Traffic Situation Visualization based on Video Composition

Cheng-You Hsieh, Yu-Shuen Wang
National Chiao Tung University

Abstract

Vehicle detectors (VDs) are usually distributed in a road network to detect macroscopic traffic situations. These detectors provide global information such as vehicle flows, average speed, and road occupancy. Given that the collected statistic data are difficult for citizens to interpret, we visualize the data by providing users with realistic traffic videos. To achieve this aim, our system collects the surveillance videos and VD data that represent the traffic situation of a position. It then builds the connection between these two types of data. Considering the distribution of VDs is much denser than that of surveillance cameras, for those road segments with a VD but without a surveillance camera, one can utilize our system to synthesize videos for visually depicting the traffic situations over there. That is, we estimate vehicle flows from a video and apply the regression model to build the mapping between the flows and VD data. After that, given by a VD dataset, our system retrieves videos that match the VD data and seamlessly composes them to synthesize a traffic video. The evaluations and the experimental results demonstrate the feasibility of our system.

Keywords:
vehicle detection, visualization, surveillance video, video synthesis

1. Introduction

Vehicle detectors are usually distributed in-road to acquire spatiotemporal traffic statistic. The detectors record the number and the average speed of vehicles that pass within a time span. These data are important for city planners and traffic controllers [1] because the data depict overview and details of traffic situations over time. However, interpreting traffic statistic demands expertise and is not suitable for general populations. For example, the same driving speed in a countryside and an urban city could have very different experience because of light and heavy traffic densities. Providing users with a number of vehicles that pass is also unintuitive because it depends on the number of lanes on the road. Moreover, a light traffic flow may indicate few vehicles on the road or a serious traffic jam, which easily induces misleading. Accordingly, providing an interface for general users to realize traffic situations is essential.

Since transmitting videos captured from road surveillance cameras consumes expensive load, simulation techniques are presented for traffic visualization. The methods estimate velocity and density fields over the road network, followed by applying an agent-based traffic simulator to create 3D animations. They enjoy the visualization from various viewpoints and even allow users to observe traffic from a driver's perspective. However, a simulator cannot always realize the real traffic flows because driving behaviors are often different in countries and regions (Figure 1), not to mention other conditions such as weather, rush hours, and holidays. To overcome this problem, we present an example-based system that visualizes traffic situations by synthesizing road surveillance videos.

Our goal is to visualize statistic VD data using real world materials. Specifically, we find a road segment where VD and surveillance camera are both available and extract the relations between them. For the place with a VD but without cameras, we synthesize a streaming video by composing the video clips in our database to visualize its traffic situations over time. The main advantage of this framework is generality. While a simulation technique is insufficient to create an animation that satisfies all driving behaviors, our system does not have this problem because they are already provided by road surveillance videos. Note that this framework also consumes light data transmission load because example videos are collected in advance. Only statistic VD data are transmitted when the traffic flows of a road segment are visualized.

The problems of traffic visualization in our framework are video retrieval and seamless composition. Considering the traffic situation of a video is unclear, to obtain the information, we seed particles on the video and track their motions. The number and speed of these particles that go outside the video coordinate or gather at the vanish point are recorded. After that, we compute a regression model to map the VD data and the particle flows so as to retrieve proper videos for composition when a macroscopic traffic statistic of another place is given. To prevent artifacts of video transition, we overlap consecutive videos by a number of frames and compute a surface that passes through pixels with the least distortion. Specifically, pixels on this surface should have small color variations and zero motions to avoid discontinuity artifacts and suddenly appearing or disappearing vehicles. We also apply the Poisson blending to smooth the difference of illumination conditions in videos for achieving high visual quality.

Our method synthesizes a streaming video to visualize the traffic situations of a place over time. This example-based framework can realize the traffic flows consistent with the sparse VD data at different regions. The main advantage of this framework is generality, which is able to visualize different traffic situations derived from driving behaviors, weathers, etc. We show the experimental results in Figures 6, 7, and in the accompanying video to demonstrate the feasibility of our technique.

2. Related work

Traffic visualization. Visualizing traffic information is essential to city planners and traffic controllers. Many on-line services such as SigAlert and Google Maps depict the conditions of a traffic network by colorization. The abstracted visual means look clean and neat but lack details for understanding dynamic vehicle flows. Therefore, Walton et al. [2] projected live traffic videos onto maps to provide such information. However, the method consumes heavy transmission load when too many traffic videos are displayed at a time. Another approach to achieve the aim is traffic simulation [3, 4, 5, 6, 7, 8, 9, 10]. The methods first estimate the full traffic state based on the sparse VD sensing data, followed by simulating the dynamics of all individual vehicles to create animations. These simulation techniques enjoy various viewpoints when visualizing traffic situations. But they are insufficient to simulate all kinds of vehicle kinematics and dynamics due to unknown driving behaviors in different regions. As a result, we attempt to visualize traffic situations by composing road surveillance videos. This example-based approach achieves more accurate visualization because various vehicle kinematics and dynamics are already provided by road surveillance videos.

Video textures. Video textures [11, 12] are commonly used to generate an infinite length video based on a finite video clip. The technique changes the order of video frames that are unnoticeable to viewers and plays the video forever. Considering that two video frames with very similar content is difficult to obtain, to prevent discontinuity artifacts, Kwatra et al. [13] overlapped a number of frames and computed a surface to transit one video to another seamlessly. Besides the videos with a fixed viewpoint, Agarwala et al. [14] captured videos using a panning camera and synthesized panoramic video textures to enhance visual experience. Later, Couture et al. [15] extended the panoramic video textures to a stereo version. Our streaming traffic video synthesis is inspired by these works. However, all previous methods consider local color gradients during composition. Foreground objects with large motions may suddenly appear or disappear when videos are composed by these methods.

Poisson blending. Many image and video editing techniques apply Poisson blending to smooth boundary artifacts when visual media are composed. This operation performs in the gradient field of an image and has attracted significant attention in research works [16, 17, 18, 19]. Our system also applies this operation to smooth discontinuity artifacts when transiting one video to another. Given that Poisson blending requires solving a large linear system, which consumes expensive computational cost, there were also techniques presented to improve its performance [20, 21, 22, 23].

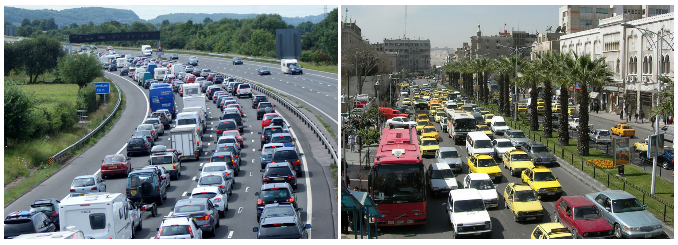
Figure 1: Left and right show traffic jams occur at different regions. Left: all vehicles are in lane and no drivers attempt to violate traffic rules. Right: vehicles cross lanes wantonly and make the traffic even worse. Simulation methods are difficult to reconstruct all types of traffic flows.

3. Algorithm

Our goal is to visualize traffic statistic using videos captured by road surveillance cameras. To achieve the aim, the first step is to build the relations between traffic statistic and videos. We search for places where VD and surveillance camera are both available, and then compute a regression model to map the detected number of vehicles and average speed to the particle flows in a video. Specifically, our system cuts a streaming road surveillance video into short clips, with each clip containing one minute, because each VD returns a traffic statistic every one minute. Because of different natures of traffic statistic and videos, we extract vehicle flows from the video and train a regression model to link these two data. This objective is achieved by computing optical flows from each video, which roughly represent the traffic flows because surveillance cameras have fixed viewpoints and most moving objects can be considered vehicles. Accordingly, given by traffic statistic at another place, our system is able to retrieve proper videos from the database and compose them together for traffic situation visualization.

Our system composes one-minute videos to a streaming video for visualizing the traffic situation of a place with a VD but without a surveillance camera. To prevent discontinuity artifacts, we reserve a number of frames at the two ends of each one-minute video. We then overlap the reserved frames and determine the smoothest surface to transit one video to another. The pixels on this surface have not only similar colors but also small motions, which can reduce discontinuity artifacts and prevent suddenly appearing or missing vehicles. Considering the illumination conditions could be different, the hard transition of videos often results in discontinuity artifacts, we smooth pixel colors to achieve a visually pleasing result.

3.1. Data acquisition

We obtain traffic statistic and road surveillance videos from Taiwan Area National Freeway Bureau. The traffic statistic depicts the passed number of large and small vehicles and the average speed of these vehicles within each minute. The road surveillance videos show the traffic situation of an area, where

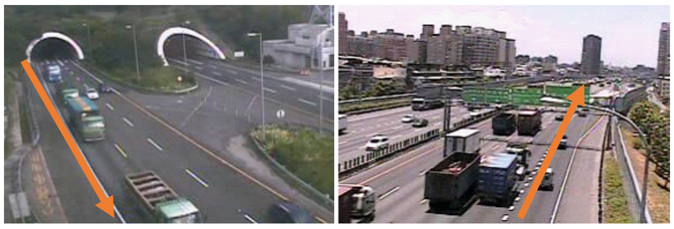

Figure 2: Vehicles may go outside the video coordinate or gather at a vanish point, depends on the flow direction.

its location and the captured time are attached. Although the original surveillance videos are of high quality, our obtained versions are of low resolution (352 × 198) and low frame rates (7 fps) due to some privacy issues. But the presented technique could be directly used for proceeding high quality surveillance videos if necessary.

3.2. Data mapping

To visualize the traffic statistic of a road segment, our system has to retrieve proper videos from the database for composition. Given that the videos in our database are attached with traffic statistic detected by a VD, when the traffic statistic of another place is obtained, the simplest way to retrieve proper videos is measuring the similarity of two traffic statistic. However, current VD systems are not perfect and the obtained traffic data are easily disturbed by noise. This naïve matching totally relies on VD statistic and may retrieve inadequate videos. Another way to achieve the aim could be counting vehicles by applying object detections, but the methods are slow and often miss vehicles that are not in the training dataset. Instead, we compute traffic flows and build a regression model to establish the relations between traffic statistic and videos.

We compute optical flows that represent pixel motions in adjacent video frames. To obtain vehicle flows, we uniformly seed particles on a video, where the initial distance is set to 10 pixels, and track their motions. Since particles on a vehicle would move, we re-seed particles whenever the region has no particles nearby (i.e., 10 pixels in our implementation) as video frames are updated. Observing that vehicle flows in a video may have different directions (Figure 2), when these particles move, they would eventually go either outside the video coordinate or toward the vanish point. We let users manually specify the direction that match the VD because this information can only be obtained by checking the camera orientation. Our system then counts the number of these particles and computes their average speed to represent the traffic situation. Apparently, detecting particles that go outside the video coordinate is straightforward. We point out that obtaining particles gathering at the vanish point is also simple. In our implementation, we find a circular region where particles in it are very slow or even still but have fast motions in the beginning to achieve the aim.

We represent a video with its corresponding traffic statistic using a high dimensional point $\mathbf{v} = (f_s, f_n, n_s, n_\ell, s)$, where f_s and f_n indicate the average speed and the number of particles that go outside the video or gather at the vanish point, n_s and n_ℓ are the numbers of small and large vehicles, respectively, and s is the average vehicle speed detected by a VD. We observe that f_s is proportional to s; and f_n is proportional to n_s and n_ℓ. While representing a pair of surveillance video and traffic statistic in a high dimensional space, we can determine a hypersurface that approximates the distribution of $\mathbf{v}$, which is formulated as

$$\begin{aligned} & u_1 f_s^2 + u_2 f_n^2 + u_3 n_s^2 + u_4 n_\ell^2 + u_5 s^2 \\ & + u_6 f_s + u_7 f_n + u_8 n_s + u_9 n_\ell + u_{10} s + u_{11} = 0. \end{aligned} \quad (1)$$

By letting $u_1 = 1$ and instituting the variables (f_s, f_n, n_s, n_ℓ, s) obtained from the database, we solve for the coefficients u_1 to u_{11} in a least squares sense. As mentioned earlier that VDs are not perfect, after we obtain the hypersurface, we compute the distance of each high dimensional point to the surface. The points where their distances larger than the standard deviation are removed. We apply this regression model and the remaining videos to synthesize traffic situations of a place that has a VD but no cameras. That is, given by a VD statistic (n_s, n_ℓ, s), we retrieve a video with the flow (f_s, f_n), where the composed high dimensional point is closest to the hypersurface.

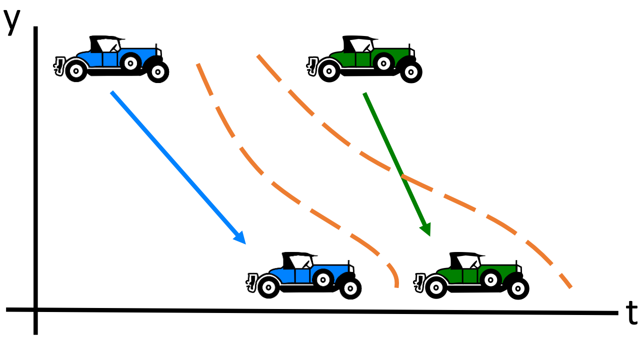

Figure 3: Suppose vehicles are moving down (y coordinate) as video frames are updated (t coordinate). We show their motions using blue and green arrows, respectively. The potential transition seams, which are rendered in orange, would partition the video into left and right segments. As illustrated, this transition seam should avoid passing through arrows to maintain vehicles appearing in only one segment. Otherwise, the (green) vehicle would suddenly appear at the point where the seam and the arrow intersect.

3.3. Traffic video synthesis

We compose video clips to a streaming video for visualizing the traffic situation of a road segment. To reduce discontinuity artifacts, we overlap the consecutive videos by 60 frames and retrieve a surface with the least distortion to transit one video to another. We then smooth pixel colors on the surface to further improve visual quality. Here we show the composition of two video clips. One can always consider the previously composed result as the first clip and the new video as the second clip to generate a streaming video.

3.3.1. Seamless surface for transition

We compute the surface in the overlapping frames for transiting one video to another. This surface should pass through the region with small color gradients to reduce discontinuity artifacts. Moreover, it has to avoid cutting through vehicle flows

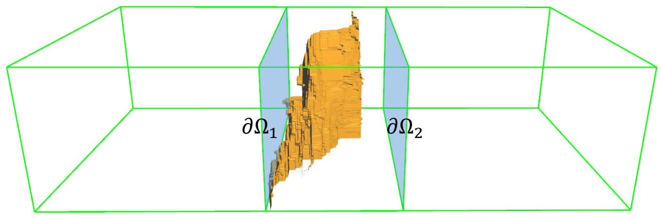

Figure 4: Our system overlaps consecutive videos Ω_1 and Ω_2, and computes the surface with the least distortion for seamless transition. The two boundaries of the overlapping frames are $\partial\Omega_1$ and $\partial\Omega_2$, respectively.

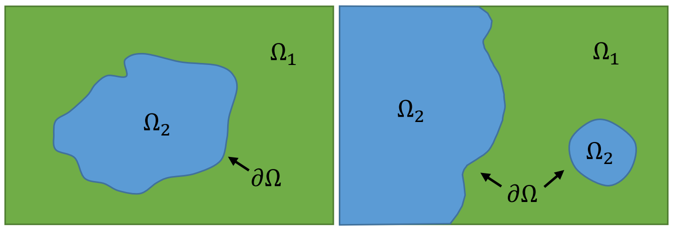

Figure 5: Left and right show different compositions of a frame. Pixels may come from Ω_1 or Ω_2. The Poisson blending can handle both situations because only color gradients in Ω_1, Ω_2, and colors on $\partial\Omega$ are considered.

to prevent suddenly appearing or missing vehicles. We show a 2D example in Figure 3 for illustration. In other words, the pixels on this surface should have similar colors and zero motions. Denote by $\mathbf{E}$ the set of edges indicating pixel adjacency, by Ω_1 and Ω_2 the overlapping pixels at the first and the second videos, respectively. Our goal is to determine a label $x = \{1, 2\}$ for each pixel p in the overlapping area by minimizing an objective function:

$$\sum_i D_d(p_i) + \sum_{\{i,j\}\in\mathbf{E}} D_s(p_i, p_j) + \sum_{\{i,j\}\in\mathbf{E}} D_v(p_i, p_j). \tag{2}$$

Specifically,

$$D_d(p_i) = \begin{cases} \omega & \text{if } p_i \in \partial\Omega_1 \wedge x_i = 2 \\ & \text{or } p_i \in \partial\Omega_2 \wedge x_i = 1 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

is a data term to constrain pixels at the first boundary $\partial\Omega_1$ belonging to the first video and at the second boundary $\partial\Omega_2$ belonging to the second video, respectively.

$$D_s(p_i, p_j) = |f_i^1 - f_i^2| + |f_j^1 - f_j^2| \tag{4}$$

is a smoothness term to achieve seamless video transition, where f_i^t is the color of pixel $p_i^t \in \Omega_t$. In other words, the surface should pass through the region that has the least color variation. Finally,

$$D_v(p_i, p_j) = \begin{cases} \omega & \text{if } |\mathrm{v}_i| > \gamma \text{ or } |\mathrm{v}_j| > \gamma \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

is a constraint to prevent the surface passing through vehicles, where v_i is the magnitude of optical flow at p_i. This constraint is important even though the region has small color variation because humans are very sensitive to dynamic and meaningful objects in a video. We set $\omega = 10000$ and $\gamma = 2$ in our implementation. Note that $\gamma \neq 0$ is due to the non-perfect detected optical flows. Clearly, the minimization of Equation 2 is a labeling problem. We apply the graph cut algorithm [24] to obtain the solution.

3.3.2. Illumination smoothing

Visual artifacts are inevitable when videos captured under different illumination conditions are composed together. To handle this problem, we search for the surveillance videos that have similar physical time to make sure their illuminations are not significantly different. We also estimate the illumination difference between consecutive clips and blend the luminance of each pixel smoothly to ease the artifacts. Specifically, we compute a background image G by averaging the overlapping frames and compare G to video frames in Ω_1 and Ω_2 to obtain the illumination changes. The formal expression is given:

$$I_d = \frac{1}{\Delta_1} \sum_{F_i\in\Omega_1} (G - F_i)_\sigma - \frac{1}{\Delta_2} \sum_{F_i\in\Omega_2} (G - F_i)_\sigma, \tag{6}$$

where F is a video frame, $(G-F)_\sigma$ is a scaler function that sums the difference of pixel luminance smaller than the standard deviation σ, and Δ is used for normalization. We point out that our system discards the pixels with large luminance deviations in Equation 6 because they often represent foreground objects.

Observing that hard transition of videos often induce discontinuity artifacts, even though they are sufficiently small, we further apply Poisson blending to smooth the boundary. Because the illumination condition of a video usually is stable, this smoothing process can be achieved in each video frame individually to save computational cost. As illustrated in Figure 5, Ω_1 and Ω_2 are the first and the second videos, respectively, and $\partial\Omega$ is the surface boundary in a frame. Our goal is to update pixel colors in Ω_2, where their color gradients are possibly retained and the constrained pixels on $\partial\Omega$ are untouched. That is, for each pixel $p_i \in \Omega_2$, we minimize

$$|N_i| f_i - \sum_{j\in N_i} f_j = \sum_{j\in N_i} \mathrm{v}_{ij}, \tag{7}$$

where N_i and f_i denote the neighbors and color of p_i, respectively, and $\mathrm{v}_{ij} = f_i - f_j$ is the color gradient. Note that we minimize the objective function

$$|N_i| f_i - \sum_{j\in N_i\wedge\Omega_2} f_j = \sum_{j\in N_i\wedge\partial\Omega} \hat{f}_j + \sum_{j\in N_i} \mathbf{v}_{ij} \tag{8}$$

for pixels where their neighbors are on $\partial\Omega$ to handle the boundary constraint, and $\hat{f}$ is a known variable obtained from Ω_1. We formulate the above equations into a linear system and solve for the pixel colors in Ω_2 in a least squares sense. Although $\partial\Omega$ could have various shapes, as illustrated in Figure 5, the Poisson blending works well in all cases. We refer readers to [16] for details.

4. Results and discussions

We have implemented the presented approach using C and run the program on a desktop PC with Core i7 3.0 GHz CPU

Retrieval by measuring VD statistics						Retrieval by using our regression model					
VD statistic		Ground truth		Deviation to the query		VD statistic		Ground truth		Deviation to the query	
# large	# small	# large	# small	# large	# small	# large	# small	# large	# small	# large	# small
19	78	14	71	6	9	13	68	21	82	1	2
18	81	17	123	3	43	24	39	21	84	1	4
18	81	8	139	12	59	26	44	12	80	8	0
17	79	15	115	5	35	15	41	17	79	3	1
17	79	17	122	3	42	15	41	11	84	9	4
20	75	13	99	7	19	23	61	17	83	3	3
16	79	4	106	16	26	13	40	24	79	4	1
20	74	17	89	3	9	23	41	13	90	7	10
20	74	24	107	4	27	20	38	16	84	4	4
18	76	15	98	5	18	16	86	11	90	9	10

Table 1: We retrieve traffic videos that have 20 large vehicles and 80 small vehicles passing through a place by measuring VD statistic and by using our regression model, respectively. Ten one-minute videos that best fit this query are listed. We also manually count the numbers of vehicles in the videos and denote the results as ground truth. clearly, the videos retrieved by using our method can better depict real traffic situations because the true numbers of large and small vehicles are closer to the query.

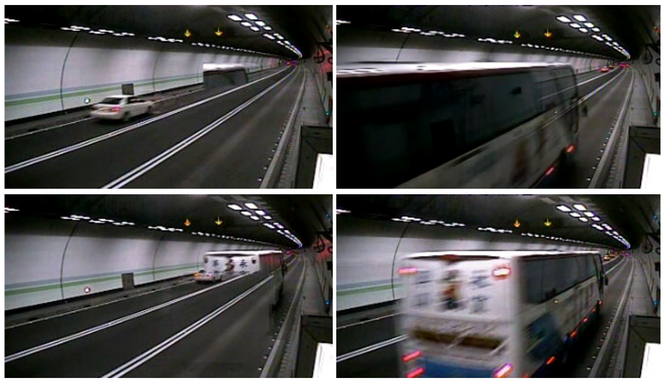
Figure 6: Left and right show the comparison without and with the flow constraint when transiting one video to another. Our system retains the persistence of each vehicle whenever it appears in the video to prevent visual artifacts.

and GeForce GTX 550 GPU. The GPU based optical flow [25, 26] method is used to extract vehicle flows in a road surveillance video. We also applied the graph cut library [24] to compute the surface that can seamlessly transit one video to another. Finally, we apply the conjugate gradient method to solve the Poisson blending problem. This process is efficient because the hard transited frames can be set as initial guess in the system to speed up the convergence. Generally, extracting flows in each video frame takes about 0.04 seconds; computing the surface to transit video clips takes 39 seconds; and smoothing transition boundaries in each frame takes about 0.2 seconds. Although the database of collected road surveillance videos is large, we point out that retrieving videos for composition is efficient. The performance is not surprising because all videos can be precomputed and only f_s and f_n that indicate average speed and number of particles will be used in this step.

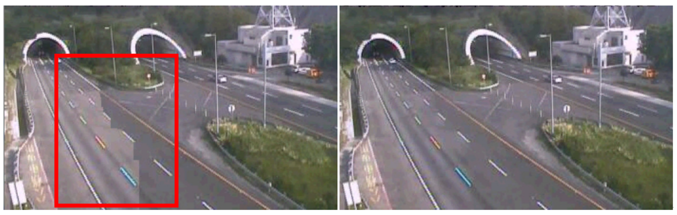
Figure 7: Left and right show the comparison without and with Poisson blending. (Left) Discontinuity artifacts occur when videos with different illuminations are composed, as highlighted by the red rectangle. (Right) The artifacts are removed by our smoothing.

4.1. Evaluations

We evaluate whether the retrieved videos well depict traffic flows. Suppose the goal is visualize the traffic situation, in which 20 large vehicles and 80 small vehicles passed a road segment in one minute. We retrieve ten videos that best fit this query by using our method and by measuring VD statistic, respectively. We then manually count the true numbers of large and small vehicles in the videos. Table 1 shows the results. As mentioned earlier, current VDs are not perfect and the detected traffic data often contain noise. Retrieving videos simply by measuring VD statistic potentially obtain improper results. In contrast, the videos retrieved by using our method can better depict real traffic situations because the true numbers of vehicles are more consistent and closer to the query. These results point out that our framework is robust against noise in VD statistic, thanks to the consideration of optical flows in traffic videos.

In addition to retrieving videos that can depict traffic situations, we tested our composition method on a variety of traffic videos. The chosen videos involve different traffic situations and weathers. Figure 6 and our accompanying video show the comparison of video transition with and without the preservation of traffic flows (Equation 5). The result in the left indicates that the surface passes through the bus because the overall

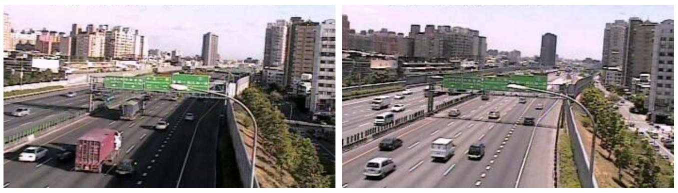

Figure 8: Left and right show different illumination conditions of a road segment. Simply smoothing the artifacts by blending luminance is not sufficient.

cost is small, although the color difference between the bus and the road is noticeable. Hence, buses suddenly appear/disappear in the composed video and the visual artifacts occur. By giving large penalties to the pixels with traffic flows, we successfully prevent this temporal artifact, as shown in Figure 6 right. We also show the comparison with and without the illumination smoothing in Figure 7 to demonstrate how the discontinuity artifacts are removed in our system. We recommend users to watch the results shown in our accompanying video because dynamic vehicle flows are difficult to appreciate in still images.

4.2. Qualitative comparisons to simulation methods

Both simulation techniques and our method aim to visualize macroscopic VD data by reconstructing traffic flows. These two approaches work on different directions and have totally different natures. The main advantage of our approach is an example based visualization, where driving behaviors that affected by road conditions, weather, and law-abiding are recorded by the videos in the database. Accordingly, our composed streaming video is able to reflect these factors during visualization. However, its main disadvantage is a fixed and unchanged viewpoint. Users can only get insight to traffic flows by watching a video instead of interacting with the flows. The simulation methods are opposite to ours. They can handle traffic-related phenomena only with assumptions. But the methods allow users to examine data from different perspectives. Users can observe traffic flows from a bird's eye view and from a driver's view to obtain global and local details, respectively. We therefore claim that our system is not presented to replace the simulation methods but to provide another strategy for traffic visualization. These two approaches are complementary and should work together to realize traffic situations of macroscopic VD data.

4.3. Limitations

Our system applies road surveillance videos of one place to depict the traffic situations of another place. A problem of this framework is of different background scene in the composed video, which may induce perceptual misleading. The simplest way to solve this problem is informing users that the video is used for visualization and asking them to neglect the background issue. Another approach is to take a picture of the place, in which the traffic situations over there will be visualized, and replace the background with content provided by the picture [27]. The achievement of this objective should consider the consistent of viewpoints and seamless blending. We plan to improve this example based traffic situation visualization system in the near future.

Another limitation of our system is large illumination changes (Figure 8). Although we have striven to smooth pixel luminance, transiting videos that have very different shadow directions still produces unusual results. Our current strategy to prevent this problem is retrieving videos with similar physical time. However, it demands a large amount of videos stored in the database to realize various kinds of traffic situations. Advanced illumination process could be used to reduce the size of a database.

5. Conclusions

We have presented a system to visualize traffic situations by composing videos in a database. Given by macroscopic VD data that depicts the vehicles passing a road segment, our system retrieves proper videos that match the data for composition. It then reduces visual artifacts by computing a surface for seamless video transition and by solving a Poisson equation for illumination smoothing. Although the visualized traffic flows are of a fixed viewpoint, by collecting road surveillance videos with different conditions, our visualization can fit various driving behaviors that a simulation method cannot achieve. Accordingly, it can cooperate with simulation techniques for users to better understand traffic situations.

Acknowledgements

We thank the anonymous reviewers for their constructive comments. We are also grateful to Taiwan Area National Freeway Bureau for kindly providing us the dataset, and flickr users (Travel Aficionado and Sam Saunders) for sharing pictures with us. This work was supported in part by the Ministry of Science and Technology, Taiwan (102-2221-E-009 -083 -MY3 and 101-2628-E-009 -020 -MY3).

References

[1] D. Wilkie, J. P. van den Berg, M. C. Lin, D. Manocha, Self-aware traffic route planning, in: Association for the Advancement of Artificial Intelligence, 2011, pp. 1521–1527.

[2] S. Walton, M. Chen, D. Ebert, Livelayer - live traffic projection onto maps, in: Eurographics 2011 - Posters, 2011, pp. 37–38.

[3] J. van den Berg, J. Sewall, M. Lin, D. Manocha, Virtualized traffic: Reconstructing traffic flows from discrete spatio-temporal data, 2014 IEEE Virtual Reality 0 (2009) 183–190.

[4] J. Sewall, D. Wilkie, P. Merrell, M. Lin, Continuum traffic simulation 29 (2) (2010) 439–448.

[5] J. Sewall, J. P. van den Berg, M. C. Lin, D. Manocha, Virtualized traffic: Reconstructing traffic flows from discrete spatiotemporal data., IEEE Trans. Vis. Comput. Graph. 17 (1) (2011) 26–37.

[6] J. Sewall, D. Wilkie, M. C. Lin, Interactive hybrid simulation of large-scale traffic, ACM Transaction on Graphics 30 (6).

[7] J. Shen, X. Jin, Detailed traffic animation for urban road networks, Graph. Models 74 (5) (2012) 265–282.

[8] D. Wilkie, J. Sewall, M. Lin, Flow reconstruction for data-driven traffic animation, ACM Trans. Graph. 32 (4) (2013) 89:1–89:10.

[9] Q. Chao, J. Shen, X. Jin, Video-based personalized traffic learning, Graph. Models 75 (6) (2013) 305–317.

[10] Q. Chao, Z. Deng, X. Jin, Vehicle-pedestrian interaction for mixed traffic simulation, Computer Animation and Virtual Worlds 26 (3-4) (2015) 405–412.

[11] A. Schödl, R. Szeliski, D. H. Salesin, I. Essa, Video textures, in: Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH), 2000, pp. 489–498.

[12] A. Schödl, I. A. Essa, Controlled animation of video sprites, in: ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2002, pp. 121–127.

[13] V. Kwatra, A. Schödl, I. Essa, G. Turk, A. Bobick, Graphcut textures: Image and video synthesis using graph cuts, ACM Trans. Graph. 22 (3) (2003) 277–286.

[14] A. Agarwala, K. C. Zheng, C. Pal, M. Agrawala, M. F. Cohen, B. Curless, D. Salesin, R. Szeliski, Panoramic video textures., ACM Trans. Graph. 24 (3) (2005) 821–827.

[15] V. Couture, M. S. Langer, S. Roy, Panoramic stereo video textures, in: International Conference on Computer Vision, 2011, pp. 1251–1258.

[16] P. Pérez, M. Gangnet, A. Blake, Poisson image editing, ACM Trans. Graph. 22 (3) (2003) 313–318.

[17] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, M. Cohen, Interactive digital photomontage, ACM Trans. Graph. 23 (3) (2004) 294–302.

[18] H. Wang, R. Raskar, N. Ahuja, Seamless video editing, in: International Conference on Pattern Recognition, 2004, pp. 858–861.

[19] J. Jia, J. Sun, C.-K. Tang, H.-Y. Shum, Drag-and-drop pasting, ACM Trans. Graph. 25 (3) (2006) 631–637.

[20] R. Szeliski, Locally adapted hierarchical basis preconditioning, ACM Trans. Graph. 25 (3) (2006) 1135–1143.

[21] A. Agarwala, Efficient gradient-domain compositing using quadtrees, ACM Trans. Graph. 26 (3).

[22] M. Kazhdan, H. Hoppe, Streaming multigrid for gradient-domain operations on large images, ACM Trans. Graph. 27 (3) (2008) 21:1–21:10.

[23] Z. Farbman, G. Hoffer, Y. Lipman, D. Cohen-Or, D. Lischinski, Coordinates for instant image cloning, ACM Trans. Graph. 28 (3) (2009) 67:1–67:9.

[24] O. Jamriška, D. Sýkora, A. Hornung, Cache-efficient graph cuts on structured grids, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 3673–3680.

[25] M. Werlberger, W. Trobin, T. Pock, A. Wedel, D. Cremers, H. Bischof, Anisotropic huber-l1 optical flow, in: Proceedings of the British Machine Vision Conference, London, UK, 2009, pp. 1–11.

[26] M. Werlberger, T. Pock, H. Bischof, Motion estimation with non-local total variation regularization, in: IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 2010, pp. 2464–2471.

[27] M. Flagg, J. Rehg, Video-based crowd synthesis, IEEE Transactions on Visualization and Computer Graphics 19 (11) (2013) 1935–1947.