DOI:10.19651/j.cnki.emt.2210460

改进 Mean Shift 目标跟踪算法实现*

刘银萍[1] 夏金锋[2] 姜 栋[2] 徐 龙[2] 严 飞[2,3]
(1.南京信息工程大学大气物理学院 南京 210044;2.南京信息工程大学自动化学院 南京 210044;
3.江苏省大气环境与装备技术协同创新中心 南京 210044)

摘 要: 目标跟踪技术目前具有较好的应用前景,但在嵌入式处理平台中面临着实时性要求高、跟踪场景复杂等情况,加上受成本和嵌入式处理平台算力的限制,其处理效果往往很难满足现实需求,因此目标跟踪等图像处理技术的落地实现是当前研究的热点内容。针对此问题,本文在 FPGA 平台实现了改进 Mean Shift 目标跟踪算法,该算法首先通过目标的概率密度分布梯度爬升来寻找目标,然后采用 Kalman 滤波的预测机制来预估下一帧搜寻计算的位置,从而减少 Mean Shift 的迭代次数。该算法实现充分利用 FPGA 能够并行和流水线处理的特点,实现了在 1 920×1 080@60 Hz 高清视频图像场景下的实时目标跟踪,其中 Kalman 滤波算法使其在较复杂场景下也能具备一定的抗遮挡干扰的能力。

关键词: FPGA;目标跟踪;Mean Shift 算法;Kalman 滤波
中图分类号: TN215 **文献标识码:** A **国家标准学科分类代码:** 510.10

Improved Mean Shift target tracking algorithmz

Liu Yinping[1] Xia Jinfeng[2] Jiang Dong[2] Xu Long[2] Yan Fei[2,3]
(1. College of Atmospheric Physics, Nanjing University of Information Science & Technology, Nanjing 210044, China;
2. School of Automation, Nanjing University of Information Science & Technology, Nanjing 210044, China;
3. CICAEET, Nanjing University of Information Science & Technology, Nanjing 210044, China)

Abstract: Target tracking technology has good application prospect at present, but in an embedded processing platform is facing complicated high real-time requirements, tracking, etc., and restricted by cost and embedded processing platform to calculate force, it is often difficult to meet the demand of reality of its processing effect, so the image processing technology such as target tracking ground implementation is a hotspot of current research content. To solve this problem, this paper implemented an improved Mean Shift target tracking algorithm on FPGA platform. The algorithm first searched for the target by the gradient climbing of the probability density distribution of the target, and then used the prediction mechanism of Kalman filter to predict the location of the next frame search calculation, so as to reduce the number of iterations of Mean Shift. The algorithm makes full use of the parallel and pipelined-processing characteristics of FPGA to realize the real-time target tracking in 1 920×1 080@60 Hz hd video image scene, and the Kalman filtering algorithm enables it to have a certain ability to resist occlusion interference in more complex scenes.

Keywords: FPGA;target tracking;Mean Shift algorithm;Kalman filter

0 引 言

目标跟踪技术是计算机视觉领域的重要分支,是指在输入的视频图像序列中自行跟踪特定目标的技术,在安防监控、赛事直播等领域具有较好的前景[1]。目前目标跟踪算法的研究处于热门发展阶段,计算机平台上的目标检测与跟踪算法无论在处理速度、抗干扰、准确度等方面都已经达到了很好的效果[2-3]。但是,在嵌入式平台等实际应用场景下的检测跟踪系统并不多[4],嵌入式平台下的算法实现需要综合考虑算法的复杂度与硬件的计算速度[5]。同时,近几年来人们对高清视频的需求越来越大,高清图像及视频的实时处理问题也随之而来,对应用场景下硬件实时处理速度、视频数据传输带宽等提出了较高的要求[6]。因此,目标跟踪等图像处理技术的落地实现是当前研究的热点内

收稿日期:2022-06-25
* 基金项目:江苏省产业前瞻与关键核心技术重点项目(BE2020006-2)、国家自然科学基金(61605083)项目资助

容。FPGA(field programmable gate array)具备的硬件逻辑资源可灵活编程与并行处理等特点，使其在图像处理领域拥有很好的应用前景，通过合理的算法优化，可使 FPGA 成为理想的视频图像处理算法的加速平台[7]。

由于以深度学习相关的目标跟踪算法复杂度较高，对硬件算力要求苛刻，所以在嵌入式平台上的目标跟踪实现仍以传统方法为主流[8]。本文针对上述目标跟踪算法落地实现过程中出现的高清、实时性和抗干扰需求问题，在 FPGA 平台上实现了改进的 Mean Shift 目标跟踪算法。但是通常 Mean shift 算法实现跟踪需要在单帧图像上进行多次的 Mean Shift 迭代计算，以此将跟踪窗口指向目标特征分布最密集处达到跟踪效果[9-10]。常用方法下的计算不可避免地需要缓存和反复读取大量图像数据，在有限的平台算力下需要占用大量处理时间[11]。因此，为了减少计算复杂度和 FPGA 实现，本文提出了结合 Mean Shift 和 Kalman 滤波的目标跟踪算法，即在跟踪算法中引入了卡尔曼滤波预测机制，每帧画面的 Mean Shift 计算使用上一帧跟踪结果的 Kalman 滤波预测值作为搜索窗口。由此就可以减少同一帧画面中 Mean Shift 的迭代搜索过程，一帧画面进行一次 Mean Shift 矢量计算就能保持对连续画面中目标的持续跟踪。同时，Kalman 滤波还能进一步地提升跟踪算法的抗干扰能力，可以应对目标完全被遮挡情况下的跟丢情况。

1　总体方案设计

本文的设计组成包含数据传输和跟踪图像处理算法两部分。图像处理模块的设计是本文主要工作内容，主要涉及 Mean Shift 目标跟踪和 Kalman 滤波两大模块。数据传输模块主要是完成 HDMI 输入输出接口和串口的功能实现。总体设计框图如图 1 所示。

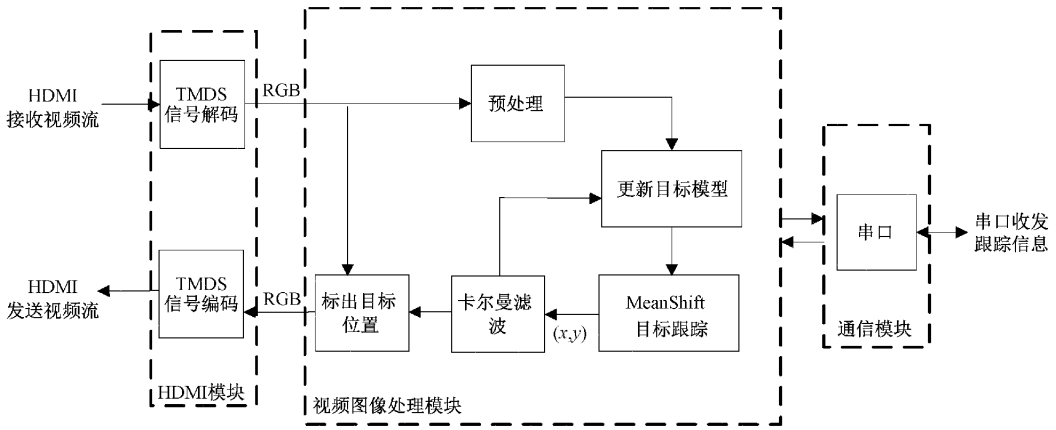


图 1　总体设计框图

逻辑功能设计的具体工作流程如图 2 所示。以 HDMI 作为视频输入方式，上位机通过 HDMI 接口给 FPGA 传输实时视频数据流。HDMI 模块会将传输过来的 TMDS 信号解码为包含 RGB 色彩像素数据的 VGA 时序。RGB 数据流输入到视频图像处理模块内，首先都会作前期的预处理操作。其后的串口模块将取出需要跟踪目标的初始质心坐标和大小，并传入下方目标跟踪部分。目标跟踪部分包含目标模型的获取、Mean Shift 目标跟踪和卡尔曼滤波算法。当前帧跟踪结果输出后，处理模块会在原 RGB 彩色图像的基础上框出目标位置，最后通过 HDMI 编码模块编码输出到显示屏。输出结果的同时，串口模块输出目标坐标到上位机。

其目标跟踪算法流程如下所述：从串口接收到目标位置、大小和图像等信息后，首先需要对第一帧画面相应位置图像做跟踪模型的初始化操作，此处目标特征使用初始区域 RGB 转 HSV 后的 H 色调的直方图建立。后续每帧画面的跟踪处理都建立在该目标直方图的反向投影图像之上，当目标未被遮挡就使用 Mean Shift 计算跟踪目标，当被完全遮挡则使用上次 Kalman 滤波的预测结果，防止

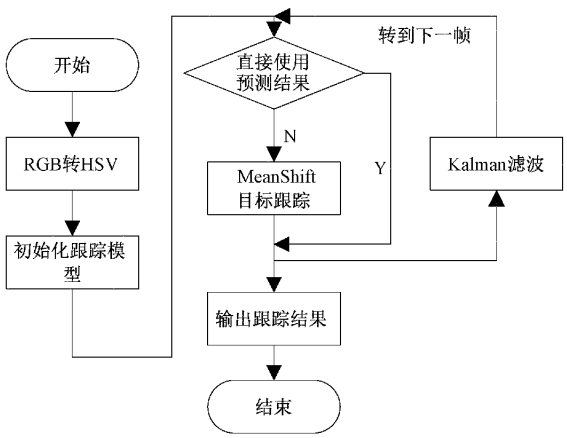


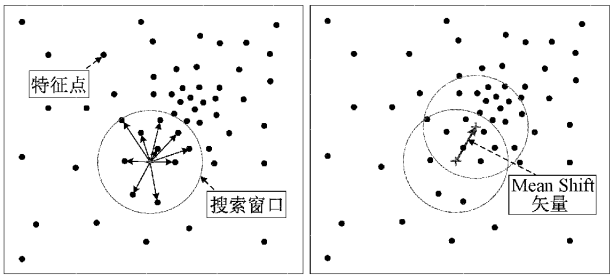
图 2　算法流程

窗口飘出丢失。每帧图像跟踪完成后，Kalman 滤波模块都将根据之前的目标位置预测出下一次目标出现的位置用于下次搜索处理。

2　基于 Mean Shift 的目标跟踪算法

该方法的主要目的是通过不断迭代计算，搜索窗口中

特征点密度最大的位置。每次先如图 3(a)所示计算偏移均值,得出特征点密集方向,后如图 3(b)更新窗口再次计算,将搜索窗口中心定位到目标中心处,实现对目标的跟踪[12-13]。



(a) 计算偏移均值　　(b) 更新搜索窗口

图 3　Mean Shift 原理示意图

2.1　颜色空间转换模块

视频及图像中常使用 RGB 颜色空间即三原色的组合来表示颜色。但这 3 个分量是亮度相关,对光照比较敏感,所以在跟踪处理中使用 RGB 色彩空间,会出现环境适应性比较差的情况。而 HSV 颜色空间将图像的色调(H)、饱和度(S)和明度(V)区分开来表示,其中色调 H 分量为色彩信息,不受光照明亮的影响,可以很好的弥补 RGB 格式的不足。本跟踪模块使用图像 HSV 中 H 分量作为处理量,所以需要将 RGB888 色彩转换为 HSV 色彩空间,其 H 计算如式(1)所示。

$$H=\begin{cases}0^\circ & \max=\min\\ 60^\circ\times\dfrac{G-B}{\max-\min}+0^\circ & \max=R,B\leqslant G\\ 60^\circ\times\dfrac{G-B}{\max-\min}+360^\circ & \max=R,G<B\\ 60^\circ\times\dfrac{B-G}{\max-\min}+120^\circ & \max=G\\ 60^\circ\times\dfrac{R-G}{\max-\min}+240^\circ & \max=B\end{cases}\tag{1}$$

其中,$\begin{cases}\max=\max(R,G,B)\\ \min=\min(R,G,B)\end{cases}$。

本功能模块的 FPGA 实现为节约资源提高算法效率,结构设计与上述图像灰度转换类似,在 VGA 时序的基础上,针对像素数据流依次对每个像素都进行一次计算操作。

具体的 FPGA 实现如图 4 所示。采用了 4 级流水线处理,其中主要包含了最大最小值判断的 MAX/MIN 模块、数据多路选择器、操作控制 CTR 模块以及多个加减乘除计算模块。其中操作控制模块根据输入的不同 RGB 值选择对应的计算公式分支,又因为式(1)中存在减法后可能出现负数,且后续除法器 Div 模块不支持负数计算的情况,所以这里还将控制 ctr0、ctr1 使减法部分转为大数减小数,最后再在 Add or Sub 模块中通过 ctr3 控制代入正负号,即选择使用减法还是加法。

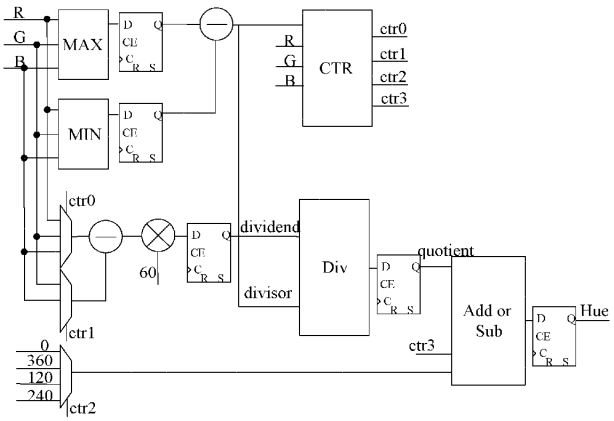


图 4　RGB 转 HSV 的 H 分量的计算模块

图 4 中的除法计算涉及转换进度相关,因此比较重要。除法器实现有多种方案,可以使用 FPGA 自带的 DSP 实现,即直接使用"/",也可以使用专用计算的 IP 核实现。本模块中除法器实现使用基于减法的除法器的算法 Verilog 实现,该算法采用左移相减的方法,计算前首先将被除数转换为高 32 位为 0,低 32 位为被除数形式的 64 位 *temp_a*,除数相反置于高 32 位,低 32 位置 0 转为 *temp_b*;计算过程为将 *temp_a* 左移 1 位(相当于商+1),如大于 *temp_b* 则 *temp_a* 减去 *temp_b* 将且加上 1;该步骤执行 32 次结束后 temp_a 的高 32 位为余数,低 32 位为商。为了计算效率,本计算模块使用组合逻辑实现。

最后,式(1)中的 H 分量的结果取值范围为 0°～360°,本模块实际需要将结果归一化到 0～255 范围内,即输出的 Hue 位宽与灰度值的一样都为 8,方便后续的统一计算。其中比较重要的就是浮点数的乘除法实现,这里依然采用转灰度模块中的浮点数转定点数方法,将数值放大计算后再缩小相同倍数。

从图 5 仿真结果可以看到,计算过程经过 4 个时钟周期即可计算出 0～255 范围的 H 分量值。

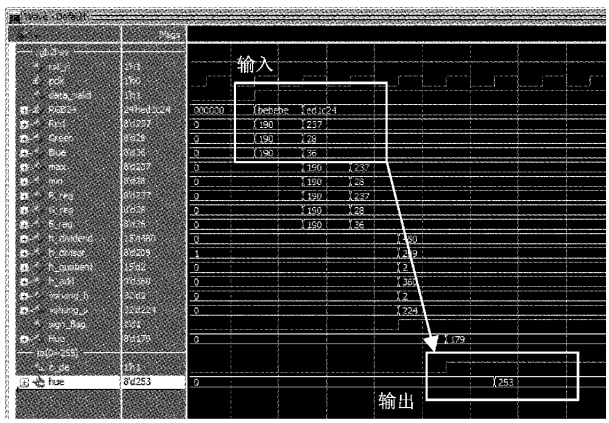


图 5　RGB 转 HSV 的 Modelsim 仿真

2.2　直方图及反向投影模块

本模块的设计主要有两个功能实现,直方图统计和直

方图反向投影。

直方图的目的是建立初始的跟踪目标色度分布模型，该目标模型是后续跟踪计算的主要参数。如图 6 所示，直方图部分的实现内容是统计初始目标（即运动检测目标）的色度分布情况，此处统计的图像范围为运动检测目标位置，统计的数据为上一节 RGB 转换 HSV 后的 H 分量数据。H 分量数值的大小在 0～255 间，则本文直方图的统计范围为 0～255，统计的就是各个色度量级的像素数量。通常，直方图统计只在跟踪过程的第一帧画面中计算，即只在第一帧模型建立目标模型。

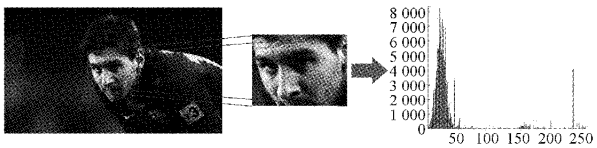


图 6　目标框选与 H 通道直方图示意图

H 色调直方图统计的 FPGA 实现从数据存储方式上看，可以有两种方案，一种用 RAM 存储、一种用寄存器存储。如使用 RAM 存储实现统计，可以例化一个存储深度为 256 的 RAM 空间，每一个地址空间都代表一个 H 色调量级，该地址存储统计的该色调的像素数量。第二种实现方案使用寄存器存储统计数据，该方案首先需要定义二维数组，定义方式如式(2)所示。FPGA 统计实现只需要在像素数据流输入时，每个像素点都按照式 3 将输入数据 *hue* 给目标寄存器加 1 操作就可完成。

$$reg\ [19:0]histogram\_cnt[255:0]; \tag{2}$$

$$histogram[hue] = histogram[hue] + 1 \tag{3}$$

其中，用 RAM 存储时要考虑到每一个像素输入后，都要进行一次 RAM 读、数据加一和写操作。并且虽然双端口 RAM 可以同时读写数据，但如果读取存储同一地址空间，数据将会出现冲突错误。因此使用该方案需要倍频像素数据时钟至少两倍且设置“写优先”和“output register”，才能无误地控制 RAM 数据的读写统计数量。而寄存器存储方法实现读写操作简单无控制时序延时，且易于实现。

综上，考虑到需要能对任意分辨率图像处理，以及 RAM 存储方式逻辑实现的繁琐，这里采用寄存器存储的方式。

统计出跟踪目标的色调分布情况后，256 个寄存器中存储了各个色调的像素数量，但其中的数据大小不确定，为了数据在同一范围内便于后续处理计算，此处需要归一化统计的像素数量到 0～255 范围内。因此直方图统计完成后还需要判断本次统计中的最大值。

该部分的实现仿真如图 7 所示，输入量包含目标的坐标信息以及按照 VGA 时序的 H 值数据流；该模块将在目标区域内统计更新直方图；右上角的表格为某一时刻的直方图统计结果；直方图完成后还将取出本次处理中的最大值，本次测试中最大值为 1 598。

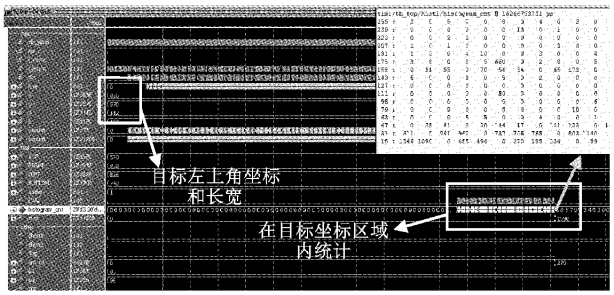


图 7　直方图实现的 Modelsim 仿真

直方图的反向投影是一种利用直方图模型计算给定图像像素点的像素分布方式。简单来讲就是用某一直方图模型寻找目标图像中与之相同的特征，反向投影后的图像每个像素点值为该图原像素值对应的直方图数据大小。简言之结果图像就是目标特征的概率密度分布图，通过直方图的反向投影，我们可以找出目标图像中与模型最相似的像素点。

本节中的反向投影实现是在直方图统计完成后的第二帧开始的，其计算如式(4)所示，每输入一个像素执行一次计算。

$$CalcBackData = histogram[hue]\frac{255}{\max} \tag{4}$$

其中，*CalcBackData* 为反向投影后的结果；*hue* 为当前输入值；*histogram*[*hue*] 为直方图中的对应数值；255/max 为归一化系数，将结果统一到 0～255 之间；max 为上面直方图统计中得出的最大值。

图 8 和 9 为直方图反向投影实现的仿真效果，可以看到反向投影数据在原像素色度数据输入后一个时钟周期就完成输出。

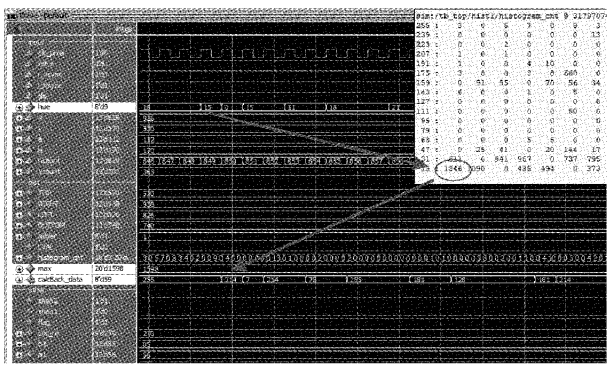


图 8　直方图反向投影实现的 Modelsim 仿真

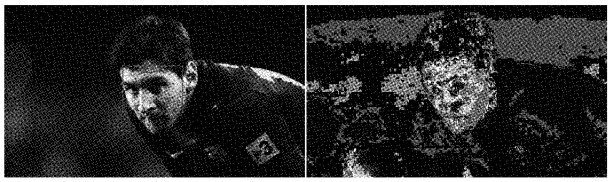


图 9　概率密度分布图 Modelsim 仿真输入与输出输出

本小节实现的直方图反向投影算法,与常见的直方图匹配、模板匹配等算法相比速度更快。本模块采用直方图获取目标特征并以此反向投影后续图像画面而显示出相似处,该方法与其他类型算法相比计算机实现效果虽然相似,但本算法无需反复读取图像数据计算即可得出目标模型在本帧的概率分布情况,而且结合 FPGA 流水线处理的优势,更能实现实时的结果输出(本模块输出整体延时仅 1 个时钟周期)。

2.3 Mean Shift 矢量计算

为在 FPGA 上实现 Mean Shift 矢量计算,本设计对跟踪算法做了计算过程的分解与针对硬件逻辑实现的优化。Mean Shift 矢量计算的最基本形式如式(5)。

$$A(f)=\frac{1}{n}\sum_{x_i\in S_r}(f_i-f) \tag{5}$$

其中,$A(f)$ 为偏移均值,即 Mean Shift 矢量;S_r 是以 f 坐标为中心,r 为半径的区域;n 是区域内特征点的总数;f_i 为表示区域内的特征点坐标;$i\in(1,n)$。

简单点说来,一次 Mean Shift 矢量计算就是将图 3 中所有从中心点指向目标特征点的向量求和,并取均值。本模块的 FPGA 实现建立在上一节直方图反向投影后的概率密度分布图基础之上,反向投影后的图像中每个像素都会有对应的取值,且数值越高的就越接近跟踪目标的特征,因此,式(5)中还需要考虑像素值对每个向量的权重影响。

在采用 Mean Shift 算法进行目标跟踪实现时,图 3 所示的特征点实际对应搜索区域图像上的每一个像素点,而搜索窗口一般也会改为矩形的形式,且为在 FPGA 实现,将公式展开分别计算 x 与 y 两个方向。

修改后的计算如式(6)、(7)所示,其中 $f_i[data]$ 为 f_i 点的对应像素值;x_i 和 y_i 为点 f_i 的横纵坐标值;x、y 为中心点横纵坐标值;1/255 为像素值的归一化系数。

$$A(x)=\frac{1}{n\times 255}\sum_{x_i\in S_r}f_i[data](x_i-x) \tag{6}$$

$$A(y)=\frac{1}{n\times 255}\sum_{x_i\in S_r}f_i[data](y_i-y) \tag{7}$$

图 10 为该方法的 Modelsim 实现仿真结果,可以看到,在一帧反向投影图像输入超过计算窗口范围后,Mean Shift 矢量的 x 与 y 值就计算出来了。

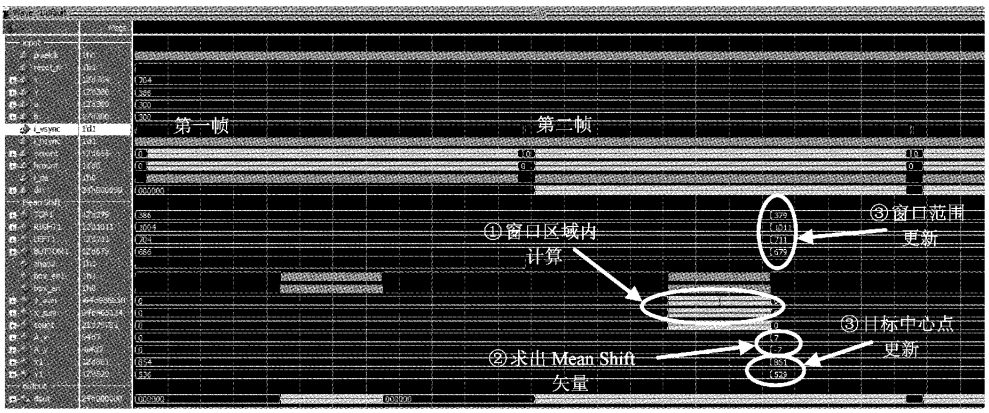


图 10　Mean Shift 矢量计算实现的 Modelsim 仿真

因为相邻两帧画面的较为相似,且目标的移动是连续的,位置改变范围不会太大,因此可以简化 Mean Shift 的流程,Mean Shift 迭代采用帧与帧之间迭代的方式,也就是说本模块的计算在一帧画面中只处理一次 Mean Shift 矢量的计算。Mean Shift 迭代的 Modelsim 仿真输出如图 11 所示。

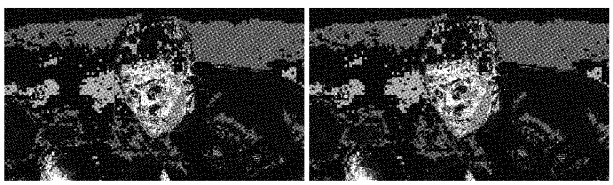


图 11　Mean Shift 迭代的 Modelsim 仿真输出

3　基于卡尔曼滤波的轨迹预测及抗干扰

Kalman 滤波即卡尔曼滤波算法基于线性系统模型,是一种对动态系统中的状态序列进行线性最优化估计的数据处理算法,可通过该算法得出当前模型的最优估计值及预测值[14]。该算法主要有“预测”和“更新”这两个处理过程。在目标跟踪处理上,滤波器会根据运动目标的当前运动位置、速度、加速度、目标大小等信息预测下次这些状态的值,“预测”后“更新”过程会通过综合观测值与之前的估计值修正目标状态[15]。本文将 Mean Shift 跟踪算法与 Kalman 滤波的预测机制相结合,既可以减少 Mean Shift 算法迭代次数,也能解决遮挡问题。

卡尔曼滤波的计算流程如图 12 所示,主要由五个公式组成:先对系统在给定初值的情况下进行预测计算,后状态更新部分会对预测结果与观测结果进行一个修正,该系统在连续时刻状态下可以预测未来的状态。

在视频图像处理的目标跟踪过程当中,相邻两帧的目标运动相隔距离很短,可以认为所做的运动是匀速运动,

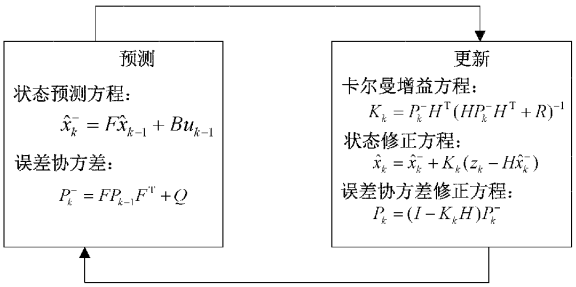


图 12　Kalman 滤波公式

因此跟踪目标的信息在卡尔曼滤波算法适用范围之内。以此目标运动模型建立的卡尔曼滤波可以提前预测到下一帧目标的大体位置，可以减少跟踪过程中搜索区域的范围，减少大量处理时间。并且，当跟踪过程当中目标出现被前景物体遮挡时，可以迭代自身预测结果，达到遮挡时处理并保证持续跟踪的效果。

本文采用目标状态向量为(x，y，dx，dy)。即认为相邻两帧图像中的目标为匀速运动，且长宽大小保持不变。并且，由于 x 与 y 轴方向上的运动可以视为互不影响的两个运算，其模型公式结构可以分解为两个一维卡尔曼，且 FPGA 可并行同时处理水平和垂直方向两个方向的计算，处理时间也能减少一半。

参照上面卡尔曼滤波处理的几个主要公式，可以推导出以下本文卡尔曼模型下的计算公式。

其运动模型为状态量选择为(x，dx)，x 表示垂直或水平方向的坐标，dx 表示目标在单位时间运动的变化量，也就是速度。取该模型目标为匀速运动，没有外部输入量，不考虑系统噪声，即状态量 $\boldsymbol{x}_k$ 为 $(x \quad \mathrm{d}x)^{\mathrm{T}}$，$\boldsymbol{u}_k$ 为 0，$\boldsymbol{\omega}_k$ 为 0。则状态方程为：

$$\boldsymbol{x}_k = F\boldsymbol{x}_{k-1} = \begin{bmatrix}1 & 1\\ 0 & 1\end{bmatrix}\begin{bmatrix}x_{k-1}\\ \mathrm{d}x_{k-1}\end{bmatrix} \tag{8}$$

观测方程为：

$$\boldsymbol{z}_k = H\boldsymbol{x}_k = [1 \quad 0]\boldsymbol{x}_k \tag{9}$$

其中，$\boldsymbol{F} = \begin{bmatrix}1 & 1\\ 0 & 1\end{bmatrix}$，$\boldsymbol{H} = [1 \quad 0]$。

1)先验估计为：

$$\hat{\boldsymbol{x}}_k^- = F\hat{\boldsymbol{x}}_{k-1} = \begin{bmatrix}1 & 1\\ 0 & 1\end{bmatrix}\begin{bmatrix}x\\ \mathrm{d}x\end{bmatrix} \tag{10}$$

其中，设其结果为 $\begin{bmatrix}l_1\\ l_2\end{bmatrix}$，则有：

$$l_1 = x + \mathrm{d}x$$
$$l_2 = \mathrm{d}x$$

2)先验估计协方差为：

$$\boldsymbol{P}_k^- = \boldsymbol{F}\boldsymbol{P}_{k-1}\boldsymbol{F}^{\mathrm{T}} + \boldsymbol{Q} = \begin{bmatrix}1 & 1\\ 0 & 1\end{bmatrix}\begin{bmatrix}p_{11} & p_{12}\\ p_{21} & p_{22}\end{bmatrix}\begin{bmatrix}1 & 0\\ 1 & 1\end{bmatrix} + \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} = \begin{bmatrix}p_{11} + p_{21} + p_{12} + p_{22} + 1 & p_{12} + p_{22}\\ p_{21} + p_{22} & p_{22} + 1\end{bmatrix} \tag{11}$$

其中，设了经验值 $\boldsymbol{Q} = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$，设先验估计协方差结果为 $\begin{bmatrix}p_{11}^- & p_{12}^-\\ p_{21}^- & p_{22}^-\end{bmatrix}$ 且 $\boldsymbol{P}_{k-1} = \begin{bmatrix}p_{11} & p_{12}\\ p_{21} & p_{22}\end{bmatrix}$，分解公式则有：

$$\begin{aligned}p_{11}^- &= p_{11} + p_{21} + p_{12} + p_{22} + 1\\ p_{12}^- &= p_{12} + p_{22}\\ p_{21}^- &= p_{21} + p_{22}\\ p_{22} &= p_{22} + 1\end{aligned} \tag{12}$$

3)卡尔曼增益为：

$$\boldsymbol{K}_k = \boldsymbol{P}_k\boldsymbol{H}^{\mathrm{T}}(\boldsymbol{H}\boldsymbol{P}_k\boldsymbol{H}^{\mathrm{T}} + R)^{-1} = \frac{\begin{bmatrix}p_{11}^-\\ p_{21}\end{bmatrix}}{p_{11}^- + 1} \tag{13}$$

其中，取了 R 为 1，设结果为 $\begin{bmatrix}k_1\\ k_2\end{bmatrix}$，则分解公式有：

$$\begin{aligned}k_1 &= \frac{p_{11}^-}{p_{11}^- + 1}\\ k_2 &= \frac{p_{21}^-}{p_{11} + 1}\end{aligned} \tag{14}$$

4)修正估计为：

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k(z_k - \boldsymbol{H}\hat{\boldsymbol{x}}_k^-) = \begin{bmatrix}l_1\\ l_2\end{bmatrix} + \begin{bmatrix}k_1\\ k_2\end{bmatrix}\left([1 \quad 0]\begin{bmatrix}x_k\\ \mathrm{d}x_k\end{bmatrix} - [1 \quad 0]\begin{bmatrix}l_1\\ l_2\end{bmatrix}\right) = \begin{bmatrix}l_1 + k_1(x_k - l_1)\\ l_2 + k_2(x_k - l_1)\end{bmatrix} \tag{15}$$

其结果就为修正后的：$\begin{bmatrix}x\\ \mathrm{d}x\end{bmatrix}$

5)更新后验协方差为：

$$\boldsymbol{P}_k = (\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H})\boldsymbol{P}_k^- = \left(I - \begin{bmatrix}k_1\\ k_2\end{bmatrix}[1 \quad 0]\right)\begin{bmatrix}p_{11}^- & p_{12}^-\\ p_{21}^- & p_{22}^-\end{bmatrix} = \begin{bmatrix}p_{11}^- - k_1p_{11}^- & p_{12}^- - k_1p_{12}^-\\ p_{21}^- - k_2p_{11}^- & p_{22}^- - k_2p_{12}^-\end{bmatrix} \tag{16}$$

设其结果为：$\begin{bmatrix}p_{11} & p_{12}\\ p_{21} & p_{22}\end{bmatrix}$，则有：

$$\begin{aligned}p_{11} &= p_{11} - k_1p_{11}\\ p_{12} &= p_{12}^- - k_1p_{12}^-\\ p_{21} &= p_{21}^- - k_2p_{11}^-\\ p_{22} &= p_{22}^- - k_2p_{12}^-\end{aligned} \tag{17}$$

通过分析上述公式可以发现，虽然卡尔曼滤波公式主要是进行矩阵计算，但经过优化拆解分析后，其主要运算还是由一系列的乘除法、加减法构成。因此，FPGA 逻辑设计可以根据图 12 的流程建立状态机，并运用 FPGA 的浮点数计算 IP 实现本算法。

FPGA 在处理一般简单计算时可以选择浮点数转定点数来解决小数的计算，该方法也可以达到一定的计算精

度,可以保证 FPGA 资源的合理利用。但是,在卡尔曼滤波中,相比较定点数处理计算公式的复杂度,使用 Xilinx 平台的浮点数计算 IP 精度更高且更易实现。本文选择 32 bit 的单精度浮点数计算,如图 13 所示,其中有 1 bit 是符号位,8 bit 是指数位,23 bit 是尾数位。

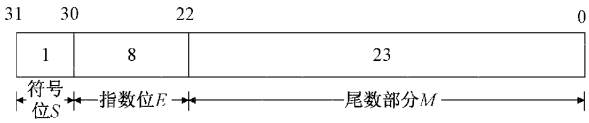


图 13　单精度浮点数数据格式

其代数形式为:

$$X = (-1)^s \times 1.M \times 2^{(E\ \ bias)} \tag{18}$$

如图 14 所示,由于输入和需要输出外部使用的坐标数据都是以定点数方式传输的,因此在卡尔曼滤波模块的两端添加定点/浮点转换模块。而中间部分则为滤波器的 5 个公式部分。

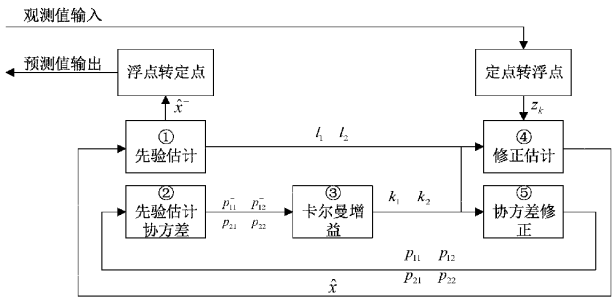


图 14　卡尔曼滤波整体模块框架

4　验证与分析

4.1　卡尔曼滤波

Kalman 滤波 5 个计算模块实现完成后,采用正弦信号作为输入信号测试该模块的预测效果。仿真时,外部文件每输入一次模拟坐标位置的正弦信号采样值,计算模块就做一次“预测”计算,并将仿真过程中的结果数据将输出到外部的文件中。由图 15 可以看到,经过多次迭代后的 Kalman 滤波模块可以较为准确的“预测”正弦信号的轨迹。

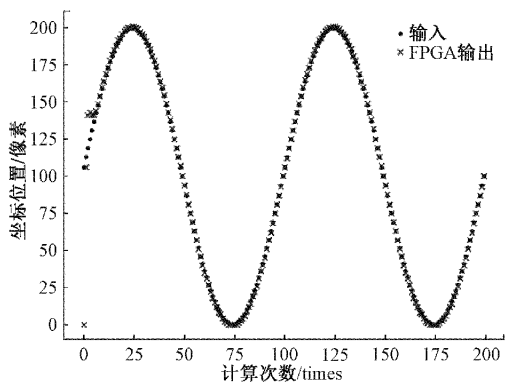


图 15　FPGA 仿真结果与输入对比

既然能够完成水平或垂直单个方向的卡尔曼滤波,再次通过模块分时复用的方法,就可实现目标坐标的完整卡尔曼滤波过程。如图 16 仿真所示,输入为一系列坐标数据,卡尔曼滤波模块每次都能在 out_vld 拉高时输出结果。

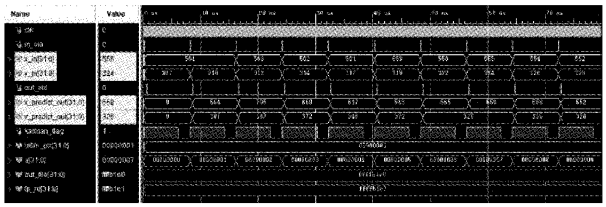


图 16　卡尔曼滤波模块仿真

如图 17 和图 18,其横纵坐标值代表目标在画面中的位置坐标值。通过对比输入与输出结果,可以看到通过多次迭代后,FPGA 计算可以按照预期预测下一步数据,整体效果良好。再与 MATLAB 实现卡尔曼滤波结果对比可发现,FPGA 计算结果仅仅是四舍五入了小数部分,两者结果几乎完全一致。

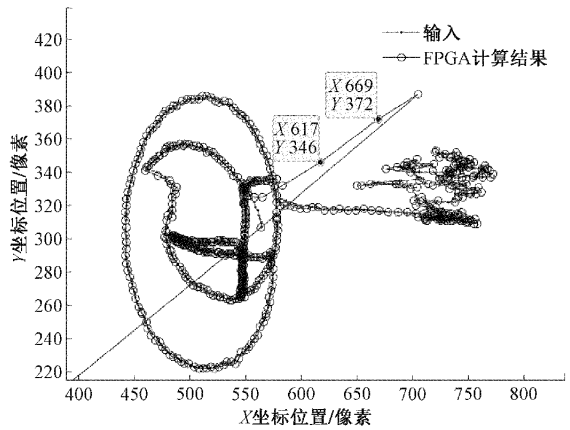


图 17　FPGA 仿真结果值与输入对比

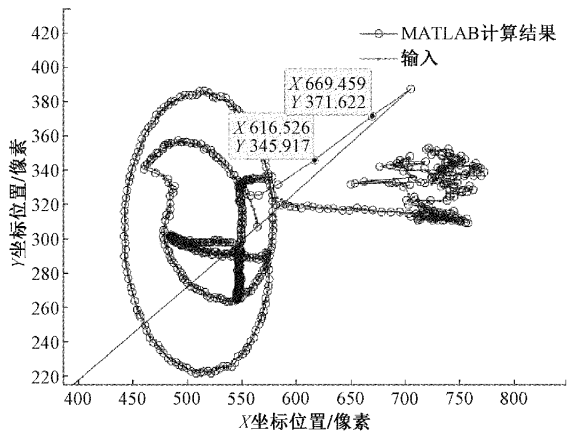


图 18　MATLAB 计算结果与输入对比

4.2　跟踪结果测试

为了测试本设计的目标跟踪性能效果,本部分做了跟踪测试实验。使用上位机控制 FPGA 的 HDMI 接口输入

视频，跟踪处理后结果输出显示到另外一个显示器。如图 19 所示，左边显示器为上位机控制界面，右边显示屏为跟踪结果输出画面，测试视频参数选择为 1 920×1 080@60 Hz，可以看到该可以实现该测试视频场景下的跟踪。

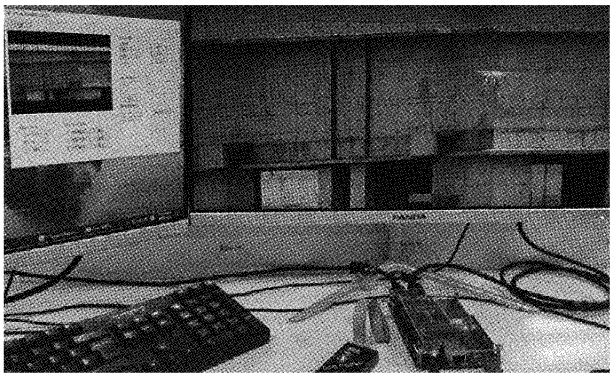


图 19　跟踪测试平台及结果输出

图 20 所示为测试对应的结果和重叠率波形图，显示了每帧跟踪结果框与实际目标位置的重合比例，纵轴 100 指跟踪结果与实际目标完全重合，0 表示跟踪结果框与实际目标位置框无重合（跟丢）。可以看到该算法可以连续持续地跟踪视频中的目标。

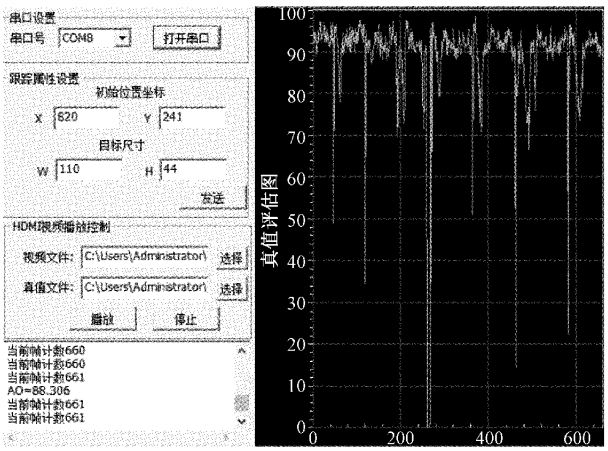


图 20　跟踪结果

5　结　　论

视频图像的目标跟踪研究具有广阔的应用前景，本文 FPGA 平台上的设计实现了高清图像下的实时运动目标检测和跟踪，且有一定的抗干扰能力和较好的准确性。但同样的，本 FPGA 平台下的算法实现也存在着一些缺点。首先，虽然跟踪过程中有模型更新机制，但在一些场景下会出现更新过度或更新过少导致丢失的情况。其次此次的跟踪实现只支持单目标的跟踪。综上所述，本设计在场景适应能力上还有很大的进步空间，未来工作将会在提高目标模型准确度和多目标场景适应性方面开展。

参考文献

[1]　益争祝玛，尚振宏，刘辉，等. 基于多特征融合的运动目标跟踪方法[J]. 仪表技术与传感器，2019(11)：95-99.

[2]　ZHANG Y，WANG T，LIU K，et al. Recent advances of single-object tracking methods：A brief survey[J]. Neurocomputing，2021，455：1-11.

[3]　高赟，徐子钦，王涛，等. 采用多边形质心的相关滤波跟踪位置校正方法[J]. 仪器仪表学报，2021，42(5)：159-172.

[4]　PANDEY J G. An embedded FPGA-SoC framework and its usage in moving object tracking application[J]. Design Automation for Embedded Systems，2021，25(3)：213-236.

[5]　HE W，ZHANG J，LIN，et al. A low-cost high-speed object tracking VLSI system based on unified textural and dynamic compressive features [J]. IEEE Transactions on Circuits and Systems II：Express Briefs，2020，68(3)：1013-1017.

[6]　YASUKAWA S，OKUNO H，ISHII K，et al. Real-time object tracking based on scale-invariant features employing bio-inspired hardware [J]. Neural Networks，2016，81：29-38.

[7]　石海林，付敏，朱革，等. 基于 FPGA 的近场无透镜图像检测系统设计[J]. 仪表技术与传感器，2021(7)：81-86.

[8]　YU X，LI B，DONG M，et al. Target Detection and Tracking System Based on FPGA[J]. IOP Conference Series Materials Science and Engineering，2020，793(1)：012008.

[9]　刘美枝，杨磊，高海. 结合角点特征的 CamShift 目标跟踪算法研究[J]. 山西大同大学学报(自然科学版)，2019，35(5)：14-18.

[10]　严飞，夏金锋，马可，等. 基于 Mean Shift 算法的目标跟踪系统设计[J]. 电子测量技术，2020，43(23)：6-11.

[11]　SONG Z，DENG W，LENTDECKER G，et al. Study of the retina algorithm on FPGA for fast tracking[J]. Nuclear Science and Techniques，2019，30(8)：1-8.

[12]　鲁国智，石晶辉，彭冬亮，等. 基于 FPGA＋DSP 架构的嵌入式视觉跟踪系统[J]. 机电工程，2017，34(4)：416-420.

[13]　高文，朱明，刘剑，等. 基于 DSP＋FPGA 框架的实时目标跟踪系统设计[J]. 液晶与显示，2014，29(4)：611-616.

[14]　李鲁明，赵鲁阳，唐晓红，等. 基于模糊卡尔曼滤波的姿态估计算法[J]. 仪表技术与传感器，2019(4)：100-105.

[15] 徐畅. 基于 FPGA 的单目标跟踪系统设计[D]. 南京:南京理工大学, 2016.

作者简介

刘银萍,博士,主要研究方向为安全工程及应用技术。
E-mail:yinpingliu@nuist.edu.cn

夏金锋,硕士研究生,主要研究方向为基于 FPGA 的硬件加速处理技术。
E-mail:jf_xia@163.com