sensors

Article

A GNSS-Based Crowd-Sensing Strategy for Specific Geographical Areas

Chuan-Bi Lin [1], Ruo-Wei Hung [2], Chi-Yueh Hsu [3] and Jong-Shin Chen [1,*]

1 Department of Information and Communication Engineering, ChaoYang University of Technology, Taichung 413310, Taiwan; cblin@cyut.edu.tw
2 Department of Computer Science and Information Engineering, Chaoyang University of Technology, Taichung 413310, Taiwan; rwhung@cyut.edu.tw
3 Department of Leisure Services Management, Chaoyang University of Technology, Taichung 413310, Taiwan; cyhsu@cyut.edu.tw
* Correspondence: jschen26@cyut.edu.tw

Received: 2 June 2020; Accepted: 22 July 2020; Published: 27 July 2020

Abstract: Infectious diseases, such as COVID-19, SARS, MERS, etc., have seriously endangered human safety, economy, and education. During the spread of epidemics, restricting the range of activities of personnel is one of the options for the prevention and treatment of infectious diseases. A global navigation satellite system (GNSS), it can provide accurate coordinates of latitude and longitude to targets with GNSS receivers. However, it is not common to use GNSS coordinates to represent positions in social life. For epidemic management, it is important to know the locations (and addresses) of targets, especially in social life. When there are many targets, it is not easy to efficiently map these coordinates to locations. Therefore, we propose a GNSS-based crowd-sensing strategy for specific geographical areas that can be used to calculate how many targets are in specific geographical areas or whether a target is in a specific area. This strategy is based on the coordinates of latitude and longitude provided by GNSS to find the locations of these coordinates. As simulated data, the data records containing latitude and longitude in a well-known social networking service platform are used. The strategy is also available for mining hot spots or hot areas.

Keywords: epidemic management; GNSS/GPS; infectious disease; isolation; social networking service; hot spot

1. Introduction

A satellite navigation system with global coverage is called a global navigation satellite system (GNSS). It allows small electronic receivers to find its position in longitude and latitude coordinates. Current international GNSS standards for International Civil Aviation address only two core constellations: the U.S. Global Positioning System (GPS) and the Global Navigation Satellite System (GLONASS) [1,2]. These systems allow small electronic receivers to determine their location with the values of longitude and latitude to high precision using time signals transmitted along a line of sight by radio from satellites. Moreover, GPS receivers released in 2018 that use the L5 band [3] can have much higher accuracy. Many studies, that have used different techniques to improve observation precisions of GNSS positioning [4–6], can support this study. In this study, the corresponding locations of the latitude and longitude coordinates provided by a GNSS are evaluated. We also assume there is a large number of targets. Each target has a GNSS receiver that can acquire its current latitude and longitude coordinates and delivers its coordinates to a server. The coordinates of targets can be acquired by a computer manner. Our proposition is to assume that, when spaces of specific areas are large, such as New York City, Wuhan City China, and Taipei City, the number of targets to evaluate their locations, such as the population of a city, is also large.

During the spread of an epidemic, such as COVID-19 [7–9], it is an effective option to limit the movement range of people in order to control the epidemic. Determining the locations of targets can help determine whether people are in quarantine. There are two topics in this research. The first item is whether the target is within a certain range, which can be used to determine whether the quarantined person is in the isolation zone. The second item is the number of targets is in a specific area, thereby, it can be used to determine whether too many people are likely to cause infectious diseases. For example, in Taiwan, the four days from 2 April to 5 April 2020, are a consecutive holiday. During this period, there were a large number of tourists. The National Health Command Center (NHCC) in Taiwan [9] estimates that there were 1.5 million visits in 11 scenic spots through the telecommunications operator signal. Since the average stay time of these visits exceeded 15 min, the command center was worried about triggering a cluster infection of COVID-19 and urgently issued a national police call to the 11 scenic spots to call for evacuation, as well as the health management of tourists.

In general, a location with latitude and longitude is termed as a geographical point. There are very many geographical points generated by various social network services (SNSs) of the platforms, such as Facebook, Twitter, Google, and Foursquare. Check-in for some targets is a location-based service. It provides a mechanism to record users who have visited these geographical points. Among these platforms, Facebook is the largest one with regard to the number of users. In Facebook, these points are termed as "places". In other words, the check-in places are the locations that people actually visited. So, using these points as experimental samples to explore the crowd distribution is reasonable and credible. The Facebook penetration rate in Taiwan is the highest in the world. Here, the number of daily users reached 13 million of approximately 23 million. Accordingly, Taiwan is an appropriate selection as the experimental area, where the main island has an area of 35,808 square kilometers.

In light of above discussions, this study proposes solutions to the following:

1. For real-time coordinates of targets, this strategy can be used to determine the number of targets in the specified area in time and can also be used to determine whether the target is in the specified area.
2. For historical coordinates of targets, this strategy can be used to determine areas in which the targets can easily gather or which locations are the hot spots.

The rest of the paper is organized as follows. In Section 2, the research related to our study is introduced. In Section 3, the system architecture and the problems are described, in which the proposition is formalized. In Section 4, the crowd-sensing strategy is proposed. In Section 5, a demonstration is given by taking the area of Taiwan Island as the experimental area and the Facebook check-in places as the targets and examples. In Section 6, a discussion about the performance is described. Finally, a conclusion is given in Section 7.

2. Related Work

Accurate latitude and longitude positioning supports our research to evaluate the corresponding social locations. In [4–6], there are different techniques to improve observation precisions of GNSS positioning. In [10], the authors focused on the integrated methodology of GNSS and device-to-device measurements. The simulation and experimental results demonstrated that the integrated methodology outperforms the nonintegrated one. In [11], a two-step approach is studied, namely, computing first the Fisher Information Matrix (FIM) for the channel parameters, and then transforming it into the FIM of the position, rotation, and clock-bias. The analysis demonstrated the advantages of the hybrid positioning in terms of (1) localization accuracy, (2) coverage, (3) precise rotation estimation, and (4) clock-error estimation. In [12], this study presented a wideband/multiband quad-antenna system for 4G/5G/GPS metal-frame mobile phones. The merit of the proposed antenna system is that a quad-antenna system is achieved under the condition of a metal frame and, without using any decoupling structure, the desired bands for 4G/5G/GPS are covered.

The Internet environment has generated a large number of geographical points, such as Facebook check-in places [13–15], Google Maps places, Foursquare check-in places, etc. These places in Social Networking Services (SNS) are special kinds of geographical points. However, each SNS point contains not only a geographical coordinate and some contents to introduce this point. Accordingly, there are many studies focused on these points with their contents. In [16], the paper aimed to assess the role that interactive technology can play in enhancing urban governance to meet social needs. In [17], a real-time Google Maps-based arterial traffic information system for urban streets is presented. In [18], the authors proposed the reuse of up-to-date and low-cost place data from social media applications for land use mapping purposes by Foursquare place data. In [19], the study aimed to explore Foursquare mobility networks and investigate the phenomena of clustering venues across the cities. In [20], the study aimed to inform on how scientific researchers could utilize data generated in location-based social networks to attain a deeper understanding of human mobility. In [21], the authors proposed to find the geographical points related to a special folk belief. In [22], the author utilized the Markov Chain model with their proposed activity detection method to predict the activity category of the user's next check-in location. In [23], an urban tourism check algorithm is proposed. It can find those who are tourists and find out where the people come from and the route of their visit. In [24], the structure analysis of place networks is explored, in which vertices of geographic places, while the links between places are formed, are based on the user's check-in history. In [25], a new feature fusion-based prediction approach is proposed, based on carefully designed feature extraction methods. If these points can be mapped to social locations, not only coordinates, these applications could have further enhancements. To acquire the social locations of geographical points is also a promising topic, such as in epidemic management. In computational geometry, this is the point-in-polygon (PIP) problem [26–30]. These studies provide methods to evaluate that a point is inside an area or is not-inside an area without error. In [29], a computer-friendly method was proposed for the PIP problem.

In [30], the authors developed a PIP algorithm that can evaluate that a point is inside or is not-inside an area. Based on this PIP algorithm, the method to find all points within a specific geographical area was developed. It first plans a range that can cover the entire specific area. Then, it finds all the points in the planned range. For each point, the PIP algorithm is used to calculate whether this point is in the specified area. Because the PIP method is verified, this method can be applied to evaluate all the points in the specified area without error. However, this method is inefficient when the area is large with a large number of points. The planned range is often much larger than the specific area. This method selects a large number of points that are not in this specific area and confirms their locations by PIP. If the planned range can be smaller, the points that need to be confirmed by PIP will be reduced a lot. In addition, if certain points are known in the area or known outside the area, there is no need to confirm through PIP. In this way, the performance will be improved a lot.

3. System Architecture and Problem Description

The architecture of the system assumes that the target has a GNSS receiver. It can convert the received signal into coordinate data and send it to storage, such as databases, according to the transmission mechanism of the wired or wireless network. The target will be represented by a geographic point, referring to a point that has its latitude and longitude coordinate. In geometry, a point with coordinate is termed as a geographical point. Moreover, a geographical area (a specific area) is a polygon. A polygon is composed of edges with geographical points. These edges enclose a measurable interior [21]. Moreover, for a geographical point p, $x(p)$ and $y(p)$ are used to represent the coordinate of p. For a geographical area A with n points, a_0, a_1, … , and a_n are used to represent these n points, where $a_0 = a_n$.

The architecture of this system can be simplified into a geographic area A and a set P of geographical points. Because the area of area A is large and the size of set P is also very large, it is impossible in the field of computers to load all the points of P into the variables in the programming language and then calculate whether each point is in the range of A. It must be planned to only take points in part of the area in A at a time and these points are affordable by computers. The type of this area is chosen as a circle. There are two reasons. The first reason is that some well-known platforms (servers), such as Facebook, also provide geographical points in a circle manner. The other reason is to follow the access method of storages. The storage in general is a database system. To retrieve some records from the storage, it must be obtained through the access mechanism of the database system, such as structured query language (SQL). The circle-area manner can be mapped to the corresponding SQL commands. However, the configuration of the circle manner is not easy to fully cover a geographical area. Here the cellular architecture using a regular hexagonal configuration, instead of the circular configuration, can be applied. The cell architecture is shown in Figure 1, in which Figure 1a provides the layout of a cell and Figure 1b provides the relative positions of cell c and its six neighbors. If the radius of the circle is r, then the length of the regular hexagonal side is r. For convenience, a cell is used to represent a regular hexagon. There are six cells around a cell, termed as the neighbors of this cell. For a cell c, cells c_{N0}, c_{N1}, … , and c_{N5} are used to represent the six neighbors of cell c. The relative positions of cell c and its six neighbors can be expressed by an oblique coordinate architecture with 60 degrees (i.e., the angle between the x-axis and the y-axis is 60 degrees). Moreover, for cell c with radius r, Table 1 provides the coordinates of each vertex. Table 2 provides the coordinates of each neighbor.

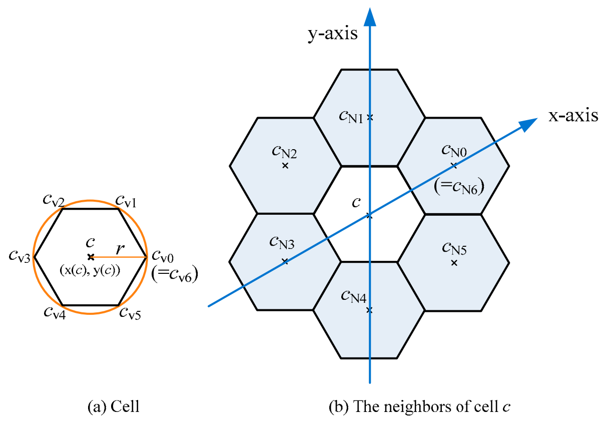


Figure 1. Cell architecture, a cell is a regular hexagon area. (**a**) For cell c with radius r, c_{V0}, c_{V1}, c_{V2}, c_{V3}, c_{V4}, c_{V5} are its six vertices. Moreover, c_{V6} represents vertex c_{V0}. (**b**) The neighbors of cell c represent as c_{N0}, c_{N1}, c_{N2}, c_{N3}, c_{N4}, c_{N5}. Moreover, cell c_{N6} represents cell c_{N0}.

Table 1. The vertices of cell c. For a cell c with radius r, the coordinates of each vertex are as follows.

Vertex	Coordinate
c_{V0} $(=c_{\mathrm{V6}})$	$(x(c) + r, y(c))$
c_{V1}	$(x(c) + 0.5r, y(c) + 0.5\sqrt{3}r)$
c_{V2}	$(x(c) - 0.5r, y(c) + 0.5\sqrt{3}r)$
c_{V3}	$(x(c) - r, y(c))$
c_{V4}	$(x(c) - 0.5r, y(c) - 0.5\sqrt{3}r)$
c_{V5}	$(x(c) + 0.5r, y(c) - 0.5\sqrt{3}r)$

Table 2. The neighbors of cell c. For a cell c with radius r, the coordinates of each neighbor are as follows.

Neighbor	Coordinate
c_{N0} (=c_{N6})	$(x(c) + 1.5r, y(c) + 0.5\sqrt{3}r)$
c_{N1}	$(x(c), y(c) + r)$
c_{N2}	$(x(c) - 1.5r, y(c) - \sqrt{3}r)$
c_{N3}	$(x(c) - 1.5r, y(c) - \sqrt{3}r)$
c_{N4}	$(x(c), y(c) + - \sqrt{3}r)$
c_{N5}	$(x(c) + 1.5r, y(c) - 0.5\sqrt{3}r)$

Moreover, if expressed by the geographic coordinate system (i.e., use latitude and longitude values to describe a coordinate with cell c), the offset coordinates of the six neighbors are shown in Table 2. Notably, the distance r is transformed by the Cartesian coordinate. Every point that is expressed in ellipsoidal coordinates can be expressed as rectilinear x y z (Cartesian) coordinates. Cartesian coordinates simplify many mathematical calculations. The Cartesian systems of different data are not equivalent. The distance between the points of longitude 121 and latitude 21 to the point of longitude 122 and latitude 21 is about 103 km. The distance between the points of longitude 122 and latitude 21 to the point of longitude 122 and latitude 22 is about 111 km. If it is configured with a radius of 1 km, r can be set to about 0.009 (=1/111) near this area.

A cell c is a regular hexagon area, defined as (c, r), where ($x(c)$, $y(c)$) is the geographical coordinate, and r is the length of the regular hexagonal side. Moreover, V(c) = $\{c_{V0}, c_{V1}, c_{V2}, c_{V3}, c_{V4}, c_{V5}\}$ is defined as the set of six vertices and NB(c) = $\{c_{N0}, c_{N1}, c_{N2}, c_{N3}, c_{N4}, c_{N5}\}$ is defined as the set of six neighboring cells. Table 1 provides the coordinates of each vertex in $V(c)$ and Table 2 provides the coordinates of each neighbor in $NB(c)$.

Hereafter, the word "geographical point" is simply termed as a "point" and the word "geographical area" is simply termed as an "area". Evaluating whether a point is inside or is not-inside is the PIP problem. It can count the intersections of the polygon with the ray of this point. A ray of a point is starting from this point to any fixed direction. That the number of intersections between the polygon and the ray is odd indicates this point is inside this polygon. Otherwise, it indicates this point is not-inside this polygon. This method can evaluate the locations of points with respect to an area without error. To implement this method, it must first be able to determine whether two lines are intersected. In [29], a computer-friendly method was proposed to do it. Suppose there are two lines. The first line crosses over both point o_1 and point o_2. The other line crosses over both point o_3 and point o_4. First, ρ, as shown in (1), can be evaluated. If the value of ρ equal 0, these two lines are parallel. Next, κ_1 and κ_2, as, respectively, shown in (2) and (3), can be evaluated. If the value of κ_1 is between 0 and 1, the intersection is between o_1 and o_2. If the value of κ_2 is between 0 and 1, the intersection is between o_3 and o_4. In other words, if edge (o_1, o_2) and edge (o_3, o_4) are intersected, the value of κ_1 is between 0 and 1 and the value of κ_2 is also between 0 and 1.

$$\rho = (x(o_1) - x(o_2)) \cdot (y(o_3) - y(o_4)) - (y(o_1) - y(o_2)) \cdot (x(o_3) - x(o_4)) \tag{1}$$

$$\kappa_1 = ((x(o_1) - x(o_3)) \cdot (y(o_3) - y(o_4)) - (y(o_1) - y(o_3)) \cdot (x(o_3) - x(o_4)))/\rho \tag{2}$$

$$\kappa_2 = -(((x(o_1) - x(o_2)) \cdot (y(o_1) - y(o_3)) - (y(o_1) - y(o_2)) \cdot (x(o_1) - x(o_3)))/\rho \tag{3}$$

The overlap of an area A with a cell c, where the space of A is far greater than the space of c, is also considered. There are two overlapping cases between A and c. The first case is the space of A is overlapped with the space of c. Non-first relationship is the second relationship. The first case can be evaluated when all six points are inside A and there are no intersection points among the edges of c and the edges of A.

As shown in Figure 2, an area is composed of a convex polygon with six points, a_0, a_1, … , and a_5. The rays of p_1, p_2, and p_3 within this area, respectively, have 2, 1, and 0 intersection(s). Since the ray of a point has odd intersections within the area, it means that the point is inside the area; otherwise, the point is not-inside the area. Then, both point p_1 and point p_3 are not-inside the area and point p_2 is inside the area. Continually, there are three cells, c_1, c_2, and c_3. Since all vertices of c_1 are in this area, two vertices of c_2 are in this area, and no vertices of c_3 are in this area, both c_1 and c_2 overlap with this area and c_3 does not overlap with this area.

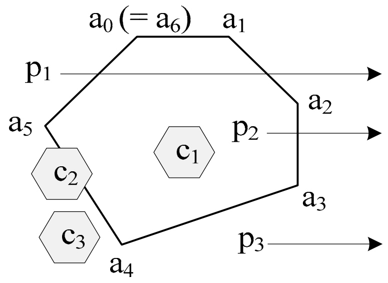


Figure 2. Example of an area, points, and cells. This area is a polygon composed by points a_0, a_1, ... , and a_5. To this area, point p_1 and point p_3 are not-inside and point p_2 is inside, both cells c_1 and c_2 are overlapped and c_3 is not overlapped. Moreover, cell c_1 is also termed as inside this area.

4. Crowd-Sensing Strategy

Given an area $A = \{a_0, a_1, \ldots, a_n\}$ and a large number of points P, where P are stored in a database and can be accessed by using SQL commands, this strategy includes two step cell allocation and point acquisition. It can achieve the evaluation all of points in P inside area A by applying the two steps.

4.1. Cell Allocation

The first step applies algorithms EEI, PIA, and CAO to achieve cell allocation. Algorithm 1 provides algorithm EEI. It is an Edge–Edge Intersection algorithm, where inputs are two edges (o_1, o_2) and (o_3, o_4) and the output is a value of 0 or 1. If edge (o_1, o_2) and edge (o_3, o_4) are intersected, it results by returning 1. Otherwise, it results by returning 0. In this algorithm, it first calculates the value of ρ according to (1). If the value of ρ is 0, the two edges are parallel. If the value of ρ is not 0, it calculates the values of κ_1 and κ_2. If both κ_2 and κ_1 are between 0 and 1, it indicates that the two edges have an intersection.

Algorithm 1 EEI (o_1, o_2, o_3, o_4)

```
1.   {
2.     $\rho$: = $(x(o_1) - x(o_2)) \cdot (y(o_3) - y(o_4)) - (y(o_1) - y(o_2)) \cdot (x(o_3) - x(o_4))$;
3.     if $\rho = 0$ then return 0;
4.     else
5.     {
6.       $\kappa_1$: = $((x(o_1) - x(o_3)) \cdot (y(o_3) - y(o_4)) - (y(o_1) - y(o_3)) \cdot (x(o_3) - x(o_4)))/\rho$;
7.       $\kappa_2$: = $-(((x(o_1) - x(o_2)) \cdot (y(o_1) - y(o_3)) - (y(o_1) - y(o_2)) \cdot (x(o_1) - x(o_3)))/\rho$;
8.       if ($\kappa_1 \geq 0$ and $\kappa_1 \leq 1$) and ($\kappa_2 \geq 0$ and $\kappa_2 \leq 1$) then return 1;
9.       else return 0;
10.    }
11.  }
```

Algorithm 2 provides the Point Inside Area (PIA) algorithm, whose inputs are a point p and an area A and its output is a value of 0 or 1. If point p is inside area A, it results by returning value 1. Otherwise, it results by returning value 0. In this algorithm, variables o_1 and o_2 are used to represent an edge of area A and variables o_3 and o_4 are used to represent the incremental extension from p toward the x-axis until it is greater than the coordinate values of x(o_1) and x(o_1), as in line 6 (i.e., x(o_4) is set as (max(x(o_1), x(o_2)) + 1). Variable count is used to record the number of intersections between the extension line of p and the edge of A. If the number of intersections is odd, it means that p is inside A, otherwise p is not-inside A.

Algorithm 2 PIA (p, A)

```
1.   {
2.     $count$: = 0;
3.     for $i$: = 0 to $n - 1$ do
4.     {
5.       $o_1$: = $a_i$; $o_2$: = $a_{i+1}$;
6.       $o_3$: = $p$; $o_4$: =(max(x($o_1$), x($o_2$)) + 1, y($p$));
7.       if (EEI($o_1$, $o_2$, $o_3$, $o_4$) == 1) then $count$: = $count$ + 1;
8.     }
9.     if ($count$ % 2 == 1) then return 1;
10.    else return 0;
11.  }
```

Algorithm 3 provides the Cell-Area Overlap (CAO) algorithm, whose inputs are a point p and an area A and its output is a value of 0 or 1. In this algorithm, each vertex of the cell is taken out independently, and is then calculated for whether it is inside A. The variable count is used to record how many vertices of cell c are inside A. If any of the six vertices of cell c are inside A, c and A are overlapping. Otherwise, c and A are non-overlapping.

Algorithm 3 CAO (c, A)

```
1.   {
2.     $count$: = 0;
3.     for $i$: =0 to 5 do
4.     {
5.       $p$:= $c_{vi}$;
6.       if (PIA($p$, $A$) == 1) then $count$: = $count$ + 1;
7.     }
8.     if ($count$ > 0) then return 1;
9.     else return 0;
10.  }
```

Algorithm 4 provides the Cell Allocation (CA) algorithm, whose inputs are an area A and a cell c_s. The allocation starts with cell c_s, called the seed cell, and then expands to its neighboring cells NB (c_s). Then, the neighbors of c_s continue to extend the allocation of cells. The allocation is done until area A is completely covered by cells. In this algorithm, variable C records the allocated cells and variable P records the reference cells. In each iteration (in lines 5 to 14), each cell c_p will be selected in sequence from P. Then, the neighbors of the c_p are calculated using NB(c_p) (in line 7). For each cell, c_{NB} of NB(c_p), if cell c_{NB} is overlapped with A and c_{NB} is not included in C, c_{NB} will be added into C (i.e., C: = C∪{c_{NB}}). Additionally, c_{NB} is added to N (i.e., C: = ∪{c_{NB}}). At end of the iteration, the value of N is assigned to p to start a new iteration. Therefore, this algorithm is stopped when P is empty (in line 3) and then it returns the allocated cells C.

Algorithm 4 CA (c_s, A)

```
{
  $P$: = {$c_s$}, $N$: = ∅, C:= {$c_s$};
  while ($P$≠∅)
  {
    for each $c_p$∈ $P$ do
    {
      $NB$:= NB($c_p$);
      for each $c_{NB}$∈ $NB$ do
      {
        if (CAO($A$, $c_{NB}$) == 1 && $c_{NB}$ ∉ C ) then
        {
          $N$:= $N$∪{$c_{NB}$}, C:= C∪{$c_{NB}$};
        }
      }
    $P$: = $N$, $N$: = ∅;
  }
 return $C$;
}
```

As shown in Figure 3, the irregular area is composed of 2055 points, and the numbers on cells index the order of cell allocation. In this example, c_i is used to represent the cell, numbered as i. Initially, cell c_0 is allocated. In the first iteration, the six neighbors c_1, c_2, ... , and c_6 are included in C. Herein, C contains seven cells, c_0, c_1, c_2, ... , and c_6, and P contains six cells, c_1, c_2, ... , and c_6. In the second iteration, cells c_1, c_2, ... , and c_6 will be the reference cells. For instance, the neighbors of c_1 contains cells c_7, c_8, c_2, c_0, c_6, and c_9. However, c_2, c_0, and c_6 are already included in C. Only c_7, c_8, c_9 are included in C. Continually, in the third iteration, cells c_{19} to c_{32} are included to C and in fourth iteration, cells c_{33} to c_{37} are included in C. In the fifth iteration, cells c_{33} to c_{37} are the reference cells P. Since all of the non-overlapped neighbors of c_{33}, c_{34}, ... , or c_{37} are included in C, no cell are included in C and P is empty (in line 3). Finally, the allocation is finished by returning the allocation C.

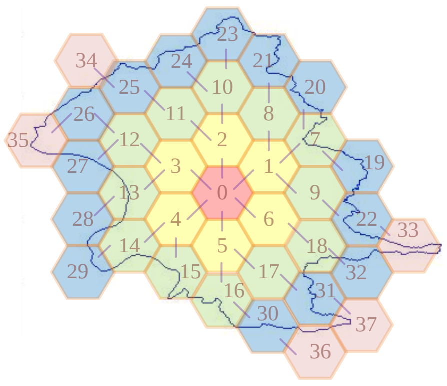


Figure 3. Example of Cell Allocation to an area. The allocation is started from c_0. In each iteration, some cells will be allocated. After five iterations, the allocation is finished.

4.2. Point Acquisition

Points in the local database can be obtained with the similar SQL command in (4), where 'P' is the table that stores all of geographic points, 'x' and 'y' are the fields for storing latitude and longitude values, and r is the radius of the circular area. In addition, the SQRT (v) function is used to calculate the square root of the value v, and the POWER (v, n) function is used to calculate the nth power of the value v.

$$\text{SELECT * FROM `}P\text{` WHERE SQRT(POWER(`x` } - \text{ x(c),2) + POWER(`y` } - \text{ y(c), 2))} \leq r \tag{4}$$

Algorithm 5 provides the Cell Inside Area (CIA) algorithm, whose inputs are a cell c and an area A and its output is a value of 1 or 0 that, respectively, indicates c inside A or c not-inside A. In this algorithm, each edge of c and each edge of A will be calculated to the number of intersections. If the number of intersections is 0, c is inside A, otherwise c is not inside A.

In summary of the above algorithm, the Crowd-Sensing (CS) algorithm is purposed. Algorithm 6 provides the CS algorithm. It is a Crowd-Sensing algorithm, where the input includes a seed of cell c_s, an area A, and all of points P and its output is a set P_A of all points inside A. In this algorithm, it first calculates the cell allocation C by applying CA (c_s, A), where C is the set of cells that can fully cover the space of area A. Then, for each cell c in C, it first extracts the geographic points Pc included in cell c from the database. There are two cases to deal with Pc. The first case based on cell c is inside area A, and Pc is directly included in P_A. The other case is based on cell c not-inside A. In this case, every point p in Pc will be evaluated with area A. If point p is inside area A, then p is included to P_A. For example, in Figure 3, because cells c_0, $c_{1,\ldots,}$ c_6, $c_{8,}$ $c_{9,\ldots,}$ c_{12} are inside this area, the points in this range are included in P_A. Other cells are not-inside this area, so the points in this range must be evaluated. In particular, our strategy only needs to evaluate the points in the outermost cells. In fact, the range of points to evaluate is very small. In general, if an area is allocated thousands of cells, only hundreds of cells are evaluated.

Algorithm 5 CIA (c, A)

```
1.   {
2.     $count$: = 0;
3.     for $i$: =0 to 5 do
4.     {
5.       for $j$: =0 to $n$-1 do
6.       {
7.         $o_1$: = $c_{Vi}$, $o_2$: = $c_{Vi+1}$, $o_3$: = $a_j$, $o_4$: = $a_{j+1}$;
8.         if (EEI($o_1$, $o_2$, $o_3$, $o_4$) == 1 ) $count$: = $count$ +1;
9.       }
10.    }
11.    if ($count$ == 0) return 1;
12.    else return 0;
13.  }
```

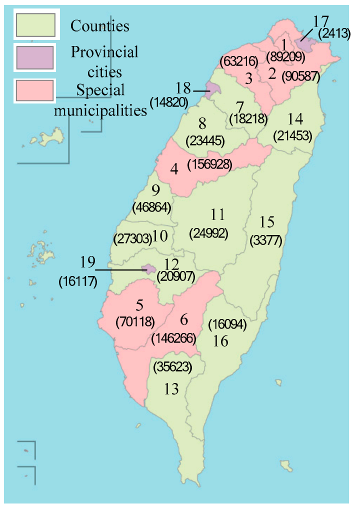
Algorithm 6 CS (c_s, A)

```
1.  {
2.    $P_A$: = Ø;
3.    C: = CA ($c_s$, $A$);
4.    for each c ∈ C do
5.    {
6.      $P_C$: = (SELECT * FROM 'P' WHERE SQRT(POWER('x'-x($c$),2) + POWER('y'-y($c$), 2)) <= r)
7.      if (CIA($c$, $A$) == 1) $P_A$: = $P_A \cup P_C$;
8.      else
9.      {
10.       for each $p \in P_C$ do
11.       {
12.         If (PIA($p$, $A$) == 1) then $P_A$: = $\cup\{p\}$;
13.       }
14.     }
15.   }
16.   return $P_A$;
17. }
```

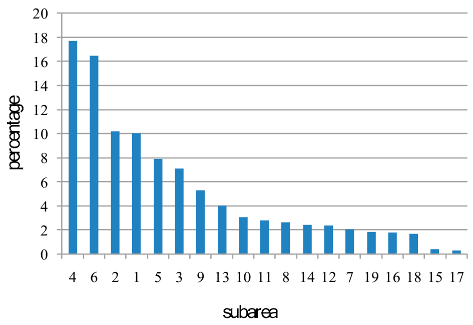
5. Demonstration

The geographical area of experiment is mainly Taiwan Island that is located between 120 degrees to 122 degrees east longitude and 22 degrees to 25 degrees north latitude. It has an area of 35,808 square kilometers with 23.7 million inhabitants. The points are based on the check-in places of Facebook. The points were acquired from Facebook platform in January 2017, for a total of 1,112,188. We used these points as targets and try to find the targets in specific areas. This area currently contains 6 special municipalities, 10 counties and 3 provincial cities. We demonstrate the results according to the 19 subareas. Figure 4 provides the distribution of 19 subareas, the special municipalities are numbered as 1–6, counties are numbered as 7–16, and provincial cities are numbered as 17–19. Each subarea is composed of hundreds to thousands of vertices. For instance, subarea 1 is composed of 2055 vertices. For each subarea, the proposed strategy was applied to evaluate the number of points inside it.

In Section 5.1, we first display the points in each area. In addition, the population density and the space density with points were also considered to find hot areas. In Section 5.2, we then display the points in each spot. The numbers of points in spots are also used to find hot spots. In addition, the distribution of spots are used to display the crowd distribution. In Section 5.3, we present the contributions of this study. In Section 5.3, we discuss the differences between this study and the previous study.



(a)



(b)

Figure 4. The experimental area with 887,950 points. This area currently contains 6 special municipalities, 10 counties and 3 provincial cities. These subareas are numbered as 1–19: (**a**) The number of points in each subarea. (**b**) The point ratio of each subarea to 16 subareas.

5.1. Points in Areas

In Figure 4a, the value in parentheses is the number of points of the subarea. For example, there are 156,928 points in the area numbered as 4. Moreover, there are a total of 887,950 points in these 19 areas, of which subareas 4, 6, 2, 1, 5, and 3 are the hot areas with the most points. Figure 4b provides the point ratio of each subarea to 16 subareas. There are 616,324 (69.41%) points in these six hot areas. Especially, all of the six hot areas are special municipalities. These results show that, through the coordinates, we can accurately calculate the targets in a specific administrative area, rather than only the information of the latitude and longitude coordinates. Then, the population and the space of hot area are taken into consideration.

Table 3 provides the population, space, points of each hot area, population density, and space density. For example, in subarea 1, the number of the population is 2,687,629, the space is 271.8 square kilometers, and the number of points is 89,209. Therefore, its population density is 3.32% (i.e., the value of 89,209/2,687,629 × 100%) and its space density is 328.21 (i.e., the value of 89,209/271.8). Among these six hot areas, subarea 1 especially has a much higher space density than the other five hot areas. For a specific area with positioned targets, the population and the space can be taken into consideration. It can get the proportion of the targets to the number of the population and the spatial proportion of the target object. This result can be used to determine whether the targets are too crowded in this area based on population or space. It can be used as a control for crowds, such as prohibiting more targets from entering this area or evacuating some targets from leaving this area. It can help to keep the number of targets in this area under control for social applications, such as traffic or epidemic control.

Table 3. Population density and space density based on number of points.

Subarea	Population	Space (km^2)	No. of Points	Population Density (#/Population × 100%)	Space Density (#/Space)
1	2,687,629	271.8	89,209	3.32	328.22
2	3,984,051	2052.57	90,587	2.27	44.13
3	2,171,127	1220.95	63,216	2.91	51.78
4	2,778,182	2214.90	156,928	5.65	70.85
5	1,886,267	2191.65	70,118	3.72	31.99
6	2,777,873	2951.85	146,266	5.27	49.55

#: number of points.

Figure 5 provides the distribution of points for subarea 1 and its neighborhood. The range has 93,549 points, where there are 89,029 points inside subarea 1 and 4520 points not-inside subarea 1. Most of the points are gathered in subarea 1. Moreover, the points are concentrated between longitude 121.45 to 121.6 and latitude 24.95 to 25.01. However, there are almost no geographic points between latitude 25.1 and 25.22. Obviously, the distribution of points is very uneven. Then, we continue to calculate the locations of the hot spots to reveal the concentrations.

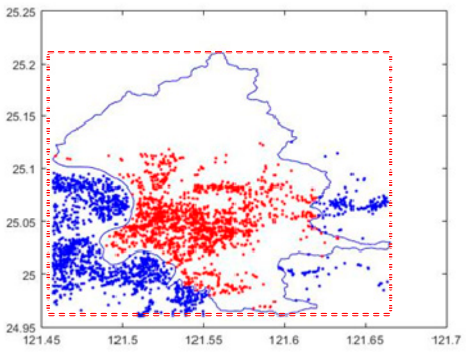


Figure 5. Point distribution of subarea 1 and its neighborhood. The range is longitude 121.45 to 121.67 and latitude 24.95 to 25.22. The number of points in this range is 93,549. Randomly, 1600 points inside subarea 1 and 1600 points not-inside subarea 1 are marked on this range.

5.2. Points in Spots

The area is divided into small areas, termed as spots, with a range of 0.01 degrees of latitude and longitude, and the area is slightly larger than 1 square kilometer. As shown in Table 4, there are a total of 15,573 spots. There are 15,347 spots whose numbers of points are less 1000. However, there are 12 spots whose numbers of points are more than 4000. Table 5 provides the details of the 12 hot spots, including the locations, the coordinates, and the numbers of points. The 12 hot spots are in subarea 4, subarea 6, subarea 9, and subarea 19. Moreover, subarea 4 contains six hot spots, subarea 6 contains four hot spots, subarea 9 contains one hot spot, and subarea 19 contains one hot spot. Figure 6 shows the distribution of points in hot spot one. There are 6024 points in this range of about one square kilometer. It means when we are in this hot spot, we are easily exposed to these targets (points). The analysis of hot spots helps to understand whether the target is concentrated in a certain small range, and it is also easy to monitor these spots.

Table 4. Spot statistics with points.

No. of Points	Spots	Percentage
less than 1000	15,347	98.55
1000–2000	152	0.98
2000–3000	49	0.31
3000–4000	13	0.08
4000–5000	4	0.03
5000–6000	7	0.04
6000–7000	1	0.01
Total	15,573	100

Table 5. The first 12 hot spots.

Hot Spot	Location	Coordinate	No. of Points
1	subarea 6	120.3, 22.67	6024
2	subarea 6	120.3, 22.62	5731
3	subarea 4	120.64, 24.18	5527
4	subarea 6	120.3, 22.63	5506
5	subarea 4	120.66, 24.16	5428
6	subarea 4	120.68, 24.16	5154
7	subarea 4	120.68, 24.15	5152
8	subarea 6	120.3, 22.64	5043
9	subarea 4	120.65, 24.16	4855
10	subarea 19	120.44, 23.48	4544
11	subarea 4	120.68, 24.14	4424
12	subarea 9	120.54, 24.08	4056

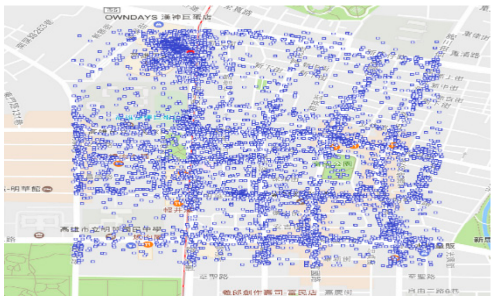
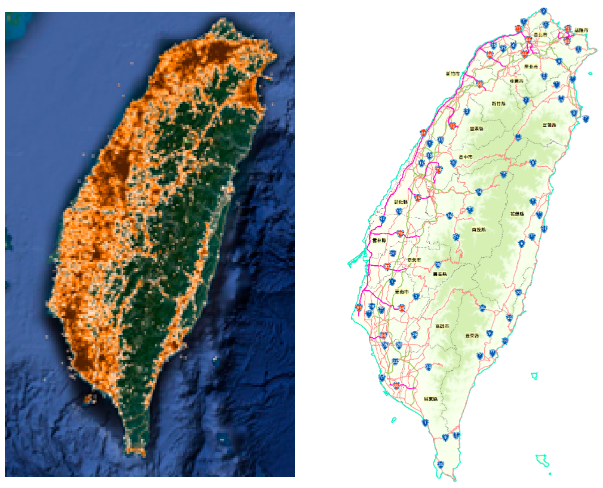


Figure 6. Distribution of points in hot spot one with 6024 points.

The first 10,000 spots, according to the number of geographic points, are taken into consideration. Then, these spots are classified into 10 groups, where each group is represented using brown color. The darker color represents more geographic points in this spot. The layout of the 10,000 spots is shown in Figure 7a. Comparing the relative positions of the 19 subareas in Figure 4a, it shows that hot spots are almost concentrated in the range of subarea 1 to subarea 6. These six areas are six municipalities in which the population densities are more than other administrative areas. Figure 7b shows the distribution of Highway in this area. The distribution of spots is consistent with the distribution of highway. The geographic points of this experiment are the check-in places of Facebook in Taiwan Island. These places are the historical records of Facebook users who have been to these places for a long time. Because Facebook users are very numerous in this experimental area, these points represent almost every point covered by social activities in the area. Therefore, areas with convenient transportation and high urbanization will have more geographic points. We indicated the spot, according to the number of geographic points, as a spot height map. This map, as shown in Figure 7a, coincides with the highway distribution and administrative area distribution of this area. These facts reveal that our research is a crowd-sensing strategy.



(**a**) (**b**)

Figure 7. (**a**) Distribution of spots and (**b**) distribution of highways. The layout of (**a**), respectively, coincides with the layout of (**b**) and the layout of Figure 4a.

5.3. Summary

In [29], an error-free method is presented that can calculate whether a point is inside a polygon. For this, we planned the PIA algorithm, which can also evaluate whether a point is within an area or not without error. Therefore, the efficiency of this type of method is determined by how many points are needed to apply the PIA algorithm and not the accuracy. We take an example, shown in Figure 5, to illustrate the difference between our strategy and [30]. In [30], it first plans a rectangle range that can cover this area. The range is marked as a dashed line. It then finds all points in this range. In our experimental environment, there are 95,349 points. So, the number of executing PIA algorithms is 95,349. In our strategy, we planned a set of cells that could cover this area. The range of the cells is similar to the example shown in Figure 3 that is slightly larger than this area. In fact, we planned with a cell of about 1 square kilometer. This area is about 271.8 square kilometers, as shown in Subarea 1 of Table 3. In this example, the number of cells is about 350 and the number of points within these cells is about 90,000. These cells are classified into inside cells and not-inside cells through the CIA algorithm. Most points are inside cells. The points are surely also in this area without confirming locations by performing PIA. Only a few points located in the outermost cells (not-inside cells) need to confirm their locations. In the examples of Sections 5.1 and 5.2, the points that need to execute PIA rarely exceeds 10%.

The previous study [30] is very time-consuming to find out the points in large-scale areas, especially when the number of candidate points is very large. The main contribution of our study is that the points that PIA needs to confirm are reduced to very few. Therefore, our strategy has the ability to efficiently handle large-scale areas, such as countries and cities. In the computer field, when the amount of data is large, it is not feasible to process all the data at once. It is necessary to transfer to the batch manner. The design of cells is adaptive to this. It is also convenient to be implemented for epidemic prevention management.

6. Discussion

Let A be a specific area and P be the set of points. Our proposed strategy provided a solution to acquire all points in P inside A. The PIA algorithm can be used to evaluate whether point p is inside or not-inside area A. The time complexity is O(n_A), where n_A is the number of vertices (or edges) of area A. The reason is that it needs to count the number of intersections among the ray of p and the n_A edges. The simple method to acquire all of points inside area A is to evaluate each point of P by PIP. The candidate points, which are points needing the evaluation of PIA, are all of the n_P points. The time complexity is O($n_A \times n_P$), where n_P is the number of points in set P. When n_P is a very large number, it is greatly time-consuming to acquire these points. In [30], an enhanced strategy was proposed. It first evaluates the boundary (i.e., a space of rectangle). This rectangle is fully covered by the space of area A. The points within the rectangle are evaluated. In our strategy, we allocate a set of cells to fully cover area A. The total space of these cells is slightly larger than the space of area A. Then, the cells are divided into inside cells and not-inside cells. Only points within not-inside cells must be evaluated. The not-inside cells are the cells that have intersections with area A. In fact, only points near the edges of area A are evaluated. The number of candidate points is reduced to a very small number. Therefore, our strategy is efficient to acquire the targets in an area.

7. Conclusions

In this study, Taiwan Island and the administrative areas are used as geographic areas, and check-in places on Facebook are used as geographic points to verify the proposed strategy. These are the actual data. Due to the fact that the spot height map, respectively, matches the distribution of administrative areas and matches the distribution of highways, it verifies the practicability of our strategy. This strategy mainly provides two techniques. The first technique is used to calculate whether the geographic points are in a specific area. The second technique is used to calculate the number of points in a specific area. In our demonstration, we first, respectively, analyzed the numbers of points of the 16 administrative areas. Then the populations of the administrative areas and the spaces of the administrative areas were included in the discussion. This provided these results: the number of geographic points per unit area and the ratio of geographic points to the population. This is the category of hot areas.

Because the numbers of geographic points in geographic areas are not enough to represent the distribution of geographic points, we planned a space of about 1 square kilometer as a unit, termed as a spot, and calculated the number of geographic points in each spot. This is a hotspot category. Of course, the size of a spot depends on the actual situation. Based on the above discussion, our strategy is a GNSS-based crowd-sensing strategy for specific areas. This research is very useful in many fields. We used non-real-time and real-time GNSS coordinates for the applications. Non-real-time coordinates (i.e., historical records), can be used to know where hot areas hot spots can develop. It can be deployed in advance for this area to avoid or mitigate future events. For real-time coordinates, it can be used to know where it is becoming a hot area or where it is becoming a hot spot. It can be deployed ahead of schedule to avoid or mitigate ongoing events. The acquisition of coordinates may have privacy constraints. To acquire current location information with user consent may be available. Observing privacy constraints to do more epidemic prevention management is the goal of our future work.

Author Contributions: Conceptualization, C.-B.L. and J.-S.C.; methodology, R.-W.H.; software, C.-Y.H.; validation, C.-B.L., R.-W.H., C.-Y.H., and J.-S.C.; formal analysis, R.-W.H.; investigation, C.-B.L. and J.-S.C.; writing, C.-B.L. and J.-S.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: This research was partially supported by the Ministry of Science and Technology, Taiwan (ROC), under contract no.: MOST 108-2410-H-324-007 and MOST 108-2637-E-324 -004.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. *International Civil Aviation Organization Annex 10 to the Convention of International Civil Aviation*; International Civil Aviation Organization: Montreal, QC, Canada, 2007; Volume I.
2. Hegarty, C.J.; Chatre, E. Evolution of the Global Navigation Satellite System (GNSS). *Proc. IEEE* **2008**, *96*, 1902–1917. [CrossRef]
3. Leclère, J.; Landry, R., Jr.; Botteron, C. Comparison of L1 and L5 Bands GNSS Signals Acquisition. *Sensors* **2018**, *18*, 2779. [CrossRef] [PubMed]

4. Zhang, Z.; Li, B.; Shen, Y.; Gao, Y.; Wang, M. Site-Specific Unmodeled Error Mitigation for GNSS Positioning in Urban Environments Using a Real-Time Adaptive Weighting Model. *Remote Sens.* **2018**, *10*, 1157. [CrossRef]
5. Li, T.; Zhang, H.; Gao, Z.; Chen, Q.; Niu, X. High-accuracy positioning in urban environments using single-frequency multi-GNSS RTK/MEMS-IMU integration. *Remote Sens.* **2018**, *10*, 205. [CrossRef]
6. Cai, C.; Pan, L.; Gao, Y. A precise weighting approach with application to combined L1/B1 GPS/BeiDou positioning. *J. Navig.* **2014**, *67*, 911–925. [CrossRef]
7. Park, S.W.; Cornforth, D.M.; Dushoff, J.; Weitz, J.S. The time scale of asymptomatic transmission affects estimates of epidemic potential in the COVID-19 outbreak. *Epidemics* **2020**, *31*, 100392. [CrossRef] [PubMed]
8. Wigginton, N.S.; Cunningham, R.M.; Katz, R.H.; Lidstrom, M.E.; Moler, K.A.; Wirtz, D.; Zuber, M.T. Moving academic research forward during COVID-19. *Science* **2020**, *368*, 1190–1192. [CrossRef] [PubMed]
9. Wang, C.J.; Ng, C.Y.; Brook, R.H. Response to COVID-19 in Taiwan: Big Data Analytics, New Technology, and Proactive Testing. *JAMA* **2020**, *323*, 1341–1342. [CrossRef] [PubMed]
10. Yin, L.; Ni, Q.; Deng, Z. A GNSS/5G integrated positioning methodology in D2D communication networks. *IEEE J. Sel. Areas Commun.* **2018**, *36*, 351–362. [CrossRef]
11. Destino, G.; Saloranta, J.; Seco-Granados, G.; Wymeersch, H. Performance Analysis of Hybrid 5G-GNSS Localization. In Proceedings of the 2018 52nd Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, USA, 28–31 October 2018; pp. 8–12.
12. Huang, D.; Du, Z.; Wang, Y. A Quad-Antenna System for 4G/5G/GPS Metal Frame Mobile Phones. *IEEE Antennas Wirel. Propag. Lett.* **2019**, *18*, 1586–1590. [CrossRef]
13. Feitelson, D.G.; Frachtenberg, E.; Beck, K.L. Development and Deployment at Facebook. *IEEE Internet Comput.* **2013**, *17*, 8–17. [CrossRef]
14. Lin, H.-T. Applying location based services and social network services onto tour recording. In Proceedings of the 2012 Ninth International Conference on Computer Science and Software Engineering (JCSSE), Bangkok, Thailand, 30 May–1 June 2012; pp. 197–200.
15. Chen, J.S.; Hsu, C.Y.; Yang, C.Y.; Wei, C.C.; Ciang, H.G. A data mining method for Facebook social network: Take "New Row Mian (Beef Noodle)" in Taiwan for example. In Proceedings of the 2017 IEEE 8th International Conference on Awareness Science and Technology (iCAST), Taichung, Taiwan, 8–10 November 2017; pp. 165–169.
16. Liu, H.K.; Hung, M.J.; Tse, L.H.; Saggau, D. Strengthening urban community governance through geographical information systems and participation: An evaluation of my Google Map and service coordination. *Aust. J. Soc. Issues* **2020**, *55*, 182–200. [CrossRef]
17. Wu, Y.-J.; Wang, Y.; Qian, D. A google-map-based arterial traffic information system. In Proceedings of the 2007 IEEE Intelligent Transportation Systems Conference, Seattle, WA, USA, 30 September–3 October 2007; pp. 968–973.
18. Spyratos, S.; Stathakis, D.; Lutz, M.; Tsinaraki, C. Using Foursquare place data for estimating building block use. *Environ. Plan. B Urban Anal. City Sci.* **2017**, *44*, 693–717. [CrossRef]
19. Novović, O.; Grujić, N.; Brdar, S.; Govedarica, M.; Crnojević, V. Clustering Foursquare Mobility Networks to Explore Urban Spaces. In *World Conference on Information Systems and Technologies*; Springer: Cham, Switzerland, 2020; pp. 544–553.
20. Noulas, A.; Scellato, S.; Mascolo, C.; Pontil, M. An empirical study of geographic user activity patterns in foursquare. In Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media, Catalonia, Spain, 17–21 July 2011.
21. Huang, Y.F.; Chen, J.S.; Lin, C.B. A Specific Targeted-Place Mining Method for a Famous Social Network: Take Wang-Ye Worship in Taiwan for Example. In Proceedings of the 2018 15th International Symposium on Pervasive Systems, Algorithms and Networks (I-SPAN), Yichang, China, 16–18 October 2018; pp. 263–266.
22. Xia, T.; Shen, J.; Yu, X. Predicting human mobility using sina weibo check-in data. In Proceedings of the 2018 International Conference on Audio, Language and Image Processing (ICALIP), Shanghai, China, 16 July 2018; pp. 380–384.
23. Yang, K.; Wan, W.; Xia, T.; He, X. Urban tourism research based on the social media check-in data. In Proceedings of the 4th International Conference on Smart and Sustainable City (ICSSC 2017), Shanghai, China, 5 June 2017; pp. 1–3.

24. Ding, X.; Xu, J.; Chen, G. Exploring structural analysis of place networks using check-in signals. In Proceedings of the 2013 IEEE Global Communications Conference (GLOBECOM), Atlanta, GA, USA, 9–13 December 2013; pp. 3194–3199.
25. Han, Y.; Yao, J.; Lin, X.; Wang, L. GALLOP: GlobAL feature fused LOcation Prediction for Different Check-in Scenarios. *IEEE Trans. Knowl. Data Eng.* **2017**, *29*, 1874–1887. [CrossRef]
26. Ding, J.; Wu, K.; Guan, H.; Wang, D.; Rui, T. Point-in-polygon algorithm based on monolithic calculation for included angle of half plane continuous chains. In Proceedings of the 2010 18th International Conference on Geoinformatics, Beijing, China, 18–20 June 2010; pp. 1–4.
27. Kularathne, D.; Jayarathne, L. Point in Polygon Determination Algorithm for 2-D Vector Graphics Applications. In Proceedings of the 2018 National Information Technology Conference (NITC), Colombo, Sri Lanka, 2–4 October 2018; pp. 1–5.
28. Ochilbek, R. A new approach (extra vertex) and generalization of Shoelace Algorithm usage in convex polygon (Point-in-Polygon). In Proceedings of the 2018 14th International Conference on Electronics Computer and Computation (ICECCO), Kaskelen, Kazakhstan, 29 November–1 December 2018; pp. 206–212.
29. Antonio, F. Faster line segment intersection. In *Graphics Gems III (IBM Version)*; Morgan Kaufmann: Burlington, MA, USA, 1992; pp. 199–202.
30. Chang, S.C.; Huang, H.Y.; Huang, Y.F.; Yang, C.Y.; Hsu, C.Y.; Chen, J.S. An efficient geographical place mining strategy for social networking services. In Proceedings of the 2019 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW), Ilan, Taiwan, 20–22 May 2019; pp. 1–2.