计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2019,28(9):271−277 [doi: 10.15888/j.cnki.csa.007069]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

联合 YOLO 和 Camshift 的目标跟踪算法研究①

韩　鹏, 沈建新, 江俊佳, 周　喆

(南京航空航天大学 机电学院, 南京 210016)
通讯作者: 沈建新, E-mail: cadatc@nuaa.edu.cn

摘　要: 为了解决传统目标跟踪算法在有遮挡后无法准确跟踪的问题, 提出了将 YOLO 和 Camshift 算法相联合的目标跟踪算法. 基于 YOLO 网络结构来构建目标检测的模型, 在模型构建之前, 采用图像增强的方法对视频帧进行预处理, 在保留视频帧中足够图像信息的同时, 提高图像质量, 降低 YOLO 算法的时间复杂度. 用 YOLO 算法确定出目标, 完成对目标跟踪的初始化. 根据目标的位置信息使用 Camshift 算法对后续的视频帧进行处理, 并对每一帧的目标进行更新, 从而可以保证不断调整跟踪窗口位置, 适应目标的移动. 实验结果表明, 所提的方法能够有效地克服目标被遮挡后跟踪丢失的问题, 具有很好的鲁棒性.

关键词: YOLO 算法; Camshift 算法; 图像增强; 目标跟踪; 遮挡

引用格式: 韩鹏,沈建新,江俊佳,周喆.联合 YOLO 和 Camshift 的目标跟踪算法研究.计算机系统应用,2019,28(9):271–277. http://www.c-s-a.org.cn/1003-3254/7069.html

Research on Target Tracking Algorithm Based on YOLO and Camshift

HAN Peng, SHEN Jian-Xin, JIANG Jun-Jia, ZHOU Zhe

(College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)

Abstract: In order to solve the problem that traditional target tracking cannot be accurately tracked after occlusion, a target tracking algorithm combining YOLO and Camshift algorithm is proposed. Building a model of target detection using YOLO network structure, before the model is constructed, the image frame is preprocessed by image enhancement method, while maintaining sufficient image information in the video frame, improving the image quality and reducing the time complexity of the YOLO algorithm. The target is determined by the YOLO algorithm, and the initialization of the target tracking is completed. According to the position information of the target, the Camshift algorithm is used to process the subsequent video frames, and the target of each frame is updated, so that the position of the search window can be continuously adjusted to adapt to the movement of the target. The experimental results show that the proposed method can effectively overcome the problem of tracking loss after the target is occluded, and has good robustness.

Key words: YOLO algorithm; Camshift algorithm; image enhancement; target tracking; occlusion

目标跟踪是目前计算机视觉的一项基本核心技术, 在民用和军事的许多领域都有广阔的应用前景. 目前, 目标跟踪在智能视频监控、智能交通[1]、军事应用、图像检索等领域得到了充分的发展. 然而, 图像是从三维空间到二维平面的投影, 本身丢失了部分信息, 而且运动目标不是一个一成不变的信号, 它在跟踪过程中会发生旋转、位移、缩放、遮挡等复杂变化, 都在一定程度上影响目标跟踪的准确性, 增加了目标跟踪的难度[2].

为了实现有效的目标跟踪, 提高目标跟踪的鲁棒

① 基金项目: 江苏省研究生科研创新计划 (KYCX18_0317)
Foundation item: Graduate Innovation Program of Jiangsu Province (KYCX18_0317)
收稿时间: 2019-03-08; 修改时间: 2019-04-02; 采用时间: 2019-04-04; csa 在线出版时间: 2019-09-05

性, 卷积神经网络的算法已经成功应用于目标跟踪领域. 目前常见的算法有 R-CNN (Region-based Convolutional Networt)[3]、Fast R-CNN[4]、Faster R-CNN[5]、YOLO (You Only Look Once)[6]、SSD (Single Shot MultiBox Detectoer)[7]等. R-CNN 的基本思想是利用选择搜索算法 (Selective Search) 在图像中提取可能包含目标候选区域, 然后用 CNN 提取特征, 实现目标跟踪. 但是 R-CNN 计算量大, 步骤繁琐, 运行缓慢. Fast R-CNN 在 R-CNN 的基础上进行了两点改进, 一是在网络的最后一个卷积层后加了一个 ROI (Region of Interest) 池化层, 二是把目标边框的回归和分类这两个任务的损失函数合在一起训练, 使得训练的效率更高, 运行速度更快.

从 R-CNN 和 Fast R-CNN 以及到后来的 Faster R-CNN 等一系列算法都是基于候选区域的. 而不用候选区域的算法也有很多, 比较典型的是基于回归思想的 YOLO 和 SSD 算法.

这些基于卷积神经网络的目标跟踪算法其实更侧重于目标物体的检测, 在每一帧图像中进行滑动窗口进行遍历, 虽然包含了目标所在的所有可能位置, 但是缺乏对视频帧中目标运动信息的连续获取. 传统的视频跟踪算法如 Camshift (Continuously Adaptive Meanshift)[8]算法的实时性比较强, 但是当场景中出现目标被遮挡时, 跟踪效果受到较大影响, 跟踪目标甚至会丢失.

针对目标被遮挡的问题, 本文联合 YOLO 和 Camshift 的目标跟踪算法的长处, 用 YOLO 算法不断更新目标框, 克服目标遮挡的干扰, 改进传统的 Camshift 跟踪算法.

1 视频帧预处理及检测算法

通过图像增强的方法对视频的首帧进行预处理, 通过 YOLO 算法处理图像增强后的首帧图像, 检测出目标, 利用 Camshift 算法对目标框进行跟踪.

1.1 图像增强

图像增强的方法有很多, 按照不同作用域可分为空域和频域增强, 空域有灰度变换、直方图均衡化和空域滤波等, 频域增强有高通、低通和同态滤波等[9]. 由于在目标的检测跟踪中, 光照会影响图像的整体观感, 导致目标的部分信息丢失, 无法辨认目标等问题, Retinex[10]作为空域增强算法的一种, 在动态压缩、颜色不失真和边缘增强三个方面达到平衡, 所以本文利用 Retinex 算法对视频帧进行自适应的增强.

Retinex 算法将图像看作是由入射图像和反射图像共同构成的, 即:

$$S(x,y) = R(x,y) \times L(x,y) \tag{1}$$

其中, $S(x,y)$ 为视频帧初始的图像; $R(x,y)$ 表示反射分量, 即图像的内在属性; $L(x,y)$ 表示入射分量, 决定图像像素能够达到的动态范围. Retinex 的本质就是得到能够表示图像内在信息的反射分量, 去除入射分量. 在计算过程中, 将上式取对数来方便计算.

$$\log S = \log(R \times L) = \log R + \log L \tag{2}$$

令 $s = \log S, r = \log R, l = \log L$, 即:

$$r = s - l \tag{3}$$

在 Retinex 算法中, 利用高斯函数和初始图像进行卷积来表示入射分量, 即:

$$r(x,y) = \log S(x,y) - \log[F(x,y) * S(x,y)] \tag{4}$$

其中, $r(x,y)$ 是输出图像, *运算是卷积运算, $F(x,y)$ 是中心环绕函数, 即:

$$F(x,y) = Ke^{-(x^2+y^2)/c^2} \tag{5}$$

$$\iint F(x,y)dxdy = 1 \tag{6}$$

其中, K 是归一化因子, c 是高斯环绕的尺度. Retinex 算法中的卷积运算是对入射图像的计算, 通过计算像素点与周围区域在加权平均的作用下, 估计图像中光照变化, 去除 $L(x,y)$, 保留 $R(x,y)$ 信息.

使用 Retinex 算法对视频帧进行图像增强处理, 图 1 展示了部分图像经过 Retinex 处理前后的对比效果.

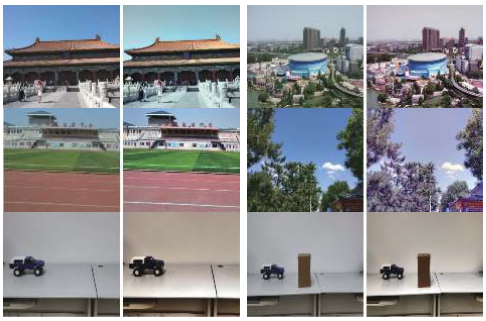

图 1 图像增强前后对比图

1.2 YOLO 算法目标检测

YOLO 检测算法将目标检测看作一个单一的回归问题, 直接在图像中找到目标的边界框. 将图像输入卷

积神经网络进行特征提取. 不同卷积层的特征图如图 2 所示.

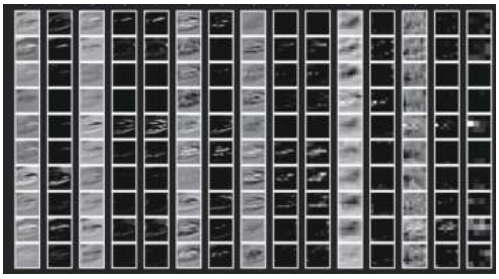

图 2 不同卷积层特征图

将特征图输入检测网络进行检测, 判断目标的类别并提取目标的边界框, 并对边界框采用非极大抑制进行选取, 得到最终的边界框. YOLO 算法的检测框架如图 3 所示.

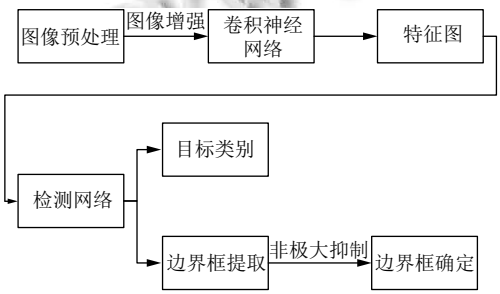

图 3 YOLO 检测框架图

YOLO 算法将图像分为 $S\times S$ 的网格, 如图 4 所示, 如果目标的中心落入其中的一个单元格中, 该单元格负责检测目标.

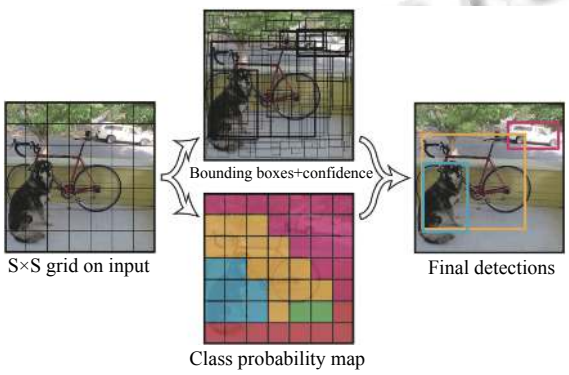

图 4 YOLO 检测步骤

在每个单元格中, YOLO 会预测 B 个目标边界框, 每个边界框包含 5 个预测值: x,y,w,h 和置信度. 如果该单元格中存在目标, 置信度分数等于预测框与真实框的交并比, 如果该单元格中不存在目标, 则置信度分数为零. 表达式如下:

$$Confidence = Pr(Object)\times IOU_{pred}^{truth} \tag{7}$$

其中, IOU_{pred}^{truth}表示预测框与真实框的交并比, 交并比用来表示在目标检测中对目标定位是否准确. 表达式如下:

$$IOU_{pred}^{truth} = \frac{(box(pred)\cap box(truth))}{(box(pred)\cup box(truth))} \tag{8}$$

YOLO 算法的网络结构图如图 5 所示, 网络中有 24 个卷积层和两个全连接层, 网络的设计借鉴了 Google Net 的思想, 在每个 1×1 的降维层之后再连接一个 3×3 的卷积层, 来代替 Inception 结构[11]. 由于在目标跟踪时, 只需要判断是前景还是背景, 不需要进行目标类别的判断, 所以可以将全连接层去掉, 用 Softmax[12]分类器进行简化, 如图 6 所示, 将最后一层卷积层的输出作为 Softmax 分类器的输入, 将图像检测分为前景和背景, 将检测为前景的目标作为候选区域, 为下面的目标跟踪做准备.

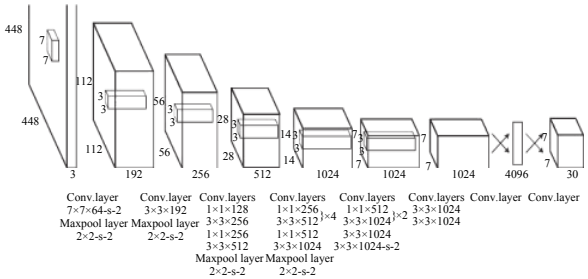

图 5 YOLO 算法网络结构图

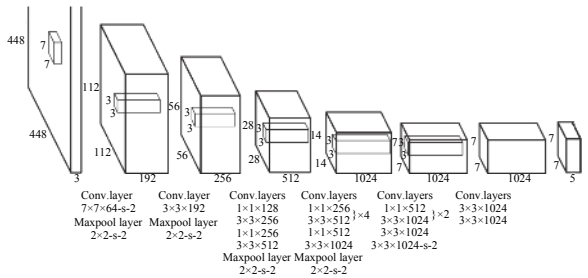

图 6 简化后 YOLO 算法网络结构图

2 联合跟踪算法

本文基于目标的卷积特征, 在 Camshift 的算法框

架下对目标进行检测跟踪. 首先采用图像增强的方法对视频帧进行预处理, 然后通过 YOLO 算法检测图像中的目标, 通过非极大抑制确定目标框, 再利用 Camshift 算法对目标进行跟踪, 确定目标下一次迭代的新位置, 为视频的下一帧选择候选目标. 当目标被遮挡时, 使用 YOLO 算法对目标进行更新.

2.1 Camshift 跟踪算法

Camshift 算法, 又称连续自适应均值漂移 (Meanshift) 算法. 它的基本思想是在视频图像上的每一帧都做 Meanshift 算法, 将上一帧的结果 (跟踪窗口的中心和大小) 作为当前帧的初始值, 依次这样迭代下去, 达到目标跟踪的效果.

Camshift 算法如图 7 所示.

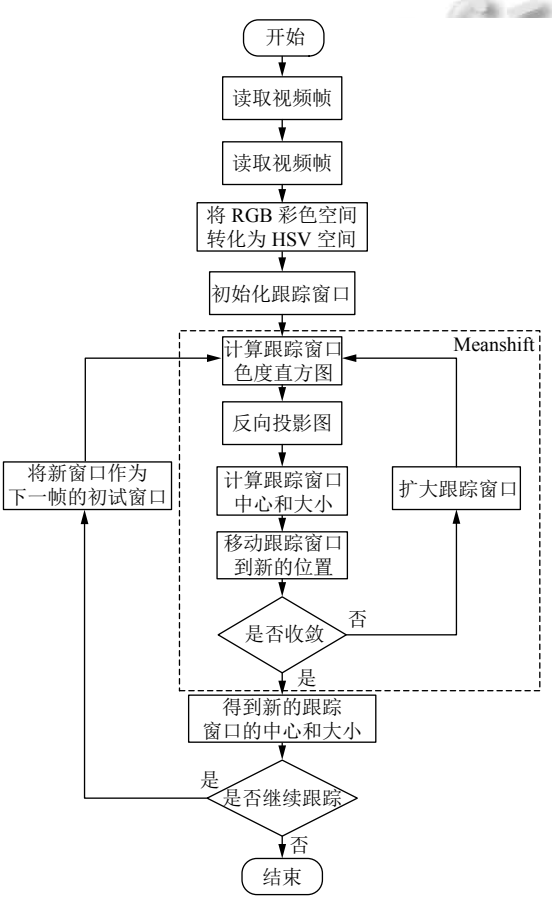

图 7 Camshift 算法流程图

Camshift 算法进行目标跟踪, 首先提出用 HSV 色彩空间中色度分量的直方图的反向概率投影来作匹配, 具体分为下面几个步骤: (1) 确定初始目标和区域; (2) 计算出目标区域的色度分量直方图; (3) 利用直方图计算输入图像的反向投影图; (4) 利用 Meanshift 算法在反向投影图上进行迭代, 直到其达到最大迭代次数或者收敛, 保存窗口的零阶矩、一阶矩和二阶矩; (5) 从第四步中计算出新的窗口中心和大小, 以此为初始值, 进行下一帧的目标跟踪 (即跳转至第二步).

Camshift 算法在 Meanshift 算法的基础上, 主要是能根据中心位置和窗口大小进行更新, 达到跟踪窗口自适应的目的, 具体流程如下:

1) 通过反向投影图计算初始跟踪窗口的颜色概率分布 $I(x,y)$;

2) 计算跟踪窗口的零阶矩、一阶矩和二阶矩:

$$M_{00}=\sum_x\sum_y I(x,y) \tag{9}$$

$$M_{10}=\sum_x\sum_y xI(x,y) \tag{10}$$

$$M_{10}=\sum_x\sum_y xI(x,y) \tag{11}$$

$$M_{20}=\sum_x\sum_y x^2I(x,y) \tag{12}$$

$$M_{02}=\sum_x\sum_y y^2I(x,y) \tag{13}$$

$$M_{11}=\sum_x\sum_y xyI(x,y) \tag{14}$$

3) 计算中心位置:

$$x_c=\frac{M_{10}}{M_{00}} \tag{15}$$

$$x_c=\frac{M_{10}}{M_{00}} \tag{16}$$

4) 计算跟踪窗口的长度和宽度:

$$l=2\sqrt{\frac{(a+c)+\sqrt{b^2+(a-c)^2}}{2}} \tag{17}$$

$$h=2\sqrt{\frac{(a+c)-\sqrt{b^2+(a-c)^2}}{2}} \tag{18}$$

其中,

$$a=\frac{M_{20}}{M_{00}}-x_c^2, b=2\left(\frac{M_{11}}{M_{00}}-x_cy_c\right), c=\frac{M_{02}}{M_{00}}-x_cy_c \tag{19}$$

2.2 结合目标区域像素值进行遮挡判断

为了当利用 Camshift 算法估计的跟踪框与运动目标检测结果偏差值较大时, 则利用目标区域的像素值与目标完整的像素值进行比较, 并利用目标的速度对

目标位置进行预估计, 将预估计的结果分别与跟踪框估计的结果和运动目标检测的结果相比较, 将相近的位置重新定义为运动目标位置.

在 Camshift 跟踪算法中, 将估计的跟踪框区域的像素数目与完整目标的的比值定义为目标遮挡参数. 通过这个比值来判断是否需要用 YOLO 算法进行目标检测, 重新确定跟踪框位置.

$$\alpha_t = \frac{I_t}{I_0} \tag{20}$$

式中, I_t 与 I_0 对应第 t 帧目标区域与完整目标的像素值. 当 α_t=0 时, 表示出现完全遮挡.

2.3 算法步骤

本文提出联合 YOLO 和 Camshift 的跟踪算法, 相对于传统 Camshift 算法在跟踪精度和鲁棒性都有较大的提高. 算法步骤如下:

1) 初始化. 利用 YOLO 算法初始化视频首帧;

2) Camshift 跟踪. 利用 Camshift 算法跟踪目标, 同时判断目标是否发生遮挡现象;

① 有遮挡: 若$\alpha_t < \beta$,其中β为遮挡阈值, 则认为目标被遮挡. 在判断出目标被遮挡后, 考虑目标的速度不是突变的, 一般处于匀速运动或者匀加速运动, 利用遮挡前的 n 帧图像所跟踪的位置信息, 二次拟合出位置与帧数的变化关系, 利用这个关系进行遮挡时的位置预估; 在目标再次被检测出来时, 需要用 YOLO 算法进行目标更新.

② 无遮挡: 若$\alpha_t \geq \beta$, 则认为目标没有被遮挡, 用 Camshift 算法继续跟踪.

联合 YOLO 算法和 Camshift 算法如图 8 所示.

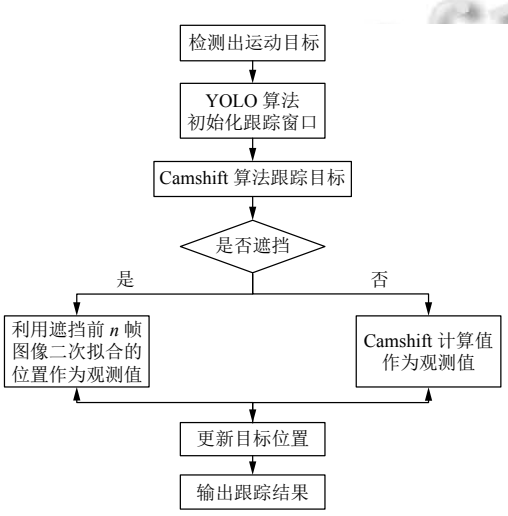

图 8 联合 YOLO 和 Camshift 算法流程图

3 实验仿真对比

实验硬件平台是 Intel(R) Core(TM) i7-6700HQ 2.60GHz CPU,8GB 内存的 PC 机, 以 Python3.6+ OpenCv3.4.2 和 Tensorflow 框架为开发平台对本文提出的跟踪算法进行实现. 选用遥控车在无遮挡和有遮挡情况下的视频进行实验仿真. 视频帧大小为 1280×720, 共 180 帧, 帧率为 30 帧/s. 实验跟踪界面如图 9.

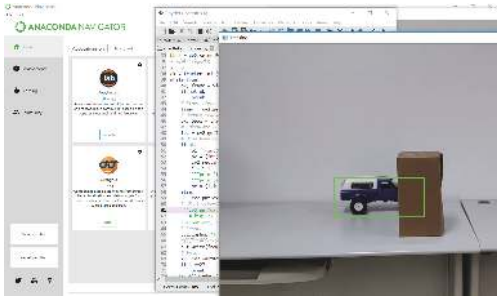

图 9 跟踪界面

3.1 定性结果对比

为了直观比较改进算法的效果, 将传统 Camshift 算法、KCF(Kernel Correlation Filter)[13]算法和本文联合 YOLO 和 Camshift 跟踪算法进行实验对比, 并分别选取部分视频帧的跟踪结果进行对比分析.

实验 1: 该视频选取遥控车未被遮挡的情况. 背景环境与遥控车的颜色信息相差较大, 对跟踪的干扰小. 图 10 至图 12 分别是遥控车未被遮挡视频序列的第 95、104、109 帧, 用矩形框来跟踪遥控车. 传统 Camshift 算法在简单背景下和未被遮挡时能够跟踪遥控车, KCF 算法也能跟踪遥控车, 但是这两种算法的目标跟踪结果不准确, 跟踪矩形框选定范围较大. 相较而言, 本文联合 YOLO 的 Camshift 算法能够准确地实现遥控车的稳定跟踪.

(a) 第95帧 (b) 第104帧 (c) 第109帧

图 10 传统 Camshift 算法的跟踪效果

(a) 第95帧 (b) 第104帧 (c) 第109帧

图 11 KCF 算法的跟踪效果

(a) 第95帧 (b) 第104帧 (c) 第109帧

图 12 联合 YOLO 和 Camshift 算法的跟踪效果

实验 2: 该视频选取遥控车在中途被纸箱遮挡的情况. 背景环境依旧与遥控车的颜色信息相差较大, 对跟踪的干扰小. 图 13 至 15 分别是遥控车中途被遮挡视频序列的第 69, 89, 159 帧, 同样用矩形框来跟踪遥控车. 传统 Camshift 算法在遥控车将要被遮挡时丢失目标, KCF 算法也在遥控车被部分遮挡时丢失目标, 而本文联合 YOLO 和 Camshift 算法能够准确地实现遥控车的稳定跟踪.

(a) 第69帧 (b) 第89帧 (c) 第159帧

图 13 传统 Camshift 算法的跟踪效果

(a) 第69帧 (b) 第89帧 (c) 第159帧

图 14 KCF 算法的跟踪效果

(a) 第69帧 (b) 第89帧 (c) 第159帧

图 15 联合 YOLO 和 Camshift 算法的跟踪效果

传统 Camshift 算法在遥控车有无遮挡两种情况下目标跟踪效果如图 10 和图 13 所示. 通过对图 13(b) 和图 13(c) 的观察, 当遥控车靠近遮挡物时, 算法的跟踪结果不准确, 跟踪框发生了较大的偏移, 彻底地丢失遥控车.

KCF 算法在实验 1 和实验 2 的效果分别如图 11 和图 14 所示. 在遥控车靠近遮挡物时, 如图 14(b) 所示, 跟踪效果没有影响. 但当遥控车受到遮挡物干扰时, 如图 14(c) 所示, 跟踪效果明显下降, 丢失跟踪目标.

联合 YOLO 和 Camshift 算法在实验 1 和实验 2 的效果分别如图 12 和图 15 所示. 在实验 2 的跟踪过程中, 当遥控车受到部分遮挡 (见图 15(c)) 时, 跟踪窗口仍能基本收敛到遥控车位置, 遥控车并没有丢失, 从而证明了本文算法的优越性.

3.2 定性结果对比

为了进一步验证联合 YOLO 和 Camshift 算法在目标被部分遮挡时处理的有效性, 从精确度和实时性两个方面进行量化的结果对比.

在算法的精确度方面, 由于遥控车在水平方向运动, 中心点 Y 轴变化较小, 所以只选取中心点 X 轴的坐标变化进行定量分析, 比较跟踪结果中每帧图像中目标的中心点 X 轴坐标与手工标记的真实值, 以对比算法的跟踪精度. 跟踪结果与真实值越接近, 跟踪精度越高.

根据实验 1、实验 2 的视频帧跟踪结果, 选取中心点的 X 轴坐标, 对 3 种算法的精确度进行对比, 结果如图 16 和图 17 所示. 通过对比结果可知, 传统的 Camshift 算法在目标靠近遮挡物时, 即跟踪框包含了遮挡物, 传统的 Camshift 算法失效, 如图 17 所示, 在第 90 帧左右线段发生阶跃. 而 KCF 算法在目标大部分被遮挡物遮挡后, 也出现了失效的情况, 如图 17 所示, 在第 140 帧左右无法准确跟踪. 相较于传统的 Camshift 算法和 KCF 算法, 联合 YOLO 和 Camshift 算法在跟踪目标被遮挡时, 其中心点的 X 轴坐标误差始终保持最低水平, 即跟踪窗口能够跟踪目标区域, 其稳定性和精准度有很好的提高.

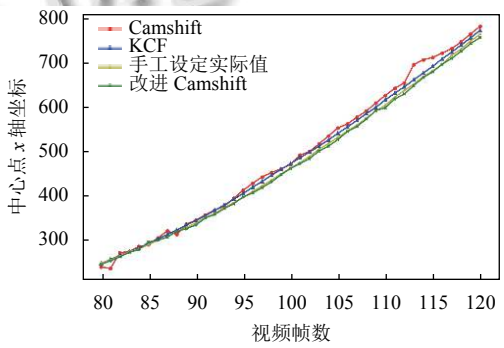

图 16 实验 1 下中心点 X 轴坐标的对比

在算法的实时性方面, 针对实验 2 的视频帧, 在有效跟踪时间内计算单帧的平均运行时间, 将其进行比较, 结果如表 1 所示.

由表 1 可知, 联合 YOLO 和 Camshift 算法采用了图像增强和 YOLO 检测, 算法耗时主要在于首帧和在

目标被遮挡时用 YOLO 算法进行目标更新, 因此相比于传统 Camshift 算法, 其复杂度更高, 单帧耗时更长, 相对应的运行速度更慢. 本文算法在仿真时能够达到 18.5 帧/s 的运行速度, 相比于 KCF 和传统 Camshift 目标跟踪算法, 算法的实时性有待提高, 但基本上满足实时跟踪的要求.

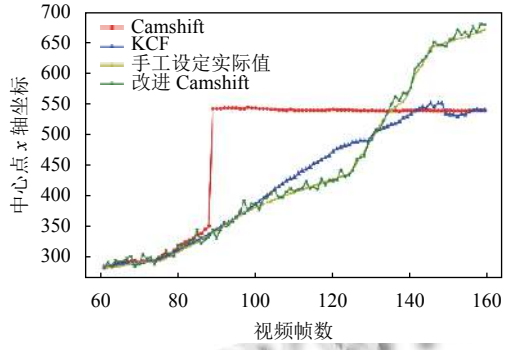

图 17 实验 2 下中心点 X 轴坐标的对比

表 1 不同算法单帧平均耗时对比

算法	单帧平均耗时 (s)
传统 Camshift 算法	0.0291
KCF 算法	0.0058
联合 YOLO 和 Camshift 算法	0.0540

4 结束语

本文为了解决了传统目标跟踪在有遮挡后无法准确跟踪的问题, 提出了一种联合 YOLO 和 Camshift 的目标跟踪算法. 在传统 Camshift 算法的框架下, 使用 Retinex 算法对视频首帧进行预处理, 并采用 YOLO 算法进行目标框确定, Camshift 进行跟踪. 通过目标检测对目标进行更新, 在部分遮挡的情况下, 仍能有效实现目标跟踪, 避免在跟踪过程中目标丢失的问题, 具有更好的抗干扰性和鲁棒性. 但它的时间复杂度较高, 下一步研究的重心是设计更加合理的卷积神经网络, 以适应目标跟踪任务, 提高运行效率.

参考文献

1 罗建豪, 吴建鑫. 基于深度卷积特征的细粒度图像分类研究综述. 自动化学报, 2017, 43(8): 1306–1318.
2 葛宝义, 左宪章, 胡永江. 视觉目标跟踪方法研究综述. 中国图象图形学报, 2018, 23(8): 1091–1107.
3 Girshick R, Donahue J, Darrell T, *et al*. Rich feature hierarchies for accurate object detection and semantic segmentation. 2014 IEEE Conference on Computer Vision and Pattern Recognition. Columbus, OH, USA. 2014.
4 Girshick R. Fast R-CNN. 2015 IEEE International Conference on Computer Vision. Santiago, Chile. 2015.
5 Ren SQ, He KM, Girshick R, *et al*. Faster r-cnn: Towards real-time object detection with region proposal networks. arXiv preprint arXiv: 1506. 01497, 2015.
6 Redmon J, Divvala S, Girshick R, *et al*. You only look once: Unified, real-time object detection. 2016 IEEE Conference on Computer Vision and Pattern Recognition. 2016.
7 Liu W, Anguelov D, Erhan D, *et al*. SSD: Single Shot MultiBox Detector. In: Leibe B, Matas J, Sebe N, et al, eds. Computer Vision-ECCV 2016. 2016. 21-37.
8 BRADSKI G R. Real time face and object tracking as a component of a perceptual user interface. Proceedings Fourth IEEE Workshop on Applications of Computer Vision. Princeton, NJ, USA, USA, 1998: 214–219.
9 郝志成, 吴川, 杨航, 等. 基于双边纹理滤波的图像细节增强方法. 中国光学, 2016, 9(4): 423–431.
10 Land EH, McCann J. Lightness and retinex theory. Journal of the Optical Society of America, 1971, 61(1): 1–11. [doi: 10.1364/JOSA.61.000001]
11 He KM, Zhang XY, REN SQ, *et al*. Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition, 2016: 770–778.
12 Yegnanarayana B. Artificial neural networks for pattern recognition. Sadhana, 1994, 19(2): 189–238. [doi: 10.1007/BF02811896]
13 Henriques JF, Caseiro R, Martins P, *et al*. High-speed tracking with kernelized correlation filters. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, 37(3): 583–596. [doi: 10.1109/TPAMI.2014.2345390]