

一种改进型 KCF 的疲劳驾驶检测方法

纪世雨

(解放军总医院医疗保障中心, 北京 100039)

摘要: 随着人民生活水平的提高, 汽车人均持有量持续上升, 从而也就导致了交通事故的增加, 根据统计可知, 大多数交通事故都是由疲劳驾驶导致的, 因此为了减少司机因为疲劳驾驶导致的交通事故, 对司机的疲劳程度检测是十分有必要的; 目前现有的疲劳检测方法虽然种类较多, 但是大多数存在着部署困难、实时性差、检测精度不高等缺点, 因此难以在实际生活中应用; 为了克服以上缺点, 在这里提出了一种基于视觉的实时疲劳检测方法; 该方法通过优化 KCF 算法实现实时人脸跟踪, 再利用现有的人脸关键点检测手段, 实现对眼睛和嘴巴的状态识别; 最后结合 SVM 实现对驾驶员疲劳状态的检测; 实验结果表明, 提出的方法具有较好的实时性, 并且精度较高, 能够满足日常需求。

关键词: 疲劳检测; 卷积神经网络; 人脸跟踪; 人脸关键点

An Improved KCF Fatigue Driving Detection Method

JI Shiyu

(Medical security center of PLA General Hospital, Beijing 100039, China)

Abstract: With the improvement of people's living standards, the per capita car ownership continues to rise, which leads to the increase of traffic accidents. According to statistics, most traffic accidents are caused by fatigue driving. Therefore, in order to reduce the traffic accidents caused by fatigue driving, it is very necessary to detect the fatigue degree of drivers. At present, there are many kinds of fatigue detection methods, but most of them have the disadvantages of difficult deployment, poor real-time performance and low detection accuracy, so they are difficult to be applied in real life. In order to overcome the above shortcomings, the real-time fatigue detection method based on vision is proposed. This method realizes real-time face tracking by optimizing KCF algorithm, and then realizes the state recognition of eyes and mouth by using the existing face key point detection methods. Finally, the driver fatigue state is detected by SVM. Experimental results show that the proposed method has good real-time performance and high precision, and can satisfy the daily needs.

Keywords: fatigue detection; convolutional neural network; face tracking; facial feature point

0 引言

根据现有资料显示, 世界上 30%~40% 的交通事故是由疲劳驾驶导致的。因此, 疲劳驾驶是交通事故中的一个重大隐患。近年来, 疲劳驾驶检测系统已成为研究的热点^[1-7]。检测方法分为主观检测和客观检测^[8]。在主观检测方法中, 驾驶员必须参与评价, 评价通过自问、评价、填写问卷等步骤与驾驶员的主观感受相联系。然后, 这些数据被用来估计疲劳驾驶

者驾驶的车辆, 帮助驾驶员相应地计划他们的时间表。然而, 客观检测方法不需要驾驶员的反馈, 因为它实时监测驾驶员的生理状态和驾驶行为特征^[9]。收集的数据用于评估驾驶员的疲劳程度。此外, 目标检测又分为接触检测和非接触检测。与接触式相比, 非接触式更便宜、更方便, 因为系统只需要利用计算机视觉技术和摄像头即可, 所以可以在更多的汽车上使用该种设备^[10]。在本研究中, 提出一种非接触式的驾

收稿日期:2021-05-21; 修回日期:2021-07-12。

基金项目:全国教育科学国防军事教育学科“十二五”规划教育部重点课题(DRA110425)。

作者简介:纪世雨(1980-),男,河南鹤壁人,学士,主要从事军事高技术应用与管理与信息工程方向的研究。

引用格式:纪世雨.一种改进型 KCF 的疲劳驾驶检测方法[J].计算机测量与控制,2022,30(1):52-59.

驶疲劳检测方法，称为 Caring。该方法只使用车载摄像头，使用每个帧图像来分析和检测驾驶员的状态。

1 系统概述

整个 Caring 检测系统的构建是基于一个商用车载摄像设备，一个云端服务器，其作用用于处理上传的数据，以及一部商业手机，用来接收云端传回的处理结果。图 1 显示了整个 Caring 检测系统的结构。

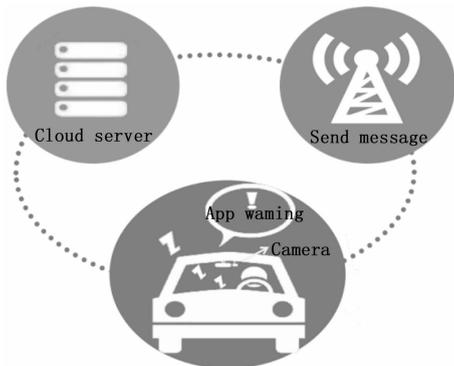


图 1 Caring 检测系统的整体架构

在开车的时候，车载摄像头捕捉到司机的头像，并将视频流实时上传到云服务器。然后，云服务器对视频进行分析，检测司机的疲劳程度。在这一阶段，主要分析了三个部分：人脸跟踪，人脸特征识别和驾驶员疲劳状态。为了满足系统的实时性要求，采用 M-KCF 算法对驾驶员的人脸进行跟踪，并基于关键点检测识别出人脸关键区域。然后，当眼睛和嘴巴的状态发生变化时，云服务器评估司机的状态。工作流如图 2 所示。最后，云服务器将结果发送到司机的手机上或应用程序中，如果司机被检测到处于昏昏欲睡状态，则通过手机或应用程序发送警告音。

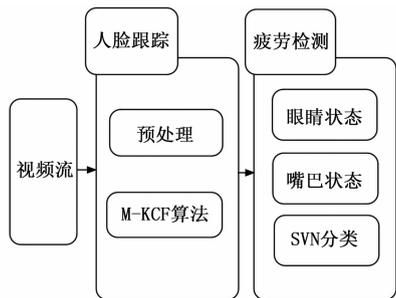


图 2 Caring 工作流程

2 基于 M-KCF 算法的脸部跟踪

在本节中，将说明使用 Caring 进行驾驶员面部跟踪的原理。由于真实环境的复杂性，视频数据的每

一帧都需要经过预处理才能满足跟踪要求。

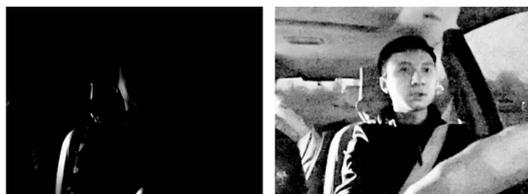
2.1 预处理

在检测过程中，驾驶室的光照强度在驾驶过程中会发生变化，这会影响图像的质量和人脸的特征。这种情况通常发生在阴天、下雨和夜晚。为了提高检测精度，采用了光照增强的方法对图像进行预处理，然后再对驾驶员的面部进行跟踪。此外，使用直方图均衡化（HE）算法来提高图像帧的亮度。

为了确定图像帧是否需要光增强，Caring 首先会评估图像的亮度。因此，第一步需要将 RGB 图像转换为 YCbCr 图像，因为在 YCbCr 颜色空间中，Y 表示亮度值^[11-12]。在此利用公式（1）计算图像中驾驶员面部周围 Y 的平均值 M ，如下所示：

$$M = \frac{\sum_{i=1}^n L}{n-i} \tag{1}$$

其中： L 表示 YCbCr 空间中每个像素的亮度值， n 和 i 分别表示图像中驾驶员面部像素的第一和最后序列号。 $n-i$ 是驾驶员面部像素的总数。如果 M 小于阈值，则图像使用 HE 算法增强图像亮度。否则，图像将保持不变。在对大样本进行计数后，将阈值设置为 60。图 3 显示了对光照不足的图像进行光照增强的前后对比。



(a) 亮度增强前 (b) 亮度增强后

图 3 光照增强前后对比

2.2 M-KCF 的原理

根据先前研究可知，KCF 算法主要是基于 HOG 特征^[13-15]。因此，在复杂的环境中，或者长时间运行，人脸检测的窗口会出现漂移，这将导致后续检测的失败，如图 4 (b) 所示。为了改进这项缺点，在此提出了一种新的人脸跟踪算法 M-KCF，来取代原有的 KCF 算法。在 M-KCF 算法中，利用了一种新的 CHOG 特征，该特征是在原有的 HOG 特征的基础上结合了 CNN 特征。接下来将具体解释 M-KCF 的原理。

2.2.1 对 CHOG 特征的提取

2.2.1.1 HOG 特征提取

在 M-KCF 算法中，CHOG 特征是人脸跟踪的



图 4 KCF 算法跟踪漂移

基本要素，其中在 CHOG 特征中，HOG 是其重要组成部分。所以需要先对 HOG 特征进行提取。为了便于计算，在首先需要将原始图像转化为灰度图像。然后，利用公式 (2) 计算图像中每个像素点的水平梯度和垂直梯度，从而得到该像素点的梯度值，最后同时得到该点的方向，公式 (2) 如下所示：

$$\begin{cases} G_x = h(x+1, y) - h(x-1, y) \\ G_y = h(x, y+1) - h(x, y-1) \end{cases} \quad (2)$$

在公式中， h ， G_x 以及 G_y 分别表示点 (x, y) 的像素值，水平梯度值和垂直梯度值。在得到计算结果后，接下来利用公式 (3) 计算点 (x, y) 的最终梯度值和其方向，具体公式如下：

$$\begin{cases} G = \sqrt{G_x^2 + G_y^2} \\ \alpha = \arctan\left(\frac{G_y}{G_x}\right) \end{cases} \quad (3)$$

在公式 (3) 中 G 为像素点在 (x, y) 位置的梯度值， α 为在该位置的方向角度。

接下来通过参考文献 [16] 可知，需要计算特征直方图。梯度方向按照敏感程度分为敏感和不敏感方向，其中敏感方向一共有 9 个，不敏感方向有 18 个。为了计算对应的直方图，需要将图像分割成若干个单元格。如果单元中的像素点属于对应的梯度方向，那么此方向的计数值将增加 1，最终每个单元都有九维和十八维特征直方图。

更重要的是，每个单元的梯度还与周围 4 个相邻

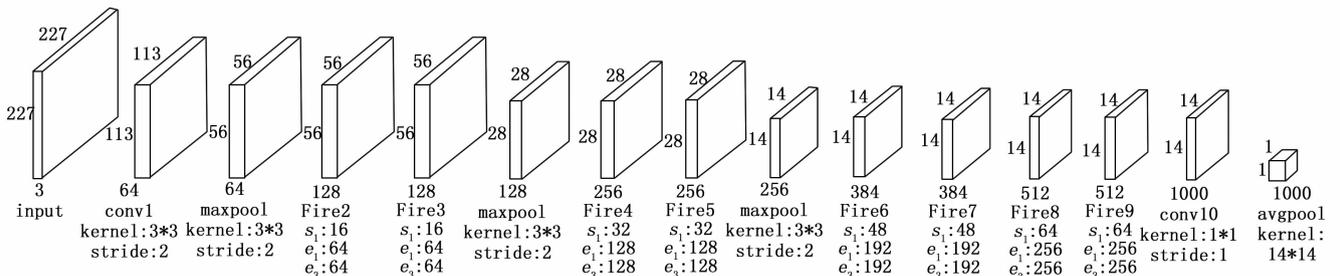


图 5 SqueezeNet 1.1 网络模型结构

单元的梯度值有关。所以需要进行归一化和截断操作，才能将这些梯度特征串联起来。具体公式如公式 (4) 和 (5) 所示：

$$N_{a,b}(j, k) = (C(j, k)^2 + C(j+a, k)^2 + C(j+a, k+b)^2 + C(j, k+b)^2)^{1/2} \quad (4)$$

在公式 (4) 中， $C(j, k)$ 表示在 (j, k) 处的单元所对应的特征向量，该向量可能为 9 或 18 维。 $N_{a,b}(j, k)$ 是归一化结果， a 和 b 是不同归一化的相关编号，共有 4 种组合，分别为 $\{-1, -1\}$ ， $\{+1, -1\}$ ， $\{+1, +1\}$ ， $\{-1, +1\}$ 。

$$F = \begin{bmatrix} T_\alpha(C(j, k)/N_{-1,-1}(j, k)) \\ T_\alpha(C(j, k)/N_{+1,-1}(j, k)) \\ T_\alpha(C(j, k)/N_{+1,+1}(j, k)) \\ T_\alpha(C(j, k)/N_{-1,+1}(j, k)) \end{bmatrix} \quad (5)$$

公式 (5) 为截断处理，在公式中 F 是在 (j, k) 处的特征向量， $T_\alpha(x)$ 表示截断操作。如果 x 中的值大于 α ，则将该值指定给 α 。

经过以上操作以后，一共可以得到 108 维特征向量，其中原来的 9 维特征向量扩展成 36 维。同理，18 维特征向量扩展到 72 维。然后将这些特征向量排列成矩阵，整个矩阵共有 4 行 27 列。接下来通过矩阵加法可以得到最后的 HOG 特征，该 HOG 特征共有 31 维。

2.2.1.2 CNN 特征的提取

由于单一的 HOG 特征会限制算法性能，造成 KCF 算法在复杂环境中，跟踪结果较差。因此 M-KCF 中还选用了 CNN 特征。相对于 HOG 特征，常规 CNN 模型提取特征的速度较慢，所以为了满足系统实时性以及保证人脸跟踪速度等相关要求，所以在此需要选用小型的 CNN 模型，在此使用的是 SqueezeNet 1.1 网络模型，整个网络模型的结构如图 5 所示。

从图 5 中可以知道，squeezeNet 的输入为 $227 \times$

227 的图像。输出为 1 000。整个网络包括 2 个卷积层，4 个池化层，以及 8 个 Fire 层，其中 Fire 层代替了传统网络中的卷积层。在 M-KCF 算法中，为了避免大量冗余的 CNN 特征，主要是选用了 Fire 5 和 Fire 9 层进行 CNN 的特征提取，具体图像如图 6 所示。

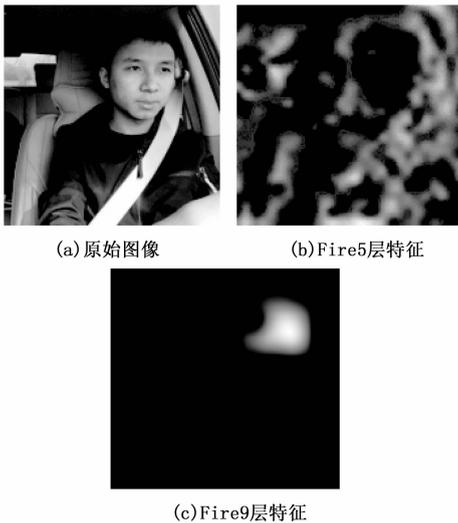


图 6 基于 CNN 的特征

2.2.2 HOG 和 CNN 特征融合

首先利用 C-KCF 算法对 $D_1 \times G_1$ 原始图像进行特征提取，得到 $D_2 \times G_2$ 的 HOG 特征和 $D_3 \times G_3$ ， $D_4 \times G_4$ 的 CNN 特征。显然，这三个特征的大小不同。为了方便后续处理，所以需要我们对它们的尺寸进行调整，使它们具有相同的尺寸。因此，调整公式如公式 (6) 所示。

$$\begin{cases} D = D_a \times \theta \\ G = G_a \times \varphi \end{cases} \quad (6)$$

在公式 (6) 中， D 和 G 分别表示标准长度和宽度。 D_a 和 G_a 分别表示三个特征的原始长度和宽度。 θ 和 φ 是尺寸因子。

类似于 KCF 算法的结构，在 M-KCF 算法中，使用通过使用核岭回归和结合每一个特征去训练它们的分类器，具体如公式 (7) 所示。

$$\vec{\alpha} = \vec{\hat{\alpha}} = \vec{y}(\mathbf{K} + \lambda \mathbf{I})^{-1} = \frac{\vec{y}}{k^{xx} + \lambda} \quad (7)$$

其中： \mathbf{K} 为核矩阵， \mathbf{I} 为单位矩阵， $\vec{\alpha}$ 为系数向量 α_i ， λ 为超参数， \vec{y} 为回归目标向量。此外，上标 \wedge 表示向量的 DFT， k^{xx} 是核矩阵 \mathbf{K} 的第一行。

训练后，我们使用每个分类器来评估每个图像样

本 z 的回归函数 $f(z)$ ， $f(z)$ 的最大值是每个特征的目标预测位置。具体计算公式如公式 (8) 所示。

$$f(z) = \mathbf{F}^{-1}(k^{xx} \odot \hat{\alpha}) \quad (8)$$

在这里 \mathbf{F}^{-1} 表示逆 DFT。因此，一共获得了三个跟踪结果。为了获得 MC-KCF 算法的最终结果，我们为 δ_1 ， δ_2 和 δ_3 这三个特征的结果设置了不同的权重。使用权重和基于 HOG 和 CNN 特征的预测位置来计算 MC-KCF 算法 T 的整个响应值。计算公式如下：

$$T = \delta_1 \times f(z_{hog}) + \delta_2 \times f(z_{fire5}) + \delta_3 \times f(z_{fire9}) \quad (8)$$

在公式 (8) 中， δ 的共域为 $[0, 1]$ 。当 $\delta = 0$ 时，对应特征的响应值不是最终结果；否则，当 $\delta = 1$ 时，对应特征的响应值是整个响应值。从响应值中，我们得到了驾驶员面部的位置。三种特征的权重不同会影响跟踪精度。因此，统计了 1 000 个不同权重比的三个特征的结果，结果如图 (7) 所示。从图中可知，当 $\delta_1 : \delta_2 : \delta_3$ 的比值分别为 0.57 : 0.14 : 0.29 时，性能最佳。所以在 Caring 系统中，比率设置为 0.57 : 0.14 : 0.29。

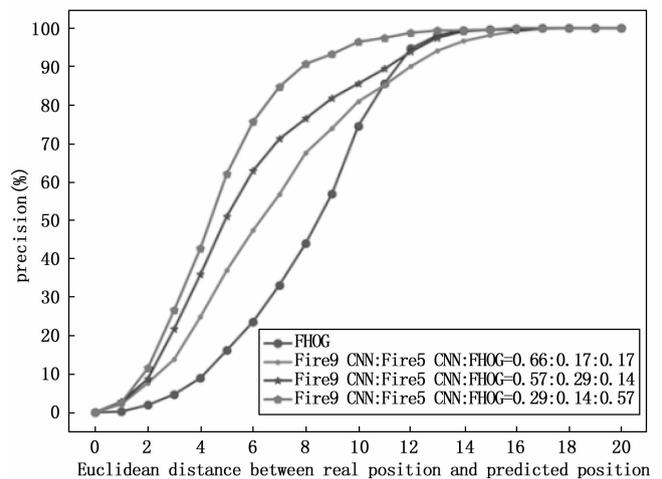


图 7 不同权重比的结果

由于 CNN 维数为 384，是大于 HOG 维数的。并且由于在视频中，检测对象连续两帧的变化较小。所以为了提高实时性，间隔 N 帧更新一次模型。从而提高系统的实时性。在 Caring 系统中， N 的值设为 3，整个过程如图 8 所示。

2.2.3 M-KCF 算法校准

如上所述，KCF 算法无法自动获得第一视频帧的跟踪目标。除此以外，由于存在被检测者不在摄像机范围内，所以需要我们对 M-KCF 算法进行校准，以

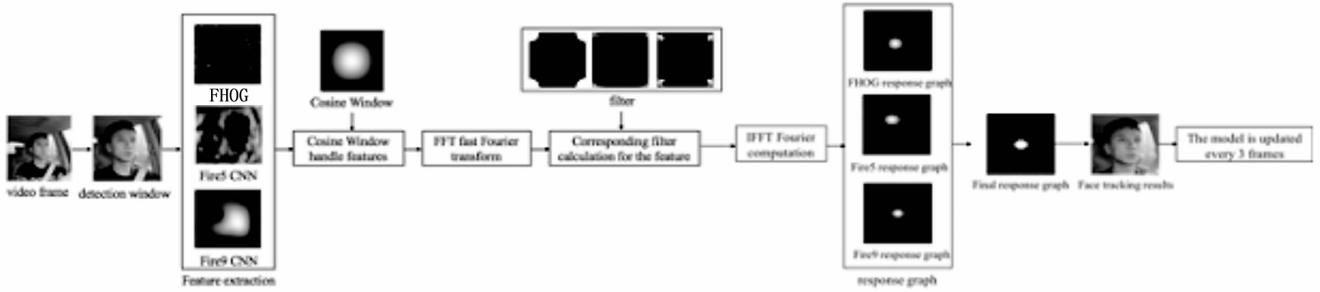


图 8 M-KCF 算法流程

防算法无法跟踪驾驶员的面部。在此使用 MTCNN 定期校准 M-KCF 算法。

云服务器对视频帧进行预处理后,判断当前图像是否为第一帧。如果是,云服务器将使用 MTCNN 算法在图像中定位人脸;否则,云服务器将继续判断跟踪时间跨度是否超过 10 s;如果回答是,云服务器将使用 MTCNN 算法重新定位人脸并重置跟踪时间。如果系统评估当前图像不是第一帧并且跟踪时间的持续时间小于 10 s,则 Caring 将使用 MC-KCF 算法使用结果来跟踪驾驶员的面部,以更新下一帧对驾驶员面部的搜索范围。整个算法流程如图 9 所示。

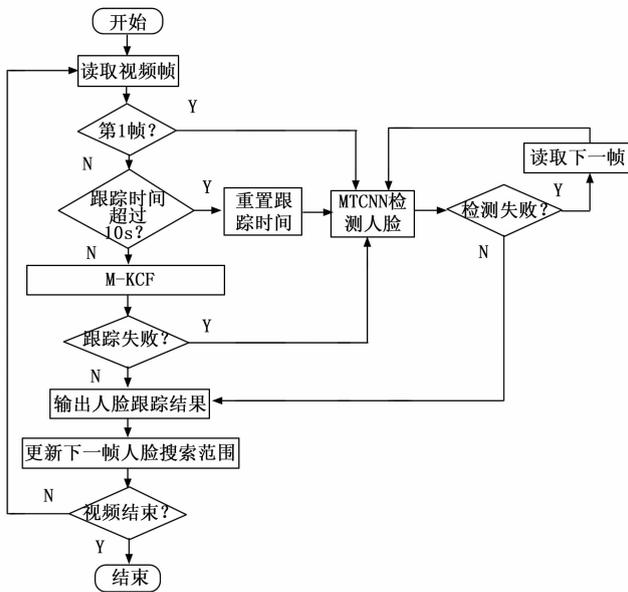


图 9 M-KCF 算法校准过程

3 司机疲劳状态判断

Caring 在用 M-KCF 算法对人脸进行实时追踪后,接下来需要进行司机疲劳状态的判断。在 Caring 中主要是通过脸部关键点确定嘴巴和眼睛状态,从而实现对疲劳的检测。具体方法将在本节中详细

说明。

3.1 脸部关键点

疲劳检测的第一步是对每帧图像进行人脸关键点检测。在此使用了基于 OpenCV 自带的 Dlib 库,该库在人脸地标识别过程中使用了方向梯度直方图和线性支持向量机。从面部提取 68 个坐标,包括眼睛、眉毛、鼻子、嘴和下颌的点。在 IBUG300-W 数据集^[17]上训练了一个预先训练过的检测器,以从任何一张脸上找到 68 个点,如图 10 所示。

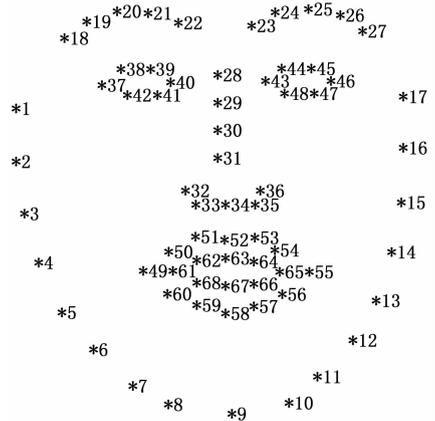


图 10 人脸 68 个点示意图

3.2 疲劳判断

使用图 11 中的眼睛标记检测哈欠,使用图 12 中的嘴巴坐标检测眼睛睁闭,分别利用公式 (9) 和 (10) 计算眼睛长宽比 (EAR) 和嘴巴长宽比 (MAR)。

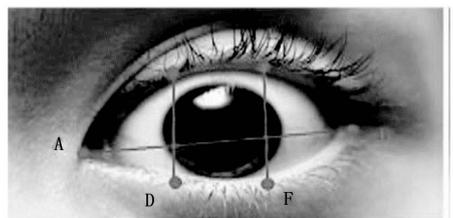


图 11 眼睛标记

$$EAR = \frac{\|CD\| + \|EF\|}{2 \times \|AB\|} \quad (9)$$

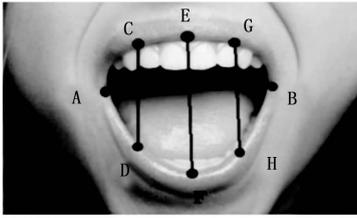


图 12 嘴巴标记

$$MAR = \frac{\|CD\| + \|EF\| + \|GH\|}{3 \times \|AB\|} \quad (10)$$

在公式 (9) 和公式 (10) 中, AB , CD , EF , GH 分别为图 11 和图 12 中的点的坐标之间的距离。可以用公式 (11) 进行计算。

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (11)$$

当眼睛睁开时, EAR 的值几乎是恒定的, 当眼睛闭上时, EAR 值变为零。当嘴闭上时, MAR 值为零, 但当嘴大开时, MAR 值增加。

3.3 SVM 分类方式

支持向量机 (SVM) 是一种有监督的学习方法, 它从一组标记的训练数据中生成输入输出映射函数。在开始的时候, 我们使用 EAR 和 MAR 的阈值来测试睡眠系统: 如果 EAR 值小于 EAR 阈值或者如果 MAR 值大于 MAR 阈值的帧数, 那么 Caring 系统会认为驾驶员处于疲劳状态。虽然由于人脸特征在不同用户之间是不同的, 设置全局阈值并不容易, 但一旦自然的表情如说话、微笑、惊讶等, 往往会使 EAR 变小、MAR 增加, 因此采用支持向量机 (SVM) 代替阈值作为分类学习算法。

然后利用 EAR 和 MAR 特征对支持向量机进行训练, 对驾驶员的状态进行分类, 可以检测出清醒、打呵欠和闭目 3 类。为了避免在驾驶员闭目时将眨眼视为驾驶员的微睡眠状态, 计算了一个新的特征, 即闭眼持续时间, 并给出了眨眼和微睡眠两类。因此, 我们可以识别每一个眨眼, 并将其与午睡或瞌睡区分开来。如果预测到打哈欠或微睡眠, 则会激活警报提醒驾驶员, 并在手机屏幕上显示通知。

4 实验与评估

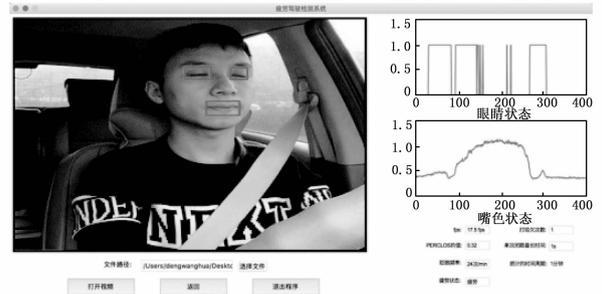
4.1 数据集和实验设置

图 13 展示 Caring 系统的原型, 整个系统是由一台型号为 S650 的 360 行车记录仪充当车载摄像头, 一台 Alienware M15 笔记本充当云服务器, 其配置为英特尔 I7 处理器, 16GB 的 DDR4 内存, 以及英伟达 GTX

2060OC 6G 显卡, 一台 iPhone 7 手机充当接收端。



(a) Caring 系统原型设置



(b) Caring 实时视频检测界面

图 13 Caring 系统展示

整个实验中, 采集了 10 名志愿者的驾驶数据, 每段视频时长 2 小时。其中志愿者需要在视频中模拟疲劳, 清醒等状态, 尽可能接近真实驾驶中司机会有有的状态。同时采用原始 KCF 算法的数据集来评估 M-KCF 算法。

4.2 M-KCF 算法性能评估

M-KCF 算法检测结果如图 14 所示, 从图 14 中可以发现 M-KCF 算法比原始 KCF 算法在检测过程中更加契合人脸, 效果较好。



(a) 男性志愿者

(b) 女性志愿者戴眼镜

图 14 M-KCF 和 KCF 检测结果对比

此外为了进一步评估 M-KCF 算法性能, 在此还利用目标边界的预测值与实际值之间的欧氏距离来对算法进行评估。将 M-KCF 算法与其他使用不同场景的跟踪算法进行了比较。主要的测试场景是快速运动, 目标在视野中消失, 目标旋转。如图 15

(a) 所示, 将每个场景中的平均测试结果计算为最终实验结果。

从图 15 (a) 可以看出, M-KCF 算法具有最佳的跟踪精度。在复杂环境下, M-KCF 的准确度接近 90%。驾驶环境中的人脸跟踪比其他环境中的人脸跟踪更简单, 因为驾驶员的面部移动更少, 速度也更平均。此外, 面部在视野中是可见的。图 15 (b) 显示了 M-KCF 测试性能和其他跟踪算法的结果, 显示当欧氏距离在 20px 内时, M-KCF 算法产生最佳性能, 精度达到约 95%。

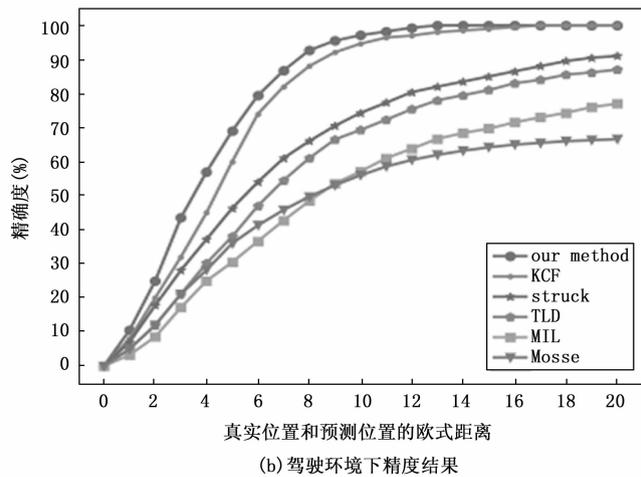
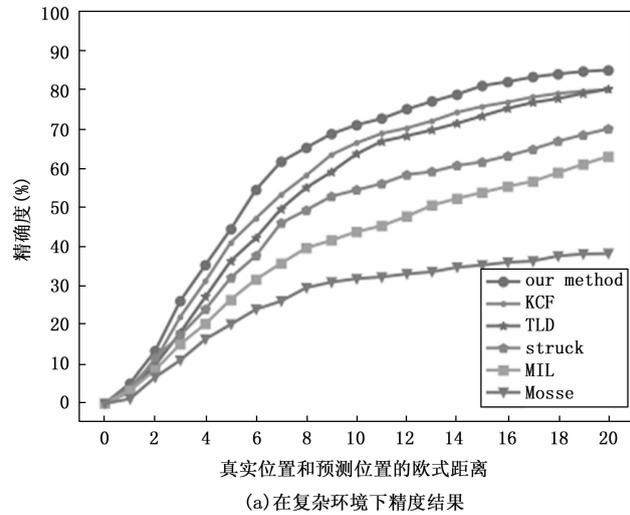


图 15 M-KCF 算法与其他算法对比

如表 1 所示, 进一步比较了不同方法的速度。KCF 算法虽然速度最快, 但其精度比 M-KCF 和 KCF+CNN 算法差。M-KCF 算法对人脸跟踪的精度最高, 比 struck 算法的精度高出近 20%, 但速度略低于 KCF 算法。M-KCF 算法每秒可处理 26 帧视频, 满足了系统的要求。因此, 认为 M-KCF 算

法性能更好, 并提供了速度和精度的实际要求。

表 1 与其他算法的性能对比

方法	视频尺寸	人脸尺寸大小	精确度/%	FPS/fps
MTCNN	1 280×720	240×360	92.9	4
KCF			91	187
DSST			85	6
Struck			75	8
KCF+CNN			92.5	27
M-KCF			96	26

4.3 Caring 性能评估

由于目前没有公开的疲劳驾驶数据集, 所以在此只能利用现在有的最新相似的疲劳检测手段与 Caring 对比, 对比结果如表 2 所示。

表 2 疲劳驾驶检测方法对比结果

方法名	主要手段	精确度/%
文献[18]	HOG+SVM+CNN	90.2
文献[19]	Adaboost+ERT+CNN	95
文献[20]	AdaBoost	92
Caring	SVM+Dlib	96

从表中可以看到, Caring 是所有 4 种检测方法中, 精确度最高的, 尤其是相对于文献 [11], 精确度提高了近 4%。所以 Caring 检测方法能够满足日常检测需求。

5 结束语

为了实现对驾驶员的疲劳程度检测, 因此基于人脸跟踪和关键点检测的基础上, 提出了一个实时疲劳检测方法 Caring。在该方法中设计了一种新人脸跟踪算法 M-KCF 来跟踪驾驶员的面部, 该算法主要是利用 CNN 和 MTCNN 对原有的 KCF 算法进行了改进。然后 Caring 基于人脸关键点提取了驾驶员疲劳特征, 利用 SVM 进行分类, 最终实现了对疲劳的检测。

参考文献:

[1] 邹昕彤, 王世刚, 赵文婷, 等. 基于眼睛与嘴部状态识别的疲劳驾驶检测 [J]. 吉林大学学报 (信息科学版), 2017, 35 (2): 204-211.

[2] 张 丞, 何 坚, 张 岩, 等. 基于脑电与眨眼频率的可穿戴疲劳驾驶检测系统 [J]. 计算机工程, 2017, 43 (2): 293-298+303.

[3] DUA M, SHAKSHI, SINGLA R, et al. Deep CNN models-based ensemble approach to driver drowsiness

