Tracking Vehicles in the Presence of Occlusions

Channa Meng[1], John Morris[2*] and Nattawoot Suwannata[1]

[1] Faculty of Engineering, Mahasarakham University, Maha Sarakham, Thailand
[2*] K-RIS, King Mongkut's University of Technology, Ladkrabang, Thailand

meng.mengchanna.channa@gmail.com, john.m@msu.ac.th[*] and nattawoot.s@msu.ac.th

Abstract. *Thailand loses about 25,000 people every year to road trauma: this motivated this study, which designed and evaluated a simple system to discourage 'Red Light Running' – failure to observe red traffic lights. Our system is simple, cheap and flexible – consisting of a single camera, a portable computer and the ability to send images to a mobile phone using the public network. However, to be flexible, cameras were set only 1-2m about the road, which caused many occlusions to be observed – the major challenge for the system software. A rule-based system was used to resolve most occlusions. In our tests, vehicles were completely and correctly tracked in 83% of frames. This was sufficient to allow images of 95% of Red Light Runners to be transmitted to a monitoring station and potentially stopped.*

Keywords: Vehicle tracking, occlusions, traffic violations, red lights

1. Introduction

The world-wide cost of road trauma is truly horrifying: WHO reports 1.25 million deaths annually [1]. In a relatively small country, Australia (pop 24 million), the annual cost is estimated to exceed $US15 billion (2015 data) with each fatality costing more than $US3 million [2]. This cost is despite stringent law enforcement and significant testing of drivers, which keeps the fatality rate down to 5.4 per 100,000. However, in Thailand, where enforcement is slack and training negligible, fatalities climb to 36.2 per 100,000 and cost about 24,000 lives annually [3]**.** Thailand ranked second in the world on this metric in 2013 – with little variation since - so attempts to improve the situation are very relevant in Thailand [3].

A breakdown of road accident causes lists speeding and careless lane changing as the major causes – see Fig. 1. Although ignoring red lights (red light running or RLR) accounts for only 1.5% of total accidents, most RLRs generate high speed side impacts which cause injuries in 48% of crashes, compared to 30% for other crashes, their cost is disproportionately higher [4]. Extrapolating US statistics (260,000 crashes for 750 fatalities) to Thailand's 24,000 annual fatalities [3], suggests that, annually, there may be ~8 million crashes of which ~100,000 would be caused by RLRs. Thus, this project which aims to educate drivers to obey traffic signs – in particular red lights – has the potential for significant impact in Thailand and other similar countries with high road accident fatality rates.

Thus we designed a simple system which could be used to train drivers to obey traffic rules and save many lives every year. Our design aims required the system to be

- Cheap: *i.e.* use cheap, readily available cameras and other components,
- Portable: easy to setup,

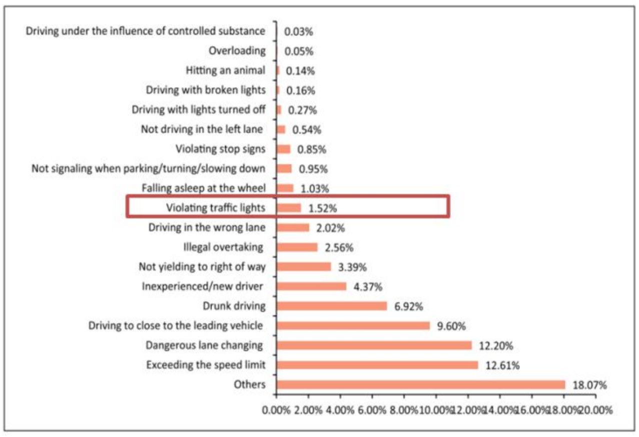


Fig. 1 Causes of traffic accidents in Thailand: source Royal Thai Police, 2015 [3]

- Flexible: not dependent on road markings (or their absence),
- Effective in a variety of scenarios
  - camera able to be positioned within a 20^{o} to 90^{o} angle to the traffic flow,
  - multiple lanes of traffic and
  - not affected by road-side clutter, *e.g.* random fixed objects, banners, flags *etc.* and
- Require minimal changes to current police practices.

Thai Police currently use road-blocks as the primary tool for detecting vehicle violations, so our system positions a camera at the monitored traffic light and transmits an image to a road-block that may be 500m further down the road. Fig. 2 shows how our system might be set up for a typical intersection.

2. Background

The key task in our system is extracting the RLRs from images captured by a *single* camera. It must be able to separate an RLR from the background and track it to ensure that it did, in fact, 'run' the red light, *i.e.* it was moving in a forbidden area when a traffic light turned red. The general problem of object tracking has been extensively studied and is usually separated into (a) modeling the (static or slowly changing) background to remove it, (b) detecting relevant objects remaining and (c) tracking them. A brief review of existing work in these areas follows accompanied by some comments on our reasons for choosing effective algorithms from the wide slate of algorithms used in similar and related applications.

2.1 Background Subtraction

The first step, BackGround Subtraction (BGS), extracts objects of interest, *i.e.* the foreground, in an image, by subtracting a background model: we can observe two types of background - static and dynamic. Historically, BGS algorithms have evolved from early simple algorithms, which simply removed pixels which match those in a static background image, to adaptive ones that allow the background to vary. Complex dynamic background models are needed to include

- Scene illumination changes, *e.g.* sunlight variation from dawn to dusk,
- Haze in the scene from rain, fog or thermal gradient changes (shimmer),
- 'Fixed' components of the scene that move with the wind, *e.g.* tree leaves and flags,
- Shadows (usually tracked with objects but strictly irrelevant and ideally removed),
- Background camouflage (causing objects to be classed as parts of the background),
- Camera jitter and
- Bootstrapping (initially static objects assigned to the background at startup).

Choudhury *et al.* surveyed work up to 2016 and evaluated several key algorithms for videos available in five commonly used video datasets: Wallflower, I2R, Carnegie Mellon, Change Detection and the Background Models Challenge [6]. Important approaches and algorithms are summarized in Table 1 and mentioned briefly in the following sections. The papers listed by Choudhury *et al.* might suggest that most of problems related background subtraction have been solved - except very fast illumination changes. However, BGS work is clearly ongoing and at least one paper [7] was found in February of 2018!

In our problem, the background for an intersection is usually slowly varying and so models which adapt to slowly changing backgrounds are adequate. One 'background' task, unique to this project, is the detecting and monitoring the red traffic light controlling an intersection - discussed separately at the end of this section.

Early basic models set the first frame as a background model and subtracted it from the current frame to extract the foreground. Subsequent frames are compared to the initial frame – or the previous one if the background varies slowly - to detect moving objects. Further improvements find the arithmetic mean of pixel values over a temporal sequence, *e.g.* the W^4 system separates people from their background in an outdoor environment using three values - minimum gray value, maximum gray value and maximum intensity difference between two adjacent frames [8] These are generally unimodal and do not work well with uncertain backgrounds [6].

2.1.1 Gaussian Mixture Models

Univariate Gaussian Models – with a Gaussian for each RGB channels – can model temporal variations in background intensities - as long as they do not oscillate. Stauffer and Grimson therefore used Gaussian Mixture Models (GMM) with each pixel represented by fixed number of Gaussians to follow illumination changes with a learning rate parameter. However, too slow a learning rate failed to handle rapid illumination changes, whereas too fast a learning rate incorporates slow moving objects into the background [9].

Zivkovic and Heijden added a new simple training technique to implement an efficient adaptive GMM model. The data for the $\boldsymbol{m}^{th}$ component depends on determined weights. With this, GMM produced an effective background model in traffic scenes; it quickly adapts to oscillating backgrounds [10]. It is available in the OpenCV library, **`BackgroundSubtractionKNN`** method [11]. Because it functioned acceptably for vehicle tracking, it was used as the basis of this work. Our decision was supported by Xu *et al.*, who assessed it in their 'promsing group', almost matching ViBe for speed [12].

2.1.2 Other BGS Models

2.1.2.1 Non-Parametric Models

Kernel Density Estimation (KDE) techniques, extended from default Gaussian distributions, adapt to unknown distributions, but they require sufficient training samples to correctly estimate the density function. The functions are expensive to compute, so several have discussed methods to reduce the kernel bandwidth (number of kernels used) without sacrificing performance: mean-shift models were used by Paccardi and Jan [13] and Elgammal et al. [14]. Mittal and Paragios also used optical flow to initialize the background model [15]. They successfully modeled moving trees in a street scenario: a common problem in our scenes, but avoided by our low camera angle: we simply cropped moving leaves from the top of each frame.

2.1.2.2 Buffer Based Subtraction

Lo and Velastin kept pixel history in a FIFO buffer and compared the buffer median with the current pixel to determine whether it is a foreground or a background pixel, which is then added to the buffer. An alternative used the medoid rather than median. Linear prediction from the buffer with a Wiener filter has also been used [16]. Wang and Suter (2007) also described a buffer consensus approach which was successful on indoor and outdoor scenes with varying illumination levels [17].

2.1.2.3 Fuzzy Models

Fuzzy models have been useful in rapidly changing environments. El Baf *et al.* found that colour space choice was important when dealing with camouflage, water surface, waving trees and fast illumination change [18]. They obtained some very high scores for specificity in Choudhury's evaluations.

2.1.2.4 Learning Models

Neural nets have been used to train the underlying density distribution of the pixel sequence and predict the next frame. Culibrk et al's Background modeling Neural Net (BNN) used 124 neurons to alter background model using color and texture detection and improved the handling of gradual illumination variation, sudden illumination variation, shadow and bootstrapping [19]. Yao and Odobez' Multilayer [20] was the only system to produce acceptable results for the time of day variation in Choudhury's evalution[8]. However, neural nets typically require significant amounts of training data [7], so they are difficult to apply to 'fresh' previously unviewed scenes.

2.1.2.5 Low Rank Sparse Decomposition

Zhou and Tao's GoDec [21] decomposed the image into a low-rank background matrix (the background model), a sparse foreground matrix and a noisy component. GoDec performed well on the camouflage data sets where the foreground objects are (deliberately) similar to background ones [12].

Authors	Model	Description	Ref
Choudhury *et al.*	**2016 Survey**	**Evaluated 11 state-of-the-art algorithms for BGS**	**2016 [6]**
Reddy *et al.*	BCCPDI	Block-based classifier cascade with probabilistic decisions	2013 [58]
Wren *et al*	Pfinder	Real-time tracking of the human body	1997 [59]
El Baf *et al*	Fuzzy Integral	Fuzzy integral for moving object detection	2006 [18]
Maddalena and Petrosino	SOBS	Self-organizing background subtraction	2008 [28]
Kaewtrakulpong and Bowden	IABMM	Adaptive background mixture model with shadow detection	2001 [60]
Yao and Odobez	Multi-Layer	Multi-layer background subtraction based on color and texture	2007 [20]
Hofmann *et al*	PBAS	Pixel-based adaptive segment	2012 [27]
Rodriguez and Wohlberg	FPCP	Fast principal component pursuit via alternating minimization	2013 [61]
Zhou and Tao	GoDec	Randomized low-rank and sparse matrix decomposition	2011 [20]
Barnich and Van Droogenbroeck	Visual B'ground Extractor	VibeGray and VibeRGB	2011 [29]
Lo and Velastion	FIFO Buffer	Compare current pixel with median in FIFO buffer	2001 [16]
Wang and Suter	FIFO Buffer	Buffer consensus	2007 [17]
Subudhi *et al.*	**2016 Survey**	Local change detection with twelve existing BGS techniques	
Stauffer and Grimson	Adapt. B'ground Mixture Models	Thresholding the error between an estimate of the image without moving objects and the current image.	1999 [9]
Zivkovic	GMM	Extension of GMM with kNearestNeighbours in OpenCV as `BackgroundSubtractionKNN`	2004 [10]
Sajid and Cheung	B'gnd model bank	Mimic human eye structure by using various colour spaces	2015 [24] 2017 [25]
Maddalena	SOBS	Self-Organizing Background Subtraction	2008 [28]
Shimada *et al.*	Bidirectional	Accuracy improved by including forward frame and accepting latency	2013 [48]
Haines and Xiang	DPGMM	Dirichlet process GMM	2013 [49]
Seidel et al	pROST	Robust online subspace tracking based on alternating minimization on manifolds	2014 [50]
Hardas *et al.*	Shadow	Distinguish between self and cast shadows	2015 [22]
Jeeva and Sivabalakrishnan	Performance	PBAS *vs* SOBS *vs* ViBe: ViBe 25% faster	2015 [26]
Xu *et al.*	Performance	GMM, KDE, Codebook, AGMM*, SACON, SOBS*, ViBe*, PBAS*: * = 'most promising', ViBe fastest after AGMM	2016 [12]

Table 1 Summary of background subtraction techniques

2.1.2.6 Shadow Removal Model

Shadows generally appear as foreground. Schemes for matching them either with the background color or textures exist. RGB colour space methods are noisy at low intensities; HSI colour spaces have been used instead with a median filter updating the existing model. Texture based techniques match features of a potential shadow to textures already present in the scene. Two stage algorithms combining both techniques have been used [10]. Hardas *et al.* distinguished two types of shadows: 'self-shadows', part of the object shadowing itself (*i.e.* part of the foreground), and 'cast shadows' - cast onto the background. Since shadows contain low brightness regions, RGB channels were normalized and then multiplied with a fixed value matrix. If the result was below a threshold, that pixel was assigned to a shadow, otherwise it was marked foreground. Post-processing morphological operations filled in foreground holes [22].

2.1.2.7 Post Processing Refinement

A foreground, extracted using a background model, might be incorrect due to a poor initial background model. Post processing checks for false alarms: steps used include median filtering and morphological operations to improve scattered noise pixels and connected components analysis to reconnect disjointed pixel. These operations can fill up camouflage gaps.

2.1.2.8 Bag of features models

Subudhi *et al.* described a bag of features containing three old features - (a) Brightness (BS), (b) Inverse contrast ration (ICR), (c) Average Sharpness (AS) - and three new) features – (d) Absolute Relative Height (ARH), (e) Local Weighted Variance (LWV) and (f) Integrated Modal Variability (IMV). They compared their results with twelve state-of-the-art algorithms and claimed that their technique was robust to noise, surface reflection, effects of uniform shading and lighting changes, with the exception of camouflaged objects [23].

2.1.2.9 Colour Spaces

Sajid and Cheung noted that the human visual system uses rods for low light imaging and cones for colour distinction and used colour spaces to mimic this: their system includes RGB, YCbCr, HSV and HIS spaces. YCbCr performs better for foreground segmentation and has been shown to be less affected by noise, shadows and illumination changes. They employ a Background Model Bank based on each channel and select a background that correlates with the input image, then in a number of steps computed on corresponding pixels decide establish a probability that each pixel is background or foreground. They argue that this is fast and effective [25] – supported by experiments showing high scores on images from the CDnet 2014 database [26].

2.1.3 Performance Evaluation

Jeeva and Sivabalakrishnan [27] assessed BGS techniques, including PBAS [28], SOBS [29] and ViBe [30]. They found that ViBe was about 25% faster than SOBS and PBAS [14]. Xu et al's [12] more extensive survey covered GMM [9], KDE [31], CodeBook 56], AGMM [10,11], SACON [17], SOBS [29], Vibe [30] and PBAS [28]. Overall, they rated AGMM, SOBS, Vibe and PBAS as the 'most promising' with Vibe exhibiting the highest frame rates for large images closely followed by AGMM. AGMM and Vibe also showed the smallest memory requirements. This supports the decision in this work to use Zivkovic's AGMM [11]. Frame rates for CodeBook were also high, but memory requirements were typically 10× those of other methods [12].

2.2 Red Light monitoring

Automatic red light detection is usually straightforward, since the red light is, by design, bright enough to be detected easily in most situations. Several techniques have been used, for example, Sooksatra and Kondo [32] used color from the CIELab model and a fast radial symmetry transform. Traffic light detection from a moving (autonomous driving) vehicle was addressed by Guo *et al* [33]: starting in the HSV space, they scanned the scene to detect candidate locations, and then used histogram of gradient features and an SVM approach to detect the light. Diaz *et al* also published a survey – targeting red light detection from moving vehicles [34], a much more difficult task than the current application, which assumes a stationary camera, and only requires the single camera to have a clear view of one of the controlling red lights.

2.3 Object modeling and tracking

Many techniques have been implemented and improved to track efficiently. An early 2006 survey by Yilmaz *et al.* [35] has been updated by recent surveys: in 2013, Smeulders *et al.* assessed trackers experimentally [36] and recently Wu *et al.* evaluated tracking performance in frames per second, success and precision [37]. They tracked single targets but also considered performance in the presence of overlap or occlusions. Further, they note that performance varied with target speed and density.

Kristan and a large team of collaborators reported on the VOT2015 challenge for object tracking algorithms [38]: VOT2015 compares single-camera, single-target, model-free, causal trackers, applied to short-term tracking; it covers 62 tracking algorithms. In VOT2015, the MDNet tracker (classified as a deep convolution neural network) was rated as the best tracker based on its adaptability and robustness [39]. However, other trackers showed better performance on individual datasets, which makes overall assessment quite complex. VOT2016 added a new sub-challenge - VOT-TIR for tracking objects in thermal images: further enhancements were made in 2018 [40].

Tracking in the presence of potentially large numbers of occlusions was a key problem in this study, as we wished to extend the work to other traffic violations. This has also been studied extensively, *e.g.* Lee *et al*. [41], Zhao *et al.* [42], Motro and Ghosh [43], and a set of multiple tracking benchmark sets focusing mainly on people, with several moving camera videos, has been generated [44].

2.3.1 Optical flow

Optical flow techniques find motion flow vectors by matching small patches surrounding each region (dense optical flow) or feature (sparse optical flow) of an image. They have been used in many applications, for example Doyle *et al.* presented a real-time pan/tilt camera object tracking based on features both in the background and the target object [45]. Kale *et al.* used motion vector estimation of optical flow for object position estimation, unaffected by image blur and a cluttered background. They segmented the image (sparse optical flow) to return a result quickly [46].

After initial trials, using optical flow to track vehicles by aerial surveillance, we found that the finding optimum parameters was a challenge and we decided to follow Occam [47] and used one of the simplest and oldest approaches – template matching – and found acceptable results – see later.

2.3.2 Kalman filter

Kalman filters have been used extensively to predict the next position of an object: they use a history of prior observations and uncertainties to find the most probable next position. For example, using Kalman filters, Li *et al.* were able to build a fully automated system which did not require any manual input [51] .An implementation in OpenCV [52] was used here to continue to predict the position of an object that has disappeared from view.

2.4 Communications

2.4.1 TCP/IP

The Transmission Control Protocol/Internet Protocol (TCP/IP) has become a very common protocol on the Internet and is now commonly available on mobile phone networks. TCP/IP connections should be reliable as checksums are attached to each packet and retry protocols in the lower network layers recover lost or corrupted packets. However, in practice, using cell phone connections, we observed many lost packets and implemented a simple retry scheme to ensure that each image was transferred correctly.

2.5 Existing systems

Several commercial systems are also now available, suited to well controlled intersections in western countries, where vehicles tend to stay in well marked lanes, not overtake within lanes, *etc.* and cameras can be aligned with lanes and monitor vehicles traveling in a single direction, *e.g.* Jenoptick's Roadstart system, which monitor up to fourlanes and relies on road loops or radar[53] and Smart Vision Technology's Smart-RLCS using a camera per lane [54]. They also generally assume a fixed camera positioned high above the road surface leading to smaller numbers of occlusions, whereas we aimed to create a small portable system with a camera set 1-2 m from the ground, so that it will view many vehicles obscured by others. Previous attempts by Klubsuwan *et al.*, in Thai conditions, detected only ~80% traffic light violations with images alone [55].

3. System design

3.1 Setup

Our system consists of a single camera attached to a PC and positioned to monitor a controlled intersection. It monitors the traffic lights themselves and does not require assistance from traffic control hardware or road loops. When it detects a red light runner, it captures images of the offending vehicle and transmits a set of images to a police post, which can be several hundred meters down the road. The total cost can be less than $US5000 – one of our design requirements that it should be available easily and cheaply. Figure 2 shows a typical multi-lane intersection and the range of positions where the camera may be placed: it only needs a clear view of the 'forbidden area' (see later) and a clear view of at least one traffic light.

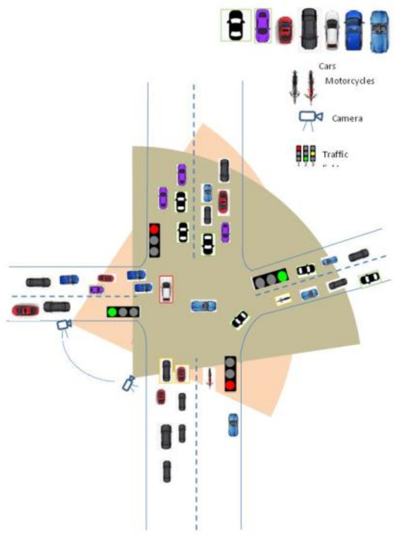


Figure 2 Overview of system at a controlled intersection showing range of positions of the camera. The solid regions show the camera field of view at the extremes of the camera position.

The software is rule-based: it builds models of all moving vehicles in the processed region and uses multiple rules to match models from one frame to the next [57].

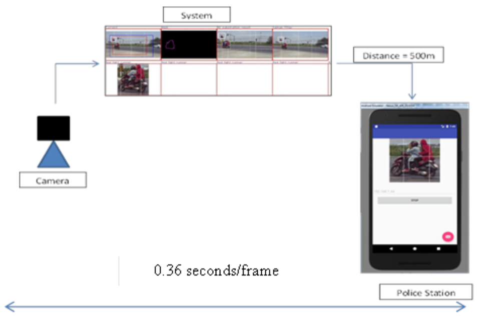


Figure 3 Overall system: camera at intersection sends RLR images to a mobile phone which may be some distance away

Figure 3 shows the system setup. After the forbidden region (red box in Figure 4) and the traffic light are identified, the program moves into automatic mode – identifying and capturing red light runners.

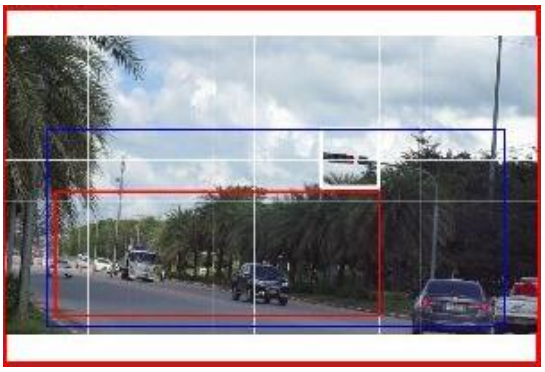


Figure 4 System setup: Red light detected in the white box. The operator clicks once to verify that this is the controlling light.
The red box marks the ***forbidden region***: no vehicle in the left lane should be moving in this area when the light is red. The blue box is the automatically added ***processed region***: vehicles are tracked in this area to capture templates of vehicles which may soon enter the ***forbidden region***.

3.2 Vehicle Detection and Tracking

The ***processed region*** reduces computation time significantly: it also prevents tracking many large, moving, but irrelevant objects, such as large tree branches and flags by the roadside. In typical intersections, the background is reasonably stable with respect to visible structures, so it is usually simple to segment the scene into background and foreground – see an example in Fig 4.

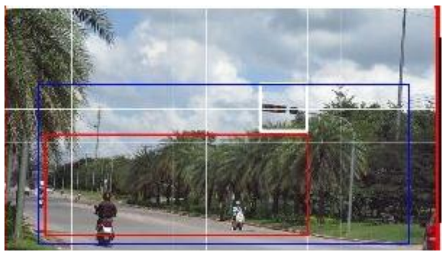
Our algorithm starts by modeling the background. Although the `backgroundSubtractionKNN` class can model a moving background, in practice, 'ghosts' of moving vehicles would remain, so we chose to select seven moving-vehicle-free frames (even in Thailand, sometimes *all* vehicles stop for a red light!) to model a more stable background and update whenever new moving-vehicle-free frames are encountered.

Initial algorithms parameter settings include:

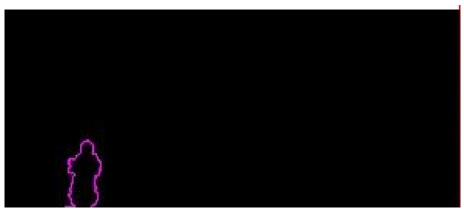
- d_{thresh}, an estimate of the maximum distance (at normal traffic speeds) that a vehicle can move, in pixels, in one frame time: d_{thresh} = 50 pixels was used for most videos.
- a_r, the acceptable contour area change from frame to frame: based on difference between a car and a motor-cycle. This ensures that a change caused by a motorcycle merging with a larger vehicle is detected.

After the background is subtracted, contours are drawn around moving vehicles (see Fig 5(b)). Each contour, that meets the criteria for a vehicle (reasonable size, position and shape), is assumed to contain a vehicle and a model of it is formed, including its position and size. A colour template containing the image inside the contour is added to the model along with other details to support matching (*e.g.* contour moments), speed estimates and the Kalman filter for it.

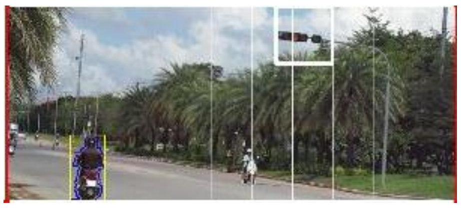
In subsequent frames, contours detected in that framc are compared with vehicle models from the previous frame. If there are simple matches, the previous model is updated. New vehicle models are created when vehicles enter the processing area and deleted when vehicles have left it. More complex situations, when vehicles ***merge*** with others (ie one vehicle becomes occluded by another) and **split** because a previously occluded vehicle is now clearly visible. For **merges**, models for both vehicles are retained (see more detail in the next section), expecting that they may split later. **Splits** can cause a new vehicle model to be created, if the vehicle was substantially occluded in previous frames. Moving leaves are removed if moving predominantly green objects are found too high in the scene. Full details of the software model and the rules applied may be found in Meng [57].



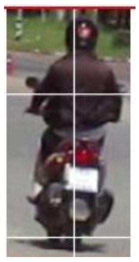
(a) Original image: light is red.



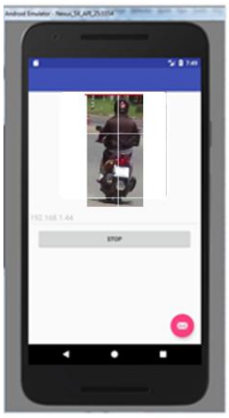
(b) After BGS, contour drawn around moving vehicle



(c) Bounding box drawn around RLR



(d) Image of RLR sent by phone to monitoring station



(e) Image received on mobile phone

Fig. 5 Steps in processing an image of a RLR

Merges are detected when a contour becomes much larger than a partially matching contour in the previous frame. In this case, the template for the foreground object is 'shrunk' by leaving only the matched area from the previous frame. The remaining fragment of the occluded vehicle is checked against another vehicle model. The occluded vehicle position is updated if sufficient of its template is visible to make a good estimate of the new position from a dead reckoning estimate.

In the final step, images of all RLRs are sent (one by one) to a mobile phone – see Fig. 5 (e).

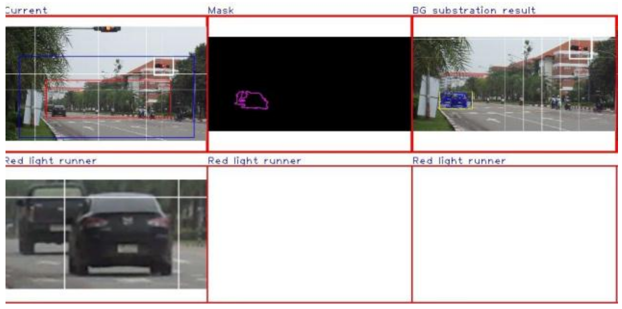


Fig 6 Merge example: the contour and bounding box of the foreground vehicle includes both vehiclesat intersection A (see Table 1)

4. Results

In this section, a summary of results for tracking *all* vehicles (not only RLRs) is presented. This is a first step towards extending this work to include other types of traffic violations, *e.g.* overtaking illegally.

The variety of intersection types, at which data was collected, is shown in Table 2.

4.1 Error counts

Table 3 shows the *overall* counts for vehicles tracked, Red Light Runners detected, events, *i.e.* situations in which vehicles occluded one another and errors. Only frames for which the light was red were processed and counted. Errors were counted for each object in each frame, *i.e.* a vehicle which was not correctly tracked for ***n*** frames was counted as ***n*** errors, and represent *total* tracking errors and not the much lower counts for incorrectly tracked vehicles. Table 3 focuses on correct tracking of *all* vehicles - including some tracked traveling in the opposite direction, which would not be classified as RLRs. No vehicles failed to be detected: errors were caused when overlaps between vehicles led to groups of vehicles (sometimes as many as four) being temporarily tracked as one. The vehicles were tracked completely in more than 83% of the frames in which they appeared. However, a much higher fraction of red light runners were identified and images captured - approaching 100% for single event red light runners (with only one failure, in 7000 + frames, due to inexperience in positioning the camera to properly view right turning vehicles). Tracking errors were caused when overlaps between vehicles led to groups of vehicles (sometimes as many as four) being temporarily tracked as one.

The tracking error counts in the final column show the good results from this system. Remember that these are per frame tracking errors and routinely a single tracking error does not cause a RLR to be missed – it will be detected correctly in other frames. An RLR will appear in the forbidden region, typically for 50 or more frames (several seconds at 15fps), thus the RLR detection rate approaches 100% even with these tracking error rates. They are strictly worst case numbers and were reported as part of an extension of this project to detect other violations - *e.g.* illegal lane changing.

Tracking individual vehicles, *i.e.* ones which did not overlap with others, caused few problems: background subtraction of vehicles close to the camera is accurate because in this environment the background was relatively stable and a reliable model was obtained. However, the low camera angle caused many vehicles to appear overlapped or occluded by others and required special processing and, as might be expected, led to some situations where tracking is not accurate. These situations may be classified as (a) merges or (b) splits and are discussed in the next sections. Algorithm details and pseudocode may be found in Meng's thesis [57].

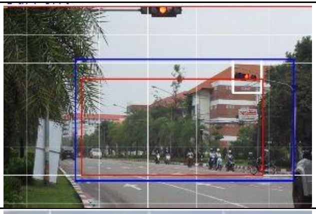
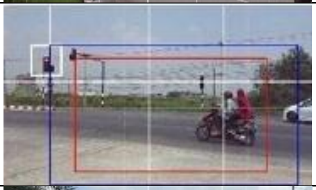
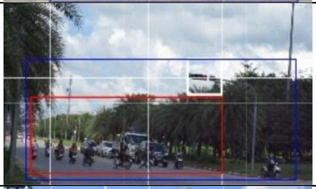
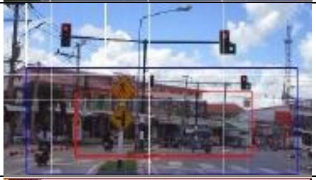
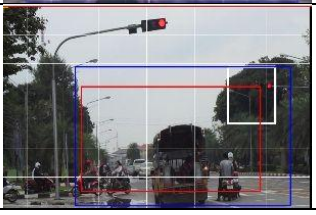
Label/ Location	View	Description	View angle
A 16:24177ºN, 103:25321ºE		Mahasarakham university's main road, 2 lanes in each direction plus side road turning to the university on the right side, well-marked road	10^o
B 16:24177ºN, 103:25321ºE		Minor road leading out of Mahasarakham city, 2 lanes in each direction + side road, almost no road markings. Note the forbidden area stops short to the left: this avoids tracking vehicles which entered on the right before the light turned red.	50º
C 16:24648ºN, 103:24590ºE		Main road in front of Mahasarakham University campus, 2 lanes in each direction + side road entering campus, well-marked road	10º
D 16:23585ºN, 103:26380ºE		Road through shopping centre near campus, 2 lanes in each direction + 2 side lanes, angled smaller road entering from left, new well-marked, multiple traffic lights	0º
E 16:24177ºN, 103:25321ºE		Mahasarakham university's main road, 2 lanes in each direction plus side road turning to the university on the left side, well-marked road	0^o

Table 2 Details of intersections used in tests

Video Location	Frame Count	Vehicle Tracked	RLR		Events		Tracking Errors	Error (%)
			GT	Imaged	Merge	Split		
A1	237	13	2	2	5	9	17	7.2
A2	278	7	4	3	2	3	6	2
A3	276	11	2	2	2	5	34	12
B	492	13	-	-	12	2	1	0.2
C1	366	22	2	0		11	38	10
C2	399	17	0	0	20	11	38	1.5
C3	398	24	2	2	12	13	25	6.3
C4	398	21	1	1	12	9	46	12
C5	398	23	1	1	14	17	67	17
D1	699	21	0	0	6	7	32	4.6
D2	809	22	4	4	22	20	53	9
D3	809	28	4	4	9	7	97	12
E1	590	37	0	0	40	33	77	13
E2	590	38	0	0	18	18	58	9.8
E3	591	41	0	0	27	20	62	10.5

Table 3 Summary of tracking errors from 7330 frames in 15 videos: locations identified with codes in Table 2.

4.2 Merges

Merges occur when a vehicle starts to occlude another going in the same or opposite direction. This is recognized when the vehicle appears to be 33% larger than it was in the previous frame. See example in Figure 6. Models for both vehicles are maintained and the background vehicle is still tracked.

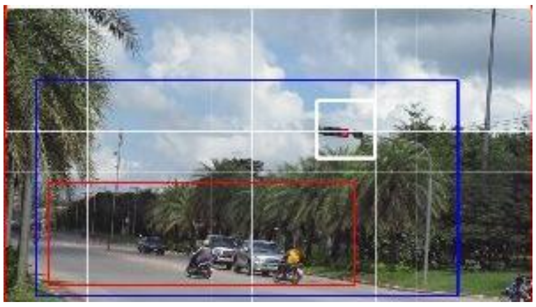


Figure 7 Original image (frame 4526) Marked up intersection C image: Forbidden area – red, Processed area – blue. Note that frame numbers in this and following frames are simply counts from some arbitrary video starting point.

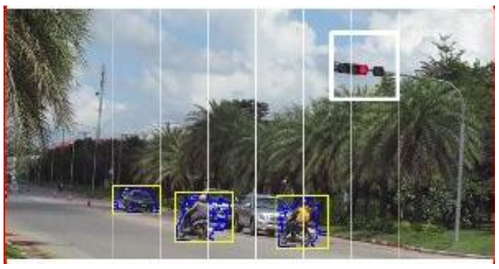


Figure 8 Moving vehicles detected after background subtraction

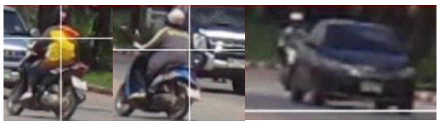


Figure 9 Vehicles detected: two left images RLRs to be sent to the monitor station; right image vehicle moving towards camera

4.3 Split

Splits usually follow merges: the vehicles that were overlapped separate and are detected as distinct separated vehicles in subsequent frames. Figures 10 - 12 show two cars coming from different directions that split after being merged in a previous frame. When two vehicles merge, models for both vehicles are retained and the original template of the occluded one is retained: so the vehicle is matched again after the split. Close vehicles entering the processing region are initially treated as one vehicle, but partial matching of templates enables them to recognized as distinct vehicles after some frames. An example of this is seen in figure 11 (centre): the bounding box initially covered both the motorcycle and the car behind it but is now split.

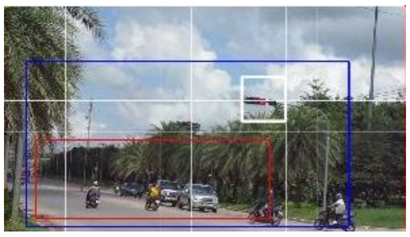


Figure 10 Original image (frame 4546): In this scene, the two cars in the opposite lane are stationary (presumably waiting for red light) and thus were removed as part of the background. The third car is still moving.

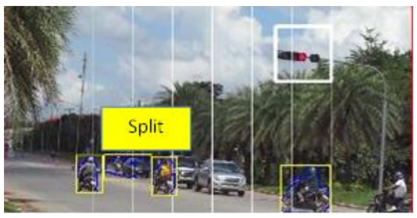


Figure 11 Splitting after Merging: Several frames after figure 7: the motorcycle on the left has split (its bounding box is now much smaller). The vehicle behind has stopped and removed by background subtraction. The two motorcycles are correctly located in their bounding boxes.

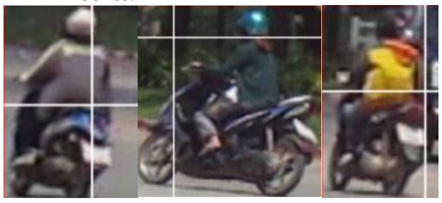


Figure 12 Vehicle images after bounding boxes correctly assigned after a split

4.4 Kalman filter

Tracking may 'lose' a vehicle when it becomes mostly occluded by one in front. Our system tracks a vehicle when it is only partially occluded and there is a sufficient visible template to match it from a previous frame, but eventually this template becomes too small for reliable matching. The model for each vehicle retains a history of known positions in previous frames and uses this history to predict the next position using a Kalman filter. As an example, a vehicle was 'lost' in frame #5328 (Figure 13), but the Kalman filter predicted the position successfully based on information captured in previous frames. Overall, this lead to a 91% success rate.

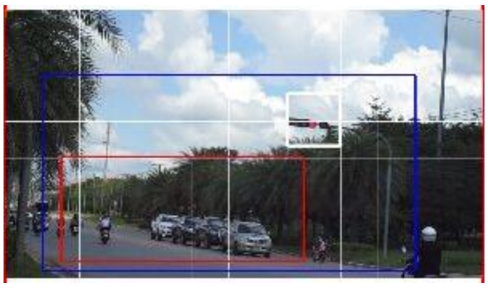


Figure 13 Frame 5328: Original image

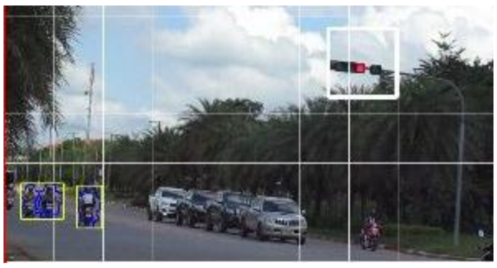


Figure 14 Frame 5332 – Bounding boxes correctly placed after BGS and contouring alone; two left motorcycles were tracked together because they were close in many previous frames.

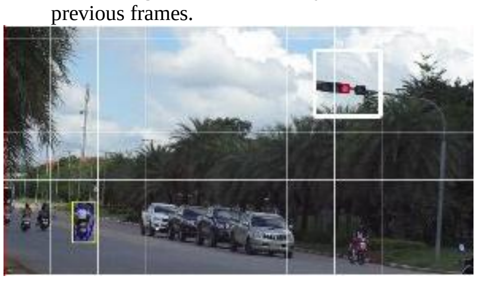


Figure 15 Frame 5334 After BGS, the vehicle was lost.

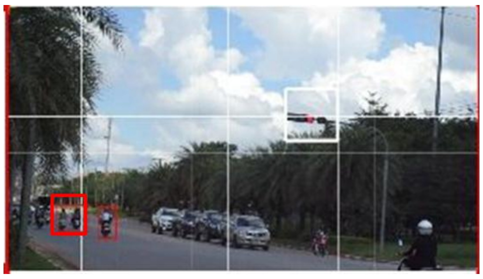


Figure 16 Frame 5334 Kalman Filter used to estimate bounding box (bold red) in correct place.

One interesting phenomenon showed the effectiveness of the Kalman filter. At a distance, the vehicle becomes small: the bounding box shrinks due to the perspective effect and it becomes part of the adaptive background and simply disappears. See example in Figure 15. However the system continues to track it until it is predicted to leave the processing area and it was observed to 'reappear', *i.e.* captured in a subsequent frame as a separator object. This is a consequence of the background model, which consists of a number of Gaussians and a probability that any pixel, which has a colour close to that of the background, may fall into the background class in one frame, but then re-appear in a subsequent frame.

4.5 Transmission to host

Figures 5(d,e) shows a small image of a Red Light Runner successfully transferred from the PC at the intersection to a mobile phone. The latency was estimated to be only 0.36s - well below the time needed (~30 s) for an RLR to travel 500m to a remote monitoring station. The mobile phone images are clear enough to allow correct identification of the RLR when stopped. In fact, several RLR images can be transmitted before the RLR reaches the monitor station. Further, a full scene image showing the RLR, in the forbidden region while the light is red, can also be sent. There is also sufficient time that the variable and unpredictable latency in the public shared network is not likely to cause many messages to be lost. In practice, we observed that several lines of an image would be lost in transmission and a simple retry protocol for lost image lines was built into our software.

5. Conclusions

We described a system for detecting Red Light Runners (RLRs) which is simple, cost effective and very flexible using only a single camera attached to a computer and a link using the telephone network to a phone. The camera could be positioned only a small height above the ground, but this led to many occlusions being observed. However, our system completely tracked vehicles correctly in 83% of frames and then successfully transferred images of RLRs to a mobile phone using the mobile network 96% of the time. The key steps for each captured image were (a) adaptive background subtraction, (b) contouring the moving objects and forming a model for each one including a position, bounding box, image template, velocity and data for a Kalman filter. When occlusions were detected our software used a rule-based approach to resolve many merges (where one vehicle appeared in front of another) and splits (when a vehicle no longer occluded one behind it). The Kalman filter improved tracking: for example, we observed vehicles in the distance that were 'lost' in the adaptive background: the Kalman filter was able to track the vehicle when it returned from the image noise.

The simplicity and low cost of this system makes it particularly relevant to a developing country like Thailand and potentially many similar countries.

5.1 Future work

The challenge of this research is to determine whether it can be used to educate drivers to respect road rules in general – not only the RLRs. Assessment could start with determining RLR rates before the system was used and then in several intervals after it had been used for some time and its use became widely understood. The basic strategies, used to detect RLRs, could also be extended to discouraging, for example, poor lane changing behaviour and failure to stop at pedestrian crossings.

6. Acknowledgements

Channa Meng thanks the Princess Maha Chakri Sirindhorn's Education Projects for Cambodia for a scholarship to complete this work.

7. References

[1] WORLD HEALTH ORGANIZATION. Global Health Observatory (GHO) data: Road safety https://www.who.int/gho/road_safety/en/

[2] ECONOMIC CONDITIONS AND AUSTRALIAN AUTOMOBILE ASSOCIATION, Cost of Road Trauma in Australia, Australian Automobile Association, 2015.

[3] ROYAL THAI POLICE (2013), *loc. cit.* WHO and Bloomberg Initiative for Global Road Safety. Road Safety Institutional and Legal Assessment Thailand. http://www.searo.who.int/thailand/ areas/rs-legal-eng11.pdf, 2015..

[4] RETTING, R., WILLIAMS, A., GREENE, M. Red-Light Running and Sensible Countermeasures: Summary of Research Findings. *Transp. Res. Rec. J. Transp. Res. Board*, vol. 1640, p. 23–26, Jan. 1998.

[5] WORLD HEALTH ORGANIZATION, "Global status report on road safety," *Inj. Prev.*, p. 318, 2013.

[6] CHOUDHURY, S.K., SA, P. K., S. BAKSHI, S., B. MAJHI. An Evaluation of Background Subtraction for Object Detection Vis-a-Vis Mitigating Challenging Scenarios, *IEEE Access*, vol. 4, p. 6133–6150, 2016.

[7] BABAEE, M., DINH, D., RIGOLL, G. A deep convolutional neural network for video sequence background subtraction. Pattern Recognition, vol 76, p. 635-649, 2018.

[8] HARITAOGLU, L.,. HARWOOD, D., AND L. S. DAVIS, L. W4: Real-time surveillance of people and their activities. IEEE TPAMI, vol. 22(8), pp. 809–830, 2000.

[9] STAUFFER, C., GRIMSON, W. Adaptive background mixture models for real-time tracking. IEEE Conference on Computer Vision and Pattern Recognition, pp. 2246–2252, 1999.

[10] ZIVKOVIC, Z., VAN DER HEIJDEN, F. Efficient adaptive density estimation per image pixel for the task of background subtraction, Pattern Recognit. Lett., vol. 27(7), pp. 773–780, 2006.

[11] OpenCV: cv::BackgroundSubtractorKNN Class Reference." [Online]. Available: http://docs.opencv.org/trunk/db/d88/classcv_1_1BackgroundSubtractorKNN.html. [Accessed: 02-Sep-2017].

[12] XU, Y., DONG, J,. ZHANG, B., XU, D. Background modeling methods in video analysis: A review and comparative evaluation, CAAI Trans. Intel. Technology, 1, pp 43-60, 2016.

[13] PICCARDI, M., T JAN, T. Mean-shift background image modelling. Int Conf Image Processing, Singapore, 3399-3402, 2004

[14] ELGAMMAL, A., DURAISWAMI, R., DAVID HARWOOD, D., DAVIS, L. Background and foreground modeling using nonparametric kernel density estimation for visual surveillance, *Proceedings of the IEEE*, vol. 90(7), pp. 1151-1163, 2002.

[15] MITTAL. A., PARAGIOS, N. Motion-Based Background Subtraction using Adaptive Kernel Density Estimation. IEEE Conf Comp Vision Pat Recog, pp. 302-309, 2004.

[16] LO, B., VELASTIN, S., Automatic congestion detection system for underground platforms. *Proc. 2001 Intl Symp on Intelligent Multimedia, Video and Speech Processing*, Hong Kong, pp. 158-161, 2001.

[17] WANG, H., SUTER, D. A consensus-based method for tracking: Modelling background scenario and foreground appearance. Pattern Recognition, 40(3), pp. 1091-1105, 2007.

[18] EL BAF, F., BOUWMANS, T., VACHON, B. A Fuzzy approach for background subtraction. ICIP 2008: pp. 2648-2651, 2008.

[19] CULIBRK, D., MARQUES, O., SOCEK, D., KALVA, H. FURHT, B. Neural Network Approach to Background Modeling for Video Object Segmentation. *IEEE Trans Neural Networks*, vol. 18(6), pp. 1614-1627, 2007.

[20] YAO, J., ODOBEZ, J. Multi-layer background subtraction based on color and texture. IEEE Comp Vision and Pattern Recognition, pp 1-8, 2007.

[21] ZHOU, T TAO, D. GoDec: Randomized low-rank & sparse matrix decomposition in noisy case. Intl Conf on Machine Learning, pp 33-40, 2011.

[22] HARDAS, A., BADE D., WALI,V. Moving Object Detection using Background Subtraction, Shadow Removal and Post Processing, IJCA Proceedings on International Conference on Computer Technology, pp. 975–8887, 2015.

[23] SUBUDHI, B., GHOSH, S., SHIU, S., GHOSH, A. Statistical feature bag based background subtraction for local change detection. Inf. Sci. (NY)., vol. 366, pp. 31–47, 2016.

[24] SAJID, H. CHEUNG, S. Background subtraction for static & moving camera. ICIP 2015: 4530-4534, 2015.

[25] *ibid.* Universal Multimode Background Subtraction. IEEE Trans Image Processing, vol 26(7), 3249-3260, 2017.

[26] WANG, Y, JODOIN, P., PORIKLI, F., KONRAD, J., BENEZETH, Y., ISHWAR, P. CDnet 2014: An Expanded Change Detection Benchmark Datase. Proc IEEE Conf. Computer Vision and Pattern Recognition Workshops, pp 393-400, 2014

[27] JEEVA, S., SIVABALAKRISHNAN, M. Survey on background modeling and foreground detection for real time video surveillance. *Procedia Comput. Sci.*, vol. 50, pp. 566–571, 2015.

[28] HOFMANN, M., TIEFENBACHER, P., RIGOLL, G. Background Segmentation with Feedback: The Pixel-Based Adaptive Segmenter. IEEE Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 38-43, 2012

[29] MADDALENA L., PETROSINO, A. A Self-Organizing Approach to Background Subtraction for Visual Surveillance Applications. IEEE Trans Image Process., vol 17(7), pp 1168-1177, 2008.

[30] BARNICH, O., VAN DROOGENBROECK, M. ViBe: A universal background subtraction algorithm for video sequences. IEEE Trans. Image Process., vol. 20(6), pp. 1709–1724, 2011.

[31] LEE, J., PARK, M.. An Adaptive Background Subtraction Method Based on a Kernel Density Estimation. Sensors, vol 12, pp 12279-12300, 2012.

[32] SOOKSATRA, S. T. KONDO, T. Red traffic light detection using fast radial symmetry transform. 2014 11th Intl Conf Electrical Engineering/ Electronics, Computer, Telecom and Inf Tech, ECTI-CON 2014, 2014.

[33] GUO, H., GAO, Z., YIANG, X. JIANG, X. Modeling Pedestrian Violation Behavior at Signalized Crosswalks in China: A Hazards - Based Duration Approach. Traffic Injury Prevention, vol. 12, 96-103, 2011.

[34] DIAZ, M., CERRI, P., PIRLO, G., FERRER, M. A Survey on Traffic Light Detection. New Trends in Image Analysis and Processing - ICIAP 2015 Workshops, 201-208, 2015.

[35] YILMAZ, A., JAVED, O., SHAH, M. Object tracking: A survey. ACM Comput. Surv. 38 (4), 13, 2006.

[36] SMEULDERS, A. CHU, D., CUCCHIARA, R., CALDERARA, DEHGHAN, A., SHAH, M. Visual Tracking: An Experimental Survey. IEEE TPAMI, vol. 36, pp. 1442-1468, 2013.

[37] WU, Y., LIM, J., YANG, M.. (2015). Object Tracking Benchmark. IEEE TPAMI, vol. 37(9). pp.1834-1848, 2013. doi: 10.1109/TPAMI.2014.2388226.

[38] KRISTAN, M. *et al.*, The Visual Object Tracking VOT2015 Challenge Results. Visual Object Tracking Workshop IEEE Intl Conf. Computer Vision Workshop (ICCVW), 2015.

[39] NAM. H. HAN, B. Learning multi-domain convolutional neural networks for visual tracking. Proc IEEE Conf. Computer Vision Pattern Recognition, pp 4293-4302, 2016.

[40] KRISTAN, M. *et al.* The Visual Object Tracking (VOT2018) Challenge Results," ECCVW, pp. 1–27, 2018.

[41] LEE, B., LIEW, L., CHEAH, W., WANG, Y. Occlusion handling in videos object tracking. IOP Conf. Series Earth Env Science. vol. 18 (1), 2014.

[42] ZHAO S, ZHANG S, ZHANG L. Towards Occlusion Handling: Object Tracking With Background Estimation. IEEE Trans Cybern., vol. 48(7), pp. 2086-2100, 2018.

[43] MOTRO, M., GHOSH, J. Measurement-wise Occlusion in Multi-object Tracking, arXiv preprint arXiv:1805.08324. 2018.

[44] MILAN, A., LEAL-TAIXE, L., SCHINDLER, R., CREMERS, D., ROTH, S., REID, I. Multiple Object Tracking Benchmark. https://motchallenge.net, Accessed 23.10.2018.

[45] DOYLE, D., ALAN L. JENNINGS, A., BLACK, J. Optical flow background estimation for real-time pan/tilt camera object tracking. Measurement, vol. 48, pp 195–207, 2014.

[46] Kale, K., Pawar, S., Dhulekar, A. Moving Object Tracking using Optical Flow and Motion Vector Estimation. 4th Intl Conf. Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions), pp 1-6, 2015.

[47] OCCAM, W. Occam's Razor, https://en.wikipedia.org/ wiki/Occam's_razor. Accessed 7 Nov, 2018.

[48] SHIMADA, A., NAGAHARA, H., TANIGUCHI, R. Background modeling based on bidirectional analysis. Computer Vision Pattern Recognition, pp. 1979–1986, 2013.

[49] HAINES, T., XIANG, T, Background Subtraction with DirichletProcess Mixture Models. *IEEE Trans Pattern Analysis Machine Intelligence*, vol. 36(4), pp. 670-683, 2014. doi: 10.1109/TPAMI.2013.239.

[50] SEIDEL, F., HAGE, C., KLEINSTEUBER, M. pROST: a smoothed ℓ_p-norm robust online subspace tracking method for background subtraction in video. Machine Vision and Applications, vol. 25(5), pp. 1227--1240, 2014.

[51] LI, X., WANG, K., WANG, W., LI, Y. A Multiple Object Tracking Method using Kalman Filter. Inf. Autom., vol. 1(1), pp. 1862–1866, 2010.

[52] OPENCV: cv::KalmanFilter Class Reference. [Online]. Available: https://docs.opencv.org/3.4/ dd/d6a/classcv_1_1KalmanFilter.html. [Accessed: 07-Nov-2018].

[53] JENOPTICK AG. Flexible systems and services for Traffic Safety. 2016. [Online]. Available: https://www.jenoptik.com/products/traffic-safety-systems/combined-speed-redlight-enforcement-monitoring. [Accessed: 08-Nov-2018].

[54] SMARTVISION TECHNOLOGY CO. LTD. Red Light Camera (RLC). www.smartvisoncompany.com/ /redlightcamera.html [Accessed: 08-Nov-2018]

[55] KLUBSUWAN, K., KOODTALANG, W., MUNGSING, S. Traffic violation detection using multiple trajectories evaluation of vehicles. *Proc. - Int. Conf. Intell. Syst. Model. Simulation, ISMS*, vol. 5 (12), pp. 220–224, 2013.

[56] K. KIM, T. CHALIDABHONGSE, D. HARWOOD, AND L. DAVIS, Real-time foreground background segmentation using codebook model. Real Time Imaging, vol. 11(3), pp. 172–185, 2005

[57] MENG, C. Traffic Violation Detection. MEng thesis, Faculty of Engineering, Mahasarakham University, 2018.

[58] REDDY, V. SANDERSON, C., LOVELL, B. Improved foreground detection via block-based classifier cascade with probabilistic decision integration. IEEE Trans. Circuits Syst. Video Technol., vol. 23(1), pp. 83–93, 2013.

[59] WREN, R., AZARBAYEJANI, A., DARRELL, T., AND PENTLAND, A. Pfinder: Real-time tracking of the human body. IEEE TPAMI, vol. 19(7), pp. 780–785, 1997.

[60] KAEWTRAKULPONG, P. BOWDEN, R. An Improved Adaptive Background Mixture Model for real-time tracking with shadow detection. Video-Based Surveillance Systems, Springer, pp. 135–144, 2002.

[61] RODRIGUEZ. P. B. WOHLBERG, B. Fast principal component pursuit via alternating minimization. Proc. IEEE Int. Conf. Image Process. pp. 69–73, Sep 2013.

Biography

Channa Meng was born in Cambodia in 1991. She graduated with a B.A. from Mahasarakham University in 2013 and expects to complete an ME from the same university in 2018. She currently works as a software engineer with Inet-logistics in Khon Kaen, Thailand.

John Morris was born in Newcastle, Australia and graduated with a PhD from the University of Sydney. He has had teaching and research positions in Canada, Japan, Australia, USA, Korea, China, New Zealand and now Thailand. He specializes in real-time stereo vision (hardware and software), but recently, his work has focused on imaging everything from drying chilli and household plants to suicidal motorists who cannot see 'red'.

Nattawoot Suwannata was born in 1971 at Maha Sarakham, Thailand. He graduated with a PhD in Electrical Engineering from Khon Kaen University, Thailand. He currently works as assistant professor in Electrical Engineering at Faculty of Engineering, Mahasarakham University, Thailand. He specializes in digital signal processing and special instrument design.