An Intent-Based Automated Traffic Light for Pedestrians

Christian Ertler Horst Possegger Michael Opitz Horst Bischof
Institute of Computer Graphics and Vision, Graz University of Technology, Austria
{christian.ertler,possegger,michael.opitz,bischof}@icg.tugraz.at

Abstract

We propose a fully automated, vision-based traffic light for pedestrians. Traditional industrial solutions only report people standing in a constrained waiting zone near the crosswalk. However, reporting only people below the traffic light does not allow for efficient traffic scheduling. For example, some pedestrians do not want to cross the street and walk past the traffic light, or just wait for another person to arrive. In contrast, our system leverages intent prediction to estimate which pedestrians are actually going to cross the road by analyzing both short-term and long-term trajectory cues. In this way, we can decrease the waiting times and pave the road for optimal and adaptive traffic light scheduling. We conduct a long-term evaluation in a European capital that proves the applicability and reliability of our system and demonstrates that it is not only able to replace existing push-button solutions but also yields additional information that can be used to further optimize traffic light scheduling.

1. Introduction

Automated traffic light controllers are widely used to optimize traffic flow and adapt the scheduling of traffic lights based on traffic density. Common methods include skipping of green phases for turning lanes or crosswalks if no vehicle or pedestrian, respectively, is waiting. However, while vehicles waiting on a specific lane can be easily detected by various sensors (*e.g.* inductive loops, radar) which scan only a limited local area, pedestrians often need to manually interact with the system by pressing a push-button. In many cases, pedestrians do not realize that manual interaction is needed; after a long time of waiting, they tend to cross the road prematurely, putting themselves and other traffic participants in danger. Others leave after pushing the button, triggering a green phase which unnecessarily slows down motorists. These are just two exemplary scenarios where a fully automated traffic light controller for pedestrians can help to improve the safety and efficiency of traffic. Further, a system that is able to report information about the density of pedestrians heading towards the crosswalk can be very useful to optimize traffic light scheduling. For example, by allowing larger crowds more time to safely clear the road.

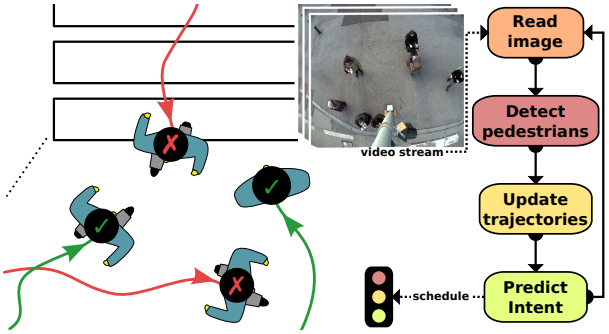

Figure 1. **Left:** Illustration of a typical crosswalk situation (top view) with people walking around and waiting near the crosswalk; pedestrians who want to cross are shown with tick marks (✓); cross marks (✗) indicate a different destination. **Right:** Program flowchart of the proposed system.

Traffic light control poses multiple challenges for a vision-based system. Since it is an outdoor application, our system must robustly handle different illumination conditions during day and night and must work in all weather conditions all year round. Further, it must be real time capable on rather limited hardware certified for continuous industrial operation; this usually excludes state of the art GPUs which are needed for fast inference of CNN models, especially in the field of object detection.

In this work, we propose an automated, intent-based traffic light control system that does not only recognize the presence of pedestrians but also their intents: *Where is the person heading to and is she planning to cross the road?* In this way, the system is able to report the total amount of pedestrians, which actually intend to cross at any time – without the need for manual interaction. Moreover, by predicting their intent, our system can, on the one hand, initiate the traffic light controller in advance and, on the other hand, prevent traffic flow disruptions by not reporting pedestrians which are just passing by. Figure 1 (left)

We gratefully acknowledge the support of NVIDIA Corporation with the donation of GPUs used for this research. This work was partially supported by the Austrian Research Promotion Agency (FFG) under the project *DGT - Dynamic Ground Truth* (860820).

978-1-5386-9294-3/18/$31.00 ©2018 IEEE

illustrates a typical situation with people heading towards, coming from the other side, and going past the crosswalk. The only input to our system is a video stream of a conventional RGB camera mounted on the traffic light pole observing the area in front of the crosswalk. The proposed system consists of three components: (1) a fast *pedestrian detector*; (2) a *tracking-by-detection approach* to maintain trajectories; (3) an *intent prediction model* to infer the most likely destination of each pedestrian depending on her past motion in combination with prior knowledge automatically inferred from previously observed pedestrian movements.

We show the effectiveness of the proposed system in a challenging long-term experiment with a prototype installed at a real crosswalk in a major European city; operating over the course of a month; with an average number of more than 3,000 pedestrians crossing per day. Our evaluation shows that our system can not only reliably replace the push-button in challenging outdoor situations (*e.g.* day/night, rain, snow), but is also able to report almost all pedestrians before they even come close to the push-button. We furthermore conduct an ablation study to investigate the influence of our model components.

2. Related work

Pedestrian detection is a well-studied task in computer vision and crucial for our application. Traditional detectors (*e.g.* [4, 7]) operate on handcrafted features and are either applied in a sliding-window manner or rely on precomputed object proposals (*e.g.* [28]). Recent detectors are driven by the success of CNNs (*e.g.* [17, 24, 25]). These are trained on large-scale datasets resulting in better generalization capabilities. As we need a robust and reliable detector to handle our challenging outdoor setting, we make use of such a CNN-based detector. CNNs are typically designed for fast inference on GPUs. Recent works study **low-cost CNN architectures** which shall allow fast inference on general purpose CPUs (*e.g.* [10, 29]). However, these architectures still rely on highly specialized instruction sets, which are often not available in industrial settings. Thus, we use a well-studied architecture [16] and implement a compressed version for fast inference on CPUs.

We use **multiple object tracking** to fuse the detections into individual trajectories over time. Over the past decades, tracking approaches primarily focused on pedestrian surveillance (*e.g.* [18]) and were recently dominated by offline methods (*e.g.* [2, 20]), which process batches of frames or even the whole image sequence at once. Our time-critical application domain, however, requires online tracking (*e.g.* [15, 23]) to infer the object states solely based on observations up to the current frame.

As detectors became more powerful, most tracking approaches rely on the **tracking-by-detection paradigm** (*e.g.* [23, 27]), which poses tracking as an association problem to robustly link corresponding detections between consecutive frames. We follow this path and especially use only geometric features for the association step, similar to [22, 23]. This is computationally cheaper and allows us to allocate more resources to the pedestrian detector.

Motion analysis methods are also closely related to our intent-based system. Their task is to model either long-term or short-term movement of pedestrians in order to predict future actions (*e.g.* [8, 12, 13, 14]). In contrast to our work, these methods often rely on additional sensor data and focus on driver assistance systems where predictions of short-term actions (*e.g.* rapid change of direction, stopping) are of main interest. Other approaches model the interaction of people in a crowd as their motion is often highly correlated (*e.g.* [1, 9]). While this is a plausible cue for short-term action prediction, we are more interested in the destination (*i.e.* the intent) of pedestrians and not in the exact path towards it.

Another related task is **crowd counting** with the goal to estimate the total amount of people given an image of a crowd without localizing individuals (*e.g.* [19, 30]). In contrast, we do not simply count the number of all the people in the scene, but only those with the intent to cross.

There are several **commercial vision-based pedestrian detectors** for traffic light control (*e.g.* FLIR or AGD[1]). However, they typically operate within a rather small waiting zone (*e.g.* AGD's stereo sensor observes $2 \times 3\,\mathrm{m}$), require different image modalities – like depth or thermal infrared – and lack the ability to predict the pedestrian's intent to schedule the traffic light in advance.

3. An intent-based pedestrian traffic light

Our traffic light controller captures the surroundings of the crosswalk by a camera mounted on top of the traffic light pole to achieve a sufficiently large field-of-view. We implement efficient detection and tracking approaches to obtain pedestrian trajectories in real-time. By analyzing these trajectories, our system infers the intent (*i.e.* the most probable destination) of each pedestrian in terms of predefined exit regions. This intent information – in combination with the resulting number of people requesting to cross the road – can then be used to schedule the traffic light. Figure 1 (right) provides a schematic overview of the system. In the following, we describe each component in more detail.

3.1. Pedestrian detection

In a first step, we detect pedestrians in each new frame of the camera stream. We build the detector upon the Single Shot MultiBox Detector (SSD) [17], and use a compressed version of AlexNet [16] as a backbone network, which gives

[1] See `www.flir.com` or `www.agd-systems.com`.

a good trade-off between runtime and accuracy; our implementation utilizes the AVX2 instruction set to obtain real-time performance on Intel CPUs. In the following, we describe our modifications to the original architecture.

We make the backbone network fully-convolutional by removing the last fully-connected layer `fc8` and converting the layers `fc6` and `fc7` to convolution layers ($6 \times 6 \rightarrow 4096$ and $1 \times 1 \rightarrow 4096$, respectively). In order to reduce the computational complexity, we further compress `fc6` and `fc7` to $6 \times 6 \rightarrow 128$ and $1 \times 1 \rightarrow 128$, respectively, using a feature distillation approach similar to [26].

Like in [17], we add two additional feature extraction layers `conv6_1` ($1 \times 1 \rightarrow 256$) and `conv6_2` ($3 \times 3 \rightarrow 512$) on top of `fc7` to cover pedestrians at all scales. Finally, box prediction is done on top of `conv4`, `conv5`, `fc7` and `conv6_2`. We feed images of size 400×400 to the detector and follow the training procedure described in [17]. The backbone network is initialized with our distilled model parameters and fine-tuned on application-specific samples observed from our highly elevated viewpoint.

The detector output for frame i is a list of axis-parallel bounding boxes $(\mathbf{c}_n^{(i)}, w_n^{(i)}, h_n^{(i)}),\ \ n \in [1, N]$, where $\mathbf{c}_n^{(i)} = (x_n^{(i)}, y_n^{(i)})^\top$ denotes the center, $w_n^{(i)}$ the width and $h_n^{(i)}$ the height of the n-th bounding box.

3.2. Tracking-by-detection

As we use most of our computing resources to obtain accurate and reliable detections, we can leverage the tracking-by-detection paradigm to link detections into pedestrian trajectories. The imaging conditions in our outdoor setting yield a substantial variance to the visual appearance of pedestrians, highly depending on solar altitude, weather conditions, light reflections or low light situations during night-time. Furthermore, our elevated viewpoint constrains visual features mostly to the rather small head and shoulder region of a pedestrian. Therefore, similar to [22, 23], our approach does not use a visual appearance model and only relies on motion cues to associate detections to trajectories.

We model the target dynamics of each trajectory by a Kalman filter [11]. A constant velocity model is usually powerful enough for real-world pedestrian surveillance scenarios (*e.g.* see findings of [20]), due to the object inertia and sufficiently high frame rates. In our initial tests, higher-order motion assumptions (such as acceleration- or curvature-based models) did not notably improve the localization. Thus, we apply the computationally more efficient constant velocity assumption to design the Kalman filter.

More formally, we determine the cost of a detection being assigned to a trajectory t_k by the Euclidean distance $\|\hat{\mathbf{c}}_k^{(i)} - \mathbf{c}_n^{(i)}\|_2$. Here, $(\hat{\mathbf{c}}_k^{(i)}, \hat{\mathbf{v}}_k^{(i)})^\top = \mathbf{F}\,(\mathbf{c}_k^{(i-1)}, \mathbf{v}_k^{(i-1)})^\top$ is the trajectory's state estimate at frame i, where $\mathbf{c}_k^{(i-1)}$ denotes the previously observed location of the k-th pedestrian, $\mathbf{v}_k^{(i-1)}$ her velocity, and $\mathbf{F}$ the filter's state transition matrix. Given the motion-based matching costs, we then solve for the optimal detection-to-trajectory assignment using the Hungarian algorithm [21]. Thus, we update the k-th trajectory as $t_k^{(i)} = t_k^{(i-1)} \cup \mathbf{c}_n^i$ if the n-th detection in frame i gets assigned to it. Afterwards, we correct the filter state with the new observation.

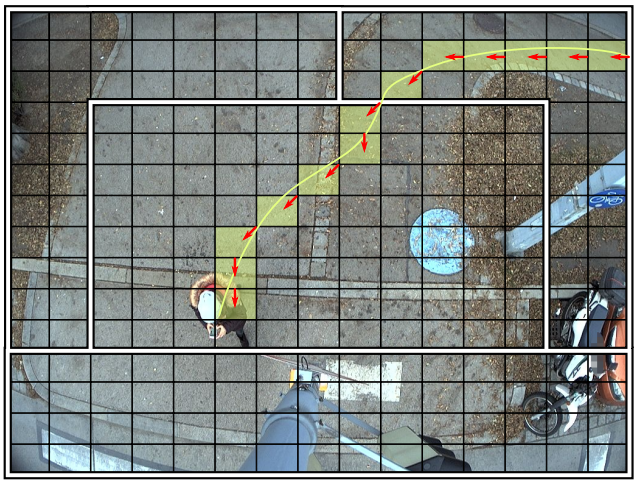

Figure 2. Illustration of the trajectory rasterization and exit regions. The colored cells show the occupation of the trajectory as given by the Bresenham algorithm; the arrows indicate the cell transitions; the $R = 3$ exit regions at the border are visualized by white lines.

For trajectory management, we follow standard strategies (*e.g.* [15, 20, 23]). More precisely, we tentatively initialize trajectories for each unassigned detection and decide after a few frames whether the corresponding trajectory should become active (*i.e.* if subsequent confident detections could be assigned to it) or if it was caused by a false positive of the detector. Trajectories are terminated once the pedestrian exits the field-of-view or no detections could be assigned to it for several consecutive frames.

3.3. Intent prediction

To estimate a pedestrian's intent given her trajectory, we divide the border of the image into R disjoint areas denoting potential exit regions (see Figure 2). Each exit region r is defined by a polygon represented via its line segments $\mathcal{L}_r = \{\mathbf{l}_{(r,1)}, \mathbf{l}_{(r,2)}, \dots\}$. We predict the intent of a pedestrian as the most likely exit region depending on the result of a dynamics-based prediction and a global trajectory prior. This allows us to identify the number of pedestrians actually requesting to cross the road at any time.

Dynamics-based extrapolation. We leverage the motion model of pedestrian k maintained by the tracker and extrapolate the location predictions by repeatedly applying the fil-

ter for ψ steps to estimate the future position $\tilde{\mathbf{c}}_k^{(i+\psi)}$ as

$$(\tilde{\mathbf{c}}_k^{(j)}, \tilde{\mathbf{v}}_k^{(j)})^\top = \mathbf{F}_k \, (\tilde{\mathbf{c}}_k^{(j-1)}, \tilde{\mathbf{v}}_k^{(j-1)})^\top, \tag{1}$$

with $j \in [i+1, i+\psi]$ and initially $\tilde{\mathbf{c}}_k^{(i)} = \mathbf{c}_k^{(i)}$, $\tilde{\mathbf{v}}_k^{(i)} = \mathbf{v}_k^{(i)}$. Given this position estimate, we compute the distance to each exit region r as

$$d_{k,r}^{(i)} = \min_{\mathbf{l} \in \mathcal{L}_r} \frac{\phi(\tilde{\mathbf{c}}_k^{(i+\psi)}, \mathbf{l})}{D}, \tag{2}$$

where D is the length of the image diagonal and $\phi(\cdot, \cdot)$ is the distance between a point and a line segment. We set $d_{k,r}^{(i)} = 0$ if $\tilde{\mathbf{c}}_k^{(i+\psi)}$ lies inside $\mathcal{L}_r$. Then, we weight the distance by a Gaussian kernel to get the similarity score $s(d) = e^{-d^2/(2\sigma^2)}$ and normalize these similarity scores over all exit regions to obtain the likelihood that pedestrian k will continue towards exit r as

$$p_{k,r,\text{EX}}^{(i)} = \frac{s\left(d_{k,r}^{(i)}\right)}{\sum_{r'} s\left(d_{k,r'}^{(i)}\right)}. \tag{3}$$

Here, we exploit the inertia assumption (*i.e.* pedestrians tend to walk rather constantly towards their desired destination), which provides a valid short-term estimate about the movement of the pedestrian. However, in real-world scenarios, motion dynamics alone are not always sufficiently reliable especially shortly after initializing a trajectory, during fast directional turns, or because detection output can be noisy such that the bounding box centers slightly jitter. All these factors may lead to temporarily incorrect estimates of the pedestrian's motion direction.

Global motion model. To overcome these issues, we additionally maintain a statistical global motion model that allows us to reason about the plausibility of a pedestrian's destination given her observed trajectory in combination with previously tracked people. This model is constantly updated with observed trajectories of people leaving the scene, thus allowing the model to improve over time. More formally, we want to model the probability $p_{k,r,\text{GM}}^{(i)} \equiv \mathbb{P}(r \mid t_k^{(i)}, \boldsymbol{\Theta})$ of pedestrian k leaving the scene at exit region r conditioned on her trajectory $t_k^{(i)}$ and the model parameters $\boldsymbol{\Theta}$.

To this end, we discretize the tracking region by a regular $W \times H$ grid and rasterize the trajectory using the Bresenham algorithm [3] to get a path through the cells of this grid, allowing cell transitions in an 8-connected neighborhood (see Figure 2). Each cell stores the *exit distribution* conditioned on the cell transition. In particular, we store the number of previously observed trajectories which passed from the neighboring cell d_in through this cell and terminated at exit region r. By storing these votes for each cell and exit region separately in $\boldsymbol{\Theta} = \theta_{(x,y,d_\text{in},r)}, \ \forall (x \in [1, W], y \in [1, H], d_\text{in} \in [1, 9], r \in [1, R])$, we can compute the priors

Algorithm 1 Global motion model.

procedure UPDATE($\boldsymbol{\Theta}$, t_k, r) ▷ Update when pedestrian k exits the field-of-view
  $\mathcal{T} \leftarrow$ RASTERIZE(REMOVEOUTLIERS(t_k))
  for $j \in [2, |\mathcal{T}|]$ **do** ▷ Iterate rasterized trajectory
    Map cell transition $\mathbf{c}_\mathcal{T}^{(j-1)} \rightarrow \mathbf{c}_\mathcal{T}^{(j)}$ to get d_in
    $x, y \leftarrow \mathbf{c}_\mathcal{T}^{(j)}$
    $\theta_{(x,y,d_\text{in},r)} \leftarrow \theta_{(x,y,d_\text{in},r)} + 1$ ▷ Increment cell transition vote for r
  end for
end procedure
procedure PREDICT($\boldsymbol{\Theta}$, t_k) ▷ Predict global motion prior for pedestrian k
  $\mathcal{T} \leftarrow$ RASTERIZE(REMOVEOUTLIERS(t_k))
  for $r \in [1, R]$ **do** ▷ Init accumulators
    $f_r \leftarrow 0$
  end for
  for $j \in [2, |\mathcal{T}|]$ **do** ▷ Iterate rasterized trajectory
    Map cell transition $\mathbf{c}_\mathcal{T}^{(j-1)} \rightarrow \mathbf{c}_\mathcal{T}^{(j)}$ to get d_in
    $x, y \leftarrow \mathbf{c}_\mathcal{T}^{(j)}$
    for $r \in [1, R]$ **do** ▷ Accumulate cell transition votes
      $f_r \leftarrow f_r + \frac{\theta_{(x,y,d_\text{in},r)}}{\sum_{r'} \theta_{(x,y,d_\text{in},r')}}$
    end for
  end for
  for $r \in [1, R]$ **do** ▷ Normalize votes
    $p_{k,r,\text{GM}}^{(j)} \leftarrow$ SOFTMAX(f_r)
  end for
end procedure

$p_{k,r,\text{GM}}^{(i)}$ for a trajectory at any frame i. This is done by accumulating these votes along the new trajectory and taking the softmax over all exit regions. To reduce the impact of noisy detection results, we additionally simplify the trajectory using the Ramer-Douglas-Peucker algorithm [5] as a preprocessing step, which discards the majority of outliers. More details can be found in Algorithm 1.

Combined prediction. We combine the predictions of both models to get the final intent prediction $p_{k,r,\text{COMB}}^{(i)} = \lambda \, p_{k,r,\text{EX}}^{(i)} + (1 - \lambda) \, p_{k,r,\text{GM}}^{(i)}$, where λ is a weighting parameter. In order to make a binary decision about the intent for the traffic light controller, we store the predicted exit regions for the last few seconds and report a crossing request only if the majority of the decisions within this time range are consistent.

4. Experimental results

We discuss results gathered throughout a long-term study at a highly frequented crosswalk in a European capital. Two cameras were mounted on top of the traffic light poles capturing the surroundings of the crosswalk at each side of the road. Both scenes are divided into $R = 3$ exit regions (*e.g.* see Figure 2). We recorded (1) the trajectory of each pedestrian; (2) our system's intent predictions densely for each time step of each trajectory; and (3) the timestamp of each manually pressed push-button.

We first compare our detector to other detectors on our application-specific dataset in terms of runtime and accuracy; then we show ablation results on our intent prediction methods using the recorded trajectories; finally, we compare our system's decisions to the existing push-button system that requires manual interaction of the pedestrians.

Method	$\mathbf{Acc}_{0.2}$	$\mathbf{Acc}_{0.5}$	$\mathbf{Acc}_{0.65}$	$\mathbf{Acc}_{0.8}$
EX MODEL	0.645	0.488	0.232	0.036
GM MODEL	0.977	0.969	0.924	0.492
COMBINED MODEL	**0.978**	**0.970**	**0.932**	**0.712**

Table 1. Predicting the intent (*i.e*. the correct exit region) with different motion model components. $\mathbf{Acc}_\rho$ is the accuracy at confidence threshold ρ.

4.1. Detector performance

The performance of the underlying detector is crucial for our system. We need accurate detections at a high frame rate, which results in usually small location changes between frames to ensure reliable trajectory matching. We use an application-specific dataset covering 2,050 annotated images from our highly elevated viewpoint using a 60/10/30 split for training/validation/testing. As we need the detector to be robust against arbitrary changes in weather and illumination conditions, we make sure to cover a diverse set of images (*e.g*. day/night, sunny/cloudy/rainy/snowy, *etc*.).

Figure 3 (left) shows a comparison of various detectors in terms of detection quality [6] and frame rate. Our detector achieves a good trade-off between speed and quality and is competitive to MobileNet [10] (which was published after our long-term experiment) in terms of accuracy. All further experiments and trajectories are based on the detections of our compressed AlexNet detector as it was used during the long-term study. We note that the system would benefit from the MobileNet detector.

4.2. Ablation study

To validate the algorithmic choices for the trajectory-based intent prediction, we conduct ablation experiments on a dataset of 80,000 validated example trajectories recorded during our long-term study. Since intent prediction is essentially a multi-class classification problem, we compute accuracy over $R = 3$ classes, each corresponding to an exit region $r \in [1, R]$. The prediction for a trajectory t_k is denoted as $\hat{r}_k = \arg\max_r c_{k,r}$. Here, $c_{k,r}$ denotes the prediction confidence that t_k ends in exit region r, defined as

$$c_{k,r} = \frac{1}{|t_k|} \sum_{j=\alpha_k}^{\alpha_k+|t_k|} \mathbb{1}\left[p^{(j)}_{k,r,m} > 1/R\right], \tag{4}$$

where α_k is the birth time of trajectory t_k and $m \in \{\text{EX}, \text{GM}, \text{COMB}\}$. By thresholding the prediction's confidence $c_{k,\hat{r}}$, we compute the average accuracy and report the results in Table 1.

While motion extrapolation (EX MODEL) alone performs poorly across all thresholds (mainly because of position jittering and its short-term assumptions), the global motion model (GM MODEL) yields good accuracy up to high confidence thresholds with a drop at 0.8. The main reason for this is that the GM MODEL needs a few cell transitions in order to deliver a reliable prediction, especially if the first few transitions yield ambiguous predictions. If we combine the two (COMBINED MODEL), the accuracy at very high thresholds – and therefore the prediction's robustness – increases significantly. This indicates that the system model parts are complementary.

4.3. Long-term proof of concept

To confirm the system's applicability, we compare it to the existing push-button solution with manual pedestrian interaction. To this end, we match push-button triggers to crossing requests as reported by our system based on the recorded timestamps. Thus, we are able to compute the recall as a function of the maximum allowed delay ϵ between a button trigger and a reported crossing request. Our system achieves recall values greater than 0.985 for a reasonable $\epsilon \leq 5\,\text{s}$ (details in the supplementary material). Moreover, we show in Figure 3 (middle) that we are able to reliably report crossing requests 3 to 4 s before the pedestrian reaches the push-button, which allows our system to react earlier.

An isolated evaluation of recall is not sound as we could easily maximize it by reporting crossing requests in a regular fashion. However, manually checking each trajectory to calculate precision would be very time-consuming. To still provide some insights, we provide a statistical comparison of prediction and button trigger frequencies in Figure 3 (right). While the absolute values are not directly comparable, since on average only every 10th pedestrian presses the button, we see that the distributions of crossing requests over time are highly correlated. Further, we observe peaks at 7am, 12am and 5pm, which correspond to people going to work, to lunch and back home, respectively. A more detailed evaluation including a manual evaluation of recall and precision can be found in the supplementary.

5. Conclusions

We aim to decrease the waiting times at crosswalks by proposing a real-time capable visual system for automated, intent-based traffic light control. It is able to decide whether a pedestrian actually plans to cross the road or not. This decision is based on the pedestrians' recorded movement. Our experiments validate our algorithmic choices and show that the method is superior to existing push-button solutions: We can reliably predict crossing requests before pedestrians reach the push-button and can, therefore, replace the manual push-button system. In contrast to the latter, we additionally report the number of waiting people and pave the road for further optimization of traffic light scheduling.

References

[1] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese. Social LSTM: Human Trajectory Prediction

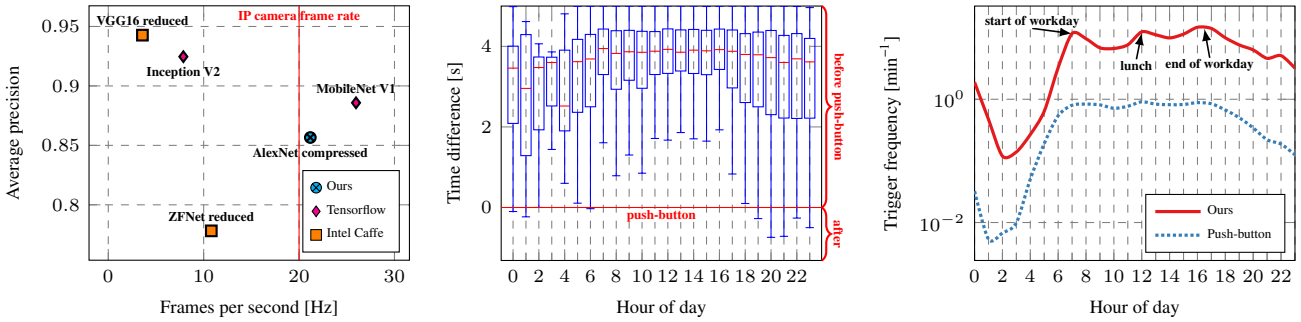

Figure 3. **Left:** Trade-off between runtime and quality for various detectors. Runtime was measured on an industrial PC with an Intel i7-6700 CPU. **Middle:** Time differences between triggers of our system and corresponding push-button triggers. Our system triggers on average 3 to 4 s before the pedestrian pushes the button. **Right:** Median trigger frequency as reported by our system and the push-button.

in Crowded Spaces. In *Proc. CVPR*, 2016.

[2] H. Ben Shitrit, J. Berclaz, F. Fleuret, and P. Fua. Multi-Commodity Network Flow for Tracking Multiple People. *IEEE TPAMI*, 36(8):1614–1627, 2014.

[3] J. E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 4(1):25–30, 1965.

[4] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *Proc. CVPR*, 2005.

[5] D. Douglas and T. Peucker. Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or Its Caricature. *Cartographica*, 10(2):112–122, 1973.

[6] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. *IJCV*, 88(2):303–338, 2010.

[7] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object Detection with Discriminatively Trained Part Based Models. *IEEE TPAMI*, 32(9):1627–1645, 2010.

[8] M. Goldhammer, K. Doll, U. Brunsmann, A. Gensler, and B. Sick. Pedestrian's Trajectory Forecast in Public Traffic with Artificial Neural Networks. In *Proc. ICPR*, 2014.

[9] D. Helbing and P. Molnar. Social force model for pedestrian dynamics. *Phys. Rev. E*, 51(5):4282, 1995.

[10] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv CoRR*, abs/1704.04861, 2017.

[11] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *J. Basic Eng.*, 82(1):35–45, 1960.

[12] V. Karasev, A. Ayvaci, B. Heisele, and S. Soatto. Intent-Aware Long-Term Prediction of Pedestrian Motion. In *Proc. ICRA*, 2016.

[13] C. Keller and D. Gavrila. Will the Pedestrian Cross? A Study on Pedestrian Path Prediction. *IEEE TITS*, 15(2):494–506, 2014.

[14] J. Kooij, N. Schneider, F. Flohr, and D. Gavrila. Context-Based Pedestrian Path Prediction. In *Proc. ECCV*, 2014.

[15] M. Kristan, J. Perš, M. Perše, and S. Kovačič. Closed-world tracking of multiple interacting targets for indoor-sports applications. *CVIU*, 113(5):598–611, 2009.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Proc. NIPS*, 2012.

[17] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single Shot MultiBox Detector. In *Proc. ECCV*, 2016.

[18] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, X. Zhao, and T.-K. Kim. Multiple Object Tracking: A Literature Review. *arXiv CoRR*, abs/1409.7618, 2017.

[19] M. Marsden, K. McGuinness, S. Little, and N. E. O'Connor. ResnetCrowd: A Residual Deep Learning Architecture for Crowd Counting, Violent Behaviour Detection and Crowd Density Level Classification. In *Proc. AVSS*, 2017.

[20] A. Milan, S. Roth, and K. Schindler. Continuous Energy Minimization for Multi-Target Tracking. *IEEE TPAMI*, 36(1):58–72, 2014.

[21] J. Munkres. Algorithms for the Assignment and Transportation Problems. *JSIAM*, 5(1):32–38, 1957.

[22] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool. You'll Never Walk Alone: Modeling Social Behavior for Multi-target Tracking. In *Proc. ICCV*, 2009.

[23] H. Possegger, T. Mauthner, P. M. Roth, and H. Bischof. Occlusion Geodesics for Online Multi-Object Tracking. In *Proc. CVPR*, 2014.

[24] J. Redmon and A. Farhadi. YOLO9000: Better, Faster, Stronger. In *Proc. CVPR*, 2017.

[25] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Proc. NIPS*, 2015.

[26] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. FitNets: Hints for Thin Deep Nets. In *Proc. ICLR*, 2015.

[27] F. Solera, S. Calderara, and R. Cucchiara. Learning to Divide and Conquer for Online Multi-Target Tracking. In *Proc. ICCV*, 2015.

[28] K. E. Van de Sande, J. R. Uijlings, T. Gevers, and A. W. Smeulders. Segmentation as Selective Search for Object Recognition. In *Proc. ICCV*, 2011.

[29] X. Zhang, X. Zhou, M. Lin, and J. Sun. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. *arXiv CoRR*, abs/1707.01083, 2017.

[30] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma. Single-Image Crowd Counting via Multi-Column Convolutional Neural Network. In *Proc. CVPR*, 2016.