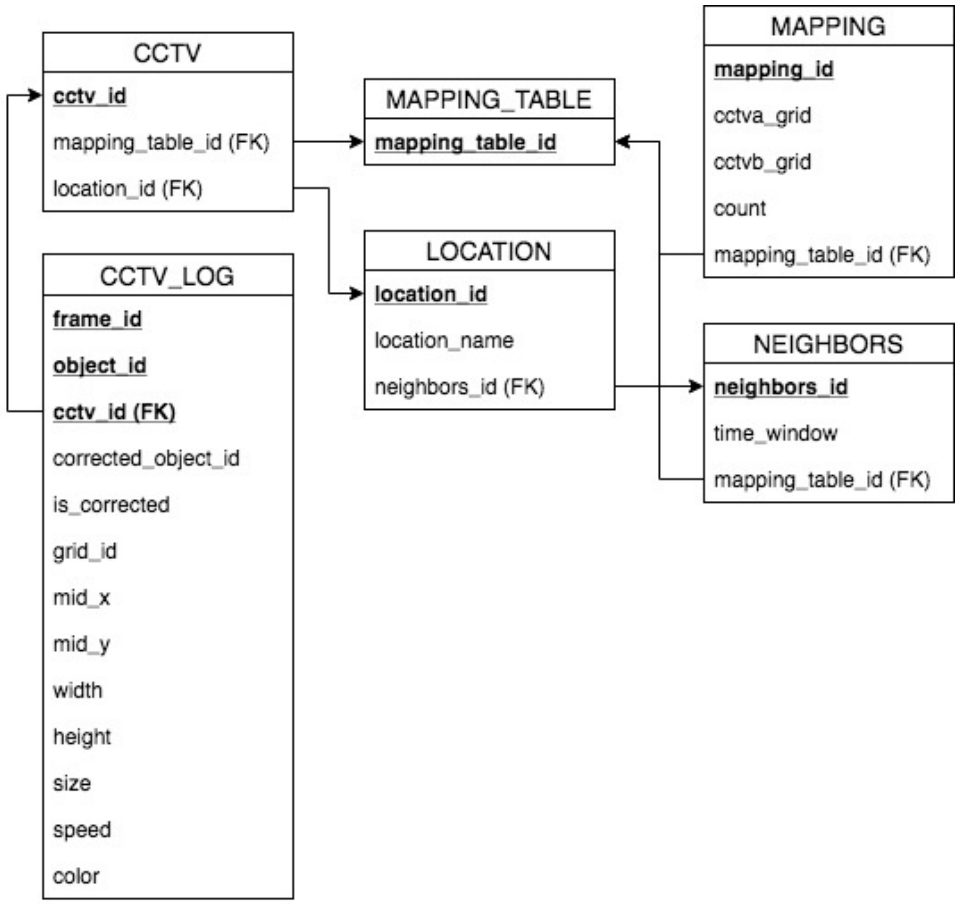


Figure 2

Relation list:

I. MAPPING_TABLE
II. MAPPING
III. CCTV
IV. LOCATION
V. NEIGHBORS
VI. CCTV_LOG

MAPPING_TABLE

Attribute	Data Type	
mapping_table_id	int	primary key

- A MAPPING_TABLE maps two CCTVs, A and B by area and time
- A.cctv_id is always smaller than B.cctv_id
- When A.location_id and B.location_id are different but connected, this indicates that they are NEIGHBORS and the value of A.location_id.neighbors_id and B.location_id.neighbors_id is the same
- The NEIGHBORS.time_window of A.location_id.neighbors_id and B.location_id.neighbors_id is the relative time window between A and B

MAPPING

Attribute	Data Type	
mapping_id	int	
cctva_grid	int	
cctvb_grid	int	
count	int	
mapping_table_id	int	

- MAPPING is created when one grid in A is the same area as one grid in B
- The grid in A is cctva_grid while the grid in B is cctvb_grid
- count is the counter for every instance that cctva_grid and cctvb_grid has the same value
- mapping_table_id is the map that this mapping represents: cctva_grid is a grid that belongs to A.mapping_table_id, the same applies to cctvb_grid with respect to B.mapping_table_id

CCTV

Attribute	Data Type	
cctv_id	int	primary key
location_id	int	foreign key
mapping_table_id	int	foreign key

- cctv_id monitors location_id
- One location_id can be monitored by two CCTVs

LOCATION

Attribute	Data Type	
location_id	int	primary key
location_name	string	
neighbors_id	int	foreign key

- Two location_ids have the same neighbors_id value indicates that they are NEIGHBORS of neighbors_id

NEIGHBORS

Attribute	Data Type	
neighbors_id	int	primary key
mapping_table_id	int	foreign key
time_window	int	

- Two location ids connected to each other are NEIGHBORS

CCTV LOG

Attribute	Data Type	
frame_id	int	primary key
object_id	int	primary key
corrected_object_id	int	
is_corrected	boolean	
cctv_id	int	foreign key
grid_id	int	
mid_x	int	
mid_y	int	
width	int	
height	int	
speed	int	
size	int	
appear	int	
color	string	

To understand the data model being used, it is required to first understand the approach this research takes on solving the problems mentioned.

In order to track an object on a map, it is necessary to know the position of the object. Since it is impossible to create a map with absolute dimensions representative to the universe of discourse due to knowledge limitations that can be extracted from CCTV logs alone, an attempt will be made to create a relational map between two CCTVs' map of vision. This relation will be called a mapping table.

To further understand what a mapping table really is, it is best to first understand what is Grid ID in a CCTV log generated by the parser in Table 1.

A CCTV footage will be divided into 100 sections of equal area called grid. The grid will then be labeled 0 to 99 as shown in Figure 4. If an object is detected by the parser in a footage, a box with width of Width and height of Height will represent the object. Values X and Y represent the middle point between the maximum X position and minimum X position of the box as well as the maximum Y position and minimum Y position of the box respectively. The position of coordinate (X, Y) in the CCTV footage will determine the Grid ID. If the coordinate (X, Y) is in the boundaries of grid 0 in Figure 4 then Grid ID = 0.

4.1 Mapping Method 1

Suppose two CCTVs A and B are monitoring a LOCATION i at two different angles. There will be an area of intersection between the area of vision of A and B. A mapping table represents the relation between the two intersected areas of vision. This can be done by assigning a relationship between all the grids in the intersected areas of vision of A and B. For example, if the grid 37 in A intersects with grid 32 in B, then a relation between grid 37 in A and grid 32 in B is created. This relation is called mapping. A mapping then can also represent the same area in the universe of discourse. This is useful in order to determine whether an object is the same person in two CCTV footages. This is because it is physically impossible for two persons to be in the same area at the same time.

It is too labor expensive to hard code a mapping table for all two CCTVs monitoring the same location. Therefore, this research attempts to create an automated procedure to perform this task using the supervised learning technique.

Suppose there is only one person in a room with two CCTV cameras A and B installed. If the person is in an area represented by the area of intersection between A and B's vision, a mapping could be created between the two grids that represent the person's position in A and B. One might argue that this is not an accurate representation of the person's real position in the universe of discourse due to two facts:
The area represented by the grids in A and B are of different sizes
Negligence of the difference of the person's size in two different footages due to the sole usage of midpoint X and Y to represent the person if there is a difference between the distance of the person to the cameras installed in the universe of discourse

To overcome this problem, instead of immediate mapping when an object is detected to be at the area of intersection between a and b's vision, a counter will be used to count the instances of the occurrence. If the count exceeds a certain threshold, then only the two grids are considered the same area in the universe of discourse.

The following is the pseudocode of this algorithm with the assumptions that:
It has already been known that a and b are installed at the same location
There is only one object in the location

In this research, this algorithm will be referred to as the single-spot learning algorithm.

Input: Two CCTV log data streams DA and DB in the form of a CCTV_LOG tuple
Output: A mapping table that contains a set of mappings between the CCTVs that generated DA and DB

```
1  for each new transaction in DA and DB:
2      check_mt_a = SELECT mapping_table_id FROM CCTV WHERE cctv_id = DA.cctv_id
3      check_mt_b = SELECT mapping_table_id FROM CCTV WHERE cctv_id = DB.cctv_id
4      if check_mt_a == check_mt_b:
5          the_mapping_table_id = check_mt_a
6      else:
7          mapping_table_id = SELECT COUNT(1) FROM MAPPING_TABLE + 1
8          INSERT INTO MAPPING_TABLE VALUES (the_mapping_table_id)
9
10     check_m = SELECT mapping_id FROM MAPPING WHERE cctva_grid = DA.grid_id AND
       cctvb_grid = DB.grid_id
11     if check_m == 0 :
12         the_mapping_id = SELECT COUNT(1) FROM MAPPING + 1
13         INSERT INTO MAPPING VALUES (the_mapping_id, DA.grid_id, DB.grid_id, 1,
           the_mapping_table_id)
14     else:
15         the_mapping_id = check_m
16         UPDATE MAPPING SET count = count + 1 WHERE mapping_id = the_mapping_id
```

Algorithm 1: Single spot learning algorithm version 1

For this research, we have implemented our own video parser using the Tensorflow Object Detection API. This API is an open-source framework built on top of Google's machine learning library Tensorflow that makes it easy to deploy object detection models. This framework however is only capable of identifying objects in a single image. Due to this limitation, we split our footages into 30 images for every second since our videos were recorded in 30 frames per second. We then run the API on every image, and recorded the results in a .csv file. We then use the data to produce a mapping table using similar method as in the single spot learning algorithm.

For the footages, we have crafted a few scenarios that may enable the algorithm to create mapping tables. We have also acted a few scenario scenes that may produce the unwanted problems mentioned in Problem 1 and Problem 2. This will enable us to test implementations on error correcting algorithms in the future. The following are three scenarios that have been acted out by us for the purpose of this research. Note that for the sake of simplicity the path drawn in Figure 5.1, Figure 5.4 and Figure 5.7 do not represent the actual path taken by the actor but the area covered.

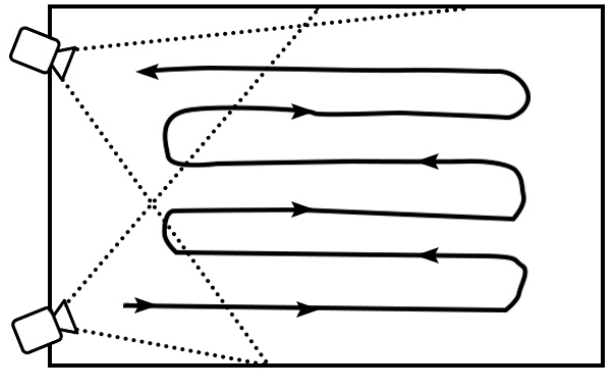


Figure 3

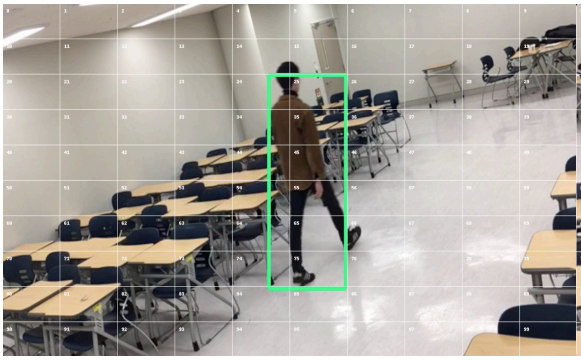


Figure 4.1

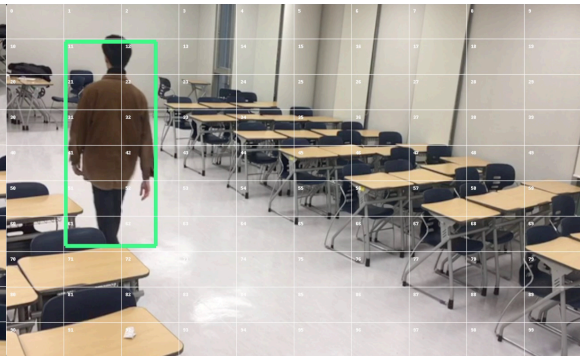


Figure 4.2

Mapping table implementation

We have tried implementing Algorithm 1 on Scenario 1 and successfully created a mapping table between two CCTVs with areas of vision as shown in Figure 6.2 and Figure 6.3 respectively.

Suppose J and K represents the grid_id of the position of the object detected in Figure 6.2 and Figure 6.3 respectively and L is the number of times the object is detected at J and K at the same frame_id. The following is the output of the mapping table implementation on Scenario 1 in the form of a set of elements (J,K): L.

(37, 32): 26, (46, 43): 15, (45, 31): 13, (36, 31): 12, (57, 57): 11, (48, 45): 11, (18, 20): 10, (47, 43): 10, (19, 21): 10,
(28, 21): 9 , (55, 55): 9 , (63, 56): 9 , (36, 30): 9 , (56, 44): 8 , (62, 67): 8 , (56, 58): 7 , (18, 21): 7 , (61, 55): 7 ,
(63, 53): 7 , (54, 42): 7 , (64, 56): 6 , (38, 44): 6 , (46, 42): 6 , (37, 43): 5 , (62, 54): 5 , (54, 52): 5 , (28, 20): 5 ,
(28, 31): 5 , (27, 30): 5 , (45, 42): 5 , (48, 46): 4 , (19, 20): 4 , (62, 64): 4 , (53, 53): 4 , (58, 46): 3 , (38, 45): 3 ,
(62, 66): 3 , (61, 66): 3 , (61, 56): 3 , (44, 42): 3 , (39, 24): 2 , (55, 54): 2 , (53, 52): 2 , (55, 41): 2 , (36, 20): 2 ,
(19, 22): 2 , (37, 31): 2 , (38, 23): 2 , (39, 23): 2 , (56, 68): 1 , (56, 57): 1 , (58, 56): 1 , (39, 34): 1 , (39, 33): 1 ,
(46, 44): 1 , (63, 57): 1 , (61, 65): 1 , (62, 53): 1 , (62, 63): 1 , (63, 63): 1 , (44, 32): 1 , (45, 32): 1 , (45, 41): 1 ,
(35, 31): 1 , (29, 22): 1 , (28, 23): 1 , (38, 33): 1 , (29, 33): 1 , (37, 44): 1

Referring to the first element in the set, the values 37, 32 and 26 will be cctva_grid, cctvb_grid, and count of a mapping of a mapping_table that maps the two CCTVs in Figure 4.1 and Figure 4.2 respectively. This mapping table could then be used to make corrections in CCTV logs that will generated by the CCTVs in the future.

4.2 Mapping Method 2

Unfortunately, there are some reasons why mapping method 1 is not applicable. Firstly, it is found that the count of mapping occurrences is insignificant. It is discovered that the object location represented by the midpoint attribute in the log is the significant factor to creating the mapping table.

If there is a grid mapped with several girds, the count of mapping occurrences is not sufficient. There are too many errors if we just use a gird to make the mapping table.
Look at this case. In this case, there are three different points and each of them represents a certain object which is captured by camera. But if we use the former mapping table it is nearly impossible to distinguish accurately.
Let the objects in cctv_a are object 1, 2 and 3 from very left respectively and in cctv_b to be object 4, 5 and 6. In the universe of discourse, object 1 is object 4, object 2 is object 5 and object 3 is object 6. However, using the former mapping table the object 13 will be mapped to object 4 which is not accurate. This kind of inaccuracy is due to the usage of the instance count method. Due to this reason we are proposing the different method to deal with this problem.

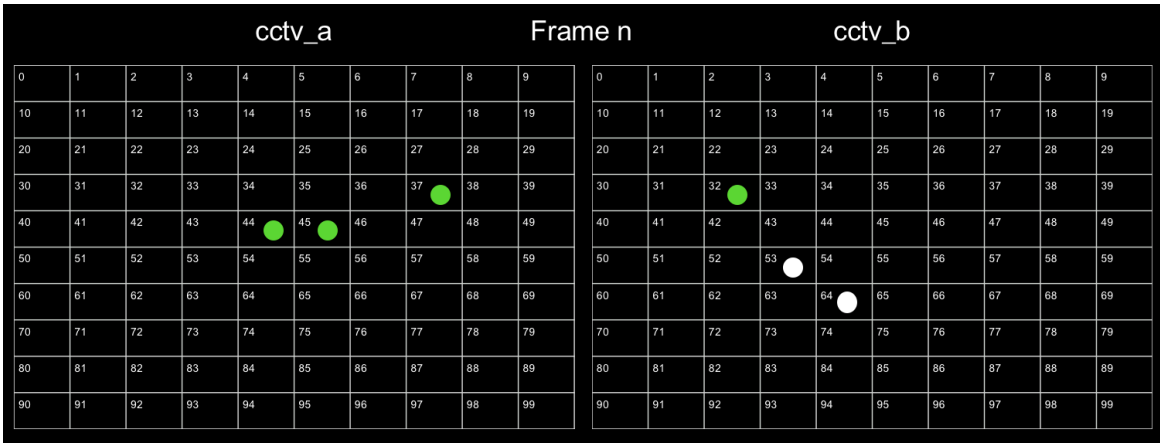


Figure 5

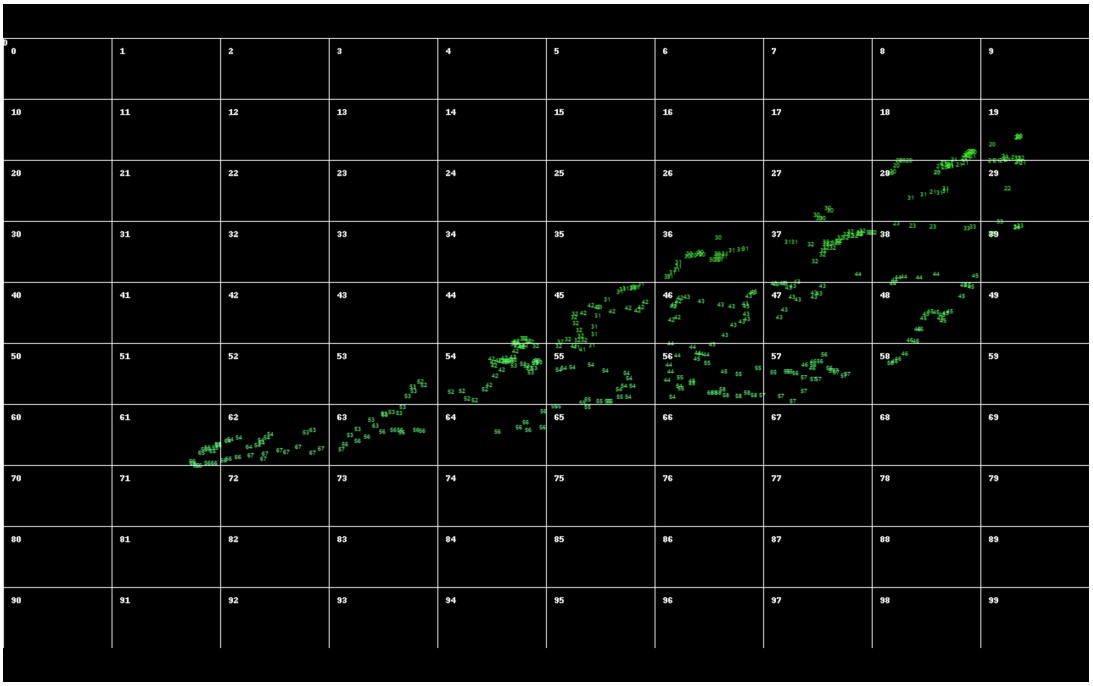


Figure 6.1

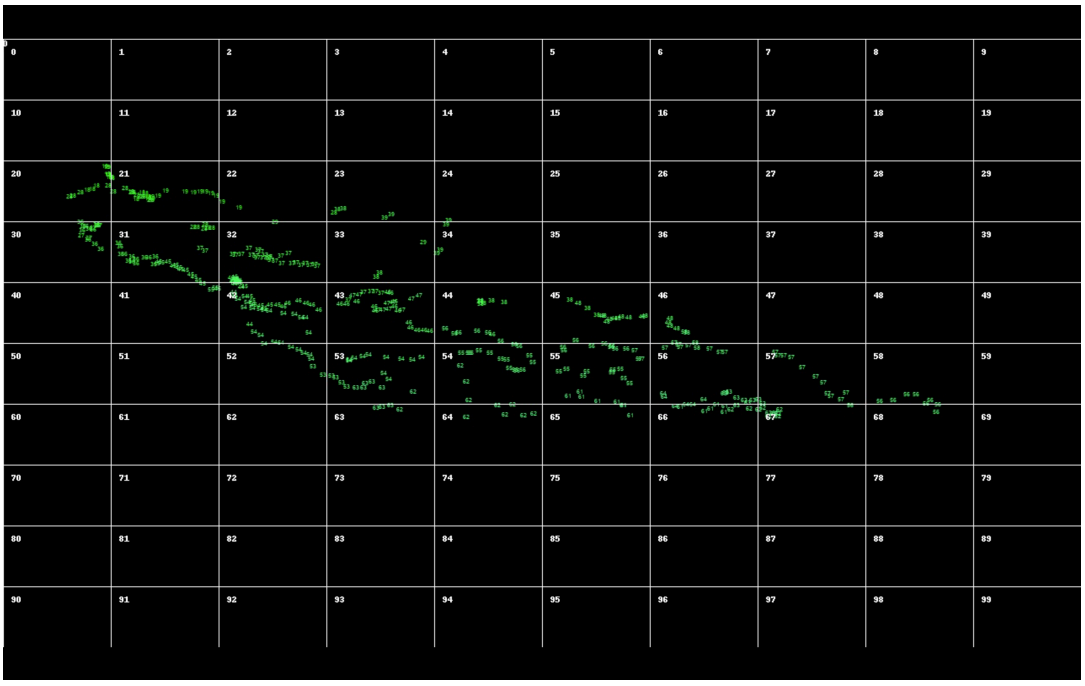


Figure 6.2

Instead of counting the instances, we recorded the object locations when the mapping is created. We discovered that in each grid where there is a mapping, there are clusters of elements with distinct labels. Therefore, it is possible to classify a space in the grid to the corresponding label according to the boundary of the clustered elements.

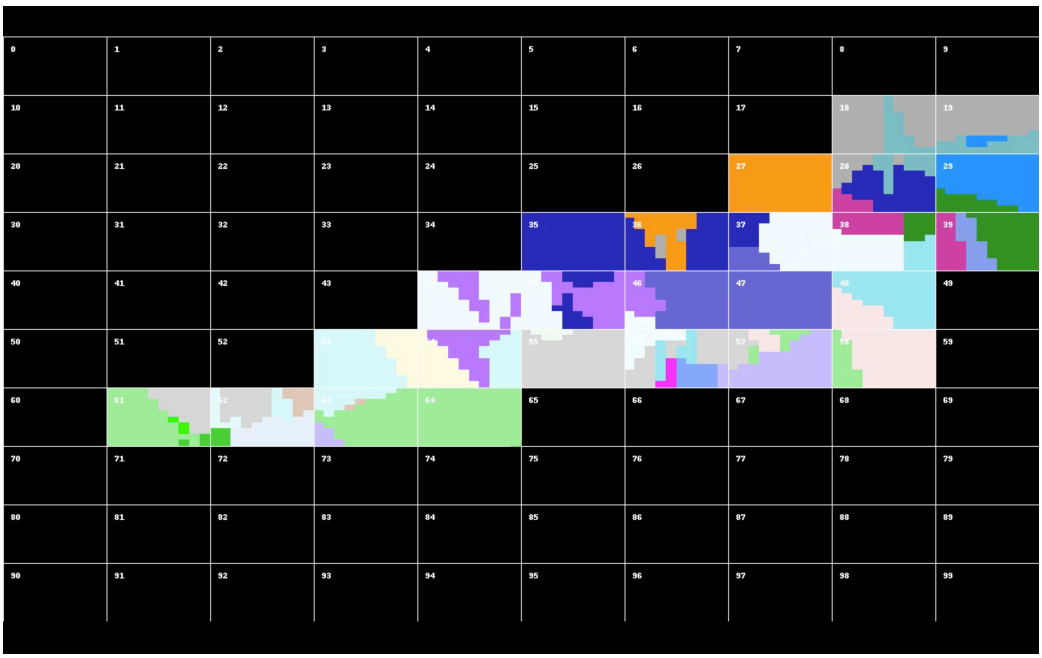


Figure 7.1

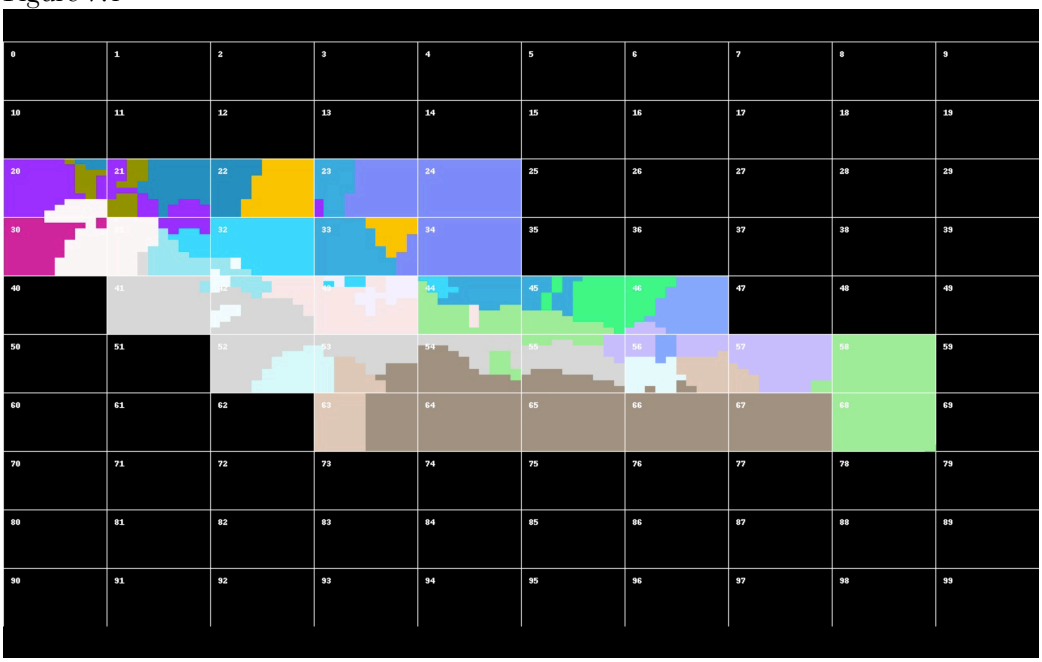


Figure 7.2

Figure 7.1 and figure 7.2 show classified space corresponding to the label. But as you can see it also doesn't look that accurate. For instance, look at this case. What if the new data is inserted like the white spot. According to our mapping table we will consider it is mapped with grid 38. But in the real world, due to the lack of the information, we cannot be sure whether it is mapped with grid 38. So we decided to make our mapping table more accurate.

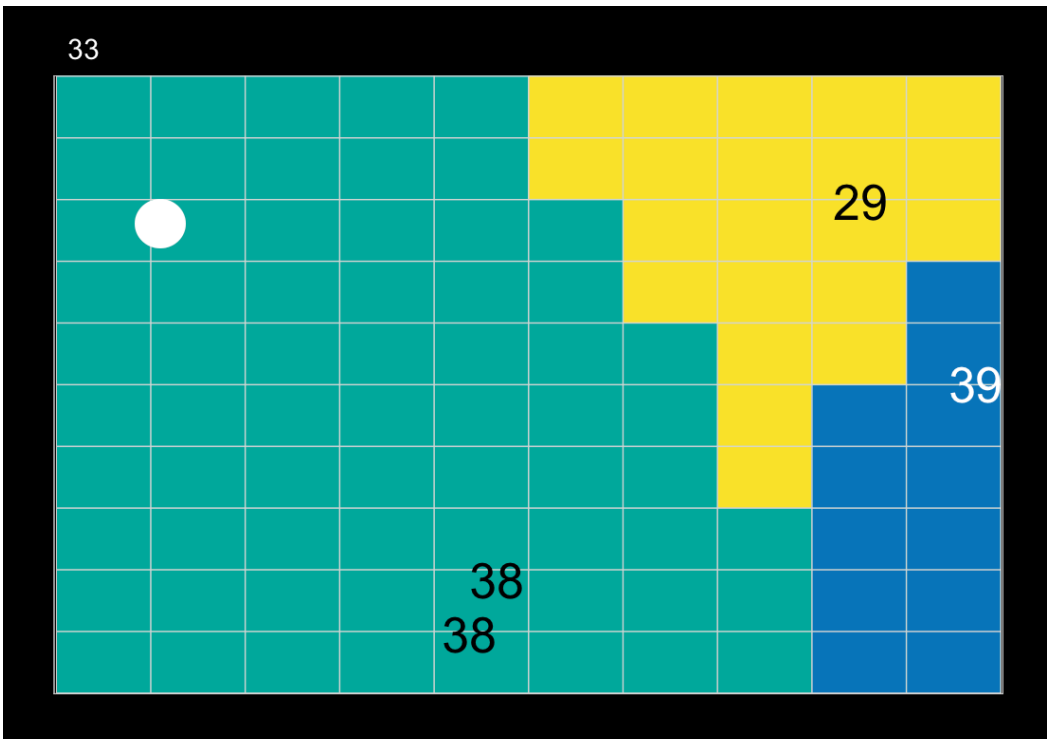


Figure 8

Therefore, instead of mapping the whole grid, we will only map a part of it as shown in Figure 8. We also created a mapping table that resembles the mentioned mapping table with higher resolution as shown Figure 9.2.

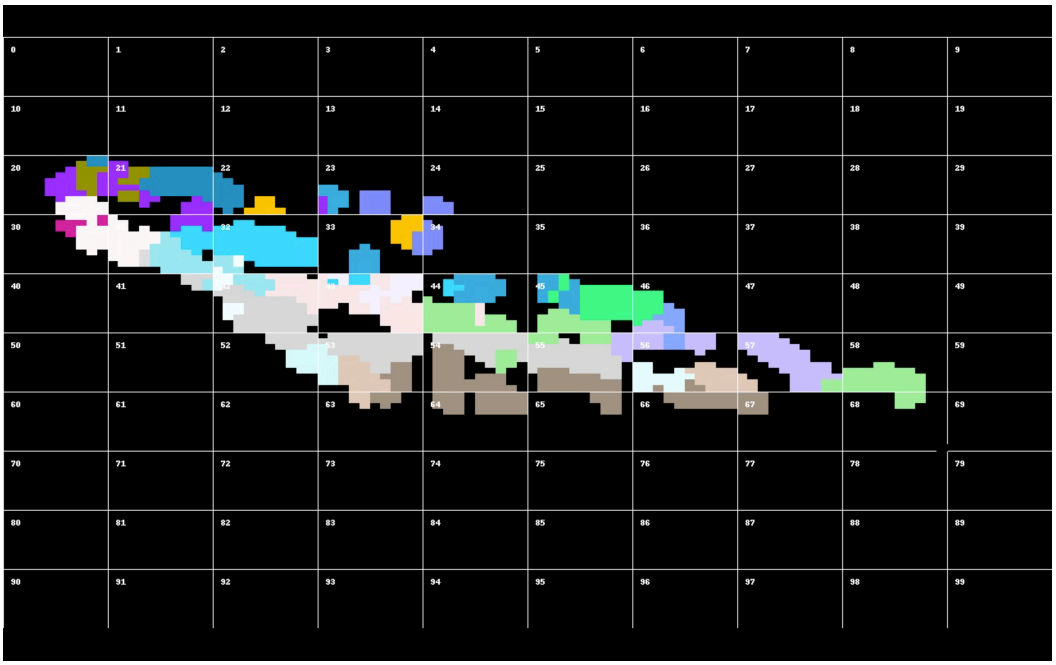


Figure 9.1

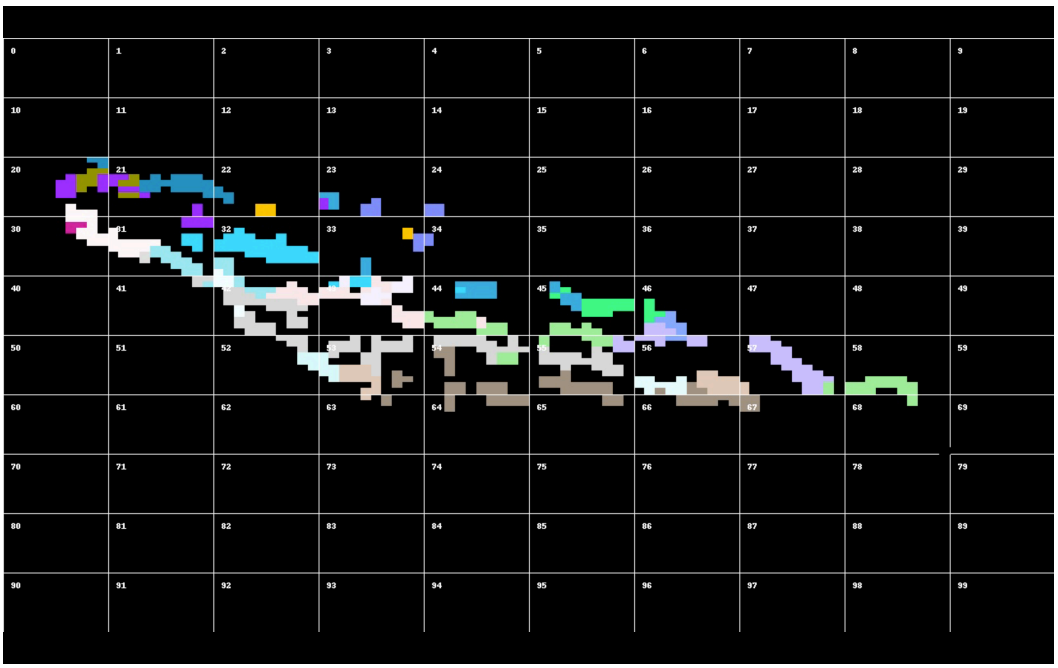


Figure 9.2

4.3 Final mapping method

The second methodology that we utilized to create the mapping table worked sufficiently well, however, there was a limitation which only allowed us to train the table when a single object is appeared. Therefore we had to find an alternative solution to enable us to train the mapping table regardless of the number of objects appearing in the same frame.

Finally, we decided to use the beauty of data. Since we started our project, our main goal has been to create an accurate mapping table. However, despite our multiple attempts, we realized that a "perfect" mapping table cannot exist. Therefore, we redesigned our whole process with 3 phases, mapping, validation and correction phases.

4.3.1 Mapping phase

Firstly, the mapping phase.
The mapping phase is slightly different from what we have done so far to get the mapping table. In this mapping phase, we are just counting the occurrence of objects regardless of the number of objects for each frame by grid. But the fact that we are using the grid to minimize the errors is same. In this new method we are using 100 x 100 size of grid different from using 10 x 10 size of grid previously. The reason that we have used smaller size of grid is 10 x 10 grid cannot reflect such complex data and our data is quite complex and it can be interpreted in many ways. Actually the pixel by pixel mapping may be an ideal way but unfortunately it requires tremendously large size of data so we couldn't try this for this project.

As we trained our mapping table every single frame, the certain phase for learning has been gone. It becomes smarter and smarter when the time goes by. Therefore, the result for the mapping phase is a mapping table which is not perfectly confirmed. We can use this when the grid is validated.

4.3.2 Validation phase

Secondly, validation phase.
This phase is for finding a valid mapping for each grid. After the mapping phase, we will get the mapping table which has an 1:1 or an 1:N relationship. For an 1:1 relationship, we don't really have to consider many things but the number of occurrences, we make it as a threshold later. In the most cases, the result was an 1:N relationship. When the relationship is an 1:N, we chose the most frequently occurred grid to map. For

instance, if certain grid is mapped into several grids from another CCTV [Figure 1]. We will choose the grid which has counted by 4 times [Figure 2]. And we will call this mapping as a "true mapping."
Therefore, the result of this phase is the mapping table which has true mappings.

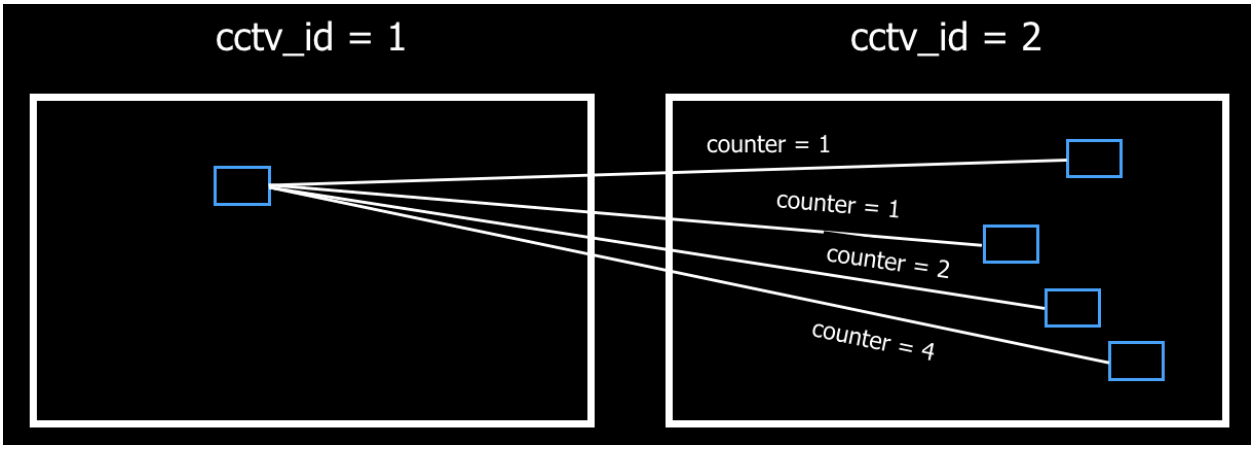


Figure 9.1

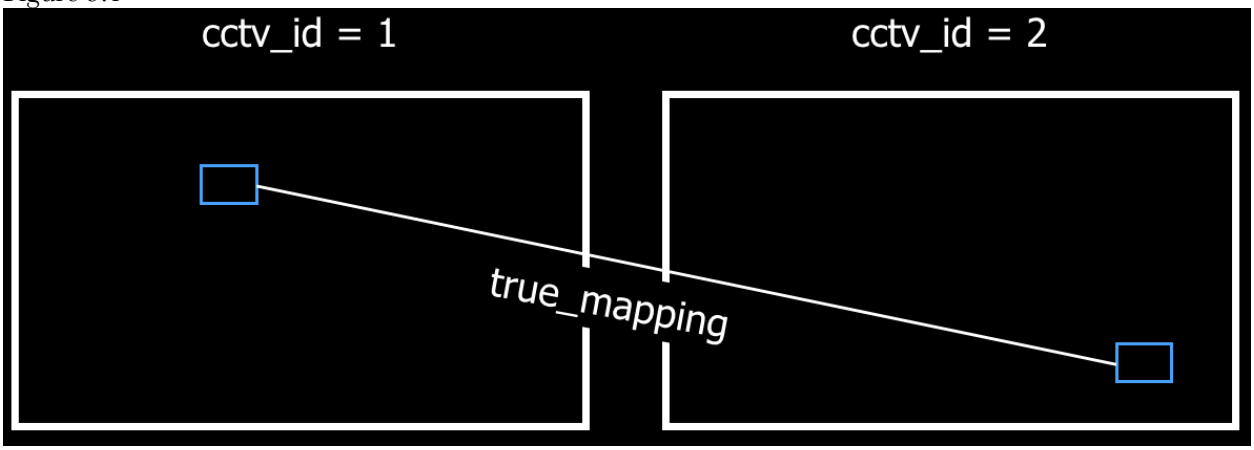


Figure 9.2

4.3.3 Correction phase

Last, correction phase.
In this phase, we will assess whether the "true mapping" is correct. In the validation phase, we found the true mappings, but we suspected whether the true mapping is always correct. And we found the cases that the true mapping may not be correct all the time. For example, if the number of occurrences is not enough compared to the others, we cannot be sure whether it happened by accident or not. In order to avoid this kind of ambiguity, we decide to use a certain number as a threshold.

It's quite obvious that every mapping table has a different threshold. So as to find this magic number, we tried to put some different arbitrary numbers. Following table shows the results that we have done to validate the mapping table.

Table 1 (One object (scene 1), real recording video)

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6
Threshold	0	1	2	3	4	5
Mapping	412	412	412	412	412	412
MAX N	5	5	5	5	5	5
1:N	76	76	76	76	76	76
Accuracy	77.88%	91.50%	99.25%	100.00%	100.00%	100.00%

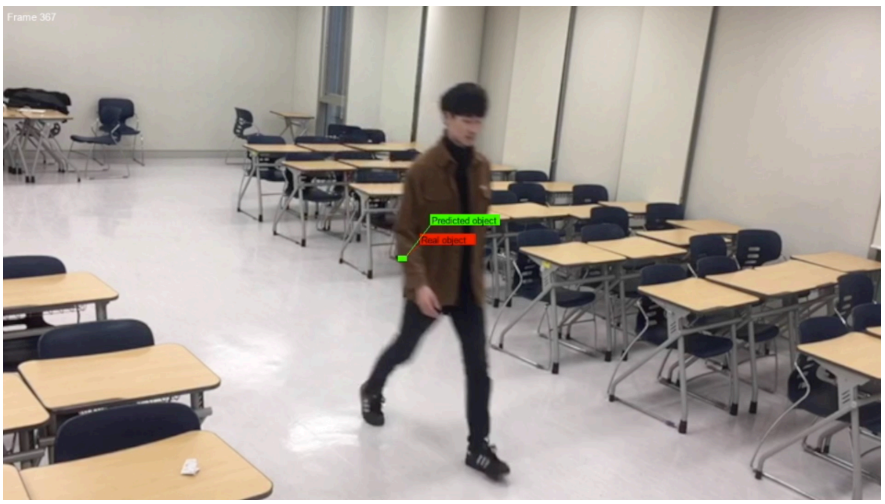
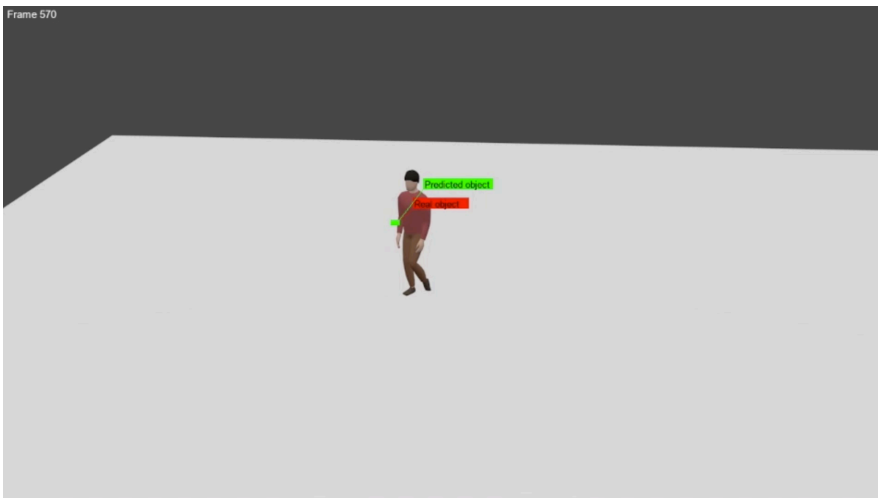


Table 2 (Two objects (scene 2), real recording video)

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6
Threshold	0	1	2	3	4	5
Mapping	420	420	420	420	420	420
MAX N	27	27	27	27	27	27
1:N	43	43	43	43	43	43
Accuracy	28.18%	27.77%	33.30%	60.14%	68.27%	100.00%

Table 3 (One object (scene 3), 3D simulation)

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6
Threshold	0	1	2	3	4	5
Mapping	942	942	942	942	942	942
MAX N	7	7	7	7	7	7
1:N	189	189	189	189	189	189
Accuracy	73.17%	90.03%	95.20%	99.25%	100.00%	100.00%



As we expected, every mapping table, which reflects different locations respectively, has a different magic number. So, using this threshold, we will assess whether the true mapping is really valid. If the mapping is true and the number of occurrence is greater than the threshold, we will consider the two objects which have been detected from two different CCTVs as a same object.

Database tables

Since our method has been changed, our database schema also changed. Following tables are the database schemas which has been changed. Surprisingly, we can distinguish multiple objects only with three tables.

CCTV_LOG

Field	Type
cctv_log_id	int
frame	int
cctv_id	int
object_id	int
x	int
y	int

GRID

Field	Type
grid_id	int
cctv_id	int
grid_index	int

MAPPING

Field	Type
mapping_id	int
grid_a_id	int

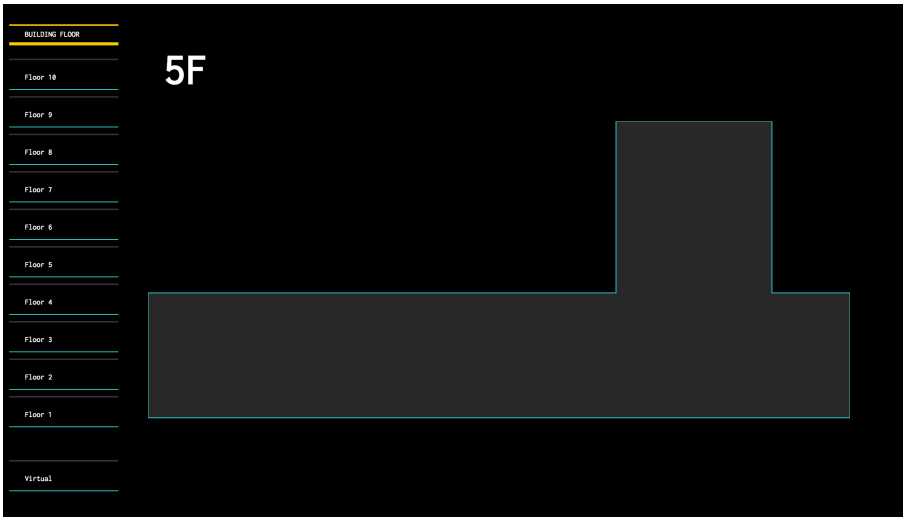
Field	Type
grid_b_id	int
counter	Int

5. Front End Implementation

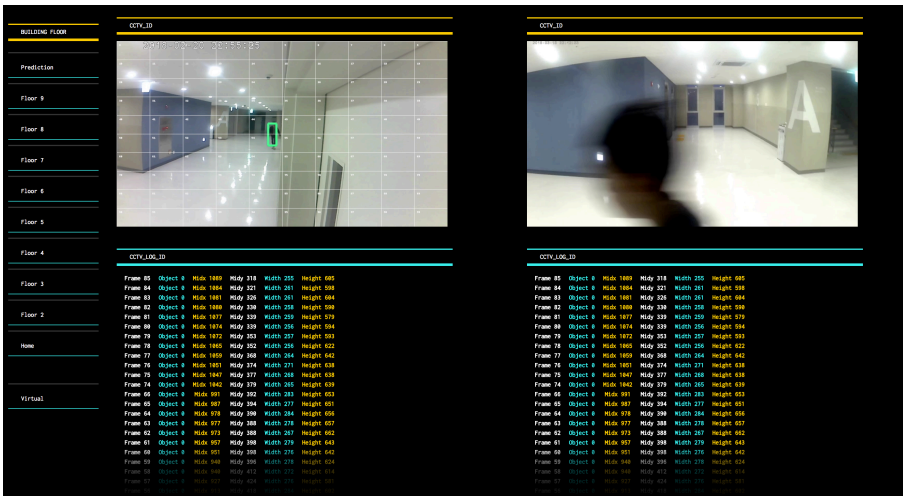
The front end web application were implemented with the following open-source Node.js dependencies: body-parser, convert-csv-to-json, express, express-validator, paper with Node.js

5.1 Page views

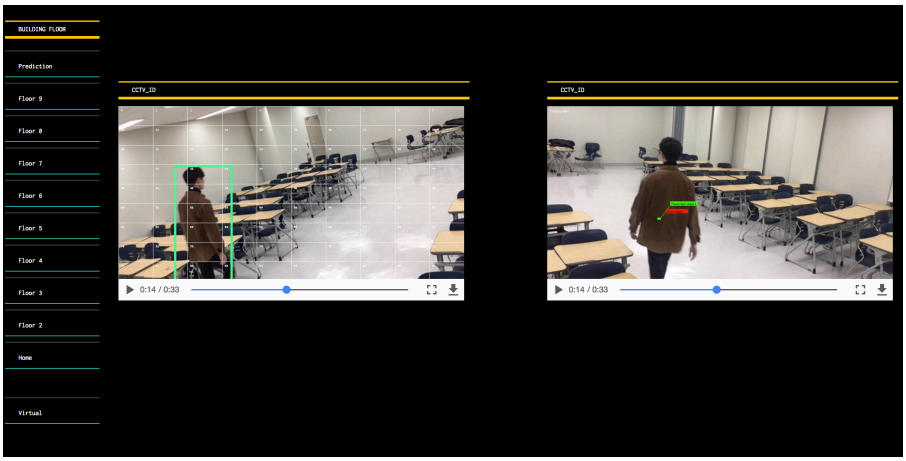
Index.ejs
Main page of our web.
We can navigate to floors with the buttons on the left pane.



floor.ejs
We can see the floor map here.
Choose a location to navigate to **camera-view** page.



camera_view.ejs
You can see the actual log data generated by our logger(parser with object detection. Implemented upon tensorflow stack)



prediction_view.ejs
This page shows the result of our prediction algorithm. Grid locations of **true-mapping** are shown along with actual location of object. Actually this page was for our own usage for further research.

6. Further Research

In addition to mapping space between two CCTV cameras, a mapping of time could be made by adding a time window relationship model between the two CCTV cameras. This could be done separately or as an upgrade to the current model used in this research.

7. Conclusion

By utilizing the intelligent logging system introduced in this research a more accurate system of CCTV system logging could be created. The database utilization approach taken by this research allows immediate correction from CCTV logs in the form of a data stream.

8. References

Tensorflow Object Detection API - Google
Park, N. H., & Lee, W. S. (2004). Statistical grid-based clustering over data streams. Acm Sigmod Record, 33(1), 32-37.
Chang, J. H., & Lee, W. S. (2003, August). Finding recent frequent itemsets adaptively over online data streams. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 487-492). ACM.