Dynamic Programming-based Pedestrian Hotspot Identification Approach

Aditya Medury (corresponding author)
Safe Transportation Research and Education Center
2614 Dwight Way #7374
Berkeley, CA 94720-7374
Tel: 510-529-8645; E-mail: amedury@berkeley.edu

Offer Grembek
Safe Transportation Research and Education Center
2614 Dwight Way #7374
Berkeley, CA 94720-7374
Tel: 510-642-5553; E-mail: grembek@berkeley.edu

4164 words text + 6 tables/figures x 250 words (each) = 5664 words

November 15, 2014

ABSTRACT
Network screening techniques are widely used by state agencies to identify locations with high collision concentration, also referred to as hotspots. However, most of the research in this regard has focused on identifying highway segments that are of concern to automobile collisions. A major difference between pedestrian and automobile hotspots is that pedestrian-based conflicts are more likely to arise in localized regions, such as near intersections, mid-blocks, and/or other crossings, as opposed to along long stretches of roadway. Hence, in order to address this issue, a dynamic programming-based hotspot identification approach is proposed which provides efficient hotspot definitions for pedestrian crashes. The proposed approach is compared with the sliding window method and the results reveal that the dynamic programming method generates more hotspots with a higher number of crashes, while covering fewer miles.

INTRODUCTION

Pedestrians, along with bicyclists, represent some of the most vulnerable users of the transportation system. In a comparison of all the crashes to have taken place in the state of California between 2005 and 2009, Grembek (*1*) found that pedestrians were 53 times more likely to get injured in a crash with a passenger vehicle, than the travelers in the vehicle itself. The relative vulnerability of pedestrians in traffic conflicts becomes even more worrisome given the recent trends in pedestrian fatalities. Between 2009 and 2012, the relative share, as well as the overall number, of pedestrian fatalities in the United States has consistently increased, even as the total number of all traffic fatalities exhibited a non-monotonic trend in the same time period (*2*). Such a trend is disconcerting as it seems to indicate that the safety concerns of pedestrians are not being adequately addressed.

One of the ways to suggest potential safety improvements to the transportation infrastructure is to identify the locations where pedestrians are most at-risk. However, individual crashes can be spatially dispersed due to natural variation in crash locations. Hence, in order to identify potentially hazardous segments of the road network, it is common practice, especially in state departments for the purpose of on-site investigation, to aggregate individual crashes to identify "hotspots" or high collisions concentration locations. The highway traffic safety literature discusses a wide variety of frameworks to identify hotspots, also referred to as network screening techniques. The objective of network screening, especially in the context of highway segments, is to identify the appropriate start and end points for hotspots along a road segment containing homogeneous traffic and similar built environment attributes.

A popular network screening technique among transportation agencies is the sliding window (SW) approach (*3, 4, 5, 6*), wherein a window of fixed length is moved along the road network in small increments until a pre-specified hotspot selection criteria, such as a critical crash count/rate threshold, is satisfied. The peak searching (PS) method involves dividing entire network into a finite number of windows of fixed length, and testing for the statistical significance of the crash count/rate within these segments by estimating the coefficient of variation (*3*). Chung et al. (*7*) developed a continuous risk profile (CRP) method to empirically determine hotspot segments based on collision data. Herein, a risk profile is generated for the entire road segment by computing a weighted moving average of collisions taking place around each location of the road segment. The resulting smoothened risk profile is compared against a baseline risk threshold, and the regions of the curve lying above this threshold are identified as the hotspots.

However, since the aforementioned techniques were primarily developed for automobile collision-based hotspots, their efficacy vis-a-vis identifying pedestrian hotspots has not been investigated in the literature yet. A major difference between pedestrian and automobile hotspots is that pedestrian-based conflicts are more likely to arise in localized regions, such as near intersections, mid-blocks, and/or other crossings, as opposed to along long stretches of roadway. For instance, if a mid-block or an intersection is an at-risk location for pedestrians, it is unlikely that a moving segment of 0.2 miles can provide a meaningful hotspot specification under such a scenario. As a result, pedestrian hotspots are more likely to be shorter and more dense in nature.

Hence, the objective of this paper is to develop a new network screening approach which can provide denser hotspot definitions for pedestrian crashes. Prior to providing a detailed description of the proposed network screening method, the next section first discusses the sliding window approach, so as to highlight the limitation of this approach for pedestrians, and consequently, better inform the formulation of the proposed framework.

SLIDING WINDOW APPROACH

The first step towards hotspot identification is to select the input parameters which define a hotspot. In the sliding window method, there are two parameters of interest which are required to be defined by the user: the hotspot window length and the minimum number of crashes per hotspot. The hotspot window length is a fixed road segment length which is used to aggregate pedestrian crashes. It is assumed that The hotspot window will have homogeneous pedestrian and road infrastructure elements so that all the crashes covered by that road segment can be expected to have occurred under similar conditions. The minimum number of crashes per hotspot defines a critical threshold which determines whether a road segment should be a hotspot or not. While frequency-based identification of hotspots does not account for the pedestrian volumes at a location, crash rates associated with any given road segment were harder to quantify since pedestrian exposure data was not available for the entire state highway system.

Once the inputs are defined, the sliding window method works as follows: a window of fixed length is moved across the entire road network so as to identify locations which meet the critical crash threshold criterion. In a slightly modified version of the traditional implementation of the sliding window approach, the window can also be moved in a manner that the starting location of a potential hotspot window is always a crash. The rationale for such an implementation is that a pedestrian hotspot should be as dense as possible. Hence any empty space, either at the start or at the end of a hotspot is excessive. Once a hotspot satisfying critical crash threshold is identified, the search for other hotspots continues from the next available crash that does not overlap with any hotspot.

The limitations of the sliding window approach, as described above, are two-fold. Firstly, owing to the fixed window length assumption, even if the hotspot is forced to be begin with a crash, it cannot be guaranteed that the hotspot definition will end with a crash as well. Consequently, there may be an arbitrary length of road segment at the end of the hotspot definition which does not cover any collision. The second limitation of the sliding window method, which has not been well documented in the literature, is that it selects hotspots on a first-come-first-serve basis. In other words, the method selects the first segment along the highway which matches the selection criteria. Such an assumption may impact the selection of more/better hotspots further downstream of that location. For instance, an overlapping hotspot which can be observed after sliding the window slightly further downstream may include more crashes.

In order to address these issues, it is important to relax the fixed window length assumption, and subsequently identify an overarching objective to guide the selection of hotspots. In the subsequent section, a dynamic programming-based algorithm is proposed which relaxes the assumption of a fixed window length, and instead interprets it as the maximum possible window length.

DYNAMIC PROGRAMMING

Dynamic programming (DP) is a decision-making framework used to solve problems involving multiple, sequential sub-problems (*8*). It involves generating and storing the solutions of each intermittent sub-problem before identifying the overall optimal solution. In the case of the sliding window method, the sub-problems are generated when a window of fixed length is moved from one crash to another as a starting point.

In proposing a dynamic programming-based hotspot identification algorithm, it is necessary to define an overarching goal for hotspot selection. In this paper, the objective of the proposed hotspot identification approach is defined to maximize the total number of crashes covered by the hotspots which are eventually chosen based on the minimum number of crashes per hotspot criterion. In addition, the proposed approach also relaxes the fixed window length assumption, and instead imposes a constraint that the size of each hotspot should be no bigger than the maximum window length prescribed by the user. An outcome of relaxing the fixed hotspot length assumption is that a hotspot can begin and end with a crash. In other words, the crashes can now be defined to be as dense as possible.

The dynamic programming approach breaks down the task of optimally identifying the hotspots across the entire road network into smaller sub-problems of identifying whether a crash i (at postmile d_i) should form an end point of a hotspot or not. In terms of the mathematical notation, let the maximum hotspot window length be defined as w, and let the minimum number of crashes per hotspot be n_{min}. Let V_i be defined as the maximum number of crashes that can be covered by hotspots from the start of the road network up to the given crash, i. Finally, let a_i be a variable that defines whether crash i represents an end point of a hotspot or not. If crash i indeed lies at the end of a hotspot, a_i identifies the location of the corresponding crash situated at the beginning of that hotspot, which can be a value between 1 (the first crash) and $i - n_{min} + 1$ (the closest crash location which meets the minimum crash number criterion). On the other hand, if a_i is chosen by the algorithm to not be a hotspot end point, it is attributed a value of zero. Using this notation, the optimization problem can be formulated as follows:

$$V_i = \begin{cases} \max_{1 \le j \le (i-n_{min}+1)}[V_{i-1}, V_{j-1} + (i-j+1)], & \text{if } (d_i - d_j) \le w, \\ V_{i-1}, & \text{otherwise,} \end{cases} \tag{1}$$

$$a_i = \begin{cases} \arg\max_{1 \le j \le (i-n_{min}+1)}[V_{j-1} + (i-j+1)], & \text{if } V_i > V_{i-1}, \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

$$i = 1, 2, \ldots, N.$$

Equation 1 states that if there does not exist any candidate hotspot with crash i as an end point, then V_i is equal to V_{i-1}. In other words, the maximum number of crashes that can be covered by hotspots from the start of the network up to crash i remains the same as the maximum number of crashes covered by hotspots up to crash $i-1$. In comparison, if there exists a candidate hotspot, (j,i), such that $d_i - d_j \le w$, then the total number of crashes covered by the addition of this hotspot is computed, which is equal to the number of crashes contained within the newly defined hotspot, $i - j + 1$, plus the maximum number of crashes that can be covered by hotspots from the start of the network up until crash $j-1$, which is equal to V_{j-1}. If such a combination yields more crashes than V_{i-1}, which represents the maximum number of crashes covered until crash $i-1$, then $a_i = j$, else $a_i = 0$. Finally, in the case that there are multiple hotspot combinations which yield the same objective function, V_{i-1}, then the optimization chooses the smallest hotspot created by the addition of crash i.

It is important to note that the algorithm introduces an element of recursion, wherein the calculation of V_i requires knowing the value of all $V_j, j < i$. Hence, in order to implement the dynamic programming algorithm, V_i is first computed for crash $i = 1$ (the first crash of the network), followed by $i = 2, 3, \ldots, N$ (the last crash of the network). Once all the computations are completed, the value of V_N indicates the maximum number of crashes that can be covered using hotspots. Thereafter, all the hotspots identified by the approach can be recreated by evaluating the values of a_i from $i = N$ to 1.

At this point, it is necessary to explain the reason behind the objective of the proposed query algorithm, which is to maximize the number of crashes covered by the selected hotspots. It is assumed that the inputs chosen by the user satisfy the minimum requirements for a region to be classified as a hotspot. Hence, it is important that chosen objective ensures that the algorithm can produce as many non-overlapping hotspots as possible that satisfy the selection criteria. However, a performance measure based on identifying the most number of hotspots cannot distinguish between the variable numbers of crashes per hotspot. Consequently, a more objective metric, which is to maximize the total number of crashes covered by the chosen hotspots, is chosen.

COMPARISON BETWEEN SLIDING WINDOW AND DYNAMIC PROGRAMMING USING TASAS CRASH DATABASE

The crash data for this study consists of non-fatal crashes involving pedestrians during the years 2005-2010, extracted from California Department of Transportation's Traffic Accident Surveillance and Analysis System (TASAS). Since the data corresponding to pedestrian and/or traffic exposure were not available for the study, the state highway system was divided into homogeneous segments on the basis of the route number, county, route suffix and prefix. For each of those segment, both sliding window and dynamic programming were used to identify the hotposts.

In order to illustrate the difference between the two approaches, different combinations of threshold levels and the hotspot window lengths were selected. Unlike the hotspot analysis for automobile collisions, where the window length is typically chosen to be 0.2 miles, this study compares the performance of sliding window and dynamic programming using $w = 0.025, 0.05$ and 0.1 miles. For each given value of w, the value of the minimum crash criterion is varied, starting from $n_{min} = 2$, until both approaches produce identical results.

Tables 1a and 1b show the total number of crashes and hotspots identified by sliding window and dynamic programming approaches, respectively. The results indicate that across all window lengths, for small values of n_{min}, the dynamic programming framework initially yields more hotspots and crashes than the sliding window approach. However, as the threshold is increased, the difference in the two approaches decreases. It can also be inferred that for a given a window length, increasing the minimum crash threshold decreases the total number of crashes covered by hotspots. Similarly, for a given crash threshold, it is expected that increasing the window length provides greater coverage to include more crashes within the hotspot.

In order to further quantify the differences between the two approaches, table 2 provides some additional statistics with regard to the average number of crashes per hotspot, average length of the hotspots as well as the total number of miles covered by the hotspots. In table 2 (a), the values in the parenthesis indicate the average hotspot length and the total number of miles covered by sliding window hotspots when excluding any extra unutilized space at the end of a hotspot. The extra space is defined as the region between the end of the hotspot and the last crash within the hotspot.

Table 2 shows that in spite of identifying a greater number of hotspots, the total miles covered by the hotspots of dynamic programming are fewer in number than the hotspots of sliding window. The same relationship largely holds with the trimmed hotspot lengths of the sliding window method, but it can be observed the difference greatly diminishes as the threshold is increased. In addition, by comparing the lengths of the trimmed sliding window hotspots (in parenthesis) with the fixed hotspot lengths, it can be observed that a large fraction of the fixed hotspot length goes unutilized in the sliding window approach. Hence, it can be argued that the dynamic program-

TABLE 1 Sliding window and dynamic programming results

(a) Sliding Window

Minimum number of crashes per hotspot, n_{min}	Fixed hotspot window length, w					
	0.025 miles		0.05 miles		0.1 miles	
	#crashes	#hotspots	#crashes	#hotspots	#crashes	#hotspots
2	2369	936	2873	1110	3522	1272
3	1122	309	1416	377	2040	523
4	592	130	797	168	1156	225
5	277	50	421	73	785	129
6	132	20	202	29	476	66
7	73	10	112	14	351	44
8	-	-	-	-	200	22
9	-	-	-	-	120	12
10	-	-	-	-	68	6

(b) Dynamic Programming

Minimum number of crashes per hotspot, n_{min}	Maximum hotspot window length, w					
	0.025 miles		0.05 miles		0.1 miles	
	#crashes	#hotspots	#crashes	#hotspots	#crashes	#hotspots
2	2383	1088	2912	1329	3569	1634
3	1137	332	1439	418	2080	625
4	604	135	801	176	1175	254
5	281	50	428	77	812	143
6	133	20	205	30	494	71
7	73	10	112	14	359	46
8	-	-	-	-	202	22
9	-	-	-	-	122	12
10	-	-	-	-	68	6

TABLE 2 Additional summary statistics

(a) Sliding Window

Minimum number of crashes per hotspot, n_{min}	Fixed hotspot window length, w								
	0.025 miles			0.05 miles			0.1 miles		
	Average number of crashes per hotspot	Average hotspot length (in miles)	Total miles covered by hotspots	Average number of crashes per hotspot	Average hotspot length (in miles)	Total miles covered by hotspots	Average number of crashes per hotspot	Average hotspot length (in miles)	Total miles covered by hotspots
2	2.53	0.025 (0.0093)	23.4 (8.713)	2.58	0.05 (0.018)	55.5 (20.54)	2.77	0.1 (0.044)	127.2 (55.61)
3	3.63	0.025 (0.0123)	7.725 (3.811)	3.76	0.05 (0.023)	18.9 (8.706)	3.9	0.1 (0.055)	52.3 (28.97)
4	4.55	0.025 (0.0142)	3.25 (1.845)	4.74	0.05 (0.026)	8.40 (4.392)	5.14	0.1 (0.061)	22.5 (13.69)
5	5.54	0.025 (0.0152)	1.25 (0.759)	5.77	0.05 (0.027)	3.65 (1.99)	6.09	0.1 (0.069)	12.9 (8.87)
6	6.6	0.025 (0.0166)	0.5 (0.333)	6.97	0.05 (0.030)	6.60 (0.881)	7.21	0.1 (0.076)	6.6 (5.05)
7	7.3	0.025 (0.0175)	0.25 (0.175)	8	0.05 (0.031)	4.4 (0.43)	7.98	0.1 (0.080)	4.4 (3.51)
8	-	-	-	-	-	-	9.09	0.1 (0.086)	2.2 (1.88)
9	-	-	-	-	-	-	10	0.1 (0.085)	1.2 (1.02)
10	-	-	-	-	-	-	11.33	0.1 (0.082)	0.6 (0.49)

(b) Dynamic Programming

Minimum number of crashes per hotspot, n_{min}	Maximum hotspot window length, w								
	0.025 miles			0.05 miles			0.1 miles		
	Average number of crashes per hotspot	Average hotspot length (in miles)	Total miles covered by hotspots	Average number of crashes per hotspot	Average hotspot length (in miles)	Total miles covered by hotspots	Average number of crashes per hotspot	Average hotspot length (in miles)	Total miles covered by hotspots
2	2.19	0.0077	8.396	2.19	0.015	20.52	2.19	0.037	54.98
3	3.42	0.0117	3.906	3.44	0.022	9.09	3.32	0.049	30.33
4	4.47	0.0139	1.873	4.55	0.025	4.34	4.63	0.055	14.01
5	5.62	0.0154	0.769	5.56	0.026	2.04	5.68	0.067	9.65
6	6.65	0.0166	0.333	6.83	0.031	0.93	6.96	0.074	5.25
7	7.3	0.0175	0.175	8	0.031	0.43	7.8	0.078	3.61
8	-	-	-	-	-	-	9.18	0.086	1.88
9	-	-	-	-	-	-	10.17	0.085	1.02
10	-	-	-	-	-	-	11.33	0.082	0.491

ming approach provides more efficient hotspot definitions than the sliding window method while identifying more crashes with a smaller footprint.

On the other hand, a comparison of the average number of crashes per hotspot indicates that the sliding window method yields a higher number of crashes per hotspot. This is to be expected since the dynamic programming method has a flexible window length. However, the difference is further exaggerated by the fact that the dynamic programming gives preference to smaller hotspots if two hotspot combinations yield the same crash coverage.

CASE STUDIES

To better illustrate the dynamics of the two network screening approaches, two illustrative case studies from the TASAS results are selected. Firstly, a short segment from US 101, as shown in figure 1, is selected. It contains 18 pedestrian crashes as indicated in the figure in the form of black dots.

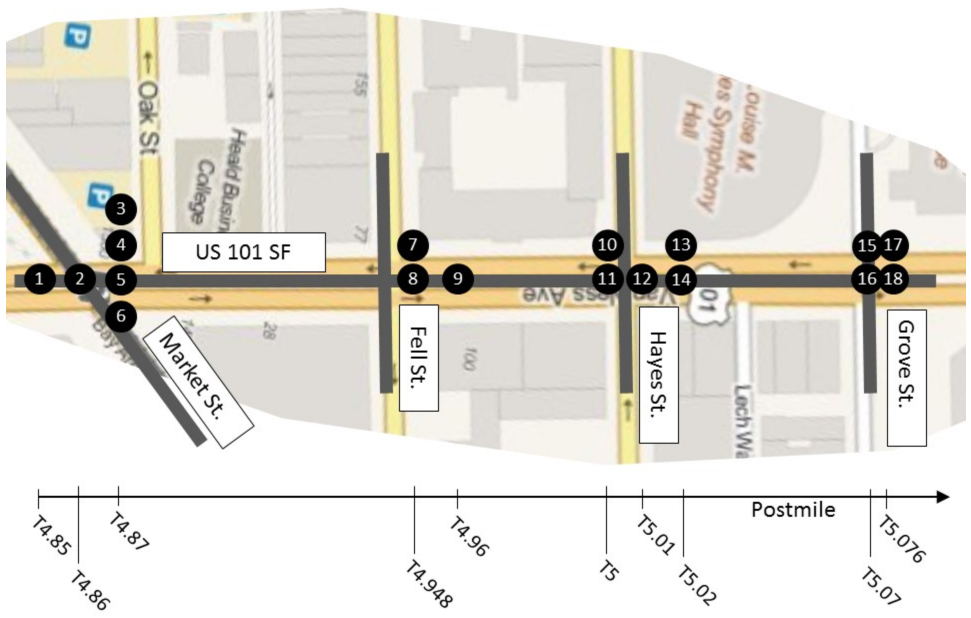
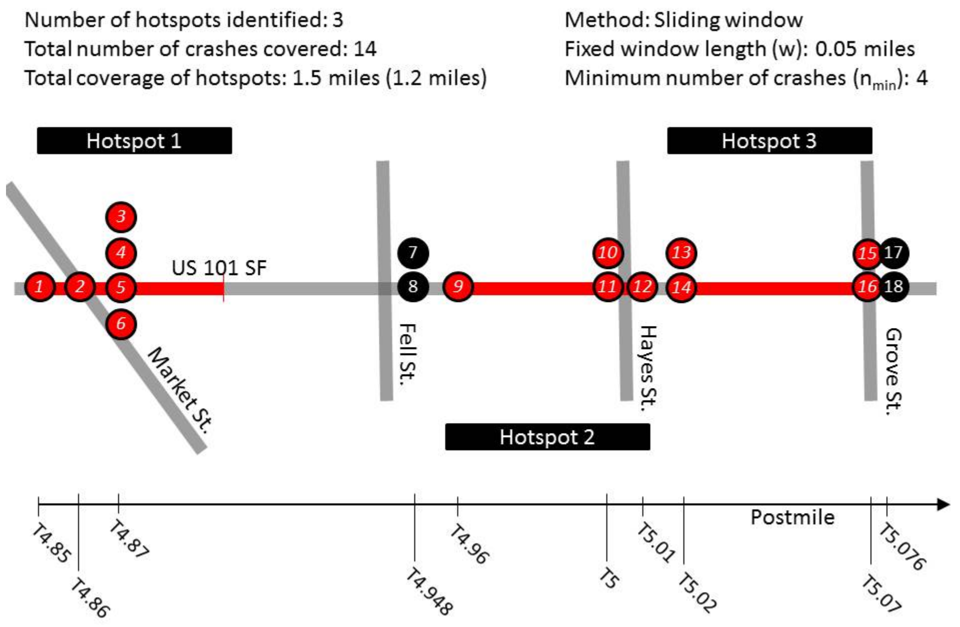
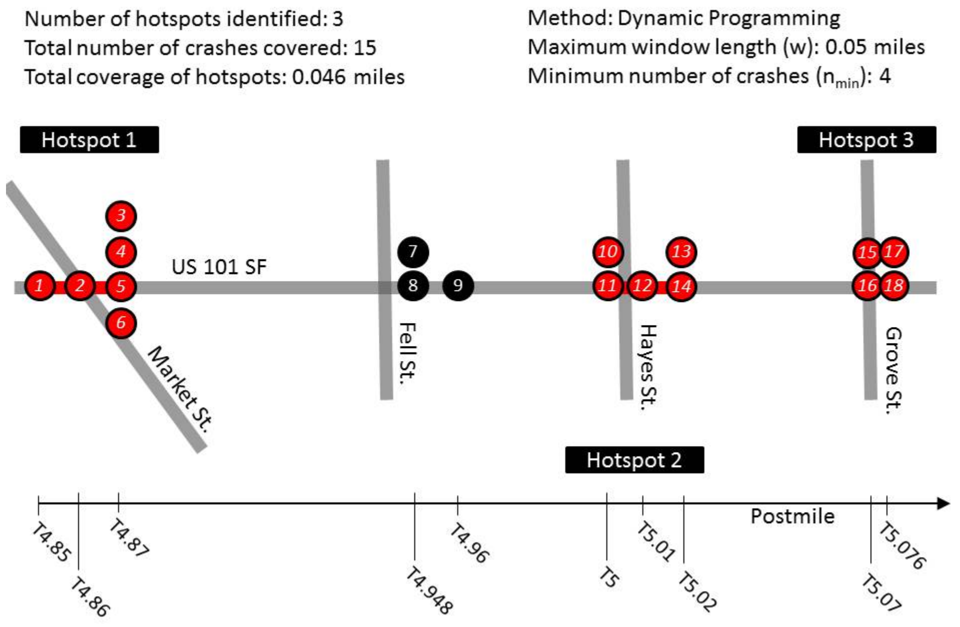


FIGURE 1 Section of 101SF involving eighteen pedestrian crashes

Figure 1 indicates that a hotspot length of 0.2 miles, and even 0.1 miles, can be too long for pedestrian collisions, as they can cover multiple intersections. Consequently, consider the hotspot identification scenario, $w = 0.05$ miles and $n_{min} = 4$. The results of implementing the two hotspot identification techniques on the road segment illustrated in figure 1 are shown in figure 2. Figure 2(a) shows the three hotspots identified using the sliding window approach as [T4.85,T4.90], [T4.96,T5.01] and [T5.02,T5.07]. The underlying process of the sliding window method can be understood as follows: the sliding window is first placed on crash 1 as the starting point. The resulting 0.05 mile segment satisfies the minimum crash criterion and is thereby selected as a hotspot. Subsequently, the window is moved to crashes 7 and 8 as starting points, but the resulting 0.05 mile segment can only accommodate crashes 7, 8 and 9. As a result, the window moves to crash


Number of hotspots identified: 3
Total number of crashes covered: 14
Total coverage of hotspots: 1.5 miles (1.2 miles)
Method: Sliding window
Fixed window length (w): 0.05 miles
Minimum number of crashes (n_{min}): 4
Hotspot 1
Hotspot 2
Hotspot 3
US 101 SF
Market St.
Fell St.
Hayes St.
Grove St.
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
Postmile
T4.85
T4.86
T4.87
T4.948
T4.96
T5
T5.01
T5.02
T5.07
T5.076

(a) Sliding Window


Number of hotspots identified: 3
Total number of crashes covered: 15
Total coverage of hotspots: 0.046 miles
Method: Dynamic Programming
Maximum window length (w): 0.05 miles
Minimum number of crashes (n_{min}): 4
Hotspot 1
Hotspot 2
Hotspot 3
US 101 SF
Market St.
Fell St.
Hayes St.
Grove St.
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
Postmile
T4.85
T4.86
T4.87
T4.948
T4.96
T5
T5.01
T5.02
T5.07
T5.076

(b) Dynamic Programming

FIGURE 2 Comparison between dynamic programming and the sliding window method for a case study involving a section from U.S. Route 101

9, which yields four crashes, and is selected as a hotspot. After the second hotspot is identified, the window is moved to crash 13 as a starting point, which yields the third hotspot. The remaining crashes 17 and 18 do not satisfy the minimum crash criterion.

In comparison, figure 2(b) shows the hotspots identified by the dynamic programming approach. Herein, it can be seen that even though both approaches identified three hotspots, the dynamic programming approach covered more crashes with a smaller hotspot coverage length. The primary contributing factors for the difference between the two results is the sliding window's first-come-first-serve assumption. In this case, since the sliding window approach first came across the hotspot [T4.96,T5.01], it overlooks a more significant clustering of crashes around Hayes St., as it partially overlapped with the already chosen hotspot.

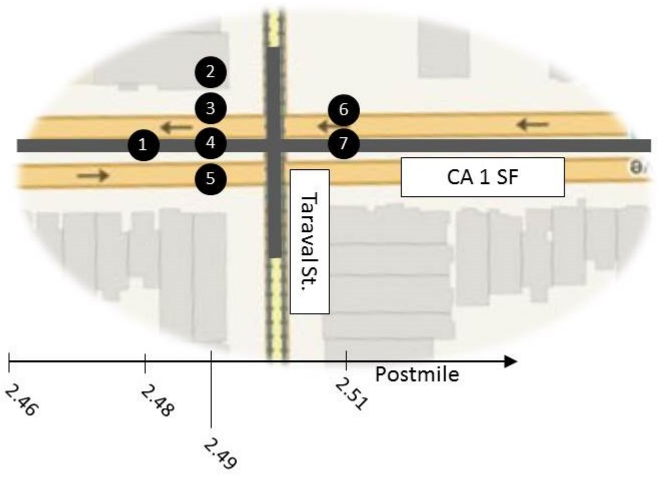
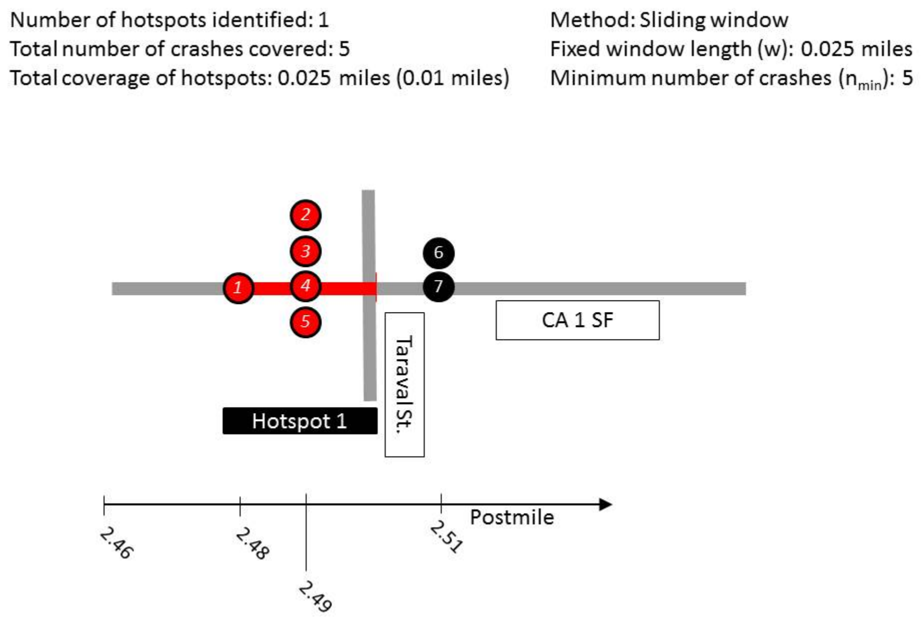


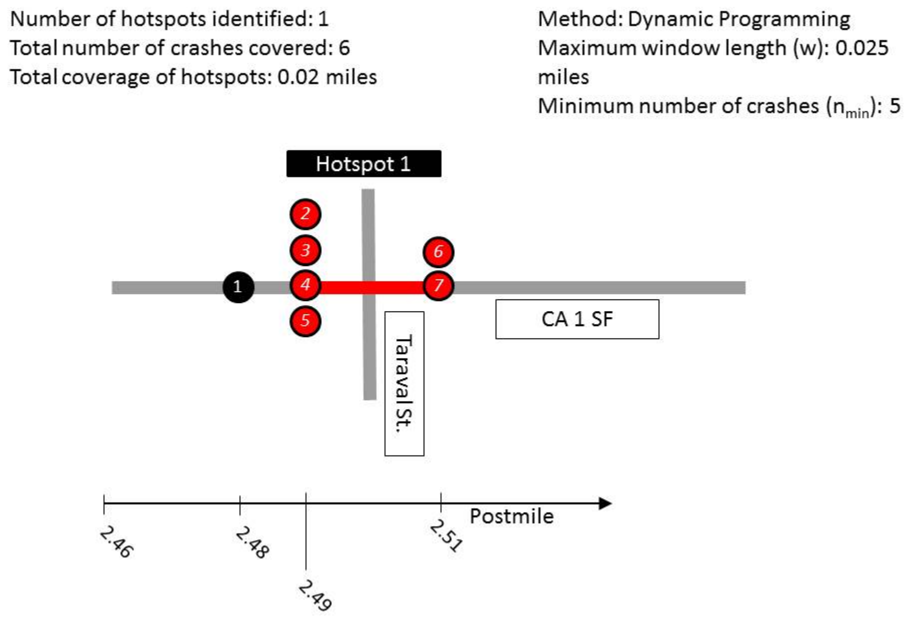
FIGURE 3 Section of California State Route 1 involving seven pedestrian crashes

Figures 3 and 4 provide another illustration of the limitations of the sliding window's first-come-first-serve hotspot selection approach. In figure 3, a short segment of California State Route 1 is selected from the results corresponding to $w = 0.025$ miles and $n_{min} = 5$. Herein, figure 4(a) shows that the sliding window method selects the segment between 2.48 and 2.49 miles as the hotspot. However, as figure 4(b) reveals, there are a greater number of crashes present between 2.49 and 2.51 miles which is not recognized by the sliding window approach.

In the two case studies described above, a useful by-product of the dynamic programming approach was a hotspot definition which was better aligned with the underlying intersections. While it is also possible that the dynamic programming method identifies some hotspots which do not overlap with the underlying intersections/mid-blocks, such instances are more likely to be a consequence of the chosen objective function, which is to identify hotspots with the maximum crash coverage. Hence, it can be argued that under circumstances which involve multiple overlapping hotspot definitions, the dynamic programming method is more likely to provide hotspots that cover more crashes and are also spatially more meaningful.


Number of hotspots identified: 1
Total number of crashes covered: 5
Total coverage of hotspots: 0.025 miles (0.01 miles)
Method: Sliding window
Fixed window length (w): 0.025 miles
Minimum number of crashes (n_{min}): 5
1
2
3
4
5
6
7
CA 1 SF
Taraval St.
Hotspot 1
Postmile
2.46
2.48
2.49
2.51

(a) Sliding Window


Number of hotspots identified: 1
Total number of crashes covered: 6
Total coverage of hotspots: 0.02 miles
Method: Dynamic Programming
Maximum window length (w): 0.025 miles
Minimum number of crashes (n_{min}): 5
Hotspot 1
1
2
3
4
5
6
7
CA 1 SF
Taraval St.
Postmile
2.46
2.48
2.49
2.51

(b) Dynamic Programming

FIGURE 4 Comparison between dynamic programming and the sliding window method for a case study involving a section from California State Route 1

CONCLUSIONS

Based on the results of the case studies and the overall TASAS pedestrian crashes comparison, it can be argued that dynamic programming is a more efficient hotspot identification algorithm than the sliding window approach. It provides shorter hotspot lengths with greater crash coverage, and is also more likely to allow hotspots to be identified in meaningful locations. In comparison, the sliding window approach is shown to be inflexible due to its first-come-first-serve hotspot selection approach and the fixed window length assumption.

An important benefit of having dense hotspots is that it allows the field engineers to focus only on regions that are relevant to the hotspot. In fact, the TASAS pedestrian crashes comparison reveals that, across all maximum window lengths and minimum crash threshold combinations, even though the dynamic programming approach produced more hotspots than the sliding window method, the total number of miles covered by the hotspots were fewer in the case of dynamic programming. This implies that the sliding window method provides inefficient hotspot definitions, as illustrated by the comparison with the trimmed hotspot lengths in table 2.

In terms of future work, the dynamic programming approach can be extended to include multiple constraints and/or other objective functions. For instance, constraints pertaining to the crash density within each hotspot can be introduced so as to differentiate between a hotspot with all crashes occurring at a single location and a hotspot with all the crashes at different locations. Since the DP approach provides a general hotspot identification framework, its efficacy can also be investigated in the context of automobile collision-based hotspots. Finally, while this study indicates that the dynamic programming framework improves upon the sliding window method, it is important to compare it with other hotspot techniques introduced in the literature, especially the continuous risk profile method.

ACKNOWLEDGEMENT

The authors would like to thank the California Department of Transportation for funding this research.

REFERENCES

[1] Grembek, O., *The relative vulnerability index: a framework for evaluating multimodal traffic safety*, 2012, URL: http://goo.gl/sccUf1.

[2] National Highway Traffic Safety Administration, *Traffic Safety Facts (Pedestrian): 2012 Data*, 2012, URL: http://goo.gl/DO5wqS.

[3] National Research Council (US). Transportation Research Board. Task Force on Development of the Highway Safety Manual and Transportation Officials. Joint Task Force on the Highway Safety Manual, *Highway Safety Manual*, Vol. 2. AASHTO, 2010.

[4] Harwood, D., D. Torbic, B. Richard K.R., and M. M. Meyer, *SafetyAnalyst: Software Tools for Safety Management of Specific Highway Sites*, 2010, URL: http://goo.gl/SoLiQZ.

[5] Grembek, O., K. Kim, O. H. Kwon, J. Lee, H. Liu, M. J. Park, S. Washington, D. Ragland, and S. M. Madanat, *Experimental Evaluation of the Continuous Risk Profile (CRP) Approach to the Current Caltrans Methodology for High Collision Concentration Location Identification*. California Department of Transportation Technical Report No. CA12-2192, 2012.

[6] Oregon Department of Transportation, *Safety Priority Index System*, 2014, URL: http://goo.gl/HHczIi.

[7] Chung, K., D. R. Ragland, S. M. Madanat, and S. M. Oh, The Continuous Risk Profile Approach for the Identification of High Collision Concentration Locations on Congested Highways. *Transportation and Traffic Theory 2009: Golden Jubilee: Papers selected for presentation at ISTTT18*, 2009.

[8] Bertsekas, D. P., *Dynamic programming and optimal control*, Vol. 1. Athena Scientific Belmont, MA, 1995.