计算机科学
COMPUTER SCIENCE

面向资源受限边缘设备的实时精确目标跟踪

张莘沂, 谭光

引用本文

张莘沂, 谭光. 面向资源受限边缘设备的实时精确目标跟踪[J]. 计算机科学, 2024, 51(11A): 231200167-9.

ZHANG Xinyi, TAN Guang. Real-time Accurate Object Tracking for Resource-constrained Edge Devices [J]. Computer Science, 2024, 51(11A): 231200167-9.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.231200167

面向资源受限边缘设备的实时精确目标跟踪

张莘沂　谭　光
中山大学智能工程学院　广东 深圳 510275
(zhangxy336@mail2.sysu.edu.cn)

摘　要　实时视频分析任务通常涉及到运行计算密集型的深度神经网络模型来实现目标跟踪。在实际应用中,将多路视频数据分析任务卸载到摄像机附近的边缘设备上进行处理变得尤为重要。然而,这些边缘设备的计算资源通常非常有限,导致目标跟踪的精度较差。这主要是由过时的检测结果、跟踪错误积累以及无法感知新目标造成的。针对上述问题,提出了一种基于预测和修正的检测跟踪框架。该框架中包含了3个核心的组件:1)预测性检测传播:通过轻量级预测模型快速更新过时的对象边界框以匹配当前帧;2)帧差修正器:基于帧差信息将出现误差的目标框回归到正确位置;3)新目标检测器:在跟踪过程中通过对帧差特征进行聚类发现新出现的目标。实验结果表明,相比基线方法,该框架在不同的交通场景中取得了19.4%到34.7%的精度提升,同时保持了实时的运行速度。

关键词:边缘设备;资源效率;目标检测;目标跟踪
中图分类号　TP391.4;U495

Real-time Accurate Object Tracking for Resource-constrained Edge Devices

ZHANG Xinyi and TAN Guang
School of Intelligent Systems Engineering,Sun Yat-Sen University,Shenzhen,Guangdong 510275,China

Abstract　Real-time video analysis tasks often involve running computationally intensive deep neural network(DNN) models for object tracking. In practical applications,offloading multi-stream video analysis tasks to edge devices near the cameras has become crucial. However,these edge devices often have limited computing resources,resulting in low tracking accuracy. This is primarily due to outdated detection results,accumulated tracking errors,and the inability to detect new object. To address these issues,a prediction-correction based framework is proposed. The framework comprises three core components:1)Predictive detection propagation,which rapidly updates outdated object bounding boxes using a lightweight prediction model to match the current frame. 2)Frame difference corrector,which refines bounding boxes based on frame difference information. 3)New object detector,which discovers newly appearing objects during the tracking process by clustering frame difference features. Experimental results demonstrate that the framework achieves accuracy improvements ranging from 19.4% to 34.7% compared to baseline methods across various traffic scenarios while maintaining real-time execution speed.

Keywords　Edge device,Resource efficiency,Object detection,Object tracking

1　引言

实时视频分析的任务旨在实时检测和跟踪视频中的感兴趣目标,分析它们的统计学运动模式,其在交通监控分析[1]、人员识别[2]、物体定位和跟踪[3]以及增强现实[4]等各个领域中发挥着日益重要的作用。考虑到数据传输成本和隐私问题,近年来在摄像头附近的边缘设备上处理视频数据成为一种新兴的趋势[5-6]。通常,视频分析需要在GPU上运行复杂的DNN模型。然而,在实际情况中,边缘设备的硬件资源通常是有限的;同时,尽管有些智能摄像头或边缘设备可能配备有足够的GPU资源,但随着视频任务量的急剧增加,其成本可能会显著增加。

因此在有限的资源约束下,一种常见的目标跟踪方法是使用基于检测的跟踪(Detection-Based Tracking,DBT)框架[7-9]。DBT框架中首先运行深度神经网络(Deep Neural Network,DNN)模型来确定视野中目标的位置和类别。在检测间隔期间,执行轻量级跟踪方法来更新目标框位置。本文在Jetson Nano上评估了标准的DBT框架的运行性能,其中目标检测任务通过在GPU上部署YOLOv5模型[10]来完成。表1列出了不同模型大小和输入图像大小设定下的检测时间。虽然随着模型压缩程度的增大(如YOLOv5n),运行效率也会明显提升,但同时检测精度也会较大幅度地下降,尤其是在高清视频的检测任务中。例如,YOLOv5s(s代表小)模型需要899 ms来检测1920×1080视频帧中的目标。这样的延迟几乎是不可接受的,因为在此期间目标可能已经发生了显著移动,使得检测结果已经过时。对于DBT框架中的目标跟踪任务,通常采用光流跟踪方法,其延迟的主要来源包括:1)特征点提取,在跟踪窗口开始时进行,耗时140 ms;2)特

基金项目:国家自然科学基金(62372488)
This work was supported by the National Natural Science Foundation of China(62372488).
通信作者:谭光(tanguang@mail.sysu.edu.cn)

征点匹配，平均每帧耗时 11 ms；3）更新物体边界框，平均每帧耗时 14 ms。显然，在低端设备上，光流跟踪仍然会引入不可忽视的延迟。

表 1 Jetson Nano 上的 YOLOv5 目标检测延迟

Table 1 Time consumption of object detection on Jetson Nano using YOLOv5

模型	图像大小	预处理/ms	推理/ms	NMS/ms	每帧耗时/ms
s	1 280	7.9	350.9	2.57	390.0
s	1 680	21.5	649.9	1.40	716.3
s	1 920	42.5	817.5	1.11	899.1
m	1 280	10.4	809.5	1.16	858.7
m	1 680	57.5	1474.7	0.63	1 587.3
m	1 902	CUDA 显存不足			

现有研究中，有几种可能的方法可以改善检测和跟踪性能。

1）帧过滤：Reducto[6]，NoScope[11]和 Glimpse[8]这类方法专注于过滤掉内容变化不相关或不显著的帧。这些方法可以作为视频分析的预处理步骤使用。

2）感兴趣区域：最近的方法，如 FlexPatch[7]和 EAAE[4]，专注于识别帧中可能包含感兴趣目标或跟踪失败频繁发生的区域。这种方法需要对区域分割和分割处理进行复杂设计。

3）模型配置调整：这种方法[12]包括动态调整系统配置参数，如图像大小和模型大小。在资源受限的系统上，可用的配置选择可能受到限制，从而限制了优化空间。

在本文特定的场景中，简单应用现有方法并不能有效解决问题。

图 1(a)给出了 DBT 的工作流程。每个时间窗口对应一次检测所需的持续时间 T。在 $t+T$ 帧的时间点，检测器已经为 t 帧产生了一个检测结果，但该结果已经过时。直接重用这些结果会导致准确性较差。因此，需要通过几个跟踪步骤将过时的结果更新以匹配当前帧，这个过程称为检测传播(Detection Propagation，DP)。这个过程为后续的持续跟踪提供了新的基础。检测传播中的一种可行的实现方案称为增量检测传播(Incremental Detection Propagation，IDP)[8]。IDP 通过缓存中间帧，并计算和分析连续帧之间的帧差变化来获得关键帧。这种方法能够消除冗余帧，并使得目标跟踪可以在缓存的子集上快速进行。

图 1(b)给出了在 30 帧的检测延迟下，将 IDP 应用于基本 DBT 工作流程时的精度变化情况。以窗口 2 为例，在标记为 B 的时刻，检测器完成了对第 1 帧的检测并发送给 CPU 上的跟踪器，跟踪器随即开始 DP 过程，将第 1 帧的检测结果快速传播到第 30 帧(在 C 范围内的绿色线)。为了方便说明，图 1(b)以框架运行的时间顺序而不是以帧序号进行对齐。通过在 3 个关键帧上运行目标跟踪，将结果传播到第 30 帧(在 A 范围内的三角形)。由于在 30 帧内发生了场景变化，DP 在第 30 帧的实际准确性会大幅下降(A)。此外，IDP 过程表现出明显的延迟。因此，在延迟期间的跳帧导致跟踪的准确性不可避免地降低(在 C 范围内的橙色线)。在 D 时刻，DP 过程完成，精度得到一定的补偿。后续在持续跟踪过程中，现有的目标不断移动，新的目标可能出现，这进一步导致跟踪的准确性逐渐下降。

基于以上观察，需要解决 3 个问题来提高跟踪准确性。

1）减少长跨度 DP 引起的准确性损失(如 A 所示)，同时最小化 DP 本身的时间开销(如 C 所示)；

2）减少随时间累积的跟踪错误；

3）处理在跟踪过程中新出现或离开场景的对象。

为了解决这些问题，本文提出了一种基于预测和修正的跟踪方法，主要贡献包括 3 个部分。

第一个部分为“预测性检测传播器”，可以快速更新旧的物体目标框以匹配当前帧。通过设计了一个极其轻量级的模型，根据物体的历史轨迹预测其未来位置。该模型在 Jetson Nano 的 CPU 上运行，仅需 2.52 ms 就能完成推理。

第二个部分为“帧差校正器”。作为一种低级特征，帧差对物体运动的响应非常敏感。然而，帧差图像通常会显示严重的碎片化和噪声，使得其利用非常困难。本文将其作为一个修正器，并使用了一种基于帕累托前沿的搜索算法，以提高修正算法的搜索效率。

第三个部分为“新目标检测器”。在较长的检测窗口期间，跟踪算法将持续错过新的目标，加剧了准确性损失的问题。现有的解决方法，如文献[7]和文献[9]，通常依赖于消耗 GPU 资源的检测模型。而本方法利用帧差捕获新出现的物体只需要消耗少量 CPU 资源。

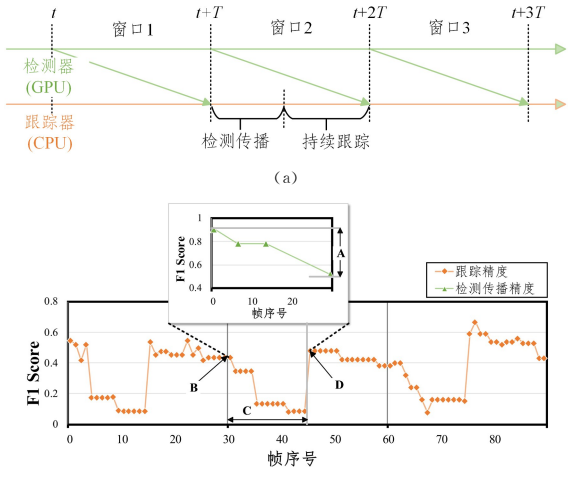


图 1 DBT 框架及特性

Fig. 1 DBT framework and characteristics

与表现最佳的基准方法相比，本方法在平均 F1 准确性上有显著的改进，在 4 种交通场景取得了 19.4%到 34.7%的精度提升，整体平均 F1 分数从 0.491 提高到 0.623。

2 相关工作

2.1 大规模视频分析

回顾性视频分析涉及对录制的视频内容进行离线分析，从存档的镜头中提取有价值的信息。为了应对计算需求所带来的瓶颈，一种典型的方法是使用轻量级模型来过滤掉无关的帧[11,13-14]。现有的一些研究[15-16]侧重于查询模型的设计和优化，而其他工作[17-18]则利用时空信息从跟踪轨迹的角度解决问题。

实时性视频分析对视频数据进行实时在线处理并识别特定的事件或活动，如运动检测、入侵检测或异常行为检测。它

涉及多个子任务，其中对象检测和跟踪是基本任务之一。实时视频分析对计算需求可能很大，特别是在同时处理大量实时视频流时。VideoStorm[19]和 VideoEdge[20]等解决方案通过分析视频特征并调整多个配置参数来实现高效的资源管理。CASVA[21]通过自适应地调整配置来解决视频流中复杂的动态变化。Yuan 等[22]提出了一种新颖的解码器，它自适应地选择关键帧，并在其他帧中重用它们的推断结果。

2.2 边缘辅助的视频分析

边缘辅助视频分析的主要目标是在数据源附近处理视频数据。根据资源可用性和任务特性，计算可以在靠近摄像头的设备上进行，也可以在具备加速器的摄像头内部进行。某些任务也可以由边缘和云端协同处理。

按照计算部署的位置，该领域的现有工作可以分为 3 种方法。

1）设备上的系统：Reducto[6]和 FlexPatch[7]提取轻量级的图像线索，如帧差异和光流，这些线索与资源受限的设备兼容。

2）以边缘为主的系统：LAVEA[23]采用任务选择和请求优先级机制来调整边缘节点之间的工作负载，Vitrack[24]利用马尔可夫模型捕捉丢失的对象并恢复整个轨迹，Elf[25]和 Adamask[26]将选定的感兴趣区域卸载到服务器以节省带宽。

3）边缘-云协同系统：最近的一些工作[5,27]专注于为边缘设备训练专有的检测模型，并定期在云端进行重新训练和更新。

2.3 目标检测与跟踪系统优化

在视频处理领域，优化检测和跟踪性能一直是持续关注的焦点，研究工作主要集中在以下几个方面。

1）跟踪优化：主要关注对移动物体的有效和高效的持续锁定，从而提高跟踪的准确性。Dong 等[28]通过使用强化学习深度 Q 网络自适应搜索最佳超参数组合来优化跟踪。CLNet[29]将跟踪任务视为分类问题，利用初始帧和受场景变化影响的帧的潜在特征来促进快速调整。

2）基于预测轨迹的优化：利用物体的历史位置提取潜在的时空移动模式。Deo 等[30]考虑轨迹预测中的横向和纵向变异性，通过使用学习的离散策略在采样地图上遍历。Choi 等[31]将车道级别和车辆间交互整合起来生成多个潜在的未来轨迹，并使用分层评分系统对其进行推理评分。

3）基于帧差异的优化：利用低计算成本的帧差异提取潜在信息。Glimpse[8]在差异较小的帧中跟踪物体，并将检测任务委托给具有显著变化的帧的服务器。Corsel 等[32]将帧差异明确地作为运动线索融入 YOLOv5 中，增强对移动物体的检测。Lv 等[33]利用帧差异技术有效地识别新物体的潜在区域。

3 基于预测与修正的系统框架

本文在传统的基于检测的跟踪框架上进行改进。在终端设备的 GPU 上运行基于深度学习的目标检测模型，同时在其 CPU 上运行轻量级光流法跟踪器。图 2 给出了一个视频检测窗口中的分析过程，窗口的长度由目标检测的持续时间决定。在窗口开始时，进入检测传播阶段，接收到来自过去窗口的延迟检测结果。本文提出使用一个预测性检测传播器来更新旧的检测结果，以便在更新的结果基础上执行后续的跟踪。在持续跟踪阶段，使用帧差修正器来调整由检测传播和跟踪生成的边界框，以提高精度。同时，使用新目标检测器在跟踪过程中发现新出现的对象，而不依赖于常规的目标检测模型。

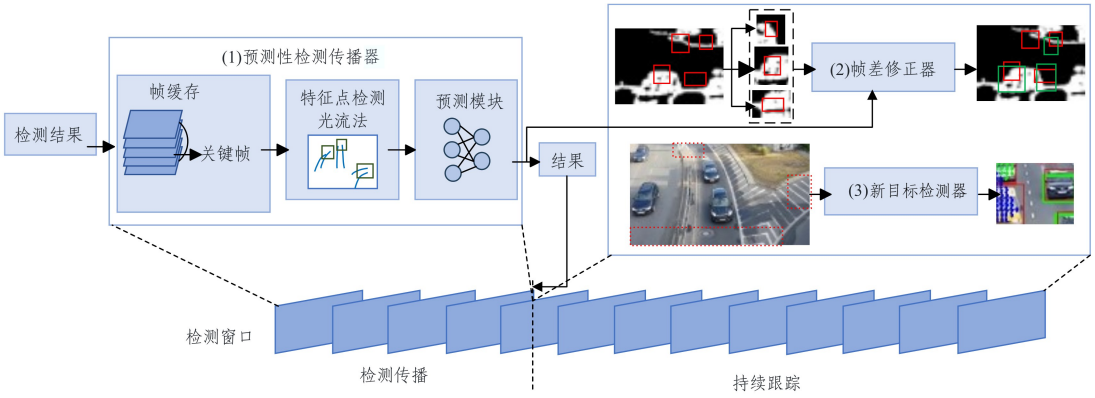


图 2 基于预测与修正的系统框架

Fig. 2 Prediction and correction based framework

预测性检测传播模块实现了更快、更准确的 DP。具体而言，在对窗口的起始帧进行目标检测后，将该窗口中的所有帧存储到缓存中。在帧缓存中，按照固定间隔 δ 选择 m 个关键帧。在这些关键帧上，提取每个物体目标框内的特征点，使用光流法在关键帧上跟踪这些特征点并更新目标框位置，以获得每个物体的历史位置序列，这些序列将作为轻量级预测模型的输入。该模型预测每个物体在当前帧的位置。根据实验确定 m 和 δ 的最佳值分别为 2 和 5，即当检测延迟为 30 帧时，选择第 5 帧和第 10 帧作为 DP 关键帧。一般来说，本文的 δ 值比文献[8]中的 δ 值小。

帧差修正器模块有助于减小由于预测和跟踪不准确性而产生的目标框误差。这些目标框连同帧差异图像一起用于生成候选框列表，然后从中搜索最佳解。考虑到解空间可能较大，本文提出了一种基于帕累托前沿的多目标优化算法，以快速找到最优解。

新目标检测器模块首先根据图像中可能频繁出现或消失对象的新目标区域计算出区域内的帧差。如果帧差值超过预定义的阈值，说明出现了新目标。然后使用聚类算法确定新物体的位置和数量。此外，对于正在从场景中消失的物体，如果边界框内的帧差低于预定义的阈值，这

些物体就可以被确定为离开了画面。

3.1 预测性检测传播

在静止相机拍摄的视频流中，物体通常呈现出规律的移动模式。然而，现有的目标跟踪方法并没有充分利用这些轨迹模式。本文利用这种规律性，以更高效的方式预测物体的位置。现有的方法通常通过深度学习模型结合复杂的道路网络结构和车辆交互行为模式来预测未来的轨迹[34-35]。虽然这些方法可以实现高精度的预测，但它们依赖于大量的 GPU 资源，给边缘系统的部署带来了挑战。相反，基于概率统计预测方法计算开销较低，但在复杂场景中的表现不佳。因此，本文提出了一种轻量级模型来学习场景特征并进一步预测物体的轨迹，即使在 CPU 上其也能高效运行。这个预测模型在每个窗口内的 DP 过程中运行。

本系统对每个特定场景中的对象的历史数据进行离线分析。首先进行相机校准来获取鸟瞰图中的物体的世界坐标，使用基于两个消失点的相机校准方法[36-37]来确定世界坐标。具体而言，首先手工标记两对平行车道线（在图 3 中用蓝色实线表示，这些线在消失点处相交），以及一条已知长度的线（在图 3 中用蓝色虚线表示）。使用这些信息即可估计相机的内参和外参矩阵。随后，将内参矩阵和外参矩阵相乘以获得投影矩阵 $\boldsymbol{P}$。投影矩阵 $\boldsymbol{P}$ 用于将坐标从图像坐标系 $[x_I^t, y_I^t]^T$ 转换到世界坐标系 $[x_W^t, y_W^t]^T$，转换方法如式(1)和式(2)所示。

$$\lambda_1 [x_I^t, y_I^t]^T = \boldsymbol{P} [x_W^t, y_W^t]^T \tag{1}$$

$$\lambda_2 [x_W^t, y_W^t]^T = \boldsymbol{P}^{-1} [x_I^t, y_I^t]^T \tag{2}$$

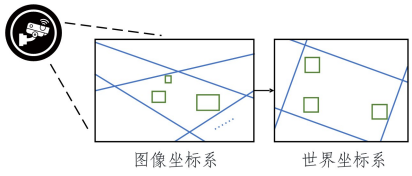


图 3 BEV 坐标标定过程

Fig. 3 BEV coordinate calibration process

接下来就可以基于轨迹预测模型预测边界框的未来位置。虽然 RNN[38]，Transformer[39] 和 VAE[40] 在轨迹预测中被广泛使用，但它们通常需要更多的参数来拟合数据。对于特定的任务，本文探索性地使用了一个快速的基于 CNN 的模型，我们发现其同样可以在满足准确性的前提下保证快速性。

轨迹预测模型的架构如图 4 所示。首先在选定的 3 个历史帧上传播检测结果，获得 N 个检测到的物体的目标框序列。每个目标框的信息包括其在图像坐标中的位置、宽度、高度和类别。随后，使用式(1)将图像转换为世界坐标。接下来将边界框的相应宽度、高度和类别与世界坐标连接起来，创建一个 $N\times3\times5$ 维的输入矩阵。其经过全连接层(FC)以提取位置序列的特征。为了确保卷积层的感受野覆盖所有的历史信息，模型使用了两层一维卷积操作，并进行了对称填充。然后，使用一个 FC 层将潜在特征转换为 4 个未来帧的轨迹，得到一个 $N\times4\times4$ 维的矩阵。每个预测的物体由其在世界坐标中的中心、宽度和高度表示。最后，使用式(2)将世界坐标映射到图像坐标。预测模型的训练损失使用预测目标框和真实值之间的均方误差(MSE)进行衡量。模型运行时从预测结果中选择与当前帧最接近的一个作为最终结果。为了减小预测误差的影响，这些结果将通过帧差修正器进行优化。最终的结果为后续的持续跟踪提供了基础。

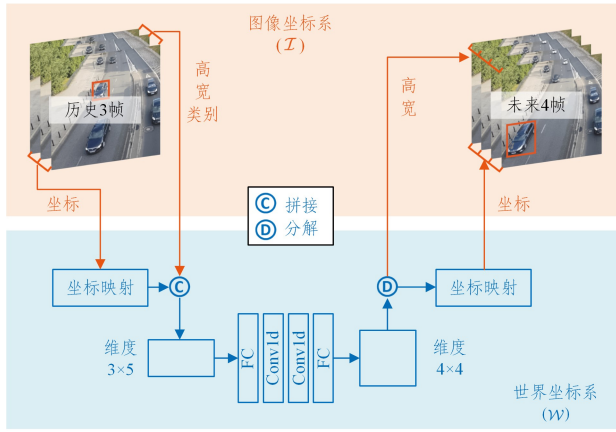


图 4 轨迹预测模型架构

Fig. 4 Schema of trajectory prediction model

3.2 帧差修正器

通过分析帧间像素变化，帧差法能够提供物体在场景中如何出现、移动或消失的信息。现有的使用帧差的目标检测方法[41-42]仍然难以处理帧差破碎和物体重叠等问题。此外，现有的方法需要较大量的计算资源，不适用于低端设备的实时计算。

为了在克服帧差限制的同时实现低计算复杂度，本文提出了一种不使用复杂的帧差处理而是基于简单的帧间帧差进行修正的算法。

算法 1 基于 Pareto 前沿的帧差修正器算法

输入：初始边界框 $B^0=[x_1^0, x_2^0, y_1^0, y_2^0]$

输出：最优边界框 $B^*=[x_1^*, x_2^*, y_1^*, y_2^*]$

1. 扩大 B^0 以得到区域 A，并计算二值帧差 D；
2. 提取 Sobel 边缘特征 S_x 和 S_y；
3. $L_x=\{S_x$ 中的强边缘的 x 坐标$\}$
4. $L_y=\{S_y$ 中的强边缘的 y 坐标$\}$
5. 生成候选框集合

$$\{B'\}=\{[x_1', x_2', y_1', y_2'] \mid x_1', x_2' \in L_x, y_1', y_2' \in L_y\}$$

6. for each 候选框 in $\{B'\}$ do
7. 计算 $O_c(B')$，$O_r(B')$ 和 $O_m(B')$
8. end
9. 通过计算式(4)获得帕累托最优解集 C；
10. $B^*=\arg\max_{B\in P}(O_c(B)+O_r(B)+O_m(B))$

在每个检测到的物体上执行边界框校正算法。选择各自边界框内特征点运动方差较大的物体。方差阈值经过实验经验性总结为 25。给定一个初始边界框 B^0，其 x 轴和 y 轴范围表示为 $x_1^0<x_2^0$，$y_1^0<y_2^0$。B^0 的扩展区域为 $A=\{p_{x,y} \mid x\in[x_1^0-\delta_x, x_2^0+\delta_x], y\in[y_1^0-\delta_y, y_2^0+\delta_y]\}$。其中，$\delta_x$ 和 δ_y 分别设置为 B^0 宽度和高度的一半。然后，通过计算 A 中每个像素的灰度图像在帧 i 和 $i+1$ 之间的绝对值差异 $p_{x,y}$ 来获得 A 的二值图像 $D=d_{x,y}$。如果 $p_{x,y}$ 超过阈值，则二值图像中对应位置 $d_{x,y}$ 的值为 1，否则为 0。

$$d_{x,y}=\begin{cases}1, & p_{x,y}>\text{threshold}\\ 0, & p_{x,y}<\text{threshold}\end{cases} \tag{3}$$

通常情况下，真实边界框的边界与帧差图像中物体的轮廓接近。首先，使用 Sobel 算子分别在帧差图像的 x 和 y 方向上获取梯度 S_x 和 S_y，并保留梯度大于一定阈值（在本文实

验中为254)的强边缘。将它们的 x 轴和 y 轴坐标分别存储在两个列表L_x和L_y中。遍历这些列表以生成潜在的候选框,这些框定义了解空间B'。如果采用穷举搜索策略,将会生成大量的候选框,使得算法不适用于实时场景。因此需要一种快速的算法来搜索最优解。

本文提出了一个多目标优化问题,其涉及3个目标函数,分别是$O_c(B')$,$O_r(B')$和$O_m(B')$。其中,$O_c(B')$定义为二值图像在候选框B'内部值为1的像素总和与该区域内值为1的像素总和之比;$O_r(B')$定义为二值图像在候选框B'内部值为1的像素总和与候选框B'的面积之比;$O_m(B')$定义为候选框B'与初始框B^0的 IoU。该多目标优化问题可以定义为:

$$\begin{aligned}&\text{Maximize } F(B')=[O_c(B'),O_r(B'),O_m(B')]\\&\text{s.t. } B'\in\{B'\}\end{aligned}\tag{4}$$

3个目标函数较高的值对应于更准确的候选框。因此可以确定 Pareto 最优集 C,它对应于解空间的 Pareto 边界。最后将 C 中所有目标函数之和最高的解作为最终的最优解。

图5给出了提出的3个目标与目标框准确度之间的相关性,准确度以与地面真实(Ground Truth,GT)框的 IoU 衡量。图5中的每个点对应于一个候选框的3个目标值,这些候选框是通过在GT框的基础上进行不同的移动和缩放生成的。图5右侧的颜色图表示候选框与GT框之间的 IoU,从蓝色到黄色表示数值逐渐增加。图中有黑色边缘的点表示帕累托前沿曲面。在这个前沿曲面内部,进一步通过找到最大化目标值之和来找到唯一最优解,如图5中的红色箭头所示。

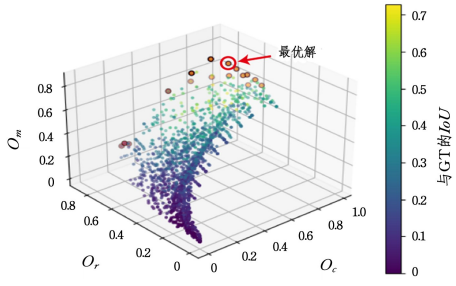


图5 候选框的解空间和帕累托边界

Fig. 5 Solution space and Pareto frontier of proposal boxes

3.3 新目标检测

基于光流的跟踪方法无法检测新目标。然而,在静态摄像头场景中,新目标经常出现在特定区域。因此,可以利用帧差信息推断新目标的出现。

首先分析每个场景,找到目标经常进入或离开画面的区域。然后,定期监测这些区域的帧差,以确定它们是否超过预定义的阈值,并以此作为后续处理的触发器(见图6(a))。注意,如果跟踪中的目标存在于这些区域内,则屏蔽这些目标框内部的帧差。

新目标检测的主要挑战在于确定新对象的数量和类别,同时减轻相机振动和背景运动等因素的影响。本方法基于两个观察结果:1)属于同一目标的像素在一段时间内位置上接近并具有相似的运动模式;2)属于前景目标的像素通常呈现平滑的运动,而属于背景的像素通常呈现不规则的运动或保持静止。

首先,以固定间隔提取帧差特征,并对原始帧差图像进行降采样以提高效率。结果如图6(b)所示。帧差大于阈值的像素被稀疏提取并添加到光流跟踪的特征点列表中进行短期跟踪,获得一些特征点序列,将其中静止或非平滑移动的点滤除。

为了确定新目标的数量,对选出的序列使用基于密度的聚类算法(DBSCAN)[43]。根据第一个观察结果,将每个序列的初始位置和移动量拼接起来作为聚类的输入,聚类得到几个特征点的簇。最后,用目标框将每个簇的所有特征点包围起来,得到新目标的位置。使用帧差修正器对这些目标框进行校正。为了确定新目标的类别,分析历史视频帧中各类别目标框的出现频率和形状,并将每个新目标与最可能的类别进行匹配。上述过程如图6(c)所示。

(a) (b) (c)

图6 新目标检测器说明

Fig. 6 Illustration of new object detector

4 实验

4.1 实验设置

4.1.1 数据集

本文实验中使用了4个真实世界的数据集,其中2两个为标准的公开交通监控基准数据集[33-44],另2个为新收集的交通视频[1)]。这4个数据集简称为 OnRamp,Intersect,Express 和 UrbMixed。它们具有不同的交通密度和车辆速度。具体而言:

1)OnRamp 是一个包含快速移动车辆的匝道场景,相对于摄像机距离较近;

2)Intersect 是一个从低角度观察的十字路口场景,包含低速车辆和密集的交通容量;

3)Express 记录了高速公路上稀疏的车辆流,相对于摄像机距离较远;

4)UrbMixed 是一个城市场景,包含混合的交通流,车辆的速度各不相同,并且有一些静止的车辆停在路边。

使用 YOLOv5x[10] 作为标准生成真实值,因为该模型具有较大的参数量和较高的检测精度,能够表现出与人类视觉几乎相当的性能。

4.1.2 基线方法

本文方法与以下代表性方法进行比较。

1)MARLIN[9]:它使用深度神经网络模型进行目标检测,并使用基于光流的跟踪器在检测延迟期间进行目标位置更新。检测模型仅在画面出现显著变化时触发,以减少计算开销。

2)MARLIN+IDP[8-9]:这是一个 MARLIN 的改进版本,它通过 IDP 将旧的检测结果传播到当前帧。

为了确保公平比较,本实验在 Jetson Nano 上实现了 MARLIN 和 MARLIN+IDP 的并行化版本,采用了基于

1) https://pan.baidu.com/s/19HXXoM3O-foBrwDi45AC8w?pwd=erry

GPU 的检测和基于 CPU 的跟踪的范式。

4.1.3 性能指标

本文使用平均 F1 分数作为评估跟踪精度的指标。其定义如式(5)所示：

$$F1=\frac{2 \cdot precision \cdot recall}{precision+recall} \tag{5}$$

此外，交并比(Intersection over Union，IoU)及其平均值 mIoU 被用来衡量预测框与 GT 框之间的重叠程度。需要注意的是，mIoU 考虑了多类别的影响，与单一类别的 IoU 有所区别。

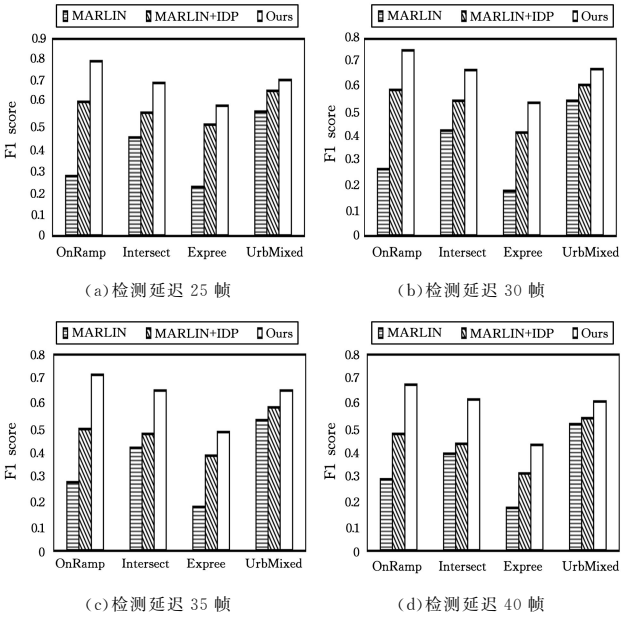
4.1.4 实现细节

本文方法使用 Python 3.7.11 和 PyTorch 1.10.0 在 CUDA 11.1 上实现，主要实验平台是 Nvidia Jetson Nano。实验展示了在严格的资源预算下通过优化可以实现的潜在性能提升。

使用 YOLOv5s[10]作为目标检测器，它是 YOLOv5 的低端版本，在所选实验平台上推理速度更快，准确性较低。

4.2 总体精度

图 7 比较了 4 个数据集上不同方法的平均 F1 分数。由于影响系统效果的一个关键因素是检测延迟，实验通过提前对所有帧进行目标检测，并在跟踪过程中使用预设延迟获取每帧的结果来模拟不同的 GPU 能力。



(a)检测延迟 25 帧 (b)检测延迟 30 帧

(c)检测延迟 35 帧 (d)检测延迟 40 帧

图 7 在多种检测延迟下的精度

Fig. 7 Accuracy under various detection delays

如图 7 所示，MARLIN 方法准确率较低，这是因为它们不执行跟踪或检测传播，使得算法严重受到过时的检测结果的影响。MARLIN+IDP 结合了检测传播，因此表现更好。然而，由于检测传播的延迟和误差，它并没有很好地解决检测延迟问题，随着延迟的增加，IDP 的效果持续减弱。本文方法在所有数据集和检测延迟下都取得了最佳性能。

以检测延迟 30 帧的情况为例，图 8 给出了本文方法和 MARLIN+IDP 方法在一个时间窗口内第 7，19，20 帧的跟踪效果及单帧的 F1 分数。可以看出，由于检测延迟较长，IDP 方法会出现明显的光流特征点匹配错误，尤其是画面近处移动速度较快的目标框会出现明显的偏移；而本文方法使用预测性检测传播结合帧差修正器，能够更准确地定位目标。同时，通过新目标检测器，本文方法检测出了画面右侧及画面上方出现的新目标。

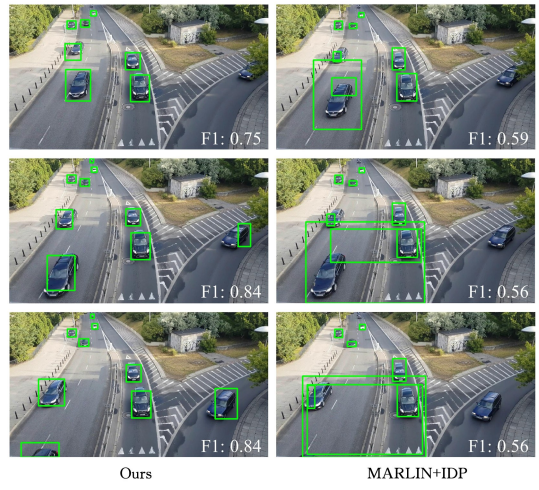


图 8 演示对比过程

Fig. 8 Demonstration process comparison

4.3 总体开销

图 9 给出了在 Jetson Nano 上分析 OnRamp 数据集上两种方法的 CPU 和内存使用情况。在测试过程中，两种方法都充分利用了 GPU，因此没有显示出来。在图 9 下侧 CPU 利用率中，蓝色和绿色曲线分别表示运行本文方法和 MARLIN+IDP 方法时 4 核 CPU 的平滑 CPU 利用率。本文方法在基线方法的基础上加入了额外模块来优化精度。额外的模块包括轨迹预测模型、帧差修正器及新目标检测器。但从运行的总体情况来看，两种方法之间的计算开销差异很小，这表明本文方法所使用的模块都是轻量级的，不会对系统造成过多额外的开销。关于内存使用情况(上半部分)，本文方法的内存消耗与 MARLIN+IDP 方法相当。具体而言，预测模型的参数大小仅为 840 kB，与整体内存占用相比可以忽略不计。在稳定状态中(第二个窗口之后)，两种方法在每个窗口中的内存使用情况都会下降，这是由于 DP 完成并释放了一些内存。这些下降点的相对位置也表明本文方法中的 DP 比基线方法中的 DP 更早完成，进一步证实了之前在案例研究中的观察结果。

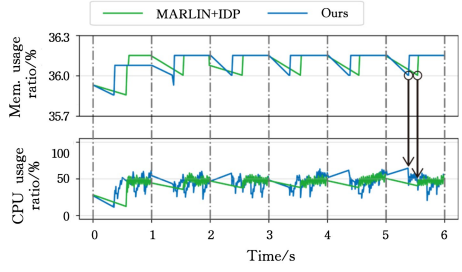


图 9 OnRamp 数据集上的 CPU 与内存使用情况

Fig. 9 CPU and memory usage on OnRamp

对于每个检测窗口，提取边界框内的特征，然后进行检测传播和持续跟踪，此处使用一种计算效率高的 Harris 角点检测算法。作为本方法和基线方法的共有部分，光流特征提取过程需要 140 ms。特征点进一步用于估计光流，每次耗时

11 ms。对于 MARLIN+IDP,一个 IDP 过程涉及 30 次帧差的计算和一个线分割问题的解决,这些操作总共需要 224 ms。表 2 列出了在 30 帧延迟和图像尺寸为 1 920×1 080 的情况下,本文方法中每个组件的运行时间。预测性检测传播过程包括 3 个步骤:从缓存中选择帧、运行预测模型和校正预测结果,总共需要 53.9 ms。这意味着本文方法的 DP 比 MARLIN+IDP 快 4 倍,因此可以更早地释放 CPU 从而进行持续跟踪。

帧差修正器和新目标检测分别需要 12.9 ms 和 8.1 ms,两者都小于 30 FPS 采样率下的正常帧间隔。这表明这两个组件可以很好地适应常规的帧率要求,而不会引入额外的延迟。

表 2 各组件的运行时间开销

Table 2 Running time of each component

组件	时间/ms
预测性检测传播	53.9
帧差修正器	12.9
新目标检测	8.1

4.4 帧差修正器效果验证

为了验证帧差修正器的效果,首先从数据集中收集了与真实值的 *IoU* 大于 0.3 的目标框。对于初始状态的 *IoU* 就较低的目标框,修正器的效果会减弱,这是因为它假设存在一个合适的初始位置,可以从中进行优化。统计数据显示,*IoU* 低于 0.3 的目标框仅占修正器处理的所有边界框的 16.8%。如表 3 所列,引入帧差修正器显著提高了 *mIoU* 和 *F*1 分数。

表 3 帧差修正器效果

Table 3 Effectiveness of frame difference corrector

数据集	*mIoU*		*F*1	
	修正前	修正后	修正前	修正后
OnRamp	0.573	0.618	0.698	0.753
Intersect	0.557	0.675	0.597	0.672
Express	0.431	0.511	0.520	0.538
UrbMixed	0.531	0.605	0.663	0.676

此外,实验评估了带有修正器和不带修正器的系统,并比较了它们的 F1 分数。在 30 帧的延迟下运行了这两种系统,结果显示带有修正器的系统在所有数据集上的 F1 得分都有所提高。

需要注意的是,在实际跟踪中,一个成功的修正可以弥补几个后续帧的准确性。因此,没有必要对特定对象进行连续的校正。

4.5 新目标检测效果验证

新目标检测器对精度的贡献取决于交通密度和平均物体的大小,因此,在 F1 分数上,它的影响可能是较小的。然而,实时视频分析通常涉及需要及时报告感兴趣事件的应用程序。因此即使新目标不经常出现,早期检测新目标也是有必要的。

首先定义量化的识别延迟为新目标首次出现在场景中到正确识别它所需的时间。这种延迟受到多种因素的影响。例如,在最坏的情况下,对于给定窗口的第 0 帧,发送检测请求并在第 1 帧出现一个对象,请求的结果在下一个窗口的第 0 帧得到,从而允许发送新的检测请求。这导致了长达 2 个窗口的延迟,DP 在这里无法发挥作用,因为它是基于当前窗口的第 0 帧运行的。此外,跟踪失败和 DP 错误也可能会延长识别延迟。从表 4 可以看出,本方法在所有情况下都比基线方法具有更短的平均延迟,从 0.98 秒到 1.85 s 不等。其中,MARLIN 方法完全不执行检测传播,因此产生的延迟最长。

表 4 新目标检测延迟

Table 4 Delay of new object detection

(s)

数据集	MARLIN	MARLIN+IDP	Ours
OnRamp	4.16	1.12	0.98
Intersect	2.32	1.99	1.36
Express	3.25	1.75	1.44
UrbMixed	3.50	2.73	1.85

4.6 BEV 坐标转换效果验证

在预测性检测传播组件中,本文提出将图像坐标系转换为俯视图,以便于轨迹预测。为了展示这种策略的好处,本实验比较了所有数据集中有无 BEV 转换的轨迹预测模型的均方误差(Mean Square Error,MSE)损失。结果显示,BEV 转换将 MSE 损失从 5.47×10^{-4} 降低到 2.78×10^{-4}。这种改进归因于 BEV 能够在真实世界中恢复车辆的真实轨迹,这些轨迹往往遵循可预测的模式。相比之下,在图像坐标系中,物体的外观会不断变形,增加了学习任务的复杂性。

结束语 本文研究了在资源受限边缘设备上实时视频分析中的检测和跟踪任务。开发了一种新颖的 DBT 系统,它包括 3 个关键组件:预测检测传播器、帧差修正器和新目标检测器。所提方法优于传统的 DBT 方法,平均 F1 准确率提高了 26.5%。

本研究主要关注固定摄像机,假设交通监控和公共监视中普遍存在的情况。后续工作计划将研究扩展到包括安装在智能车辆或机器人上的移动摄像机,增加移动性对实时性能提出了更大的挑战。

参 考 文 献

[1] CHEN J,WANG Q,CHENG H H,et al. A review of vision-based traffic semantic understanding in ITSs[J]. IEEE Transactions on Intelligent Transportation Systems,2022.

[2] YI J,CHOI S,LEE Y. EagleEye:Wearable camera-based person identification in crowded urban spaces[C]// Proceedings of the 26th Annual International Conference on Mobile Computing and Networking. 2020:1-14

[3] EMAMI P,ELEFTERIADOU L,RANKA S. Long-range multi-object tracking at traffic intersections on low-power devices[J]. IEEE Transactions on Intelligent Transportation Systems,2021,23(3):2482-2493.

[4] LIU L,LI H,GRUTESER M. Edge assisted real-time object detection for mobile augmented reality[C]// The 25th Annual International Conference on Mobile Computing and Networking. 2019:1-16.

[5] BHARDWAJ R,XIA Z,ANANTHANARAYANAN G,et al. Ekya:Continuous learning of video analytics models on edge compute servers[C]// 19th USENIX Symposium on Networked Systems Design and Implementation(NSDI 22). 2022:119-135.

[6] LI Y,PADMANABHAN A,ZHAO P,et al. Reducto:On-camera filtering for resource-efficient real-time video analytics[C]// Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications,

Techno-logies, Architectures, and Protocols for Computer Communication. 2020: 359-376.
[7] YANG K, YI J, LEE K, et al. FlexPatch: Fast and Accurate Object Detection for On-device High-Resolution Live Video Analytics[C]// IEEE INFOCOM 2022-IEEE Conference on Computer Communications. IEEE, 2022: 1898-1907.
[8] CHEN T Y H, RAVINDRANATH L, DENG S, et al. Glimpse: Continuous, real-time object recognition on mobile devices[C]// Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems. 2015: 155-168.
[9] APICHARTTRISORN K, RAN X, CHEN J, et al. Frugal following: Power thrifty object detection and tracking for mobile augmented reality[C]// Proceedings of the 17th Conference on Embedded Networked Sensor Systems. 2019: 96-109.
[10] YOLOv5[OL]. https://github.com/ultralytics/yolov5.
[11] KANG D, EMMONS J, ABUZAID F, et al. Noscope: optimizing neural network queries over video at scale[C]// Proceedings of the VLDB Endowment, 2017: 1586-1597.
[12] ZHANG S, WANG C, JIN Y, et al. Adaptive configuration selection and bandwidth allocation for edge-based video analytics [J]. IEEE/ACM Transactions on Networking, 2021, 30(1): 285-298.
[13] TCHAYE-KONDI J, ZHAI Y, SHEN J, et al. Smartfilter: An edge system for real-time application-guided video frames filtering[J]. IEEE Internet of Things Journal, 2022, 9(23): 23772-23785.
[14] MOLL O, BASTANI F, MADDEN S, et al. Exsample: Efficient searches on video repositories through adaptive sampling[C]// 2022 IEEE 38th International Conference on Data Engineering (ICDE). IEEE, 2022: 2956-2968.
[15] XU R, ZHANG C, WANG P, et al. ApproxDet: content and contention-aware approximate object detection for mobiles [C]// Proceedings of the 18th Conference on Embedded Networked Sensor Systems. 2020: 449-462.
[16] CAO J, HADIDI R, ARULRAJ J, et al. Thia: Accelerating video analytics using early inference and fine-grained query planning [J]. arXiv: 2102.08481, 2021.
[17] BASTANI F, MADDEN S. OTIF: Efficient tracker pre-processing over large video datasets[C]// Proceedings of the 2022 International Conference on Management of Data. 2022: 2091-2104.
[18] HWANG J, KIM M, KIM D, et al. {CoVA}: Exploiting {Compressed-Domain} Analysis to Accelerate Video Analytics[C]// 2022 USENIX Annual Technical Conference (USENIX ATC 22). 2022: 707-722.
[19] ZHANG H, ANANTHANARAYANAN G, BODIK P, et al. Live video analytics at scale with approximation and {Delay-Tolerance}[C]// 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17). 2017: 377-392.
[20] HUNG C C, ANANTHANARAYANAN G, BODIK P, et al. Videoedge: Processing camera streams using hierarchical clusters[C]// 2018 IEEE/ACM Symposium on Edge Computing (SEC). IEEE, 2018: 115-131.
[21] ZHANG M, WANG F, LIU J. Casva: Configuration-adaptive streaming for live video analytics[C]// IEEE INFOCOM 2022-IEEE Conference on Computer Communications. IEEE, 2022: 2168-2177.
[22] YUAN T, MI L, WANG W, et al. AccDecoder: Accelerated Decoding for Neural-enhanced Video Analytics[C]// IEEE INFOCOM 2023-IEEE Conference on Computer Communications. 2023: 1-10.
[23] YI S, HAO Z, ZHANG Q, et al. Lavea: Latency-aware video analytics on edge computing platform[C]// Proceedings of the Second ACM/IEEE Symposium on Edge Computing. 2017: 1-13.
[24] CHENG L, WANG J, LI Y. Vitrack: Efficient tracking on the edge for commodity video surveillance systems[J]. IEEE Transactions on Parallel and Distributed Systems, 2021, 33(3): 723-735.
[25] ZHANG W, HE Z, LIU L, et al. Elf: accelerate high-resolution mobile deep vision with content-aware parallel offloading[C]// Proceedings of the 27th Annual International Conference on Mobile Computing and Networking. 2021: 201-214.
[26] LIU S, WANG T, LI J, et al. Adamask: Enabling machine-centric video streaming with adaptive frame masking for dnn inference offloading[C]// Proceedings of the 30th ACM International Conference on Multimedia. 2022: 3035-3044.
[27] KONG Y, YANG P, CHENG Y. Edge-assisted on-device model update for video analytics in adverse environments[C]// Proceedings of the 31st ACM International Conference on Multimedia. 2023: 9051-9060.
[28] DONG X, SHEN J, WANG W, et al. Dynamical hyperparameter optimization via deep reinforcement learning in tracking[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2019, 43(5): 1515-1529.
[29] DONG X, SHEN J, PORIKLI F, et al. Adaptive siamese tracking with a compact latent network[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022, 45(7): 8049-8062.
[30] DEO N, WOLFF E, BEIJBOM O. Multimodal trajectory prediction conditioned on lane-graph traversals[C]// Conference on Robot Learning. PMLR, 2022: 203-212.
[31] CHOI D, MIN K W. Hierarchical latent structure for multi-modal vehicle trajectory forecasting[C]// European Conference on Computer Vision. Cham: Springer Nature Switzerland, 2022: 129-145.
[32] CORSEL C W, VAN LIER M, KAMPMEIJER L, et al. Exploiting Temporal Context for Tiny Object Detection[C]// Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. 2023: 79-89.
[33] LV X, WANG Q, YU C, et al. A Feedback - Driven DNN Inference Acceleration System for Edge-Assisted Video Analytics [J]. IEEE Transactions on Computers, 2023, 72(10): 2902-2912.
[34] GUPTA A, JOHNSON J, FEI-FEI L, et al. Social gan: Socially acceptable trajectories with generative adversarial networks [C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 2255-2264.
[35] ALAHI A, GOEL K, RAMANATHAN V, et al. Social lstm: Human trajectory prediction in crowded spaces[C]// Proceedings of the IEEE Conference on Computer Vision and Pattern

Recognition. 2016:961-971.

[36] ORGHIDAN R,SALVI J,GORDAN M,et al. Camera calibration using two or three vanishing points[C]// 2012 Federated Conference on Computer Science and Information Systems (FedCSIS). IEEE,2012:123-130.

[37] TaNG Z,WANG G,XIAO H,et al. Single-camera and inter-camera vehicle tracking and 3D speed estimation based on fusion of visual and semantic features[C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2018:108-115.

[38] CHIARA L F,COSCIA P,DAS S,et al. Goal-driven self-attentive recurrent networks for trajectory prediction[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022:2518-2527.

[39] ZHANG K,FENG X,WU L,et al. Trajectory prediction for autonomous driving using spatial-temporal graph attention transformer[J]. IEEE Transactions on Intelligent Transportation Systems,2022,23(11):22343-22353.

[40] LEE N,CHOI W,VERNAZA P,et al. Desire:Distant future prediction in dynamic scenes with interacting agents[C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017:336-345.

[41] SINGLA N. Motion detection based on frame difference method [J]. International Journal of Information & Computation Technology,2014,4(15):1559-1565.

[42] ZHENG D,ZHANG Y,XIAO Z. Deep learning-driven gaussian modeling and improved motion detection algorithm of the three-frame difference method[J]. Mobile Information Systems,2021,2021(1):9976623:1-9976623:7.

[43] SCHUBERT E,SANDER J,ESTER M,et al. DBSCAN revisited,revisited:why and how you should(still) use DBSCAN[J]. ACM Transactions on Database Systems(TODS),2017,42(3):1-21.

[44] DU K,PERVAIZ A,YUAN X,et al. Server-driven video streaming for deep learning inference[C]// Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures,and Protocols for Computer Communication. 2020:557-570.

ZHANG Xinyi, born in 1998, postgraduate. Her main research interests include object detection and tracking, and their applications in mobile computing.

TAN Guang, born in 1978, Ph.D, professor, is a member of the CCF (No. 19464M). His main research interests include design and evaluation of networked systems and machine learning.