计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2018,27(8):28−34 [doi: 10.15888/j.cnki.csa.006525]

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

基于深度学习的运动目标实时识别与定位①

童基均, 常晓龙, 赵英杰, 蒋路茸

(浙江理工大学 信息学院, 杭州 310018)
通讯作者: 蒋路茸, E-mail: jianglurong@zstu.edu.cn

摘　要: 针对人体运动目标的实时检测与定位问题, 采用深度学习的方法进行研究. 在 Caffe 框架下, 采用 SSD (Single Shot multibox Detector) 检测方法. 以 VGG16 作为基础网络模型, 增加额外特征卷积层, 提取多尺度的卷积特征. 然后对实验数据集进行迭代训练, 得到运动目标检测模型. 利用训练好的模型, 通过 2 路摄像机检测运动目标, 并双目视觉定位. 实验结果表明, 整个系统运行速度可达 40 fps, 在 10 m×10 m 的场景下, 平均定位误差在 6 cm 以内, 在速度和精度上均有很好的表现, 为大中型场景的人体运动实时检测定位问题提供了有效的解决方案.

关键词: 深度学习; Single Shot multibox Detector (SSD); 实时检测; 双目视觉定位

引用格式: 童基均,常晓龙,赵英杰,蒋路茸.基于深度学习的运动目标实时识别与定位.计算机系统应用,2018,27(8):28–34. http://www.c-s-a.org.cn/1003-3254/6525.html

Real-Time Detection and Positioning of Moving Target Based on Deep Learning

TONG Ji-Jun, CHANG Xiao-Long, ZHAO Ying-Jie, JIANG Lu-Rong

(School of Information Science and Technology, Zhejiang Sci-Tech University, Hangzhou 310018, China)

Abstract: Aiming at the issues of real-time detection and positioning of movement target, a method of deep learning is proposed. The Single Shot multibox Detector (SSD) detection method is used under Caffe framework, the VGG16 model is used as the basic network model, and the additional feature convolution layers are used to extract the multi-scale convolution features. Then the experimental data set is iteratively trained to get the motion target detection model. The moving objects are detected using the trained model and then positioned through binocular vision positioning method. The experiment results show that the system can reach 40 fps. In the 10 m×10 m scene, the average positioning error is within 6 cm. The system has sound performance both in speed and precision, which provides an effective solution for the real-time detection and positioning of human motion in large and medium-sized scenes.

Key words: deep learning; Single Shot multibox Detector (SSD); real-time detection; binocular visual positioning

1　引言

目标检测与定位是计算机视觉领域中一个重要的研究课题[1]. 检测出感兴趣目标及其位置是计算机视觉科研工作者关注的重点[2]. 传统的检测方法通过特征提取、多特征融合进行目标检测. 例如通过提取 HOG[3]、LBP[4,5]和 SIFT[6]特征, 将提取到的特征通过 SVM[7]或者 AdaBoost[8]等分类器进行分类识别. 但随着场景的变换、光照的影响以及实时性要求等使得传统方法无法满足人们的需求.

近几年深度学习由于其快速的处理能力和较高的准确率在计算机视觉中得到广泛应用, 使得在一些复杂条件下利用视觉对场景进行分析成为可能. 2012 年 Krizhevsky 等[9]采用 AlexNet 网络的卷积神经网络 (Convolutional Neural Networks, CNN) 在 ImageNet 图

① 基金项目: 浙江省重点研发计划 (2015C03023); 浙江理工大学“521 人才培养计划”
Foundation item: The Key Research and Development Program of Zhejiang Province (2015C03023); “521 Talent Development Program” of Zhejiang Sci-Tech University
收稿时间: 2017-12-01; 修改时间: 2017-12-21; 采用时间: 2018-02-26; csa 在线出版时间: 2018-07-28

像分类中取得了最好的成绩. 最初将深度学习应用到目标检测的是 Girshick[10]等提出的 R-CNN (Region-based Convolutional Neural Networks) 方法. R-CNN 将区域建议 (region proposal)[11,12]和卷积神经网络相结合, 但是在速度与精度上还无法满足人们的需求. 而后, He[9]等人提出 SPP-Net、Girshick[13]等人提出 Fast R-CNN 以及 Ren[14]等人提出的 Faster R-CNN 都是在最开始的 R-CNN 的基础上进一步的改进, 且都是基于区域建议框, 因此其速度都受到一定限制. 针对此问题, 有研究者提出了一系列的基于无区域建议的方法: 如 Redmond[15]等提出的 YOLO (You Only Look Once) 已经达到了实时检测的效果, 但是检测精度确不够理想, 随后 Liu[16]等人提出 SSD (Single Shot multibox Detector) 模型,在提高检测精度的同时也兼顾到了实时性的要求, 可以说是一个相对而言比较理想的一个算法. 本文就是基于此算法构建了人体运动检测模型.

2 系统架构

本系统首先用不同的视觉标记物对运动目标进行一定的标记, 并通学习训练相应的模型; 接着通过深度学习技术对穿戴标记物的运动人体进行检测和识别, 文中标记物为自己设计的不同颜色和纹理的帽子, 通过检测人体所佩戴的标记物来唯一识别每一个个体, 便于后面做定位, 因此只需对标记物进行训练, 当人员更换时只需佩戴相应的视觉标记物而不需要重新训练; 最后利用立体视觉定位算法对其进行定位, 从而实现人体运动目标实时检测与定位.

本视觉定位系统通过 2 路高清网络摄像机获取视频码流, 直接输出数字信号, 省去了图像采集卡将模拟信号转化为数字信号的操作. 视频实时流首先通过解码服务器将码流转换为能够处理的图像格式, 然后在分析管理服务器上进行多路视频同步、检测以及定位等操作, 将结果反馈给客户端, 分析管理服务器同时也负责接受客户端的控制指令. 视频检测流程图如图 1 所示.

2.1 多线程并行视频流采集

视频采集端采用 2 个网络摄像头进行采集, 利用 ffmpeg 中的解码库, 将相机输出的 rtsp 实时流通过解协议、解封装最终将视频解码成在 RGB 颜色空间上的图像格式, 同时利用 pthread 多线程处理库, 通过互斥锁和信号量实现多相机并行采集以及同步问题.

2.2 人体运动目标检测

目标检测部分采用深度学习框架 Caffe[17], 利用 SSD 物体检测模型进行运动人体的检测. 该方法既保证了目标检测的精度, 又保证了目标检测的速度. SSD 方法的核心是使用小的卷积滤波器来预测特征图上固定的一组默认边界框的类别分数与位置偏移量. 从不同尺度的特征图上产生不同尺度的预测, 并通过不同的宽高比来正确的进行预测. 一个单次检测器 SSD 用于多类别目标检测, 比 YOLO 速度更块, 和使用区域建议、pooling 技术的检测方法 (faster R-CNN) 一样准确. SSD 目标检测的原理如图 2 所示.

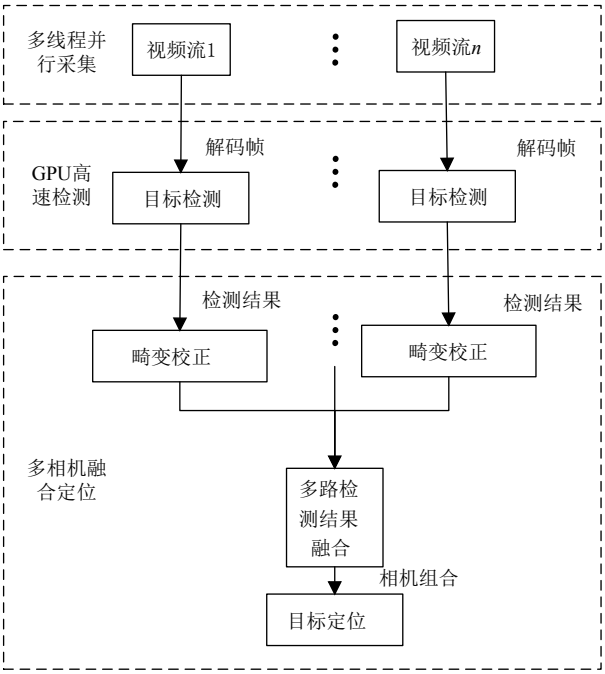

图 1 人体运动目标实时检测定位流程图

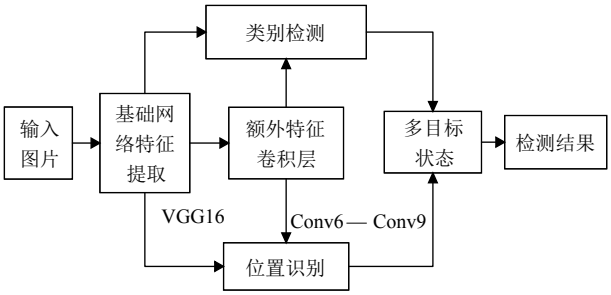

图 2 SSD 检测原理图

2.2.1 SSD 网络模型

SSD 是基于前馈卷积神经网络, 产生固定大小的

边界框集合和边界框中对象类别的分数, 然后通过非最大抑制来产生最终检测结果. 本文采用 VGG16[18]网络作为基础网络, 用于特征提取, 然后再向基础网络中添加辅助结构, 产生特定的特征用于检测[15]. 如图 3 所示, 这些特征层的尺度逐渐变小, 可以得到多个尺度检测的预测值.

2.2.2 训练

SSD 的训练与传统方法的区别在于真实标签需要与固定的检测器输出集合中的某一个特定的输出相对应 (端到端训练). 训练时, 需要建立默认框与真实框之间的一一对应关系[15]. 可以从不同位置、尺度、宽高比的默认框中选择真实的目标标签框.

2.3 双目定位

双目定位首先需要考虑摄像机的标定与图像的畸变矫正问题. 本文采用张正友的基于平面模版的标定方法[19], 然后进行去畸变处理, 得到合理的相机标定结果.

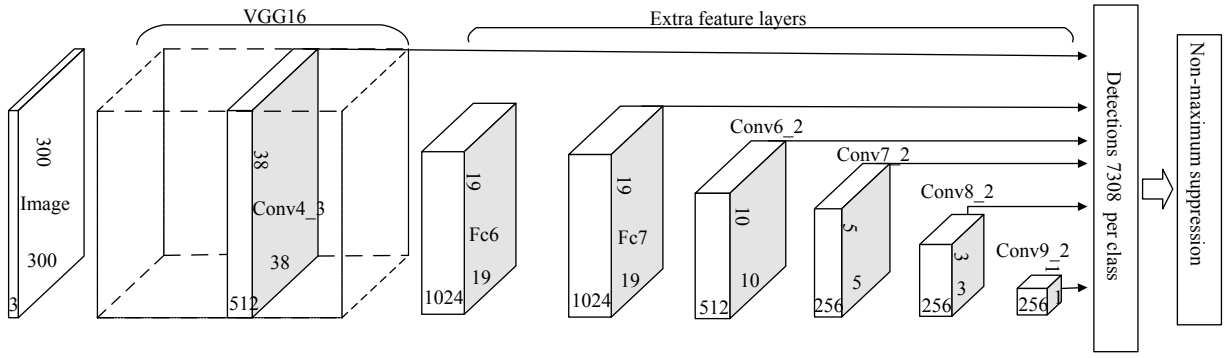

图 3 SSD 网络结构图

图 4 中o_c为摄像机的光心, z_c为光轴, (u,v)为像素坐标, $o(u_0,v_0)$为图像坐标系原点, o_co为摄像机的焦距 f, 图像坐标与像素值的对应关系为:

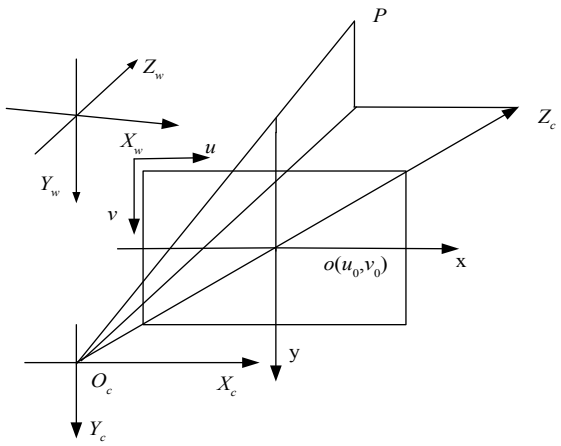

图 4 相机成像模型

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1/d_x & \gamma & u_0 \\ 0 & 1/d_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{1}$$

其中, d_x,d_y为单位像素的物理长度. 相机坐标与图像坐标的对应关系为:

$$Z_c \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} \tag{2}$$

世界坐标系与相机坐标系的对应关系为:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{3}$$

综合公式 (1)(2)(3) 可得图像坐标系 (像素表示法) 与世界坐标系的关系公式如下:

$$Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & s & u_0 & 0 \\ 0 & f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \mathbf{A}\,[\mathbf{R}|\mathbf{T}] \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{4}$$

在公式 (4) 中 (X_w, Y_w, Z_w) 为自定义的世界坐标系, (u, v) 为图像坐标系. $\mathbf{A}$ 为相机的内参矩阵, $\mathbf{R}$, $\mathbf{T}$ 为旋转和平移矩阵, 即外参矩阵.

本文自定义世界坐标, 用 Matlab 工具箱中棋盘格标定法来求得两个相机的内参数矩阵 $\mathbf{A}_1$,$\mathbf{A}_2$、以及两

个相机的畸变系数k_1,k_2, 由相机的内参矩阵和畸变系数可以得到无畸变的图片, 根据求得的无畸变图以及对应自定义的世界坐标中的 10 组对应点, 根据公式 (5) 可求出世界坐标与相机坐标的单映矩阵 **H**.

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \mathbf{H}\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix}\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} \tag{5}$$

再由地平面与图像平面之间的单应关系可以求得相机坐标系和世界坐标系之间的旋转矩阵 **R** 和平移矩阵 **T**. 设$\mathbf{H}=[h_1,h_2,h_3]$, 在已知内参数矩阵的情况下容易通过下面的公式 (6) 解得两个相机的外参数RT_1 和RT_2.

$$\begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix} = \lambda A\begin{bmatrix} r_1 & r_2 & t \end{bmatrix} \tag{6}$$

为获取目标在世界坐标中的位置信息, 需由两个摄像机对目标进行定位. 摄像机参数见表 1.

表 1 相机参数表

参数	摄像机 1	摄像机 2
内参矩阵	$\begin{bmatrix} 2562.366\,84 & 0 & 945.823\,88 \\ 0 & 2565.203\,35 & 559.174\,70 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 2562.366\,84 & 0 & 945.823\,88 \\ 0 & 2565.203\,35 & 559.174\,70 \\ 0 & 0 & 1 \end{bmatrix}$ $\begin{bmatrix} 2365.721\,47 & 0 & 884.628\,97 \\ 0 & 2367.191\,32 & 475.418\,74 \\ 0 & 0 & 1 \end{bmatrix}$
外参矩阵[**R**|**T**]	$\begin{bmatrix} -0.842\,076 & -0.542\,677 & -0.019\,283\,6 & 476.845 \\ 0.273\,430 & -0.446\,289 & 0.831\,895 & 67.311 \\ -0.464\,910 & 0.688\,297 & 0.524\,194 & 1190.81 \end{bmatrix}$	$\begin{bmatrix} 0.728\,716 & -0.653\,953 & -0.046\,887 & -138.500 \\ 0.370\,460 & 0.354\,617 & 0.870\,645 & -203.597 \\ -0.575\,961 & -0.677\,895 & 0.500\,679 & 1691.623 \end{bmatrix}$

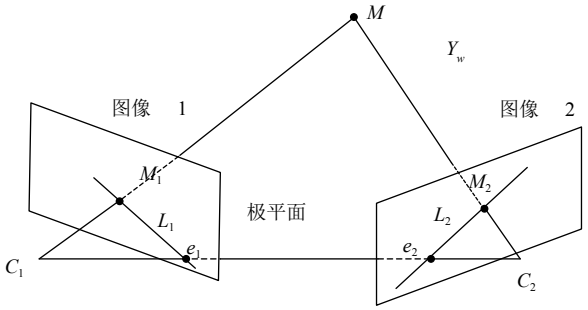

图 5 双目相机极线几何约束模型图

根据如图 5 所示的约束模型, 设两个摄像机的投影矩阵为 $\mathbf{P}_1$, $\mathbf{P}_2$:

$$\begin{cases} \mathbf{P}_1 = \mathbf{A}_1[\mathbf{R}_1|\mathbf{T}_1] \\ \mathbf{P}_2 = \mathbf{A}_2[\mathbf{R}_2|\mathbf{T}_2] \end{cases} \tag{7}$$

点m为自定义坐标下的点, 点m_1和点m_2为点m在两个相机中的投影点, 两个投影线和基线C_1C_2形成了一个三角形, 则$\begin{cases} s_1m_1 = p_1m \\ s_2m_2 = p_2m \end{cases}$, 因此利用三角形定位原理计算空间坐标点$m$.

3 实验

本文采集相机为 Dahua DH-IPC-HF8431E, 图像分辨率为 1920×1080, 训练和检测的电脑硬件配置为: CPU: Intel Core i7-6850K CPU @ 3.60 GHz; GPU: NVIDIA GeForce GTX 1080 Ti, 11 G×2.

3.1 数据集制作

本文是针对特定场景设计的一套运动目标检测与定位系统. 以一个 10 m×10 m 的室内场景中运动的人体为检测目标, 每一个运动目标佩戴一顶不同的帽子作为视觉标定物, 通过 4 路相机采集不同视角的图片数据作为样本集, 然后通过人工对采集的数据进行标注. 本文对于 10 个目标的场景的数据集包括: Images: 752 张/JPG, Labels: 752 个/XML, BoundingBoxes: 7520 个/Rectangles. 训练集有 632 张图片, 测试集有 120 张图片, 比例大概为 5:1. 验证集分为两个场景, 每个场景中都有 10 个运动目标, 共 956 张图片.

3.2 模型训练

训练时采用 VGG16 作为本次实验的基础网络, 并将最后的 fc8 层和所有 dropout[20]层去掉, 将 fc6 和 fc7 转化为卷积层[13], 将 pool5 的 2x2-s2 改为 3x3-s1, 并使用 atrous 算法填洞. 使用 GSD 对网络进行微调, 设置初始学习率为 0.000 25, 采用 multistep 的方法对学习率进行改变, gamma 设置为 0.1, 即当迭代到 30 000, 40 000, 50 000 次时, 学习率改为原来的 0.1 倍. Momentum 为 0.9, weight_decay 为 0.0005, batch_size 为 32, 总共训练 60 000 次, 总共训练时间约为 60 个小时, 最后两次训练得出的精度分别达到 96.2% 和 92.4%. 利用训练好的模型 (.caffemodel 文件) 就可以对图片视频进行

检测. 使用网络中的 conv4_3, fc7, conv6_2, conv7_2, conv8_2 和 conv9_2 总共 6 个不同尺度的特征图上进行预测置信度和位置.

通过 2.3 节方法求得相机的内参 **A** 和参矩阵 **H**, 并利用两路网络摄像机的目标检测结果进行定位.

4 结果与分析

从图 6 中的网络训练精度与损失函数中可以看到当训练迭代到 3 万次的时候, 损失函数基本保持不变, 检测精度也在 3 万次以后趋于稳定, 在 95% 左右. 利用训练好的模型对视频的每一帧进行检测. 实验统计检测识别结果如表 2.

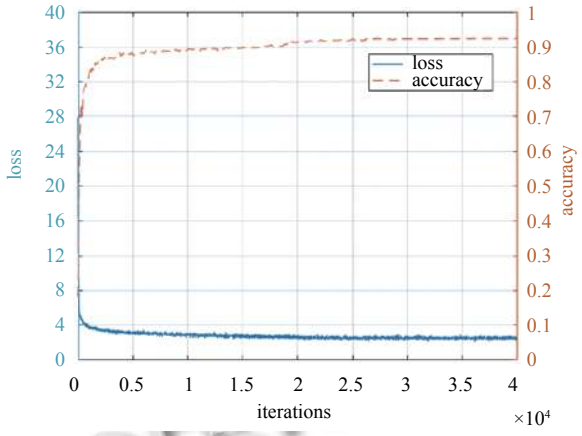

图 6 网络训练精度与损失函数图

表 2 目标识别结果 (σ=0.2)

	目标序号	目标 1	目标 2	目标 2	目标 4	目标 5	目标 6	目标 7	目标 8	目标 9	目标 10
场景 1	总帧数	752	752	752	752	752	752	752	752	752	752
	漏检数	15	11	5	14	11	11	9	16	12	8
	漏检率 (%)	2.0	1.5	0.7	1.9	1.5	1.5	1.2	2.1	1.6	1.1
场景 2	总帧数	204	204	204	204	204	204	204	204	204	204
	漏检数	9	11	12	10	13	8	7	9	17	7
	漏检率 (%)	4.6	5.4	5.9	4.9	6.4	3.9	3.4	4.4	8.3	3.4

表中的σ为置信度, 表示每个目标在检测网络中默认框与真实目标框的 IOU 大于 0.45 时默认框数量的多少. 当σ=0.6 时, 每个目标的漏检率相对较高, 当σ=0.2 时, 误检率都有明显的下降. 如表 3, 为两个场景视频中随机抽取一帧的检测情况, 场景 1 中, σ=0.6 时, 检测到 3 个目标, σ=0.2 时, 检测到 9 个目标; 在场景 2 中σ=0.6 时, 检测到 5 个目标, 当σ=0.2 时, 检测到所有目标.

表 3 不同σ检测结果图

置信度	场景 1	场景 2
σ=0.6		
σ=0.4		
σ=0.2		

通过上面的两次测试结果可知当σ=0.2 时, 检测效果最好, 因此将σ设 0.2, 利用训练好的模型对视频中的目标场景进行检测和定位, 随机选取 12 帧定位结果进行分析, 定位结果如表 4.

表 4 目标定位误差 (单位: cm)

视频帧序号		目标 1	目标 2	目标 3	目标 4	目标 5	目标 6	目标 7	目标 8	目标 9	目标 10	误差均值
1	T	542, 420	366, 435	180, 427	213, 305	366, 306	488, 306	610, 244	488, 122	366, 122	175, 125	6.8
	D	546, 416	369, 441	175, 437	209, 310	359, 309	484, 307	605, 251	488, 119	362, 126	171, 130	
2	T	671, 435	493, 430	305, 431	303, 308	490, 305	590, 302	691, 260	610, 122	488, 130	290, 122	5.9
	D	672, 439	493, 432	304, 430	209, 311	482, 306	584, 306	680, 269	601, 116	482, 138	282, 124	
3	T	555, 308	366, 305	223, 318	223, 191	366, 183	488, 183	612, 125	488, 10	368, 8	175, –3	3.8
	D	553, 304	368, 303	221, 321	223, 193	363, 183	481, 184	609, 125	487, 8	372, 5	178, –1	
4	T	552, 427	366, 427	238, 428	229, 300	360, 305	483, 305	630, 240	495, 126	366, 122	183, 112	4.7
	D	548, 423	365, 427	235, 430	224, 301	257, 305	479, 306	619, 234	492, 127	360, 122	180, 114	
5	T	458, 427	386, 427	76, 425	112, 305	272, 305	405, 305	545, 244	420, 122	385, 122	81, 130	5.5
	D	451, 422	391, 428	77, 429	111, 302	265, 304	407, 309	540, 245	413, 122	281, 125	84, 135	
6	T	559, 425	366, 432	185, 437	213, 305	366, 310	490, 305	610, 244	493, 122	366, 122	178, 124	3.1
	D	556, 421	366, 433	188, 439	211, 304	362, 311	487, 305	607, 245	492, 121	365, 121	175, 127	
7	T	549, 305	366, 305	218, 322	224, 193	366, 183	485, 183	605, 110	488, 10	366, 0	173, 0	4.0
	D	550, 303	363, 303	221, 319	222, 193	361, 185	476, 185	601, 114	488, 8	366, 2	176, –3	
8	T	660, 430	485, 427	306, 430	305, 310	488, 308	603, 308	702, 246	610, 127	488, 122	285, 127	4.5
	D	667, 433	483, 428	307, 433	299, 312	484, 311	597, 308	705, 248	614, 125	486, 132	282, 134	
9	T	449, 431	361, 462	187, 425	180, 308	362, 305	483, 305	620, 252	488, 125	356, 122	178, 127	1.5
	D	448, 433	360, 435	187, 424	181, 309	362, 305	480, 305	622, 252	489, 125	355, 122	178, 126	
10	T	544, 366	372, 336	228, 345	318, 244	372, 221	488, 244	610, 198	496, 46	376, 44	173, 42	3.8
	D	542, 366	377, 336	323, 346	319, 242	376, 220	488, 238	609, 201	498, 49	379, 41	176, 40	
11	T	549, 308	371, 300	223, 320	224, 198	361, 183	479, 188	600, 112	488, 5	366, 0	178, –10	2.6
	D	547, 309	372, 297	221, 320,	222, 192	362, 183	476, 192	598, 112	490, 4	366, 1	178, –8	
12	T	549, 442	386, 432	193, 437	203, 305	386, 305	495, 310	618, 249	493, 112	370, 122	188, 120	3.4
	D	541, 444	383, 430	196, 440	203, 303	383, 307	493, 310	616, 247	494, 108	371, 124	189, 118	
误差均值		4.5	3.9	3.7	3.9	3.8	4.9	5.3	3.7	4.6	4.1	
均方根误差		5.0	3.8	5.1	4.5	4.5	5.2	4.5	4.5	5.4	4.6	

表 4 中 T 表示真实目标的坐标, D 标表示检测定位的坐标. 从检测定位结果表中可以观察得出给个每个定位目标的平均定位误差在 6 cm 以内, 每帧检测的所有目标的平均定位误差也在 6 cm 以内, 12 帧图片的所有检测目标的平均定位误差为 4.17 cm, 均方跟误差在 4.5 左右, 检测速度可以达到 40 fps 以上, 完全可以实现实时检测的效果. 需要说明一点, 本文处理的图片大小为 1920×1080 像素, 处理图片的分辨率一般较大, 因此分辨率较大的图片进行处理实现 40 fps 左右是一个相当有参考价值的方法.

(a) 目标检测结果图 (b) 目标定位结果图

图 7 目标检测定位图

5 总结

本文利用深度学习框架, 采用 SSD 目标检测方法和双目视觉定位, 实现人体运动目标实时检测和定位. 文中采用的 SSD 模型检测方法与其他的利用神经网络的检测方法相比, 省去了区域预测和特征重采样的过程, 因此检测速度大大增加. 本文提供的定位方法平均误差在 6 cm 以内, 具有较高的精度, 且检测速度可以达到 40 fps 以上.

本文的运动目标检测与定位方法对于固定场景检测定位效果较为理想, 而针对小目标的场景, 以及运动目标遮挡比较严重的情况下, 会出现某个相机漏检和误检的情况, 可以通过以下两种方法进一步研究:

1) 通过增加多路摄像机, 两两配对, 增加融合机制, 提高检测结果;

2) 目前使用的是 VGG16 作为基础网络, 随着计算机性能的提升, 可以使用更深层次的网络, 例如 ResNet、GoogleNet 等作为基础网络或者从新设计网

络结构来提高检测效果.

参考文献

1 张慧, 王坤峰, 王飞跃. 深度学习在目标视觉检测中的应用进展与展望. 自动化学报, 2017, 43(8): 1289–1305.

2 Felzenszwalb PF, Girshick RB, McAllester D, *et al*. Object detection with discriminatively trained part-based models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2010, 32(9): 1627–1645. [doi: 10.1109/TPAMI.2009.167]

3 Taigman Y, Yang M, Ranzato M, *et al*. Deepface: Closing the gap to human-level performance in face verification. Proceedings of 2014 IEEE Conference on Computer Vision and Pattern Recognition. Columbus, OH, USA. 2014. 1701–1708.

4 Ojala T, Pietikainen M, Harwood D. Performance evaluation of texture measures with classification based on Kullback discrimination of distributions. Proceedings of the 12th IAPR International Conference on Pattern Recognition. Jerusalem, Israel. 1994. 582–585.

5 Ojala T, Pietikäinen M, Harwood D. A comparative study of texture measures with classification based on featured distributions. Pattern Recognition, 1996, 29(1): 51–59. [doi: 10.1016/0031-3203(95)00067-4]

6 Ma XX, Grimson WEL. Edge-based rich representation for vehicle classification. Proceedings of the 10th IEEE International Conference on Computer Vision. Beijing, China. 2005. 1185–1192.

7 Kazemi FM, Samadi S, Poorreza HR, *et al*. Vehicle recognition using curvelet transform and SVM. Proceedings of the 4th International Conference on Information Technology. Las Vegas, NV, USA. 2007. 516–521.

8 Freund Y, Schapire RE. A desicion-theoretic generalization of on-line learning and an application to boosting. Proceedings of the 2nd European Conference on Computational Learning Theory. Barcelona, Spain. 1995. 23–37.

9 Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Proceedings of the 25th International Conference on Neural Information Processing Systems. Lake Tahoe, NV, USA. 2012. 1097–1105.

10 Girshick R, Donahue J, Darrell T, *et al*. Rich feature hierarchies for accurate object detection and semantic segmentation. Proceedings of 2014 IEEE Conference on Computer Vision and Pattern Recognition. Columbus, OH, USA. 2014. 580–587.

11 He KM, Zhang XY, Ren SQ, *et al*. Spatial pyramid pooling in deep convolutional networks for visual recognition. Proceedings of the 13th European Conference on Computer Vision. Zurich, Switzerland. 2014. 346–361.

12 Uijlings JRR, Van De Sande KEA, Gevers T, *et al*. Selective search for object recognition. International Journal of ComputerVision,2013,104(2):154–171.[doi:10.1007/s11263-013-0620-5]

13 Girshick R. Fast R-CNN. Proceedings of 2015 IEEE International Conference on Computer Vision. Santiago, Chile. 2015. 1440–1448.

14 Ren SQ, He KM, Girshick R, *et al*. Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39(6): 1137–1149. [doi: 10.1109/TPAMI.2016.2577031]

15 Redmon J, Divvala S, Girshick R, *et al*. You only look once: Unified, real-time object detection. Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, NV, USA. 2016. 779–788.

16 Liu W, Anguelov D, Erhan D, *et al*. SSD: Single shot multibox detector. Proceedings of the 14th European Conference on Computer Vision. Amsterdam, The Netherlands. 2016. 21–37.

17 赵永科. 深度学习: 21 天实战 caffe. 北京: 电子工业出版社, 2016. 10–25.

18 Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv: 1409.1556, 2014.

19 Zhang Z. A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000, 22(11): 1330–1334. [doi: 10.1109/34.888718]

20 Srivastava N, Hinton G, Krizhevsky A, *et al*. Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 2014, 15(1): 1929–1958.