文章编号: 1001-0920(2019)01-0144-07 DOI: 10.13195/j.kzyjc.2017.1029

基于尺度自适应均值偏移优化的TLD跟踪算法

张惊雷[1,2†], 时 鹏[1], 温显斌[3]

(1. 天津理工大学 电气电子工程学院，天津 300384；2. 天津市复杂系统控制理论及应用重点实验室，天津 300384；3. 天津理工大学 计算机视觉与系统教育部重点实验室，天津 300384)

摘 要: 为解决目标在形变、遮挡和快速运动时所导致的跟踪失败,在经典TLD算法的框架下,使用尺度自适应均值偏移算法重新设计跟踪器,提出了MS-TLD算法. 通过引入颜色直方图特征和尺度自适应,跟踪器能准确跟踪形变和快速运动的目标. 设计跟踪-检测反馈机制,通过跟踪器和检测器相互校正,使新算法在目标被遮挡时具有很好的跟踪鲁棒性. 采用TB-50标准测试集进行了实验验证与评测,结果表明所提出算法有效克服了由于目标形变、遮挡和快速运动以及背景干扰所导致的跟踪失败,比TLD等4种经典算法具有更好的跟踪准确性和鲁棒性.

关键词: tracking-learning-detection；均值偏移；尺度自适应；跟踪-检测反馈

中图分类号: TP273 **文献标志码:** A

A TLD tracking algorithm based on scale-adaptive mean-shift method

ZHANG Jing-lei[1,2†], *SHI Peng*[1], *WEN Xian-bin*[3]

(1. School of Electrical and Electronics Engineering，Tianjin University of Technology，Tianjin 300384，China；2. Tianjin Key Laboratory for Control Theory and Applications in Complicated Systems，Tianjin 300384，China；3. Key Laboratory of Computer Vision and System of Ministry of Education of China，Tianjin University of Technology，Tianjin 300384，China)

Abstract: In order to solve tracking failures caused by objects deformation, occlusion and fast motion, an algorithm called mean shift-tracking learning detection(MS-TLD) under the framework of the classical tracking-learning-detection(TLD) algorithm is proposed, which reconstructs a new tracker using the scale-adaptive mean-shift method. By introducing color histogram features and scale-adaption, the new tracker can track objects with deformation and fast moving. A new tracking-detection feedback strategy for the inter-correction between tracker and detector is designed, by which the proposed algorithm has better robustness when objects are occluded. The TB-50 standard dataset is used to verify and evaluate the proposed method. The experimental results show that the proposed algorithm can overcome the tracking failures caused by objects with deformation, occlusion, fast motion, as well as background clutters, and has better tracking accuracy and robustness compared with the TLD and other 3 classical algorithms.

Keywords: tracking-learning-detection；mean-shift；scale-adaptive；tracking-detection feedback

0 引 言

目标跟踪是计算机视觉领域的重要研究方向,在监控、人机交互和医学成像方面有着广泛应用. 目标跟踪任务是在初始帧中给定目标状态,并在后续帧中对目标状态进行实时估计. 目标跟踪在实际应用中面临多方面的挑战,如外观变形、光照变化、快速运动、运动模糊、背景相似干扰、平面外旋转、平面内旋转、尺度变化、遮挡和超出视野,这使得对任意目标进行跟踪仍然是一个困难的问题[1-2].

目前目标跟踪算法可分为生成类和判别类两种. 生成类方法的主要思想是对跟踪目标区域进行建模,并在下一帧寻找与模型相似的区域作为预测区域,典型算法如Comaniciu等[3]提出的均值偏移(mean-shift)跟踪算法. 这类算法着重对目标本身的描述,但忽略了背景信息,在目标剧烈变化或遮挡时容易产生漂移. 近年来,判别类方法由于其出色的性能,逐渐成为研究的主流. 这类算法通常在第1帧就对目标区域和背景区域进行区分并作为正负样本,提

收稿日期: 2017-08-01；**修回日期:** 2017-10-27.

基金项目: 国家自然科学基金项目(61472278).

责任编委: 张维海.

作者简介: 张惊雷(1969−), 男, 教授, 博士, 从事模式识别、图像处理等研究；时鹏(1991−), 男, 硕士生, 从事图像处理、目标跟踪等研究.

†通讯作者. E-mail: zhangjinglei@tjut.edu.cn.

取特征并运用机器学习的方法训练分类器,然后下一帧进行判别. 分类器的训练使用了背景信息,因此能够更为精确地区分前景和背景,跟踪效果普遍比传统的生成类方法好. 典型算法如Hara等[4]提出的Struck算法, Kalal等[5]提出的TLD算法, Zhang等[6]提出的压缩感知跟踪算法(Compressive tracking), Danelljan等[7]提出的CN算法(Color names), Henriques等[8]提出的核化相关滤波算法(Kernelized correlation filter, KCF). 判别类方法在目标尺度剧烈变化和快速运动时,会因为学习器学习到大量背景信息而导致漂移和跟踪失败. 近些年,深度学习开始应用于目标跟踪中,典型算法如Wang等[9]提出的SO-DLT算法(Structured output deep learning tracker), Nam等[10]提出的MDNet算法(Multi-domain network), Bertinetto等[11]提出的SiamFC算法(Fully-convolutional siamese networks),但训练样本的缺失是深度学习应用在目标跟踪中面临的最大问题.

本文提出了MS-TLD(TLD+ASMS)算法. 这是基于经典TLD算法的一种改进算法,该算法摒弃了TLD算法中易受光照变化等干扰的中值流跟踪器(基于一种改进的光流法),采用尺度自适应的均值偏移[12](Scale-adaptive mean-shift for tracking, ASMS)算法设计了新的跟踪器. 新跟踪器引入了颜色直方图特征和尺度自适应,具有更好的跟踪准确度. 同时,设计跟踪-检测反馈机制,以检测器的输出来纠正跟踪器的错误,以跟踪器的输出来提升学习器与检测器的性能. 实验表明,算法不仅保留了TLD算法对目标良好的重检测能力,而且对形变、遮挡和快速运动的目标也表现出了良好的跟踪精度和实时性.

1　经典TLD算法原理

TLD(Tracking-learning-detection)是一种在线的长时间跟踪算法. 算法由3部分组成:跟踪模块、学习模块和检测模块,如图1所示.

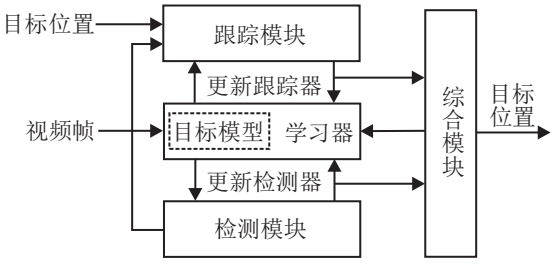


图1　TLD算法框图

经典TLD算法的跟踪器采用的是中值流跟踪法[13](Median-flow tracker), 这是一种基于光流法(Lucas-kanade tracker)的改进方法. 在t帧初始化跟踪点框,使用光流法跟踪各个点,然后将在$t+1$帧跟踪到的点再使用光流法反向跟踪到第t帧,计算t帧的初始点与反向跟踪点的欧氏距离,如果大于设定的前后向错误阈值(forward-backward error),则舍弃掉该点,利用剩余的点估计目标的移动情况. 将每个点在空间移动的中值作为目标的位移,将每个点与上一个点的位移的比率作为尺度变化,然后更新当前的框图.

基于中值流算法的跟踪器是一种短时跟踪器,它对目标的跟踪局限于前一帧和当前帧目标的位置变化,当出现跟踪失败和检测失败导致的目标丢失时,中值流跟踪器便会停止工作,直到检测器再次检测到目标. 跟踪器跟踪失败会影响学习器中目标模板的更新,降低了检测器的准确度; 同时TLD算法使用的是灰度图,更容易受到背景干扰.

检测器由3部分组成:方差分类器,组合分类器和最近邻分类器. 方差分类器通过计算搜索框与目标框的灰度值方差,将结果低于原图50%的结果抛弃,此分类器可以去除大部分背景. 组合分类器基于随机森林方法[14],对输入图像进行高斯滤波降噪,然后对图像内部随机的13对像素进行对比,产生一组13位的二进制像素编码. 通过这种方法产生10组编码,每组编码将映射到某一后验概率,然后将10组概率的结果平均,如果其结果大于0.5则通过. 最近邻分类器将通过组合分类器的图像与样本库中的样本用归一化互相关(Normalized cross correlation method, NCC)计算相似度,输出相似度大于阈值的目标.

学习器采用P-N Learning[15]半监督学习机制. P-N Learning包含两部分:P-expert和N-expert. P-expert用于识别漏检目标,即寻找正样本;N-expert用于识别误检目标,即寻找负样本. P-expert基于时间的连续性考虑,目标在当前帧的位置,必然与前一帧的位置接近,所以与目标相距位置过远的均可标记为负样本. 利用连续帧跟踪的结果产生当前目标的位置,如果该位置标记为负样本,则更改为正样本并送入训练集. N-expert基于空间性考虑,TLD是一种单目标跟踪策略,所以在每一帧中最多只存在一个可能的位置. N-expert综合跟踪器与检测器的结果,输出可能性最大的位置,并将与最可能位置不重合的区域标记为负样本,同时重新对跟踪器进行初始化.

2　MS-TLD算法

本文的MS-TLD算法是在经典TLD算法中进行了改进,整体算法框架如图2所示. 算法不仅重新设计了跟踪器,而且加入了跟踪-检测反馈机制,有效提

升了算法的跟踪精度.

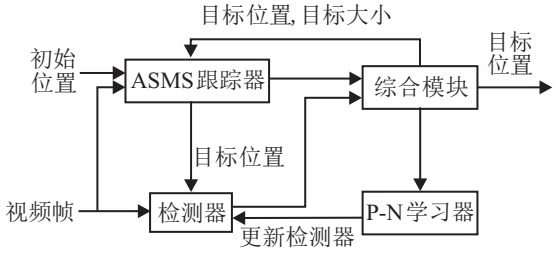


图 2 MS-TLD算法框图

2.1 尺度自适应的均值偏移算法(ASMS)跟踪器

在跟踪快速形变、尺度变化、快速运动的目标时,由于中值流跟踪法不能进行有效地跟踪,常常会导致跟踪失败. 针对此类问题,本文采用尺度自适应的均值偏移(Scale-adaptive mean-shift for tracking, ASMS)算法设计新的跟踪器. 新的跟踪器加入了颜色直方图特征和尺度估计,可以利用颜色特征和尺度特征获取更多的目标信息,具有更高的置信度. 同时尺度自适应的均值偏移算法是一种长时跟踪算法,当检测器丢失目标时,跟踪器依然可以独立运行.

尺度自适应均值偏移算法[12]是在经典的mean-shift算法框架下,加入了尺度估计和经典的颜色直方图特征,并对尺度估计进行了反向一致性检查.

2.1.1 经典mean-shift跟踪算法

经典mean-shift跟踪算法是通过分别计算目标区域和候选区域内像素的特征值概率得到关于目标模型和候选模型的描述,然后利用相似性函数度量初始帧目标模型和当前帧的候选模板的相似性,选择相似度最大的候选模型并得到关于目标模型的mean-shift向量,这个向量即是目标由初始位置向当前位置移动的向量. 由于均值偏移算法具有快速收敛性,通过不断迭代计算mean-shift向量,算法最终将收敛到目标的真实位置,实现对目标的跟踪.

根据标准的mean-shift算法,目标模型被描述为原点附近的一个在特征空间里的核密度直方图,即

$$\hat{\boldsymbol{q}}=\{\hat{q}_u\}_{u=1,2,\cdots,m},\ \sum_{u=1}^{m}\hat{q}_u=1. \tag{1}$$

在下一帧,$\boldsymbol{y}$处的候选目标模型可由如下直方图描述:

$$\hat{\boldsymbol{p}}(\boldsymbol{y})=\{\hat{p}_u(\boldsymbol{y})\}_{u=1,2,\cdots,m},\ \sum_{u=1}^{m}\hat{p}_u=1. \tag{2}$$

令$\boldsymbol{x}_i$为像素位置,n为目标像素个数,$\{\boldsymbol{x}_i^*\}_{i=1,\cdots,n}$为以目标中心为原点的像素位置,对选中的区域的灰度颜色空间均匀划分,得到由$\boldsymbol{m}$个相等区间构成的灰度直方图. 目标模型在特征$u\in\{1,2,\cdots,m\}$的概率密度可表示为

$$\hat{q}_u=C\sum_{i=1}^{n}k(\|\boldsymbol{x}_i^*\|^2)\delta[b(\boldsymbol{x}_i^*)-u]. \tag{3}$$

其中:$k(x)$为核函数;$b(\boldsymbol{x}_i)$为$\boldsymbol{x}_i$处像素所属直方图区间;u为颜色直方图索引;δ为Kronecker函数,用来判断目标区域中像素$\boldsymbol{x}_i$处的灰度值是否属于直方图中第u个单元;C为归一化系数. 因此,$\sum_{u=1}^{m}\hat{q}_u=1$.

在第t帧时,根据第$t-1$帧的目标中心位置,以$\hat{\boldsymbol{y}}_0$为搜索窗口的中心,得到候选目标的中心位置坐标$\boldsymbol{y}$,计算当前帧的候选目标区域直方图. 该区域的像素位置用$\{\boldsymbol{x}_i\}_{i=1,2,\cdots,n_h}$表示,$n_h$为候选目标区域像素个数. 使用与目标模型相同的核函数$k(x)$,候选目标概率密度可表示为

$$\hat{p}_u(\boldsymbol{y})=C_h\sum_{i=1}^{n_h}k\Big(\Big\|\frac{\boldsymbol{y}-\boldsymbol{x}_i}{h}\Big\|^2\Big)\delta[b(\boldsymbol{x}_i)-u]. \tag{4}$$

其中:h为尺度参数,C_h为归一化系数. 相似性函数用于描述目标模型和候选目标之间的相似程度,采用巴氏参数作为相似性函数,其定义为

$$\rho[\hat{\boldsymbol{p}}(\boldsymbol{y}),\hat{\boldsymbol{q}}]=\sum_{u=1}^{m}\sqrt{\hat{p}_u(\boldsymbol{y})\hat{q}_u}. \tag{5}$$

巴氏参数越大,说明两个模型越相似. mean-shift的迭代过程,是以前一帧中目标中心位置$\hat{\boldsymbol{y}}_0$作为搜索窗口的中心,不断搜索使相似函数最大的候选区域,寻找目标新的位置

$$\hat{\boldsymbol{y}}_1=\frac{\sum_{i=1}^{n_h}\boldsymbol{x}_i w_i g\Big(\Big\|\frac{\hat{\boldsymbol{y}}_0-\boldsymbol{x}_i}{h}\Big\|^2\Big)}{\sum_{i=1}^{n_h}w_i g\Big(\Big\|\frac{\hat{\boldsymbol{y}}_0-\boldsymbol{x}_i}{h}\Big\|^2\Big)}, \tag{6}$$

$$w_i=\sum_{u=1}^{m}\sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{\boldsymbol{y}}_0)}}\delta[b(\boldsymbol{x}_i)-u], \tag{7}$$

其中$g(x)=-k'(x)$,$k'(x)$为$k(x)$的导数.

2.1.2 尺度估计

假设连续帧中目标的尺度变化是各向同性的. 令$\boldsymbol{y}=(y^1,y^2)^{\mathrm{T}}$,$\boldsymbol{x}_i=(x_i^1,x_i^2)^{\mathrm{T}}$为像素位置,$N$为图像的像素个数. 图像中的目标通过一个椭圆形区域$(x_i^{*1})^2/a^2+(x_i^{*2})^2/b^2<1$描述. 目标模型特征$u\in\{1,2,\cdots,m\}$的概率密度为

$$\hat{q}_u=C\sum_{i=1}^{N}k\Big(\frac{(x_i^{*1})^2}{a^2}+\frac{(x_i^{*2})^2}{b^2}\Big)\delta[b(\boldsymbol{x}_i^*)-u]. \tag{8}$$

其中:C为归一化系数;$k(x)$为核函数,当$x\geqslant 1$时,$k(x)=0$. 令$\{\boldsymbol{x}_i\}_{i=1,2,\cdots,N}$为候选目标区域以$\boldsymbol{y}$为原点的像素位置. 使用相同的核函数,候选目标特征$u=1,2,\cdots,m$的概率密度为

$\hat{p}_u(\boldsymbol{y},h)=$

$$C_h\sum_{i=1}^{N}k\Big(\frac{(y^1-x_i^1)^2}{a^2h^2}+\frac{(y^2-x_i^2)^2}{b^2h^2}\Big)\delta[b(\boldsymbol{x}_i)-u]. \tag{9}$$

其中

$$C_h=\frac{1}{\sum\limits_{i=1}^{N}k\Big(\frac{(y^1-x_i^1)^2}{a^2h^2}+\frac{(y^2-x_i^2)^2}{b^2h^2}\Big)}, \tag{10}$$

h为候选目标的尺度参数. 通过给定的核函数和变量h, C_h可以通过下面的方式近似表示: 令n_1为目标的椭圆区域包含的像素个数, n_h为候选目标在h尺度下的椭圆区域包含的像素个数, 有$n_h = h^2n_1$. 由黎曼积分的定义可以得出

$$\sum_{i=1}^{N}k\Big(\frac{(x_i^1)^2}{a^2h^2}+\frac{(x_i^2)^2}{b^2h^2}\Big)\frac{\pi abh^2}{n_h}\approx$$
$$\int\int_{\{\frac{(x_i^1)^2}{a^2h^2}+\frac{(x_i^2)^2}{b^2h^2}<1\}}k\Big(\frac{(x_i^1)^2}{a^2h^2}+\frac{(x_i^2)^2}{b^2h^2}\Big)\mathrm{d}x^1\mathrm{d}x^2=$$
$$h^2ab\int\int_{\|\boldsymbol{x}\|<1}k(\|\boldsymbol{x}\|^2)\mathrm{d}\boldsymbol{x}. \tag{11}$$

因此$C_h \approx C\frac{1}{h^2}$, 对于任意两个$h_0$和$h_1$, 有$C_{h_1}\approx C_{h_0}\frac{h_0^2}{h_1^2}$. 目标模型和候选模型的相似度通过巴氏参数计算, 引入C_h的近似值后可得

$$\rho[\hat{\boldsymbol{p}}(\boldsymbol{y},h),\hat{\boldsymbol{q}}]\approx\hat{\rho}[\boldsymbol{y},h]=$$
$$\sum_{u=1}^{m}\Big(C_{h_0}\frac{h_0^2}{h^2}\sum_{i=1}^{N}k\Big(\frac{(y^1-x_i^1)^2}{a^2h^2}+$$
$$\frac{(y^2-x_i^2)^2}{b^2h^2}\Big)\delta[b(\boldsymbol{x}_i)-u]\hat{q}_u\Big)^{1/2}. \tag{12}$$

通过求取$\hat{\rho}[\boldsymbol{y},h]$的最大值, 前一帧的目标中心位置$\hat{\boldsymbol{y}}_0$通过迭代向新的位置$\hat{\boldsymbol{y}}_1$移动, 尺度从$h_0$变为$h_1$, 这种方式与经典的mean-shift相同.

定义

$$w_i=\sum_{u=1}^{m}\sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{\boldsymbol{y}}_0,h_0)}}\delta[b(\boldsymbol{x}_i)-u], \tag{13}$$

$$G=\sum_{i=1}^{N}w_igig\Big(\frac{(y_0^1-x_i^1)^2}{a^2h_0^2}+\frac{(y_0^2-x_i^2)^2}{b^2h_0^2}\Big), \tag{14}$$

$$\boldsymbol{m}_k(\hat{\boldsymbol{y}}_0,h_0)=$$
$$\frac{\sum\limits_{i=1}^{N}\boldsymbol{x}_iw_ig\Big(\frac{(y_0^1-x_i^1)^2}{a^2h_0^2}+\frac{(y_0^2-x_i^2)^2}{b^2h_0^2}\Big)}{G}-\hat{\boldsymbol{y}}_0. \tag{15}$$

即

$$\boldsymbol{m}_k(\hat{\boldsymbol{y}}_0,h_0)=(m_k^1(\hat{\boldsymbol{y}}_0,h_0),m_k^2(\hat{\boldsymbol{y}}_0,h_0))^{\mathrm{T}},$$

其中$g(x)=-k'(x)$. 然后可得

$$\frac{\partial\hat{\rho}[\boldsymbol{y},h]}{\partial y^1}(\hat{\boldsymbol{y}}_0,h_0)=\frac{C_{h_0}}{a^2(h_0)^2}Gm_k^1(\hat{\boldsymbol{y}}_0,h_0), \tag{16}$$

$$\frac{\partial\hat{\rho}[\boldsymbol{y},h]}{\partial y^2}(\hat{\boldsymbol{y}}_0,h_0)=\frac{C_{h_0}}{b^2(h_0)^2}Gm_k^2(\hat{\boldsymbol{y}}_0,h_0), \tag{17}$$

$$\frac{\partial\hat{\rho}[\boldsymbol{y},h]}{\partial h}(\hat{\boldsymbol{y}}_0,h_0)=$$
$$\frac{C_{h_0}}{(h_0)^2}G\Big[\frac{1}{h_0}\sum_{i=1}^{N}w_i\Big(\frac{(y_0^1-x_i^1)^2}{a^2}+\frac{(y_0^2-x_i^2)^2}{b^2}\Big)\times$$
$$g\Big(\frac{(y_0^1-x_i^1)^2}{a^2h_0^2}+\frac{(y_0^2-x_i^2)^2}{b^2h_0^2}\Big)\Big/G-$$
$$h_0\sum_{i=1}^{N}w_ik\Big(\frac{(y_0^1-x_i^1)^2}{a^2h_0^2}+\frac{(y_0^2-x_i^2)^2}{b^2h_0^2}\Big)\Big/G\Big]. \tag{18}$$

最后, 得到$\boldsymbol{y}$和h的更新值为

$$\hat{y}_1^1=\frac{1}{a^2}m_k^1(\hat{\boldsymbol{y}}_0,h_0)+\hat{y}_0^1,$$
$$\hat{y}_1^2=\frac{1}{b^2}m_k^2(\hat{\boldsymbol{y}}_0,h_0)+\hat{y}_0^2; \tag{19}$$

$$h_1=$$
$$\Big[1-\frac{\sum\limits_{i=1}^{N}w_ik\Big(\frac{(y_0^1-x_i^1)^2}{a^2h_0^2}+\frac{(y_0^2-x_i^2)^2}{b^2h_0^2}\Big)}{G}\Big]h_0+$$
$$\frac{1}{h_0}\sum_{i=1}^{N}w_i\Big(\frac{(y_0^1-x_i^1)^2}{a^2}+\frac{(y_0^2-x_i^2)^2}{b^2}\Big)\times$$
$$g\Big(\frac{(y_0^1-x_i^1)^2}{a^2h_0^2}+\frac{(y_0^2-x_i^2)^2}{b^2h_0^2}\Big)\Big/G. \tag{20}$$

2.2 改进跟踪器算法

在加入了尺度估计后, 为了防止尺度变化产生错误, 加入了对于尺度的反向一致检查. 通过计算目标从$t-1$帧到t帧的尺度和t帧到$t-1$帧尺度的误差, 当误差大于阈值时, 则认为尺度估计产生错误, 这时采用初始尺度对目标进行尺度估计.

跟踪器的算法具体步骤如下.

输入: 目标模型$\hat{\boldsymbol{q}}$, 目标起始位置$\boldsymbol{y}_0$, 目标起始大小$\boldsymbol{s}_0$;

输出: 目标在每帧的位置和大小$(\boldsymbol{y}_t,\boldsymbol{s}_t)$, 其中$t\in\{1,2,\cdots,n\}$.

对于每帧$t\in\{1,2,\cdots,n\}$, 都有:

Step 1: 通过式(9)计算候选目标模型$\hat{p}_u(\boldsymbol{y}_{t-1},h_{t-1})$.

Step 2: 通过式(19)计算目标新的位置$\boldsymbol{y}_t$.

Step 3: 通过式(20)更新目标的尺度h.

Step 4: 如果连续两帧目标位置的距离$\|\boldsymbol{y}_t-\boldsymbol{y}_{t-1}\|^2$小于阈值$\varepsilon$, 或者达到最大迭代次数maxIter, 则进入下一步; 否则返回Step 1继续进行迭代计算.

Step 5: 如果$|\log(h)|>\Theta_s$, 则说明目标尺度发生变化, 进入下一步对目标尺度进行反向一致性检查; 否则, 目标大小更新为

$$\boldsymbol{s}_t=(1-\gamma)\boldsymbol{s}_{t-1}+\gamma h\boldsymbol{s}_{t-1}.$$

Step 6: 进行反向一致性检查. 首先通过Step1 ~ Step 4计算目标从t帧到$t-1$帧的尺度变化h_{back}. 如果$|\log(hh_{\mathrm{back}})|>\Theta_c$, 说明尺度变化前后不一致, 则

目标大小更新为

$$\boldsymbol{s}_t = (1-\alpha-\beta)\boldsymbol{s}_{t-1} + \alpha\boldsymbol{s}_{\text{default}} + \beta h\boldsymbol{s}_{t-1},$$

其中$\alpha{=}c_1(\boldsymbol{s}_{\text{default}}/\boldsymbol{s}_{t-1})$.

以上算法通过C++实现. 在mean-shift迭代过程中的参数$\varepsilon = 0.1, \text{maxIter} = 15$;反向尺度检查阈值参数$\Theta_{\text{s}} = 0.05, \Theta_c = 0.1$;尺度更新中参数$c_1 = 0.1, \beta = 0.1, \gamma_1 = 0.3$;$\boldsymbol{s}_{\text{default}}$为第一帧初始化时的目标大小.

2.3 跟踪-检测反馈机制

本文算法在综合模块加入了跟踪-检测反馈机制,综合模块的目标位置作为反馈,参与跟踪器的目标位置和尺度的更新. 在目标位置更新反馈方面,当目标被遮挡和背景存在干扰时,ASMS跟踪器会产生漂移. 为了防止由于跟踪器的错误导致的跟踪失败,使用检测器对跟踪器进行校正. 当检测器未检测到目标时,将跟踪器的结果作为最终输出;当检测器仅检测到一个目标,但此目标与跟踪器的结果没有重叠时,判断此时跟踪器产生漂移,并用检测器的结果对跟踪器进行校正;当检测器检测到多个目标时,先将目标进行聚类,然后将聚类后得到的目标框和跟踪器产生的结果进行加权运算,得到最终输出,并对跟踪器进行校正. 在目标尺度更新反馈方面,检测器以初始目标尺度的10%、尺度缩放系数1.2进行21次尺度缩放,当检测器与跟踪器的输出尺度不一致时,经过综合模块的加权计算后,对跟踪器的尺度进行校正.

为了提高算法的实时性,需要尽量缩小检测器的搜索范围,以避免耗时的全图搜索. 检测器在以跟踪器反馈的目标位置为中心的两倍范围内搜索,当连续10帧检测失败时才进入整图搜索,这将显著地减少候选搜索框,提高了实时性. 综合模块得到的结果将输入学习器,更准确的跟踪结果为学习器提供更多学习样本,提升了检测器的准确度.

3 实验结果

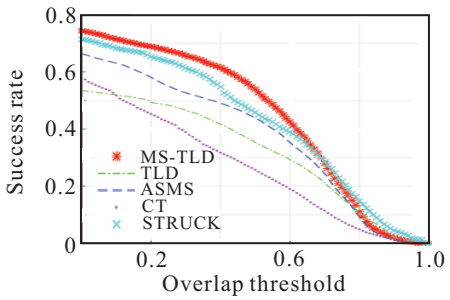
3.1 跟踪算法性能评测

为评估改进后算法的性能,本文采用TB-50的数据集对算法进行测试. 为证明算法的改进效果,同时与原TLD算法、ASMS算法、CT算法和STRUCK进行对比. 算法使用Visual Studio 2015和OpenCV3.2进行编译,运行平台为配置2.5 GHz i5处理器(4核),8 G内存的笔记本电脑.

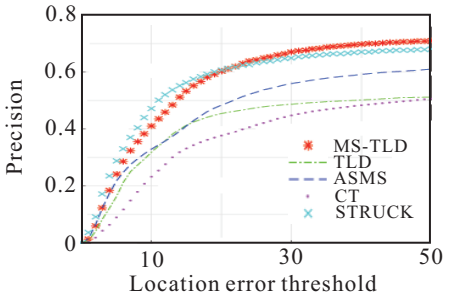
TB-50是在CVPR2013国际会议上提出的跟踪算法评测数据集,包含50个视频序列,涵盖了目标跟踪所面临的各种问题,即外观变形、光照变化、快速运动、运动模糊、背景相似干扰、平面外旋转、平面内旋转、尺度变化、遮挡和超出视野. 依照跟踪算法评测通用标准,将算法得到的目标跟踪框与人工标定的真实目标框的一致性情况进行性能评测,评测以目标跟踪框与真实目标框的重叠度和中心点的欧氏距离作为指标,综合为成功绘图和精确绘图表示. 成功绘图的x轴为两框重叠度,y轴为大于该重叠度所占的比例,通常取重叠度为50%作为典型值. 精确绘图的x轴为两框中心点的欧氏距离,y轴为小于该距离所占的比例,通常取欧式距离为20像素作为典型值. 本文对TB-50数据集中所有50个视频序列进行了评测,各算法运行的平均速度如表1所示,统计的评测结果如图3所示.

表1 各算法在TB-50数据集运行的平均速度

算法	MS-TLD	TLD	ASMS	CT	STRUCK
平均速度/fps	24.6	24.5	349.5	40.7	19.1



(a) OTB50-success plots of OPE



(b) OTB50-precision plots of OPE

图3 本文算法MS-TLD与TLD、ASMS、CT、STRUCK算法性能对比

实验测试数据包括了TB-50的全部测试视频(TOTAL)以及形变(DEF)、遮挡(OCC)、快速运动(FM)、背景干扰(BC)、尺度变化(SV)等情况下,本文算法与其他算法的测试结果对比,表2和表3分别表示测试数据中重叠度大于50%帧的占比和中心点欧氏距离小于20像素帧的占比,以这两个指标评价算法跟踪的成功率. 其中:形变部分包含Basketball、Panda等23个视频序列;遮挡部分包含Box、Tiger2等

表2 重叠度大于50%的帧占比 %

	MS-TLD	TLD	ASMS	CT	STRUCK
TOTAL	**54.87**	36.11	44.56	26.42	46.53
DEF	**43.81**	22.56	33.44	22.47	38.91
OCC	**52.05**	30.32	39.64	25.51	40.14
FM	**67.42**	40.77	57.45	27.40	52.89
BC	**46.29**	29.53	37.43	26.78	43.00
SV	**53.44**	33.47	43.13	24.25	42.38

表3　中心像素误差小于20像素的帧占比　%

	MS-TLD	TLD	ASMS	CT	STRUCK
TOTAL	59.10	44.87	47.33	36.96	**60.03**
DEF	**51.05**	31.61	39.02	31.23	47.71
OCC	**59.60**	45.99	46.80	43.76	57.41
FM	**64.80**	42.89	56.80	28.79	57.19
BC	46.07	31.83	37.73	25.81	**48.51**
SV	59.69	45.79	47.70	39.38	**60.68**

28个视频序列;快速运动部分包含Human9、Jumping 24个视频序列;背景干扰部分包含Football 、Soccer等20个视频序列;尺度变化部分包含Car4、Liquor等36个视频序列. 表格中将每种情况测试的最佳结果进行了加粗表示.

3.2　评测结果分析

图4为MS-TLD、TLD、ASMS三种算法在测试集中的几个典型序列的跟踪结果.

图 4　MS-TLD, TLD, ASMS 在若干典型视频序列的跟踪效果

在 Human9 序列中, 从第 73 帧开始, MS-TLD、TLD 和 ASMS 均跟踪失败, 但是在第 77 帧时, MS-TLD 重新检测到目标, 并重新跟踪到目标; 在 Box 序列的第 471 帧, 由于目标被严重遮挡, 导致 3 个算法跟踪失败, 但目标重新出现后, MS-TLD 和 TLD 重新跟踪到了目标, 而 ASMS 则一直跟踪失败; 在 Jumping 序列的第 79 帧中, 由于目标运动产生的模糊, 导致了 TLD 跟踪失败, 在第 97 帧, 由于背景的干扰, 导致了 ASMS 跟踪失败; 在 Panda 序列的第 91 帧, 由于目标部分遮挡, 导致 TLD 跟踪失败, 在第 568 帧, 由于熊猫路过石头导致的背景干扰使 ASMS 跟踪失败; 在 Taiger2 序列的第 28 帧, 由于复杂背景变化导致 ASMS 跟踪失败, 在第 41 帧, 由于目标形变导致 TLD 跟踪失败. 对于以上的多种情况, 本文算法都能够很

好地进行跟踪.

4 结 论

本文在经典TLD框架的基础上,使用尺度自适应的均值偏移算法重新设计了跟踪器,并设计了跟踪-检测反馈机制,提出了MS-TLD算法. 实验表明,本文算法跟踪形变、遮挡和快速运动的目标时具有更高的准确性. 本文算法在实时性方面与经典TLD算法相当,与高速的ASMS算法还有差距,如何提高算法的实时性是今后需完善的工作.

参考文献(References)

[1] Wu Y, Lim J, Yang M H. Online object tracking: A benchmark[C]. Proce of the IEEE Conference on Computer Vision and Pattern Recognition. Portland: IEEE Press, 2013: 2411-2418.

[2] Wu Y, Lim J, Yang M H. Object tracking benchmark[J]. IEEE Trans on Pattern Analysis and Machine Intelligence, 2015, 37(9): 1834-1848.

[3] Comaniciu D, Ramesh V, Meer P. Real-time tracking of non-rigid objects using mean shift[C]. Proc of IEEE Conf on Computer Vision and Pattern Recognition. Hilton Head: IEEE Press, 2000, 2: 142-149.

[4] Hare S, Golodetz S, Saffari A, et al. Struck: Structured output tracking with kernels[J]. IEEE Trans on Pattern analysis and Machine Intelligence, 2016, 38(10): 2096-2109.

[5] Kalal Z, Mikolajczyk K, Matas J. Tracking-learning-detection[J]. IEEE Trans on Pattern Analysis and Machine Intelligence, 2012, 34(7): 1409-1422.

[6] Zhang K, Zhang L, Yang M H. Real-time compressive tracking[C]. European Conference on Computer Vision. Berlin Heidelberg: Springer, 2012: 864-877.

[7] Danelljan M, Shahbaz Khan F, Felsberg M, et al. Adaptive color attributes for real-time visual tracking[C]. Proc of the IEEE Conf on Computer Vision and Pattern Recognition. Columbus: IEEE Press, 2014: 1090-1097.

[8] Henriques J F, Caseiro R, Martins P, et al. High-speed tracking with kernelized correlation filters[J]. IEEE Trans on Pattern Analysis and Machine Intelligence, 2015, 37(3): 583-596.

[9] Wang N, Li S, Gupta A, et al. Transferring rich feature hierarchies for robust visual tracking[J]. arXiv Preprint arXiv:1501.04587, 2015.

[10] Nam H, Han B. Learning multi-domain convolutional neural networks for visual tracking[C]. Proc of the IEEE Conf on Computer Vision and Pattern Recognition. Las Vegas: IEEE Press, 2016: 4293-4302.

[11] Bertinetto L, Valmadre J, Henriques J F, et al. Fully-convolutional siamese networks for object tracking[C]. European Conf on Computer Vision. Amsterdam: Springer International Publishing, 2016: 850-865.

[12] Vojir T, Noskova J, Matas J. Robust scale-adaptive mean-shift for tracking[C]. Scandinavian Conf on Image Analysis. Berlin Heidelberg: Springer, 2013: 652-663.

[13] Kalal Z, Mikolajczyk K, Matas J. Forward-backward error: Automatic detection of tracking failures[C]. The 20th International Conf on Pattern Recognition(ICPR). Istanbul: IEEE Press, 2010: 2756-2759.

[14] Ozuysal M, Calonder M, Lepetit V, et al. Fast keypoint recognition using random ferns[J]. IEEE Trans on Pattern Analysis and Machine Intelligence, 2010, 32(3): 448-461.

[15] Kalal Z, Matas J, Mikolajczyk K. Pn learning: Bootstrapping binary classifiers by structural constraints[C]. 2010 IEEE Conf on Computer Vision and Pattern Recognition(CVPR). San Francisco: IEEE Press, 2010: 49-56.

(责任编辑：孙艺红)