Recommending the Least Congested Indoor-Outdoor Paths without Ignoring Time

Vasilis Ethan Sarris
University of Pittsburgh
Pittsburgh, PA, USA
vas82@pitt.edu

Panos K. Chrysanthis
University of Pittsburgh
Pittsburgh, PA, USA
panos@cs.pitt.edu

Constantinos Costa
Rinnoco Ltd.
Limassol, Cyprus
costa.c@rinnoco.com

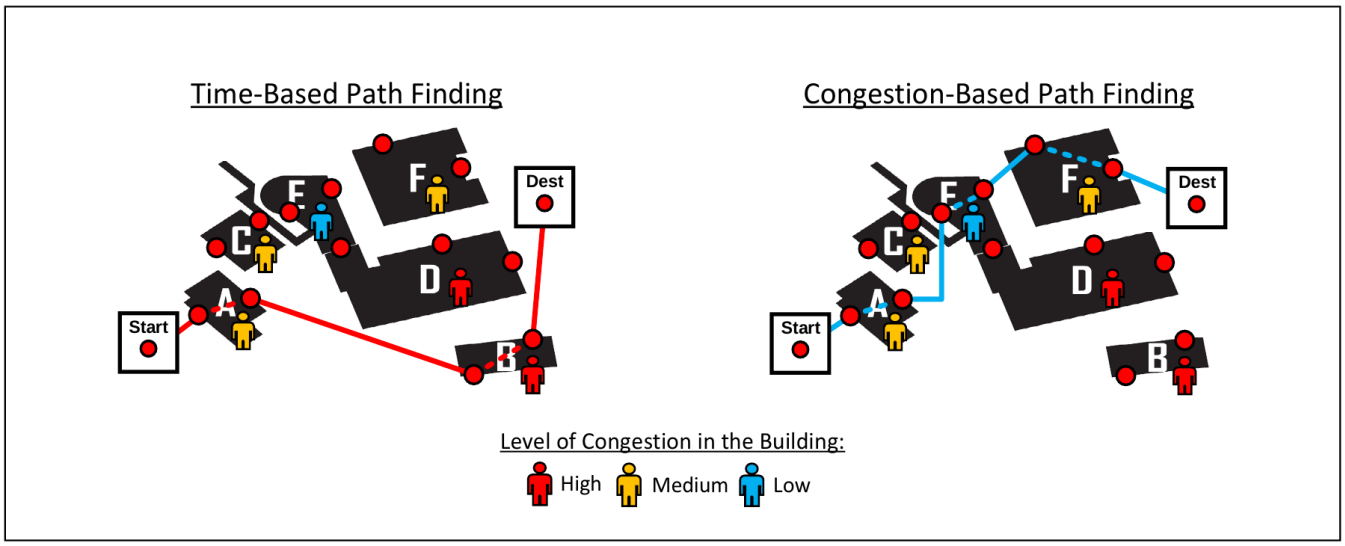


Figure 1: Time-based Path Finding may sacrifice exposure risk in order to optimize for time (left); Congestion-based Path Finding allows us to avoid this sacrifice (right).

ABSTRACT

The exposure to viral airborne diseases is higher in crowded and congested spaces, the COVID-19 pandemic has revealed the need of pedestrian recommendation systems that can recommend less congested paths which minimize exposure to infectious crowd diseases in general. In this paper, we introduce *ASTRO-C*, an extension of previous work *ASTRO*, which optimizes for minimum congestion. To our knowledge, *ASTRO-C* is the only solution to this problem of constraint-satisfying, indoor-outdoor, congestion-based path finding. Our experimental evaluation using randomly generated Indoor-Outdoor graphs with varying constraints matching various real-world scenarios, show that *ASTRO-C* is able to recommend paths with, on average a 0.62X reduction in average congestion, while on average, total travel time increases by 1.06X and never exceeds 1.10X compared to *ASTRO*.

SSTD '23, August 23–25, 2023, Calgary, AB, Canada

ACM ISBN 979-8-4007-0899-2/23/08…$15.00
https://doi.org/10.1145/3609956.3609969

CCS CONCEPTS

• **Information systems** → **Location based services**; **Spatial-temporal systems**; **Location based services**; **Mobile information processing systems**.

KEYWORDS

Pedestrian Path Recommendation, Constraint-based Path Finding, Indoor-Outdoor Graphs Generation, Indoor Congestion, COVID-19, Crowd Diseases

ACM Reference Format:
Vasilis Ethan Sarris, Panos K. Chrysanthis, and Constantinos Costa. 2023. Recommending the Least Congested Indoor-Outdoor Paths without Ignoring Time. In *Symposium on Spatial and Temporal Data (SSTD '23), August 23–25, 2023, Calgary, AB, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3609956.3609969

1 INTRODUCTION

Given the economic, environmental and quality of life impact of traffic congestion, route/outdoor recommendation systems consider traffic congestion and recommend less congested and less-time consuming alternative routes to a given destination [10]. The COVID-19 pandemic has revealed an analogous impact of congestion and pointed out for the need of pedestrian recommendation systems that can recommend less congested paths that minimize

exposure to COVID-19 and infectious viral diseases in general. This observation led us to develop *ASTRO* [6], to support *Mobile Contact Avoidance Navigation (MCAN)* [17] by implementing *CAPRIO* [7, 8].

ASTRO (Accessible Spatio-Temporal Route Optimization) is a novel constraint-satisfying indoor-outdoor pedestrian path finding algorithm, which considers a number of factors including predicted congestion when finding a least-time path that satisfies a set of user-given constraints. *ASTRO* was subsequently extended to *ASTRO-K* [20] which finds the top-k sufficiently-distinct constraint-satisfying indoor-outdoor paths. This extension allowed *CAPRIO* to disperse congestion inadvertently created by *CAPRIO* route recommendations, across a number of paths, thus mitigating its contribution to the congestion of any one area.

ASTRO and *ASTRO-K* offer a shortest path to a destination. However, this may not be the most desirable for an individual with an underlying health condition and at high-risk for a viral infection. Such an individual would prefer the least congested path even if this is longer as it would help maximize the likelihood that this high-risk user stays healthy.

In this paper, we introduce *ASTRO-C*, another extension of *ASTRO*, which computes the *minimum congestion*, constraint-satisfying Indoor-Outdoor path for a given query. This provides *CAPRIO* the ability to recommend paths for those whose primary concern is not time, but rather potential exposure to a viral infection, other crowd diseases and sensory-overloading areas (e.g., areas with loud noise, flashing lights, and/or strong smells). *ASTRO-C* selects among equally congested paths the one which is less-time consuming.

We evaluate *ASTRO-C* experimentally using randomly generated Indoor-Outdoor graphs with varying graph generation constraints to showcase various real-world scenarios. Our experimental results show that *ASTRO-C* is able to recommend paths with, an average $0.62X$ reduction in average congestion, and in the best case a reduction of $0.56X$, while on average, total travel time increases by $1.06X$ and never exceeds $1.10X$ compared to *ASTRO*. These congestion reduction gains come at the cost of execution time. Compared to *ASTRO*, *ASTRO-C* explores an average $7.55X$ more vertices (i.e., buildings), yet it scales well with the number of buildings while retaining the same interactive characteristics as *ASTRO*.

Our main contributions in this paper are as follows:

- We introduce *ASTRO-C*, an extension of *ASTRO* which optimizes for minimum congestion without ignoring time. To our knowledge, *ASTRO-C* is the only solution to this problem of constraint-satisfying, indoor-outdoor, congestion-based path finding.
- We present a random Indoor-Outdoor graph generator for modeling various real-world scenarios.
- We perform an experimental evaluation and show that *ASTRO-C* is able to return paths that significantly reduce average congestion along a path.

The rest of the paper is structured as follows. In Section 2, we review the necessary prior knowledge for *ASTRO-C*. In Section 3, we formalize the least-congestion path finding problem as well as present *ASTRO-C* in detail. In Section 4, we go in detail describing the Random Graph Generator. In Sections 5 and 6, we describe the experimental methodology and results. In Section 7, we discuss related work before finishing with our conclusions in Section 8.

2 BACKGROUND

In this section, we review the information needed to understand *ASTRO-C* . This includes a description of Indoor-Outdoor graphs (Section 2.1) which are able to represent buildings, doors and corridors within buildings, an overview of the *ASTRO* and *ASTRO-K* constraints (Section 2.2), and finally, a review of the *ASTRO* algorithm (Section 2.3).

2.1 Indoor-Outdoor Graphs

An Indoor-Outdoor Graph $G(V_o, E_o, G_{indoor})$ consists of *outdoor vertices* V_o, *outdoor edges* E_o and *Indoor Graphs* $G_{indoor}(V_i, E_i)$. Indoor Graphs G_{indoor} are bidirectional weighted graphs which are comprised of indoor vertices V_i, representing only the doors allowing people to enter and exit buildings, and edges E_i represent the paths between any two indoor vertices. Outdoor vertices are comprised of Indoor Graphs and represent buildings. Indoor-Outdoor Graphs are bidirectional weighted graphs comprised of outdoor vertices and edges represent the outdoor paths between buildings, connecting each of their entrances. The structure of an Indoor-Outdoor Graph enables *ASTRO*, *ASTRO-K* and *ASTRO-C* to find paths seamlessly traversing through both indoor and outdoor spaces.

2.2 Mobility Constraints

Mobility constraints capture *user preferences and abilities*, by tuning these constraints users can personalize the indoor-outdoor paths [14] they are recommended.

Definition 2.1. Outdoor Exposure (E) – the maximum amount of time allowed to traverse any given edge between two outdoor vertices.

Definition 2.2. Time Limit (T) – the maximum amount of time allowed to traverse the indoor-outdoor path.

Definition 2.3. Congestion (C) – the maximum amount of congestion which can be encountered while traversing between two indoor vertices. Congestion is measured by $number\ of\ people/m^2$.

Definition 2.4. Accessibility (A) – a binary constraint that represents the requirement for accessible doors, floors and corridors. For example, sufficiently wide doors (both into and out a building), with easily accessible automatic door openers and ramps.

2.3 The *ASTRO* Algorithm

ASTRO is a constraint-based variant of the A^* algorithm that optimizes for travel time. Being an optimal A^* variant means that *ASTRO* is guaranteed to have optimal edge selection when traversing the graph [3].

Given an Indoor-Outdoor graph, *ASTRO* behaves like standard A^*, traversing the graph over the outdoor vertices but then at each step also expands the current outdoor vertex v_c's Indoor Graph $G^{v_c}_{indoor}$ to find the best ordered pair of indoor vertices to use as the entry and exit for the current outdoor vertex v_c. Unlike standard A^* when an edge is expanded it may be pruned if it no longer meets the

constraints $\Pi = (E, T, C, A)$. Pruning edges in this manner allow us to avoid constructing the complete Indoor-Outdoor graph, which would be prohibitively expensive.

As mentioned above, in *ASTRO*, the unit of measure for both the exact cost function $g()$ and estimated cost heuristic $h()$ is *time* rather than distance. The exact cost $g()$ for an outdoor vertex is the sum of the time it takes to traverse to the current vertex, and the heuristic cost $h()$ is an estimate of the amount of time it will take to reach the terminal vertex from the current vertex. This change in unit of measure is accomplished by simply multiplying the distance traversed by the average walking speed of a person ($1.4m/s$) [13, 19].

$$outdoor\_time\_segment = \frac{distance}{1.4m/s} \tag{1}$$

For paths within between indoor vertices, congestion is also taken into account when calculating time. This is done by also adding the product of the distance with the predicted congestion of that distance. Since $congestion$ is a fraction, we use $1 + congestion$ to modify distance in order to maintain the monotonic nature of the equation.

$$indoor\_time\_segment = \frac{distance * (1 + congestion)}{1.4m/s} \tag{2}$$

This change from distance to time although simple is critical to the algorithm as it allows *ASTRO* to dynamically compute predicted congestion based on the estimated time of arrival.

3 LEAST-CONGESTION PATH FINDING

In this section, we formalize the problem of least congestion path finding (Section 3.1), discuss a few key assumptions we have made about this problem (Section 3.2), and present our algorithm *ASTRO-C* (Section 3.3).

3.1 Problem Definition

Definition 3.1. Given an Indoor-Outdoor graph $G(V_o, E_o, G_{indoor})$, a source outdoor vertex v_s, a terminal outdoor vertex v_t, a departure time D, and a set of constraints $\Pi = (E, T, C, A)$, find the constraint-satisfying path with the minimal summed congestion which would be experienced by the user.

The metric for "congestion experienced by the user" is the average congestion experienced at the time the user would arrive at the given indoor path. The path returned must still satisfy the given maximum congestion constraint C outlined above.

3.2 Key Assumptions

Before presenting the algorithm, let us first outline two key assumptions that are used by *ASTRO-C*.

Assumption 1: As long as a metric is monotonically increasing, it can be used as the basis for a path finding cost function.

Although typically when path finding, metrics such as distance or time are used, this does not necessarily have to be the case. Using this assumption, our cost function can be based on the sum of any metric within our path. This key idea is what enables *ASTRO-C*'s least-congestion path finding. By leveraging the predicted congestion metrics, *EpicGen* [2] produces for a given indoor path and arrival time (μ, σ, min, max), we can create a cost function which ranks paths based on the sum of the average congestion along the path as described in Equation 3.

However, the previous congestion is unable to predict future congestion with any guarantees. Thus, the major limitation of a cost function calculated this way is that we were unable to create a useful path finding heuristic with any guarantees of optimally.

$$g(path) = \sum_{e \in path.E_i} e[avg\_congestion] \tag{3}$$

$$h(path) = path[total\_time] + time\_heuristic(path) \tag{4}$$

Assumption 2: The heuristic function $h()$ can be decoupled from the cost function $g()$ and thus need not use the same metrics.

The goal of a heuristic function is to allow us to reduce our computational cost by using our knowledge of the graph to better select edges during path finding. Although we were unable to create a heuristic with guarantees for our congestion-based cost function, using this assumption, we were still able to optimize our search. *ASTRO-C* employs a cost-based search algorithm using heuristic priority, where if there are choices with equivalent cost the heuristic is able to determine priority (Algorithm 1). Using Equation 3 as the congestion-based cost function, and the time-based cost/heuristic functions of *ASTRO* as our heuristic (Equation 4), we are now able to rank equivalent paths with priority. This is most helpful in case such as comparing two different indoor paths of the same outdoor vertex.

Algorithm 1: New priority queue compare function compatible with both *ASTRO* / *ASTRO*

Input: v_1: Outdoor Vertex, v_2: Outdoor Vertex,
Output: $v_1 < v_2$

```
1: if v1[fscore] ≠ v2[fscore] then
2:     return v1[fscore] < v2[fscore]
3: else return v1[hscore] < v2[hscore]
```

3.3 *ASTRO-C*

Using the cost and heuristic functions defined in the previous section, *ASTRO-C* provides a solution to the problem of least-congestion path finding (Algorithm 2).

High-Level Idea: *ASTRO-C* contains a priority queue of outdoor vertices and at each step the least-congestion, non-closed outdoor vertex is explored. The exploration consists of constructing the vertex's indoor graph, finding the least congested indoor path, adding the outdoor vertex with the information about the outdoor/indoor path into the queue, and then setting the outdoor vertex as closed. Each outdoor vertex is only expanded once before being set to closed, in order to minimize expensive indoor graph constructions. Once the terminal outdoor vertex is the head of the

Algorithm 2: *ASTRO / ASTRO-C*

Input: v_s: Source Outdoor Vertex, v_t: Terminal Outdoor Vertex, O: Set of Outdoor Vertices, Π Set of User-Given Constraints, *toAdd* Macro: Algorithm 3
Output: p: best path

1: Initialize Priority Queue *OPEN* and Set *CLOSED*
2: $OPEN.push(v_s)$
3: **while** !$OPEN.empty()$ **do**
4: $curr \leftarrow OPEN.pop()$
5: **if** $curr = v_t$ **then return** $p \leftarrow ReconstructPath(curr)$
6: **else if** $curr \notin CLOSED$ **then**
7: $CLOSED \leftarrow CLOSED \cup curr$
8: **for** $outdoor \in \{O - CLOSED\}$ **do**
9: $IndoorGraph \leftarrow outdoor[IndoorGraph]$
10: **for** $in \in IndoorGraph$ **do**
11: $\overrightarrow{st_1} \leftarrow Status(curr[out], in, curr[time])$
12: **for** $out \in \{IndoorGraph - in\}$ **do**
13: $exitTime \leftarrow curr[time] + \overrightarrow{st_1}[time]$
14: $\overrightarrow{st_2} \leftarrow Status(in, out, exitTime)$
15: $\overrightarrow{st} \leftarrow \overrightarrow{st_1} + \overrightarrow{st_2}$
16: $g \leftarrow curr[g] + \overrightarrow{st}[g]$
17: **if** $g < outdoor[g]$ ***and*** $CheckConstraints(\Pi, \overrightarrow{st})$ **then**
18: $outdoor[g] \leftarrow g$
19: Initialize outdoor vertex *toAdd*
20: Remove any vertices with the same outdoor vertex as *toAdd* from *OPEN*
21: $OPEN.push(toAdd)$
end
end
end
end
end

queue, we have found the least-congestion path and terminate path finding.

Algorithm: First, the outdoor vertex priority queue *OPEN* with the source vertex v_s and the empty set of outdoor vertices *CLOSED* (Line 1-2) are initialized. It should be pointed out that v_s is initialized with the departure time D before being passed to *ASTRO-C*. From there, *ASTRO-C* will continue to loop (Lines 3-19) until either the destination has been found (Line 5) or the priority queue *OPEN* is empty and the set of outdoor vertices has been exhausted, meaning no path exists.

The loop starts by setting our outdoor vertex *curr* to the head of *OPEN* and then pops the head from the priority queue (Line 4). As described earlier, the current outdoor vertex is then checked to see if it is the user's destination (Line 5). If not, the algorithm continues by checking if *curr* has already been visited (Line 6). If it has, we break out and move on to the next vertex in the queue. If it has not been visited, *ASTRO-C* will proceed to construct the outdoor vertex's indoor graph (Line 8).

Once the edges have been populated with congestion values, a loop between every pair of indoor vertices finds the best outdoor + indoor path (Lines 10-19). The outdoor path is constructed using the indoor vertex *out* of *curr* as the start and the indoor vertex *in* we are currently exploring as the end (Line 11). The indoor path is constructed using that same indoor vertex *in* but now as the start and the indoor vertex *out* we are currently exploring as the end (Line 13). Once the indoor path is constructed, we join this with the outdoor path and calculate the cost (Line 15-16). The cost is then compared to the current best path within the outdoor vertex and the constraints (Line 17).

If the path passes this check, the current best path within the outdoor vertex is updated (Line 18), an outdoor vertex *toAdd* with the appropriate information is initialized (Line 19 / Algorithm 3), any vertex in the queue with the same outdoor vertex as *toAdd* is removed from the queue (Line 20), and *toAdd* is pushed to the queue (Line 21).

Algorithm 3: *toAdd* Vertex Initialization Macro

$toAdd \leftarrow OutdoorVertex(status, in, cameFrom, g, h, f)$
▷ *ASTRO*
$OutdoorVertex(\overrightarrow{st}, in, curr, g, h(out, t), g + h)$
▷ *ASTRO-C*
$OutdoorVertex(\overrightarrow{st}, in, curr, g, h(out, t), g)$

4 RANDOM GRAPH GENERATOR

In order to be able to run experiments which could replicate a wide variety of scenarios, we built a configurable Indoor-Outdoor graph $G(V_o, E_o, G_{indoor})$ generator. The graph generator has the following six configuration parameters that define the Outdoor Graph (i.e., the buildings and their positions) and the level of the congestion in the different outdoor vertices:

- N - Number of Buildings (Integer),
- P - Total Area Covered by the N Buildings (%),
- h - High Congestion Buildings (%),
- m - Medium Congestion Buildings (%),
- l - Low Congestion Buildings (%),
- *constant* - Constant Congestion ($True/False$).

The specific characteristics of each outdoor vertex's Indoor Graph G_{indoor} are randomly generated. This process, as well as the graph generation as a whole are described in the next section.

4.1 Graph Overview

The graph we generate is built upon a 2-dimensional square grid. Each pair of x and y coordinates is a place where an outdoor vertex could be generated. Each outdoor vertex is randomly determined to have between 2-5 indoor vertices, the exact location of which is generated by adding a random polar coordinate offset (r, θ) to the x and y of the outdoor Vertex. A Polar coordinate offset allows us to easily set a bound on the maximum distance away from the outdoor vertex using r, and equally partition the areas an indoor

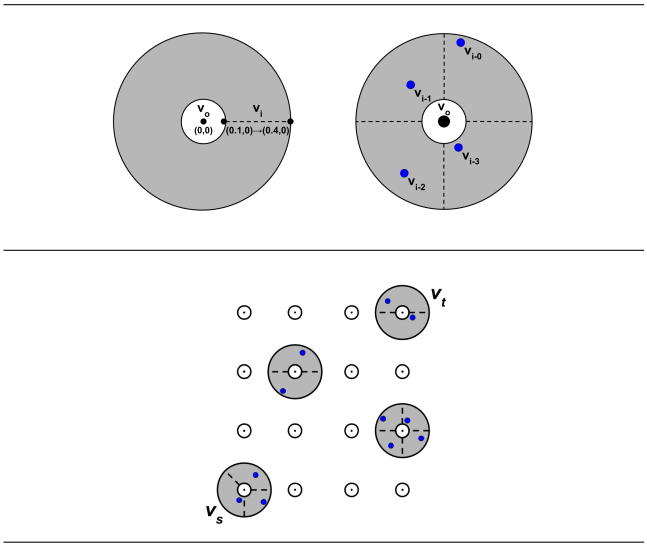


Figure 2: Visualization of the area that a new indoor vertex v_i can be placed around a coordinate (Top Left); Example of a generated Outdoor Vertex with 4 indoor vertices v_{i-n} (Top Right); Example of fully generated graph G using the graph generation parameters $N = 4, P = 0.5$ (Bottom).

vertex can be placed using θ. This is important for multitude of reasons:

(1) Setting a lower bound for r guarantees there will always be at least some indoor time traversing a building;
(2) Setting an upper bound for r guarantees our the area the indoor vertices can be located will never overlap between outdoor vertices;
(3) Assigning ranges of θ to each indoor vertex allows us to distinctly partition the space around the outdoor vertex, guaranteeing no overlapping indoor vertices.

A visualization of the total space indoor vertices can occupy, an example of a fully generated outdoor vertex, including lines to visualize the θ partitions, and a visualization of a fully generated random graph can be found in Figure 2. Please note the shape of the building is not important, what is important is that the buildings do not overlap and the doors define a possible topology.

It's important to note that since *CAPRIO* is built for the real-world coordinates, these x and y values would typically correspond to latitudes and longitudes. This means that in our generated graph the distance between two vertically or horizontally adjacent points would be around $111km$. To make our results more akin to what would be typical of an urban environment, we have scaled our experimental results down by a factor of 11100. This means, that instead of $111km$, the distance between two adjacent points is $10m$.

4.2 Graph Generation Algorithm

The generation process, using the parameters we described above, is outlined in Algorithm 4.

First, the generator initializes the new Indoor-Outdoor graph G we are generating (Line 1) and the bounds of the graph's 2D grid are calculated in such a way that ensures there are enough spaces for every building with the coverage parameter (Line 2-3). To generate the locations of the outdoor vertices an array containing all possible points in the grid is created (Line 4), and then, these values are randomly shuffled before initializing the *buildings* array using the first N values (Lines 5-6). For each value in *buildings*, the coordinates x and y are calculated (i.e., if the *bounds* are 5, and b is 14, then $x = \lfloor \frac{14}{5} \rfloor$ and $y = 14 \bmod 5$) and the outdoor vertex is added to G (Lines 8-10).

Once the outdoor vertex has been added the graph and the number of indoor vertices, *doors*, has been decided (Lines 10-11), the location and congestion of the indoor vertices can then be generated. The location of the indoor vertex is determined by a Polar offset from the outdoor vertex's Cartesian coordinates for the reasons outlined in the section above. To do this, we generate (1) a r value with a bound of $[0.1, 0.4)$ (Line 14), and (2) a θ value with a bound of $[0, \frac{360}{doors})$. To calculate the location of the indoor vertex using this offset, it is converted to Cartesian values, then adding to the location of the outdoor vertex (Line 17).

The congestion for the indoor vertex is decided by retrieving a random variable from a Gaussian distribution with a mean determined by the level of congestion the building is supposed to have and a standard deviation of 0.2, or it is given a constant value of 1 if the *constant* parameter is set (Lines 18-21). The *high congestion* buildings have a mean of $2\,people/m^2$, the medium congestion buildings have a mean of $1.25\,people/m^2$, and the *low congestion* buildings have a mean of $0.75\,people/m^2$. The values were based around the standard social distancing guidelines of $1\,person/m^2$. Once the indoor vertex's congestion is generated it is added to the outdoor vertex's Indoor Graph (Line 22).

It's important to note that Lines 18-21 simplify the congestion generation into one value, however, in practice a 24-hour time-series of 5-minute intervals is generated. The congestion used in our generated graph is static throughout the day so the one value populates the entire time-series, but in our real-world data sets this is not the case. In Section 6 of this paper, we address this shortcoming with an experiment evaluating how *ASTRO-C* performs with graphs simulating congestion patterns typical of various times throughout the day.

Table 1: Random Graph Generator Parameters

Symbol	Definition	Default	Range
N	Number of Buildings	100	25,50,100,200
P	Total Area Covered by N Buildings	75%	25%, 50%, 75%
h	High Congestion Buildings	30%	4%, 27%, 69%
m	Medium Congestion Buildings	40%	4%, 27%, 69%
l	Low Congestion Buildings	30%	4%, 27%, 69%
constant	Buildings with Constant Congestion	False	True, False

5 EXPERIMENTAL METHODOLOGY

This section provides details regarding the algorithms, random graph generator configurations, and metrics used for the evaluation of *ASTRO-C*.

Algorithms: To our knowledge, ASTRO is the current state-of-the-art in Indoor-Outdoor path-finding using predictive congestion.

Algorithm 4: Graph Generation

Input: N: Buildings, P: Coverage (%), h: High Congestion (%), m: Medium Congestion (%), l: Low Congestion (%), $constant$: Constant Congestion
Output: G: Indoor-Outdoor Graph

▷ **Step 1:** Initialize Graph and Calculate the Grid's Bounds
1: $G \leftarrow new\ IndoorOutdoorGraph$
2: $total\_area \leftarrow N/P$
3: $bounds \leftarrow \lceil \sqrt{total\_area} \rceil$
▷ **Step 2:** Generate Coordinates for Outdoor Vertices
4: $possible\_buildings \leftarrow Range(0, bounds^2)$
5: $shuffle(possible\_buildings)$
6: $buildings \leftarrow possible\_buildings[0, N-1]$
▷ **Step 3:** Generate Coordinates for Indoor Vertices
7: **for** $i \leftarrow 0; i < N; i++$ **do**
8: $b \leftarrow buildings[i]$
9: $(b_x, b_y) \leftarrow (\frac{b}{bounds}, b\ \%\ bounds)$
10: $G.add(OutdoorVertex(id = b, x = b_x, y = b_y)$
11: $doors \leftarrow rand(2, 5)$
12: $deg\_partition \leftarrow \frac{360}{doors}$
13: **for** $j \leftarrow 0;\ j < doors;\ i++$ **do**
▷ **Step 4:** Generate Offset for Current Indoor Vertex
14: $r \leftarrow rand(0.1, 0.4)$
15: $\theta \leftarrow (j * deg\_partition + rand(0, angle\_partition))$
16: $\theta \leftarrow \theta * \frac{\pi}{180}$ /* convert to radians */
17: $(d_x, d_y) \leftarrow (b_x + r * cos\theta, b_y + r * sin\theta)$
▷ **Step 5:** Generate Congestion for Current Indoor Vertex
18: **if** $i < N \times h$ **then**
$con \leftarrow gaussian(\mu = 2, \sigma = 0.2)$
end
19: **else if** $i < N(h + m)$ **then**
$con \leftarrow gaussian(\mu = 1.25, \sigma = 0.2)$
end
20: **else**
$con \leftarrow gaussian(\mu = 0.75, \sigma = 0.2)$$;
end
21: **if** $constant$ ***or*** $con < 0$ **then** $con \leftarrow 1$
22: $G[b].add(IndoorVertex(id = j, x = d_x, y = d_y, congestion = con)$
end
end

By comparing *ASTRO-C* to *ASTRO*, we are able to gain insight into the different paths that may be recommended to users of *CAPRIO*.

Methodology: A test of *ASTRO* and *ASTRO-C* on the University of Pittsburgh and University of Cyprus data sets using *CAPRIO* provided no significant insights. However, the use of generated graphs inspired by these real-world data sets allows us to perform a through sensitivity analysis, which would otherwise not be possible.

In order to provide a holistic view and gain insights into the trends, a generated a set of 10 random graphs with a given set of parameters was used for each run of an experiment. Additionally, we are providing the average of the results of these 10 graphs, which illustrates the average case given a configuration, rather than focusing only on a single case using a single configuration.

Graph Configuration: The configuration used as the default for our graph generator is $N = 100$, $P = 0.75$, $h = 0.3$, $m = 0.4$, $l = 0.3$, $constant = False$. The experimental range of values for the parameters are summarized in Table 1.

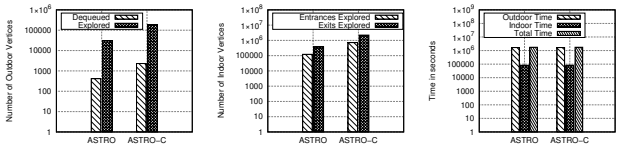
Path Finding Constraints: The path finding constraints are $E = 30m$, $T = (2 * graph\_bounds * distance)/1.4$, $C = unbounded$, and $A = False$. Although, total time is typically a smaller value and congestion is typically bound in the real-world use case, by minimizing the number of number of paths pruned via these constraints, our experimental results highlight only the differences caused by *ASTRO-C*'s modifications to *ASTRO*.

Metrics: In each trial we record a number of measurements to evaluate *ASTRO* and *ASTRO-C*. Each metric can be broken down into one of three different categories: *time, congestion*, and *computational cost*. Additionally, each metric in all three categories is accompanied by a performance ratio (*ASTRO-C* / *ASTRO*).

- Time Metrics (in *seconds*):
  - Total Time: time to traverse entire path.
  - Indoor Time: time to traverse indoor sections of a path.
  - Outdoor Time: time to traverse outdoor sections of a path.
- Congestion Metrics (in $\#\ of\ people/m^2$):
  - Average Congestion: average of all the indoor paths' average congestion.
  - Minimum Congestion: minimum indoor path average congestion.
  - Maximum Congestion: maximum indoor paths average congestion.
- Computational Cost Metrics (in *# of vertices manipulations*):
  - Outdoor Vertices Dequeued: incremented on Line 4 of Algorithm 2.
  - Outdoor Vertices Expanded:incremented on Line 8 of Algorithm 2.
  - Indoor Vertices Expanded: incremented on Line 10 of Algorithm 2 for entrances *in* and on Line 12 of Algorithm 2 for exits *out*.

6 EXPERIMENTAL RESULTS

To evaluate *ASTRO-C* performance characteristics with respect to achieved congestion reduction, computational cost, total traveling time of the recommended path, and scalability, we carried out six experiments. In each experiment, any non-specified graph generation or path finding constraints can be assumed to be the default values defined in the previous section and listed in Table 1. Recall, the path finding constraints T and C are *practically* unbounded as to minimize paths pruned via these constraints, to highlight the difference in path recommendations caused by the cost and heuristic functions.



	Outdoor Vertices Dequeued	Outdoor Vertices Expanded	Indoor Vertices Explored (Entrances | Exits)
ASTRO	416.0	31101.0	121835.0 | 377448.0
ASTRO-C	2299.0	183401.0	718355.0 | 2224968.0
Performance Ratio (ASTRO-C / ASTRO)	5.53X	5.90X	5.90X | 5.90X

	Total Time (s)	Indoor Time (s)	Outdoor Time (s)
ASTRO	1747348.85	82472.64	1664876.21
ASTRO-C	1747348.85	82472.64	1664876.21
Performance Ratio (ASTRO-C / ASTRO)	1.0X	1.0X	1.0X

Figure 3: Comparison of *ASTRO* vs. *ASTRO-C* in Constant Congestion Environments

The scalability experiments were designed using data sets based on OpenStreetMaps (OSM) [18] data for the University of Pittsburgh and University of Cyprus campuses. As an urban campus, the Pittsburgh campus consists of more than 100 buildings in an area of 0.75 square miles, whereas the Cyprus campus as a suburb campus consists of a small number of clusters of buildings/athletic installations spread over an area of 1.25 square miles.

6.1 Experiment 1: *ASTRO* vs *ASTRO-C* Paths with Constant Congestion

Our first experiment, shown in Figure 3, explores how *ASTRO* and *ASTRO-C* compare in constant congestion environments. Since *ASTRO-C* tiebreaks with the cost and heuristic functions of *ASTRO* the paths should be the same. As we can see in our results, *ASTRO-C* continues to incur additional computational cost but *does* recommend the same path.

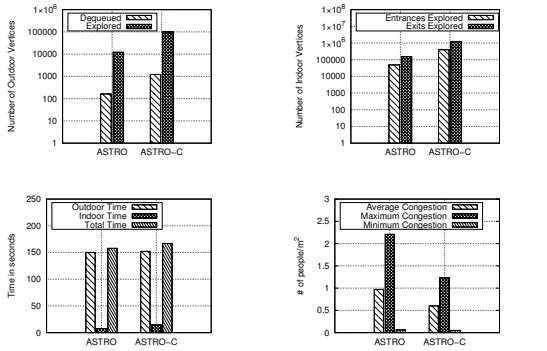
Summary: : In constant congestion environments, *ASTRO* and *ASTRO-C* will recommend the same paths.

6.2 Experiment 2: *ASTRO* vs. *ASTRO-C* Paths

Our second experiment, shown in Figure 4, compares the results of *ASTRO* and *ASTRO-C* across all recorded metrics using the standard graph generation configuration. The purpose of this experiment is to gain insight into the behaviour of the two algorithms in the general case.

Looking at our results we can see three major trends:

- First, we can see that *ASTRO-C* incurs a large computational cost. Specifically, it explores on average $7.55X$ more of vertices than *ASTRO*. This is because *ASTRO-C* does not have an optimal heuristic and is not able to prune many of the paths *ASTRO* is (as discussed in Section 3.2).
- Second, *ASTRO-C* is able to consistently recommend paths with similar travel times as *ASTRO*. Specifically, the travel time increases on average by $1.06X$ and never exceeds $1.10X$ compared to *ASTRO*.
- Finally, *ASTRO-C* is able to significantly reduce the amount of congestion across all metrics. On average, *ASTRO-C* is able to recommend paths with $0.62X$ less average congestion and in the best case it achieved a reduction of $0.56X$.



	Outdoor Vertices Dequeued	Outdoor Vertices Expanded	Indoor Vertices Explored (Entrances | Exits)
ASTRO	162.0	12381.0	49043.0 | 154915.6
ASTRO-C	1222.4	99541.0	394642.0 | 1248531.6
Performance Ratio (ASTRO-C / ASTRO)	7.55X	8.04X	8.05X | 8.06X

	Total Time (s)	Indoor Time (s)	Outdoor Time (s)
ASTRO	158.04	7.70	150.34
ASTRO-C	166.95	14.61	152.34
Performance Ratio (ASTRO-C / ASTRO)	1.06X	1.90X	1.01X

	Average Congestion ($\#\ of\ people/m^2$)	Minimum Congestion ($\#\ of\ people/m^2$)	Maximum Congestion ($\#\ of\ people/m^2$)
ASTRO	0.97	0.06	2.21
ASTRO-C	0.60	0.05	1.23
Performance Ratio (ASTRO-C / ASTRO)	0.62X	0.83X	0.56X

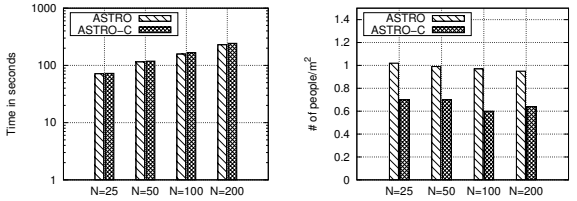
Figure 4: Comparison of *ASTRO* vs. *ASTRO-C* Paths

Summary: Consistently, *ASTRO-C* is able to recommend paths with less average congestion and similar total travel time as *ASTRO*, while retaining the same interactive characteristics as *ASTRO*.

6.3 Experiment 3: Varying the Number of Buildings

Our third experiment, shown in Figure 5, compares the recommended paths of *ASTRO-C* and *ASTRO* as the number of outdoor vertices (buildings), N, scales while keeping P (percentage of area coverage) constant to the default value. This should keep the number of options within the Outdoor Exposure E constraint similar while increasing the number of buildings in the path.

Looking at our results we can see that as N increases, there is a slight increase in the total time performance ratio but not a decrease in the congestion performance ratio like we might expect. This is likely due to the high congestion buildings becoming less of a factor in the average *ASTRO* congestion while *ASTRO-C* is finding



	Total Time (s)			Average Congestion (# of people/m^2)		
	ASTRO	ASTRO-C	Performance Ratio	ASTRO	ASTRO-C	Performance Ratio
N = 25	71.64	72.73	1.02X	1.02	0.7	0.68X
N = 50	115.42	118.87	1.03X	0.99	0.70	0.71X
N = 100	158.04	166.95	1.06X	0.97	0.6	0.63X
N = 200	229.49	243.14	1.06X	0.95	0.64	0.67X
Average	143.54	150.59	1.05X	0.98	0.66	0.69X

Figure 5: Comparison of *ASTRO* vs. *ASTRO-C* as the Number of Buildings Scales

buildings approaching the lower bound of the possible congestion values. As we described in Section 4.2, the mean for the congestion assigned to low congestion outdoor vertices was 0.75 with the default deviation of 0.2.

Summary: As N scales, *ASTRO-C* is able to more reliably approach the lower bound of congestion within the graph while incurring only a slight increase in total time.

6.4 Experiment 4: Varying the Percentage of Area Covered by Buildings

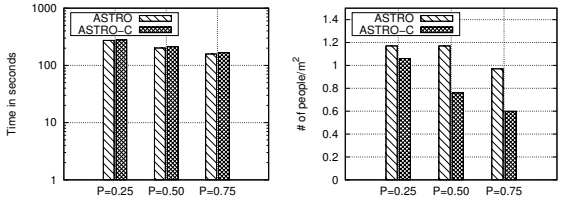
Our fourth experiment, shown in Figure 6, we vary P, the percentage of the total area covered by outdoor vertices, while using the default building values for N. Increasing this value should increase the number of vertices within the Outdoor Exposure E constraint at any given outdoor vertex. As P decrease we see the difference between the paths *ASTRO* and *ASTRO-C* recommend significantly decrease. This is because, as the number of edges satisfying the outdoor exposure constraint decreases, the more likely it is *ASTRO* and *ASTRO-C* have to choose the same edge.

Summary: As P decreases, the differences between paths recommended by *ASTRO* and *ASTRO-C* become less significant because they are more often being forced to choose the same edges.

6.5 Experiment 5: Varying the Outdoor Time Limit

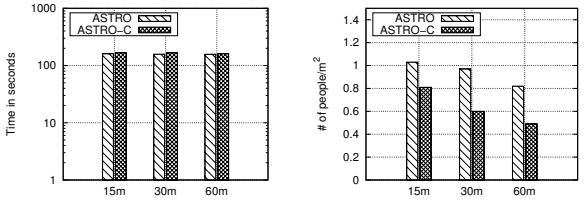
Our fifth experiment, shown in Figure 7, varies E from $15m$ to $60m$, the outdoor exposure constraint — $15m$ is the lower bound to ensure there is always a path able to be found, while $60m$ is the upper bound to ensure there is no direct path to the destination. Similar to the last experiment, increasing this value may increase the number of vertices within the outdoor exposure constraint at any given outdoor vertex while keeping the number of buildings the paths traverse through similar.

The results show at large values of E, outdoor vertices with minimal congestion indoor paths that would typically be pruned



	Total Time (s)			Average Congestion (# of people/m^2)		
	ASTRO	ASTRO-C	Performance Ratio	ASTRO	ASTRO-C	Performance Ratio
P = 0.25	273.92	278.76	1.02X	1.17	1.06	0.90X
P = 0.50	201.22	211.68	1.05X	1.17	0.76	0.65X
P = 0.75	158.04	166.95	1.06X	0.97	0.60	0.63X
Average	211.06	219.13	1.04X	1.10	0.81	0.74X

Figure 6: Comparison of *ASTRO* vs. *ASTRO-C* as the Building Density Scales



	Total Time (s)			Average Congestion (# of people/m^2)		
	ASTRO	ASTRO-C	Performance Ratio	ASTRO	ASTRO-C	Performance Ratio
15m	160.78	167.12	1.04X	1.03	0.81	0.79X
30m	158.04	166.95	1.06X	0.97	0.60	0.63X
60m	157.42	160.8	1.02X	0.82	0.49	0.60X
Average	158.75	164.96	1.04X	0.94	0.63	0.67X

Figure 7: Comparison of *ASTRO* vs. *ASTRO-C* as the Outdoor Time Constraint Scales

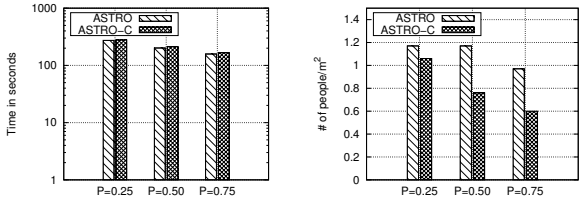
due to outdoor exposure are now available. Because of this, *ASTRO-C* is able to drastically reduce the user's exposure to congestion.

Summary: As E increases, the difference in the congestion of paths recommended by *ASTRO* and *ASTRO-C* increases as well.

6.6 Experiment 6: Simulations at Different Times of Day

Our final experiment, shown in Figure 8, tunes the parameters of high, medium, and low congestion buildings to simulate various times throughout the day. The three times we choose to simulate are 07:00, 12:00, and 20:00. The parameter values for each simulation are loosely based on what percentage of businesses would be open in an urban environment at that time. The parameters used for each simulation are as follows:

- 07:00: $h = 0.04$, $m = 0.69$, $l = 0.27$,
  - Many business have opened but not many people
- 12:00: $h = 0.69$, $m = 0.27$, $l = 0.04$
  - All businesses are open and there are many people



	Total Time (s)			Average Congestion ($\#\ of\ people/m^2$)		
	ASTRO	ASTRO-C	Performance Ratio	ASTRO	ASTRO-C	Performance Ratio
07:00	157.95	159.77	1.01X	0.91	0.69	0.76X
12:00	157.99	161.63	1.02X	1.25	0.97	0.78X
20:00	157.71	160.65	1.02X	0.62	0.46	0.74X
Average	157.88	160.02	1.02X	0.92	0.71	0.76X

Figure 8: Comparison of *ASTRO* vs. *ASTRO-C* in Simulations of Various Times Throughout The Day

- 20:00: – $h = 0.27$, $m = 0.04$, $l = 0.69$
  - Most business are close but the ones that are open typically have many people (restaurants, etc.)

In the simulations, *ASTRO-C* is able to find paths with $0.76X$ less congestion than *ASTRO*, on average. Even at peak times, when only 4% of buildings have low congestion, *ASTRO-C* is able to find paths with comparable times to *ASTRO*.

Summary: Throughout the day, *ASTRO-C* is consistently able to recommend paths than maintain an average congestion under $1\ person/m^2$ while having a similar total time to *ASTRO*.

7 RELATED WORK

In this section we discuss work strongly related to the primary contribution of this paper, *ASTRO-C*. A taxonomy of this discussion can be found in Table 2.

While there is lots of research within the realm of pedestrian navigation that use the terminology of congestion and path finding, often times these works are solving fundamentally different problems. Primary examples being research centered around pedestrian evacuation systems [11, 21–23] and pedestrian simulations [5, 12, 16]. Evacuation systems although other measuring congestion in similar ways to real-time data-dependent indoor path finding applications, the problem of finding the best exit as opposed to path finding enables solutions not practical for path finding applications. Simulations of pedestrian behaviours tend to focus on emulating how pedestrians *react* in congested spaces while traversing a path as opposed to recommending paths of least congestion. So as we can see, although these topics are similar, they are not within the scope of this paper. Congestion-aware path finding algorithms like *ASTRO-C* fall into 2 categories: *Real-Time Data-Driven* and *Predictive*.

Real-Time Data Driven Congestion: Systems in this realm are more similar to the evacuation systems and are more common in the literature. These systems often take advantage of IoT devices / wireless sensing technologies to localize users and then dynamically update the congestion of the applicable areas. Recent work using this approach include those used by Chen et al. [4] and Almari & Almari [1]. In both papers, the authors take advantage of the sensors available in their user's smartphones to interact with other nearby devices to reason about their indoor environments. In Chen et al [4], these other devices are Bluetooth Low Energy (BLE) devices while in Almari & Almari, these are other users. A less IoT-based real-time data-driven approach can be found in Damian et al. [9], which relies upon its history of previously recommended paths to update the environment and recommend least-congested paths. It is of note that the work done by Damian et al. is able to find indoor-outdoor paths and uses a multi-policy ranking systems to recommend paths, congestion is just one of their policies, not the focus of the work. The major limitation of all of these works being that there is no accounting for users not using the application.

Table 2: Taxonomy of Related Work

	Models Indoor Space	Models Outdoor Space	Congestion	Constraint-Based	Cost Metric
ASTRO	Yes	Yes	Predictive	Yes	Time
Chen et al.	Yes	No	Real-Time	No	Time
Almari & Almari	Yes	No	Real-Time	No	Time
Damian et al.	Yes	Yes	Real-Time	No	Normalized Aggregated Multi-Policy Cost
Liu et al.	Yes	No	Predictive	No	Congestion
ASTRO-C	Yes	Yes	Predictive	Yes	Congestion

Predictive Congestion: The two main techniques recently used for predicting congestion in the context of indoor path finding are based on *queuing theory* and *machine learning*. Liu et al. [15] is an example of the former. By modeling a building via the semantics of a space, they are able to construct a graph of connected queues each hyper-parameterized to simulate the congestion dynamics of a singe semantic area (i.e., predicts congestion of an entire hallway using one queue). *EpicGen* proposed in Chrysovalantis et al. [2] showcases the machine learning-based approach. Using a grid-based building model and a machine learning model trained on real-world data, this approach is able to predict congestion for any grid cell in a building at any time of the day (i.e., predicts congestion for a single cell in a hallway). In *ASTRO-C*, we adopted the machine learning based approach *EpicGen* inline with our previous work of *CAPRIO*.

8 CONCLUSIONS

The COVID-19 pandemic revealed the need for pedestrian recommendation systems that can recommend paths which minimize exposure to infectious diseases. This led us to define the least-congestion path finding problem as the exposure to viral airborne diseases is higher in crowded and congested spaces and introduce *ASTRO-C*, an extension of our previous work *ASTRO* , which optimizes for minimum congestion while still taking advantage of time as a heuristic cost. We further developed a random Indoor-Outdoor graph generator for modeling various real-world scenarios enabling detailed experimentation.

Our experimental results show that on average, *ASTRO-C* is able to recommend paths with a $0.62X$ reduction in average congestion and a total travel time increase of $1.06X$ compared to *ASTRO*. These congestion reduction gains come at the cost of increased computation. Compared to *ASTRO* , *ASTRO-C* explores an average $7.55X$ more vertices (i.e., buildings), due to having no optimal congestion path finding heuristic. Despite this it scales well in a

number of graph simulations while retaining the same interactive characteristics as *ASTRO*.

Although we were unable to construct a congestion-based heuristic, we remain hopefully that it is possible and leave this as the first major area of future work. The second area we feel has lots of potential is within enabling the random graph generator to produce realistic time-dependent dynamic congestion throughout the day by leveraging *EpicGen* [2], our experimental platform for detailed indoor congestion generation.

ACKNOWLEDGMENTS

This paper was part of the Master's project of the first author. This work was partially funded by NIH award R01HL159805. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of any of the sponsors.

REFERENCES

[1] Abdullah Alamri, Rayan Ali Alturki, and Sultan Alamri. 2022. Multi-user routing algorithm for indoor spaces – Adapted for social distancing. *Journal of King Saud University - Computer and Information Sciences* 34, 9 (2022), 7045–7058. https://doi.org/10.1016/j.jksuci.2022.06.015

[2] Chrysovalantis Anastasiou, Constantinos Costa, Panos K. Chrysanthis, and Cyrus Shahabi. 2021. EPICGen: An Experimental Platform for Indoor Congestion Generation and Forecasting. *Proc. VLDB Endow.* 14, 12 (2021), 2803–2806.

[3] Chrysovalantis Anastasiou, Constantinos Costa, Panos K. Chrysanthis, Cyrus Shahabi, and Demetrios Zeinalipour-Yazti. 2022. ASTRO: Reducing COVID-19 Exposure through Contact Prediction and Avoidance. *ACM Trans. Spatial Algorithms Syst.* 8, 2 (2022), 1–31.

[4] Lien-Wu Chen and Jun-Xian Liu. 2021. Time-Efficient Indoor Navigation and Evacuation With Fastest Path Planning Based on Internet of Things Technologies. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 51, 5 (2021), 3125–3135. https://doi.org/10.1109/TSMC.2019.2918233

[5] James C. Chu, Albert Y. Chen, and Yu-Fu Lin. 2017. Variable guidance for pedestrian evacuation considering congestion, hazard, and compliance behavior. *Transportation Research Part C: Emerging Technologies* 85 (2017), 664–683. https://doi.org/10.1016/j.trc.2017.10.009

[6] Constantinos Costa, Xiaoyu Ge, and Panos K. Chrysanthis. 2019. CAPRIO: Context-Aware Path Recommendation Exploiting Indoor and Outdoor Information. In *IEEE Intl. Conf. on Mobile Data Management*. 431–436.

[7] Constantinos Costa, Xiaoyu Ge, and Panos K. Chrysanthis. 2019. CAPRIO: Graph-based Integration of Indoor and Outdoor Data for Path Discovery. *Proc. VLDB Endow.* 12, 12 (2019), 1878–1881.

[8] Constantinos Costa, Xiaoyu Ge, Evan McEllhenney, Evan Kebler, Panos K. Chrysanthis, and Demetrios Zeinalipour-Yazti. 2020. CAPRIOv2.0: A Context-Aware Unified Indoor-Outdoor Path Recommendation System. In *IEEE Intl. Conf. on Mobile Data Management*. 230–231.

[9] Ioan Damian, Anca Daniela Ionita, and Silvia Oana Anton. 2022. Community- and Data-Driven Services for Multi-Policy Pedestrian Routing. *Sensors* 22, 12 (2022). https://doi.org/10.3390/s22124515

[10] Elham Ghaffari, Amir Masoud Rahmani, Morteza Saberikamarposhti, and Amir Sahafi. 2022. An Optimal Path-Finding Algorithm in Smart Cities by Considering Traffic Congestion and Air Pollution. *IEEE Access* 10 (2022), 55126–55135. https://doi.org/10.1109/ACCESS.2022.3174598

[11] Litao Han, Huan Guo, Haisi Zhang, Qiaoli Kong, Aiguo Zhang, and Cheng Gong. 2020. An Efficient Staged Evacuation Planning Algorithm Applied to Multi-Exit Buildings. *ISPRS International Journal of Geo-Information* 9, 1 (2020). https://doi.org/10.3390/ijgi9010046

[12] Jiun-Jia Hsu and James C. Chu. 2014. Long-term congestion anticipation and aversion in pedestrian simulation using floor field cellular automata. *Transportation Research Part C: Emerging Technologies* 48 (2014), 195–211. https://doi.org/10.1016/j.trc.2014.08.021

[13] Richard L Knoblauch, Martin T Pietrucha, and Marsha Nitzburg. 1996. Field studies of pedestrian walking speed and start-up time. *Transportation Research Record* 1538, 1 (1996), 27–38.

[14] Lucas W. Leiby, Constantinos Costa, and Panos K. Chrysanthis. 2022. Thinking Inclusively with CAPRIO. In *23rd IEEE International Conference on Mobile Data Management, MDM 2022, Paphos, Cyprus, June 6-9, 2022*. IEEE, 378–380. https://doi.org/10.1109/MDM55031.2022.00084

[15] Tiantian Liu, Huan Li, Hua Lu, Muhammad Aamir Cheema, and Lidan Shou. 2021. Towards Crowd-Aware Indoor Path Planning. *Proc. VLDB Endow.* 14, 8 (apr 2021), 1365–1377. https://doi.org/10.14778/3457390.3457401

[16] Marija Nikolić and Michel Bierlaire. 2018. Data-driven spatio-temporal discretization for pedestrian flow characterization. *Transportation Research Part C: Emerging Technologies* 94 (2018), 185–202. https://doi.org/10.1016/j.trc.2017.08.026 ISTTT22.

[17] Brian T. Nixon, Sayantani Bhattacharjee, Benjamin Graybill, Constantinos Costa, Sudhir Pathak, Walter Schneider, and Panos K. Chrysanthis. 2021. HealthDist: A Context, Location and Preference-Aware System for Safe Navigation. In *22nd IEEE International Conference on Mobile Data Management, MDM 2021, Toronto, ON, Canada, June 15-18, 2021*. IEEE, 250–253. https://doi.org/10.1109/MDM52706.2021.00050

[18] OpenStreetMap contributors. 2017. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org.

[19] R. Bohannon, A. W. Andrews. 2011. Normal walking speed: a descriptive meta-analysis. *Physiotherapy* 97, 3 (2011), 182–189.

[20] Vasilis Ethan Sarris, Constantinos Costa, and Panos K. Chrysanthis. 2022. ASTRO-K: Finding Top-k Sufficiently Distinct Indoor-Outdoor Paths. In *23rd IEEE International Conference on Mobile Data Management, MDM 2022, Paphos, Cyprus, June 6-9, 2022*. IEEE, 372–377. https://doi.org/10.1109/MDM55031.2022.00083

[21] Jesse Szwedko, Callen Shaw, Alexander G. Connor, Alexandros Labrinidis, and Panos K. Chrysanthis. 2009. Demonstrating an Evacuation Algorithm with Mobile Devices Using an E-Scavenger Hunt Game. In *Proceedings of the Eighth ACM International Workshop on Data Engineering for Wireless and Mobile Access* (Providence, Rhode Island) *(MobiDE '09)*. Association for Computing Machinery, New York, NY, USA, 49–52. https://doi.org/10.1145/1594139.1594154

[22] Qing Xiong, Qing Zhu, Zhiqiang Du, Xinyan Zhu, Yeting Zhang, Lei Niu, Yun Li, and Yan Zhou. 2017. A Dynamic Indoor Field Model for Emergency Evacuation Simulation. *ISPRS International Journal of Geo-Information* 6, 4 (Mar 2017), 104. https://doi.org/10.3390/ijgi6040104

[23] Zeyu Zhang, Hangxin Liu, Ziyuan Jiao, Yixin Zhu, and Song-Chun Zhu. 2020. Congestion-aware Evacuation Routing using Augmented Reality Devices. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 2798–2804. https://doi.org/10.1109/ICRA40945.2020.9197494