Integrating LPR with CCTV systems: problems and solutions*

David Bissessar and Dmitry O. Gorodnichy

Video Surveillance and Biometrics Section
Science and Engineering Directorate, Canada Border Services Agency
14 Colonnade Rd, Ottawa, ON, Canada, K2E 7M6
{ David.Bissessar / Dmitry.Gorodnichy} @cbsa-asfc.gc.ca

ABSTRACT

A new generation of high-resolution surveillance cameras makes it possible to apply video processing and recognition techniques on live video feeds for the purpose of automatically detecting and identifying objects and events of interest. This paper addresses a particular application of detecting and identifying vehicles passing through a checkpoint. This application is of interest to border services agencies and is also related to many other applications. With many commercial automated License Plate Recognition (LPR) systems available on the market, some of which are available as a plug-in for surveillance systems, this application still poses many unresolved technological challenges, the main two of which are: i) multiple and often noisy license plate readings generated for the same vehicle, and ii) failure to detect a vehicle or license plate altogether when the license plate is occluded or not visible. This paper presents a solution to both of these problems. A data fusion technique based on the Levenshtein distance is used to resolve the first problem. An integration of a commercial LPR system with the in-house built Video Analytic Platform is used to solve the latter. The developed solution has been tested in field environments and has been shown to yield a substantial improvement over standard off-the-shelf LPR systems.
Keywords: Video Surveillance, Video Analytics, CCTV, LPR, Performance Evaluation

Disclaimer:

Specific hardware and software products which may be identified in this paper were used in order to perform the evaluations described in this document. In no way does identification of any commercial product, trade name, or vendor, imply recommendation or endorsement by the Canada Border Services Agency, nor does it imply that the products and equipment identified are necessarily the best available for the identified purpose

1. INTRODUCTION

As a result of the increasingly growing demand for security, many countries have been deploying Video Surveillance Systems (frequently referred to as CCTV systems), as an important tool for enhancing preventive measures and aiding post-incident investigations. Within the Canadian government, many federal departments heavily use CCTV systems including the Canada Border Services Agency (CBSA) who sees video surveillance as a key technological element in protecting the country's borders [1-3].

A new generation of high-resolution IP surveillance cameras makes it now possible to apply video processing and recognition techniques to live video feeds for the purpose of automatically detecting and identifying objects and events of interest. In particular, they allow one to apply state-of-art License Plate Recognition (LPR) and Optical Character Recognition (OCR) software to the captured video surveillance images for the purpose of identifying vehicles observed passing through a checkpoint. With many commercially available automated LPR/OCR systems this application, which is of particular interest to border services agencies, still poses many unresolved technological challenges.

Crown Copyright 2011. Permission is granted to quote short excerpts and to reproduce figures and tables from this report, provided that the source of such material is fully acknowledged.
Citation: David Bissessar and Dmitry O. Gorodnichy. "Integrating LPR with CCTV systems: problems and solutions", SPIE Defence, Security & Sensing Conference, Automatic Target Recognition XXI Track, Orlando, April 25-29, 2011.

First, commercially successful LPR systems, such as those used at toll roads, are manufactured as a complete system with many built-in features such active IR illumination, special lenses and/or camera sensors, dedicated hardware, and in certain cases, additional triggers/activators to produce reliable license plate readings. Such systems however cannot be easily integrated or embedded into a CCTV video surveillance system.

Second, commercial LPR systems, including the best performing complete systems are normally deployed in "close to ideal" conditions, in which a camera is positioned so that it views the area of interest only (for example, a camera that is mounted over a traffic lane with the direction of view parallel to the lane). When operated without additional triggers/activators and under a non-zero angle of attack, the performance of commercial systems is far from perfect: there will be multiple readings obtained for a single vehicle, many of which may be noisy. Furthermore, the readings may not be even grouped by vehicles they belong to.

Third, performance evaluation of commercial LPR systems is a challenging issue in itself. Statistically significant results can only be obtained with large-scale tests or pilots conducted over a long period of time containing at least 24 continuous hours. At the same time, commercial systems can only be evaluated based on the data they capture. It will remain unknown what has NOT been captured and why, unless there is another mechanism (either manual or automatic) to know the ground truth. Without evaluation that provides the rate of both Hits and Misses of LPR, the LPR technology can not be optimally selected, nor can it be tuned.

Finally, there is the case where a license plate on a vehicle is intentionally occluded or removed. With commercial stand-alone LPR systems, such vehicles cannot be detected or will not be reported. However, it would be exactly these vehicles that would be most important to capture for enforcement and intelligence purposes.

A solution to these problems is seen in combining Video Analytics, which is the S&T area that deals with automated analysis and extraction of content from video, with traditional LPR and Optical Character Recognition (OCR) techniques, which can be applied to a raw video feed using commercial and public domain software.

In sub-optimal conditions, commercial products tend to provide a stream of License Plate Numbers (LPNs) containing various levels of noise. As shown in Figure 1, the two main types of noise are (1) multiple LPNs being generated for the same vehicle and (2) characters within LPNs being incorrectly interpreted. Adding to this difficulty is the fact that many LPR plug-in systems do not differentiate between different vehicles. Thus, further processing of LPR system outputs is desired in order to obtain a single and correct LPN for every vehicle that goes by.

The key problem of fusing noisy character strings detected by LPR/OCR software is solved by applying a post-processing filter that groups all detected LPNs into collections of similar character strings corresponding to passing vehicles referred to as "Drive-by Events". The criterion of similarity is based on the modified Levenshtein distance. The ground truth and the additional information about all passing vehicles, including those that do not exhibit detectable license plates, is detected using the Video Analytic Platform (VAP), which is the in-house built software that uses advance video recognition algorithms capable of robustly detecting moving vehicles in outdoor environments [2].

The paper discusses the two components of an integrated solution and presents experimental results. Section 2 presents the LPR fusion problem and describes the solutions developed to resolve. Section 3 describes VAP software and describes its application to LPR evaluation and detection of vehicles not bearing the license plate. Experimental results are shown in Section 4.

Figure 1. Incorrect license plate numbers obtained with commercial LPR systems: a) by dedicated hardware-based LPR system, b) by software-based CCTV plug-in LPR system.

2. FUSING NOISY LICENSE PLATE DATA TO EXTRACT GROUND TRUTH FOR PASSING VEHICLES

2.1.1 The source of noise

As noted above, commercial LPR systems show performance degradation when deployed in sub-optimal environmental conditions, exhibiting increase in the number of missed vehicles and/or multiple and noisy LPN readings. The noise can be due to ambiguous characters, plate occlusions, blurry frames, and limitation of the OCR algorithm.

Even the systems that use active IR illumination and dedicated hardware and triggers when not positioned directly above the vehicle do not show perfect performance, as seen in Figure 1.a. The performance of software-based LPR systems, which are sold as a plug-in for CCTV systems, is even much worse, since they do not commonly differentiate between different vehicles. Such systems produce a continuous stream of LPNs and associated time-stamps as shown in Figure 2.a, and contain no information about the vehicles the plates belong to.

In certain circumstances, it may be acceptable for human operator to use judgement in discerning which plates belong to the same vehicle. For a fully-automated system however, the noisy and incomplete LPR result is not acceptable.

Automatic LPN fusion offers a way to improve the LPR result, making it feasible to apply LPR within an existing CCTV system. Given a continuous stream of LPN snapshots from an LPR system as an input, the objective of the automatic LPN fusion is to refine this stream so that to produce an output stream which best approximates the ground truth of the observed vehicle traffic. LPN fusion seeks to transform the noisy stream of LPN readings into a stream consisting of a single LPN per vehicle, as shown in Figure 2.b.

Figure 2. Stream of LPN readings (a) , and the result of LPR data fusion (b,c).

This section presents two LPN fusion algorithms. Given a continuous stream of license plate numbers, many of which are different but may come from the same vehicle, the algorithms compute a single highest confidence license plate number per vehicle.

The first algorithm is the basic fusion algorithm that applies the Levenshtein Distance to group similar reads into a single label. The second algorithm builds on the basic algorithm and improves several of the limitations of the basic algorithm. Both algorithms are based on a common foundation described below. Table 1 summarizes the notations used.

Table 1. Notations

1) Snapshot s is a tuple, s=<t, r>, where:
   - t is the timestamp of snapshot s, denoted t(s);
   - r is the LPN of snapshot s, denoted r(s)
2) An infinite stream of snapshots produced by an LPR system
   S = {s_1, s_2,… }, where $\mathbf{s_i}$, are ordered chronologically
3) A set of Drive-by Events, Q = {b_1, b_2,…}, where:
   - the length of Q is q
   - Drive-by Events, b_i = {l_i, {$s_{i,1}$, $s_{i,2}$, …$s_{i,n}$}}
   - l is the best-guess LPN, denoted as l(b);
   - $s_{i,j}$ represents snapshot j in drive-by event i
   - $t(s_{i,j}) < t(s_{i,k})$ for all j < k
4) LD(s_1, s_2) is the Levenshtein Distance between s_1, s_2
5) MLD(s_1, s_2), the Modified Levenshtein Distance, between s_1, s_2
6) T_{ML}(s_1, s_2) is the applicable threshold for s_1, s_2.
7) T = {(l_1,t_2), (1_2,t_2), … (l_n,t_n)} is the threshold mapping table between a length l, and a threshold t

2.1.2 Modified Levenshtein Distance

In processing S subsequent snapshots, the similarity of consecutive snapshots s_i and s_j is determined using the modified Levenshtein distance MLD (s_i, s_j). The MLD is based on the traditional Levenshtein Distance [6] defined as the number of edits (adding, removing or changing a character from either string) required to make the strings identical.

The Levenshtein Distance metric has been used in a number of applications in pattern matching and artificial intelligence including, identification of viral sequences in DNA chains [7], examination of perceptual and acoustic differences between dialects in human languages [8], and derivation of robotic control specifications in neuro-evolutionary techniques [9].

The MLD used adds two pre-processing steps to the traditional LD to make it suitable for LPN comparison.

Pre-processing step 1: ***Preparation for prefix-suffix substrings***

Two strings are passed into the algorithm: s_1 and s_2. In the first pre-processing step, a new string s' is constructed by padding the longer of the two strings, say s_1, with l * characters on the left and right, where l is defined as the length of the smaller string s_2 minus 1, and * represents some designated character which is outside the valid alphabet for license plates.

Let $len(s_2) <= len(s_1)$, then: s' is constructed as follows:

$l = len(s_2) - 1$
*s'= s_1 padded on the left with l *characters*
*s'= s' padded on the right with l *characters*

Constructing s' in this manner allows the maximum number of overlapping prefixes and suffixes to be considered. This addresses the problem of reconciling the (erroneous) partial LPN readings which can occur for a number of reasons including vehicle motion or large angles of attack.

Pre-processing step 2: ***Iteration over substrings***

The traditional LD returns an edit distance if two substrings are not of the same length. Our algorithm requires a substring to be treated more directly to the string which contains is. In order to do this we look for the minimum distance within the longer string of all substrings of the length of the shorter string. This is done by invoking the LD on each substring in s' of the length s_2, and returning the minimum distance.

Given s' as defined in preprocessing step 1,

Let S = (set of substrings of length s2 in s') ,

then MLD(s1, s2) = $min_{(s\ in\ S)}$(LD(s,s2)

It should be noted that a property of the MLD is that if s_2 is a proper substring of s_1, then $MLD(s_1, s_2) = 0$.

The following example illustrates the concepts.

2.1.3 Example

If s_1 = ABCD123 and s_2 = 123D, $MLD(s_1, s_2)$ proceeds as follows:

1) Applying pre-processing step 1:
    a. l = len(123D) – 1
    b. s is padded to the left and right with l * characters
    c. s' is set to ***ABCD123***,
2) Applying pre-processing step 2 we iterate using the traditional ML algorithm for substrings in s':
    a. for each substring length 4 in ***ABCD123***
        i. Calculate MLD of substring and s_2
        ii. Save current minimum MLD
    b. return minimum MLD

This example returns MLD = 1 since the s_2 has a minimum LD of 1 when compared to the substring "123*" in s'.

2.1.4 Length-sensitive thresholds

The algorithms compare the MLD (s_1, s_2) to an applicable threshold, $T_{ML}(s_1, s_2)$ to determine whether or not they are related. The threshold is determined based on the length of the given LPN string so that a lower threshold is used for shorter strings.

Our algorithm examines the length of the two LPNs, using the maximum length to retrieve the applicable threshold from mapping table.

Given $T = \{(l_1,t_2), (1_2,t_2), \ldots (l_n,t_n)\}$,

Find $T_{ML}(s_1, s_2) = T[\max(len(s_1), len(s_2))]$

The algorithms presented here are not dependent on the method used to determine thresholds. Alternate threshold approaches are possible including the use of a constant value or a percentage of string length. A mapping table is used here for flexibility.

Table 2. Pseudo-code for Algorithm 1.

```
Set b to an new  drive-by event
while  true
        Set s to be the next LPN reading or null if none.
        while   s is null
                if inactivity threshold exceeded
                                        and b contains LPNs
                    finalize b
                    set b to new drive-by event
        if  fits ( s, b) is false
                finalize b
                set b to new drive-by event
        insert s into b
```

2.2 Algorithm 1: Single Event Fusion

Algorithm 1 inspects the stream S of incoming LPN snapshots s_i, and gathers consecutive similar ones into a buffer b which represents the current drive-by event. When an LPN string exceeding the similarity threshold is encountered, the current drive-by event is finalized and a new one is started.

A drive-by event is defined as a collection of LPNs that are determined to belong to the same vehicle. A *drive-by event* is represented as $b_i = < l_i, S_i >$, where the label l_i is the current best estimate of the license plate, and S_i is the collects all LPNs determined to belong to the current vehicle.

Finalizing a drive-by event b takes any action needed to record the vehicle's passage. This can include updating a system counter, or writing a file to disk for a downstream system.

2.2.1 Criteria for grouping LPN into a single event

Given this definition, we can define the MLD of an LPN s_i and a drive-by event b, and also whether s_i matches b as follows:

$$d=MLD(s_i, b) = MLD(s_i, l(b)) \quad (1)$$
$$fit(s_i, b) \text{ if } MLD(s_i, b) <= T_{ML}(s_i, l(b))$$

2.2.2 Algorithm Description

The basic algorithm continually processes the input stream item by item, calculating the MLD of the current s_i, and l(b) and comparing to applicable threshold t. If the d is below t, the LPN is added to b. If d exceeds t, s_i is deemed to belong to a different vehicle. The current b is finalized, a new drive-by event b' is created, and s_i is added to b'.

The algorithm monitors an inactivity threshold. If no LPN has been detected for a time period exceeding than the specified inactivity threshold, the current drive-by event is assumed finished, and is finalized.

The pseudo-code for this basic algorithm is shown in Table 2. The basic algorithm has two main objectives:

(1) Gather the sequential LPN readings which represent a drive-by event, and,
(2) Determine best estimate for the *actual* plate, given the set of LPNs collected in step (1)

Objective (1) is achieved using the Modified Levenshtein Distance. Objective (2) is achieved using a length-frequency heuristic, described below.

2.2.3 Labelling with Best Estimate LPN

For a drive-by event $b_i = < l_i, < s_{i,1}, s_{i,2}, ..., s_{i,n} >>$, the label l_i, represents the current *best estimate* of the actual plate given all LPNs collected so far. This label is updated when a new snapshot is inserted into b_i. The label is chosen to be the LPN, which satisfies the following conditions:

1) It's length is the closest to the expected number of characters in a license plate number, or,
2) If multiple LPNs have the same desired length, the most frequently occurring of these is selected

Depending on the application, the best-estimate LPN can be computed in a number of ways, including a frequency-based approach and robust statistics, as a median, a mean or character voting.

2.2.4 Limitations of the Algorithm 1

The basic algorithm described above has several limitations, described below.

Intervening unrelated strings

The first limitation occurs when an unrelated string is present amongst the license plate readings belonging to the same vehicle. This may occur, for example, if the upstream LPR system erroneously reads text on the vehicle's rear panel and interprets it as a license plate number. In this case, S will contain a sequence of LPN readings interrupted with the erroneously taken text. For example S might be: {"ABCD123", "BCD123", "TAXI", "ABCD123"}.

In this situation, the basic algorithm will generate 3 different vehicle drive-by events, as the erroneous text is likely to have MLD exceeding the threshold. A more sophisticated algorithm would filter out the erroneous text and place the related LPNs into the same drive-by event, thus reporting this vehicle only once.

Intermingled LPNs

The second limitation occurs when LPNs from multiple vehicles are intermingled in S. The basic algorithm makes the implicit assumption that LPN readings come in consecutively, one vehicle at a time. This may not always be the case.

When monitoring faster traffic, it is possible to use two cameras to increase roadway coverage area, which in turn increases the probability capturing one snapshot per vehicle. In this scenario, LPN readings from different vehicles may be interleaved when data from both cameras are serialized chronologically.

Two vehicles "ABCD123" and "PQRS456", captured by two cameras can produce an input stream such as {"ABCD123", "PQRS45", "BCD123", "PQRS456"}. The basic algorithm would generate 4 drive-by events, and finalize them all, thus erroneously reporting 4 different vehicles.

A more sophisticated algorithm should discern that these readings are from only 2 vehicles.

Substrings - Inability to merge

The third limitation occurs in situations when 2 LPNs previously believed to be unrelated become clearly related by a third LPN. For example, consider the situation the LPN stream produced by the upstream LPR system is {"ABCD", "123" "ABCD123"}.

Using the basic algorithm (with threshold = 3) three separate drive-by events are reported. A more sophisticated algorithm should be able to realize that the first two LPNs were from the same vehicle after seeing the subsequent LPN.

Table 3. Pseudo-code for Algorithm 2

a) The LPN processing loop in Algorithm 2 is as follows:

```
Set Q to a queue of size q
   while true
         set s to next LPN from S
         while  s is null
                  if  inactivity threshold exceeded
                           merge and close all Q
         Set b to a drive-by event  in Q where fits( s, b) is true
         if b is null
                  if Q is full
                           merge or close last
         set b to new bag
         place b at front of Q
         insert s into b
```

b) The pseudo-code for the *merge or close last* is as follows:

```
Set b to last drive-by event in Q
Set b' to drive-by event in Q where fits(l(b),b')
if b' is null or b' = b
         close b
else
         for each s in b
                  insert s into b'
remove b from Q
```

c) The *merge and close all* step can be implemented using *merge or close last:*

```
while Q is not empty
         merge or close last
```

2.3 Algorithm 2: Multiple Event Fusion

Algorithm 2 overcomes the limitations of the Algorithm 1 listed above by introducing:

1) A bounded FIFO queue Q of size q, which holds multiple buffers for concurrent drive-by events,
2) The ability to merge drive-by events.

The improvements in Algorithm 2 make it well suited to multi-camera deployments. According to Algorithm 2, LPNs s_p and s_q belonging to the same vehicle may be separated by n-1 distinct different vehicles and still be allocated to the same drive-by event.

The algorithm begins with Q empty. When the first LPN s is read from S, a new drive-by event b is created, labelled with the current LPN and added to Q. When a subsequent s' is read from S, it is compared to the labels of all active b in Q, and placed according to the criteria in 2.2.1. The label of the selected b is updated as described in Section 2.2.3.

If Q contains no b matching s', a new drive-by event b' is created, and appropriately labelled. If Q is not full, b' is added to Q. If Q is full, an attempt is made to merge existing b in Q. If no merge was possible, the last b finalized and removed from Q. After a successful merge or the removal of oldest b, b' is enqueued in Q.

When a drive-by event is removed from Q, it is finalized, and an actual drive-by event is reported. There are two situations in which a drive-by event will be finalized: (1) when Q is full or (2) when the inactivity threshold has been exceeded.

Table 3 presents pseudo-code for the main loop processing S, as well as the 2 methods removing or margining drive-by events.

3. VIDEO ANALYTIC PLATFORM (VAP) FOR LPR EVALUATION AND DETECTION OF VEHICLES NOT BEARING THE LICENSE PLATE

3.1.1 VAP Architecture

The Video Analytics Platform (VAP) is conceived and developed with the objective to enable the efficient retrieval of intelligence in video data coming from existing operational video surveillance systems. It is based on the advances in video recognition processing [3-5] and makes use of following observation on the nature of the Video Surveillance.

> *Even though a camera captures video data non-stop 24/7, in reality it is only at certain moments of time, when particular Events of Interest happen, when the captured video data need to be analyzed and when particular Details of Interest related to the event need to be extracted.*

For a user, Event and the Details of Interest are described using the English language. For example, events include "Door opens", "Car entered", "Face/Person seen" and details include "images of all people who entered", "car size and colour", "velocity" and "time and locations of persons/vehicles". In the VAP lexicon, the concepts of Event and Details of interest are defined as follows.

Definition: *Event of interest*, designated as ***E***, is an instance when certain conditions related to what is observed in video are met.

Definition: *Details of interest*, designated as ***D{ } (E)***, is set of static images and associated metadata (annotations) that is extracted and saved from video when an Event of Interest happens, of which one image with annotation is chosen to represent the Event.

Based on these definitions, the main VAP task is the following:

Task: To replace a continuous video-stream with a list of Details ***{Dj}*** that can be efficiently browsed and analyzed – by using a Video Analytic module that operates on the video-stream.

To accomplish this task VAP is made of two components: the *EventCapture* component, which serves to automatically detect a "Visual Event" from a number of video streams, and the *EventBrowser* component, which serves to display & facilitate the perusal of "Visual Details" captured from the "Visual Events".

The *EventCapture* component is a Windows MS program that has three functions. First, it taps into existing video-stream(s). Second, it embeds a VA module (software) that will process the video-steam(s). Third, it transfers the data that is extracted by a VA module to *EventBrowser*.

The *EventBrowser* component is a web application that is responsible for two main tasks. First, it stores all detected event details in a database. Second, it prepares an html-enabled web application that allows efficient viewing and analyzing of stored events.

3.1.2 Using VAP with LPR

As mentioned in the introduction, commercial stand-alone LPR systems cannot be used when a license plate on a vehicle is occluded or removed. The vehicles not bearing a license place will not be detected and reported. However, it is exactly these vehicles that are of main interest to the enforcement and intelligence communities!

VAP can be used to test and evaluate LPR systems. By running an LPR system and VAP at the same time, one can measure the rate of the false misses, which otherwise cannot be measured. The visual statistics feature of the EventBrowser can be used to populate the statistics. Figure 5 shows the snapshots of VAP running in real-life environment where vehicles are detected using Video Analytics in challenging environmental conditions.

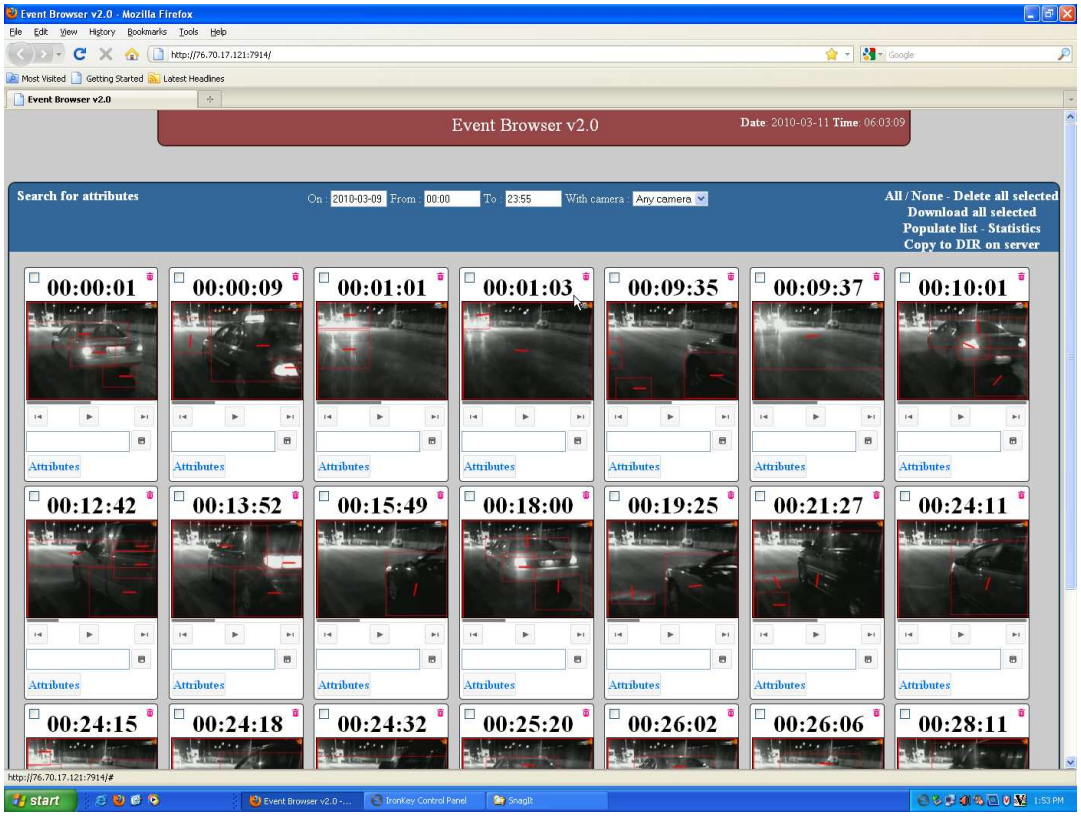

Figure 3. VAP is used to capture vehicles that pass a checkpoint.

4. EXPERIMENTAL RESULTS

4.1.1 Performance Evaluation Methodology

The LPN fusion algorithms described in Section 2 were tested in operational conditions. As depicted in Figure 4, the two off-the-shelf surveillance cameras have been connected to a commercial CCTV plug-in LPR system to monitor roadway traffic. The LPR system produces S, a continuous flow of LPN snapshots. The fusion algorithm monitors S and produces S', a filtered stream of drive-by events. Parallel to the LPR software, the VAP software was configured to capture events for all passing vehicles in order to detect cases when the LPR software fails to detect a vehicles/plate.

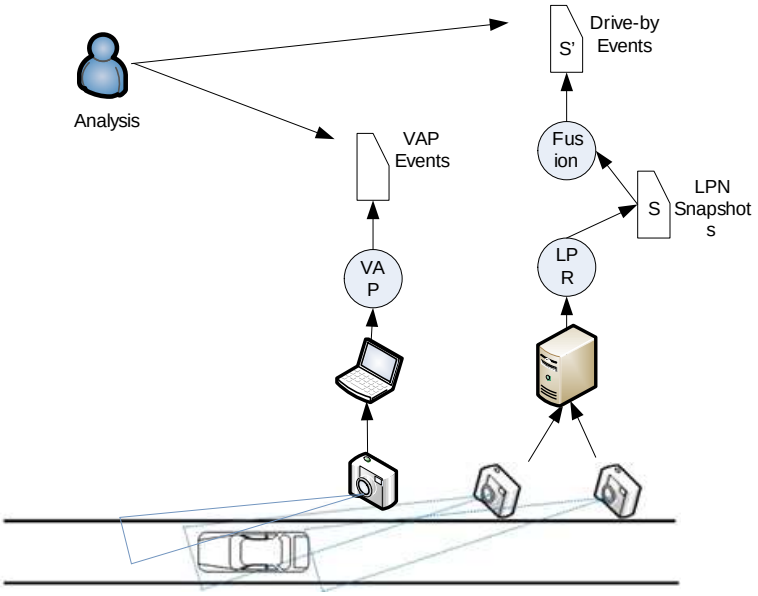

Figure 4. LPR test set-up

The system was left running for several days of which one was selected to perform the following analysis. The output of LPR was passed to the in-house built Fusion module configured with different parameters, and the filtered stream S' was manually verified against VAP-detected events for the same period to compare fusions approximation to actual traffic results.

Two types of errors, pertaining to trade-offs in choosing system parameters, are measured:

Type 1 error - *Incorrect event divisions*: This error consists of the incorrect division of LPNs which should be placed together into two or more distinct drive-by events. This errors is due to the threshold being set too high.

Type 2 error - *Incorrect merges*: This error consists of placing LPNs different vehicles being combined in the same drive-by event, in which only one number is captured. This type of error indicates that the threshold set too low.

4.1.2 Discussion of Results

The LPR system produced 4287 snapshots on one day of processing. The basic algorithm filtered these 4287 LPNs into 1049 drive-by events. Because of the limitations of the basic algorithm this number is larger than the true number of vehicles that drove by. The multiple buffer algorithm configured with 8 concurrent buffers produced 900 drive-by events. The reduced number of drive-by events indicates that the multiple buffer algorithm generated fewer Type 1 errors than the basic one.

To examine system behaviour in more detail, performance data for one hour of traffic was presented to the fusion algorithm and analyzed. Results are presented in Tables 4 and 5, and Figure 5. For this hour, the ground truth of 113 passing vehicles has been by extracted using the events detected by the VAP.

Table 4. Reducing divergence from ground truth.

Experiment	Ground Truth	LPNs	Divergence	% Reduction
LPR	113	745	632	
Algorithm 1	113	162	49	92%
Algorithm 2 (q = 8)	113	128	15	98%

Table 4 demonstrates the ability of the fusion algorithm to reduce the discrepancy between the original stream and reality. For the actual traffic of 113 vehicles, the LPR system generated 745 plate reads. The performance of the vendor system shows a 559% divergence from ground truth. With such low accuracy, the deployment of commercial CCTV plug-in LPR systems in operational environment is not feasible.

Algorithm 1 yields significant improvement in LPR performance: The 745 LPNs created by LPR are processed, producing 162 drive-by events. This represents a 92% reduction in the discrepancy between the results of the LPR system and ground truth. Applying Algorithm 2 (with 8 concurrent buffers) yields additional benefit, producing 128 drive-by events, which represents a 98% reduction of the original discrepancy.

Table 5. Behavior of Error types using different size of the Event buffer.

Number of Buffers (q)	Type 1 Errors	Type 2 Errors	Estimated Events	Ground Truth
1	49	0	162	113
2	23	1	135	113
3	19	1	131	113
4	19	1	131	113
5	19	1	131	113
6	18	1	130	113
7	17	2	128	113
8	17	2	128	113

Table 5 shows the tradeoff between Type 1 and 2 errors for different numbers of drive-by event buffers. As the number of simultaneous drive-by event buffers is increased, the number of Type 1 errors decreases, but the number of Type 2 errors increases. With more concurrently open buffers, there is a greater chance of placing an LPN into a related drive-by event. Also noteworthy, is the reduction of Type 1 error obtained simply by adding one concurrent buffer. As well, we note the relatively slow increase in the number of Type 2 errors. Experimentation with threshold values could further reduce Type 2 Errors.

Figure 5 provides another view of the effect of q on the algorithm results. As q increases, the number of Type 1 error decreases while the number of Type 2 error increases (very slightly). It appears that there are diminishing returns of setting q at a value greater than 3, as the number of Type 1 errors decreases only slightly, and Type 2 errors start to become noticeable.

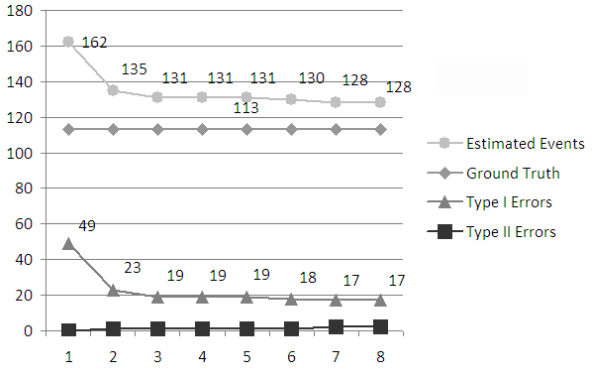

Figure 5. Effect of the size of the Event buffer (q) on fusion results.

5. CONCLUSION

This paper addresses the problem of applying LPR technology with CCTV systems for the purpose of detecting and identifying vehicles passing through a checkpoint. This problem poses many technological challenges, the main two of which are: i) noisy multiple reads that are often generated by the system, and ii) failure to detect a vehicle or license plate altogether when the license plate is occluded or not visible. This paper presents a solution to these problems. A data fusion technique based on thresholds and a MLD is used to resolve the first problem. An integration architecture which supplements a commercial LPR system with the CBSA Video Analytic Platform (VAP) is used to solve the latter. The integrated VAP-LPR-Fusion solution has been tested in operational environments and has been shown to yield a substantial improvement over commercial off-the-shelf LPR used in a stand-alone manner. The data fusion algorithm is shown to be able to filter the output of an LPR system, reducing the discrepancy between ground truth and estimated traffic by 98%.

ACKNOWLEDGEMENTS

The help of Elan Dubrofsky, Jonathon Lee, Taylor Cassidy, and Fern Corriveau in implementation and testing of the system is gratefully acknowledged.

REFERENCES

[1] Second federal department's workshop on Deploying Video Technologies for National Security (VT4NS'08). Theme: "Deploying Future-Proof Video Technology". Organized by CBSA-S&E in coordination with DNI-VACE, Ottawa, 23 October, 2008 – www.videorecognition.com/vt4ns/08

[2] Third federal department's workshop on Deploying Video Technologies for National Security (VT4NS'10). Themes: "Deploying Video Analytics" and "Faces in Video". Organized by CBSA-S&E in coordination with DRDC-CSS, Ottawa, June 3, 2010.

[3] D. Gorodnichy, "ACE Surveillance: The Next Generation Surveillance for Long-Term Monitoring and Activity Summarization". First International Workshop on Video Processing for Security (VP4S-06), June 7-9, Quebec City, Canada. NRC 48493, 2006.

[4] D.Gorodnichy, T.Mungham, "Automated video surveillance: challenges and solutions. ACE Surveillance (Annotated Critical Evidence) case study". NATO SET-125 Symposium "Sensor and Technology for Defence against Terrorism", Mainheim, April 2008.

[5] Dmitry O. Gorodnichy and Elan Dubrofsky. VAP/VAT: Video Analytics Platform and Testbed for testing . SPIE Conference on Defense, Security, and Sensing, DS226: Visual Analytics for Homeland Defense and Security track. 5 - 9 April 2010, Orlando

[6] Vladimir I. Levenshtein, *Binary codes capable of correcting deletions, insertions, and reversals*, Doklady Akademii Nauk SSSR, 163(4):845-848, 1965 (Russian). English translation: in Soviet Physics Doklady, 10(8):707-710, 1966.

[7] Bonet I. e. al "Comparing Distance Measures with Visual Methods", MICAI 2008: Advances in Artificial Intelligence 7th Mexican International Conference on Artificial Intelligence, Atizapán de Zaragoza, Mexico, October 27-31, 2008 Proceedings, Springer Berlin / Heidelberg, pp. 90-99

[8] Heeringa, Wilbert, Gooskens, Charlotte "Norwegian Dialects Examined Perceptually and Acoustically", Computers and the Humanities, 2003-08-01, Springer Netherlands, pp 293-315

[9] Trujillo, Leonardo, Olague, Gustavo, Lutton, Evelyne, Fernández de Vega, Francisco, "Discovering Several Robot Behaviors through Speciation", Applications of Evolutionary Computing, Lecture Notes in Computer Science, 2008, Springer Berlin / Heidelberg, pp. 164-174