Journal of Wuhan Polytechnic

基于 LabVIEW 的目标跟踪系统软件设计

吴丹阳，魏元焜

（辽宁机电职业技术学院 华孚仪表学院，辽宁 丹东 118009）

摘 要：在机器视觉系统中，目标跟踪是常用的功能需求。在大型的系统设计中，经常需要多任务的并行处理。利用 LabVIEW 高效的并行执行特性，给出了一种目标跟踪系统的程序设计方案。该设计方案以 LabVIEW 为软件设计平台，基于 NI Vision 机器视觉库，可快速实现目标跟踪功能。利用 LabVIEW 面向对象编程方法和队列消息处理器程序设计模式，可将目标跟踪功能与系统进行高效耦合并降低系统功能的依赖程度，可大幅提高系统的可扩展性和可维护性。实际验证表明，提出的方案可快速实现预期功能。

关键词：机器视觉；目标跟踪；LabVIEW；NI Vision；设计模式

中图分类号：TP391.41　　**文献标识码：**A　　**文章编号：**1671-931X（2023）03-0114-07

DOI：10.19899/j.cnki.42-1669/Z.2023.03.018

一、研究背景

随着机器视觉技术迅猛发展，相应的设计需求和开发工具也不断进步，随之而来对设计效率要求不断提升。早期的机器视觉设计语言以 MATLAB、C++ 为主[1~2]，通常配合 MATLAB 自带的图像处理库和 Open CV 等图像处理函数库来完成设计任务[3]。但 MATLAB 的商用成本较高，而 C++ 的语法复杂、可用的类库较少。近年来，Python 语言因其简洁、友好的编程语法受到广泛关注，基于 Python 的机器视觉库也越来越多，在一定程度上缩减了开发周期。但 Python 在界面设计上仍相对繁琐[4~5]。NI 公司的 LabVIEW 开发平台的设计初衷即为科学工作者提供简洁的界面开发环境，其所见即所得的界面设计方式和先天的多线程编程优势，可以使程序开发者摆脱繁琐的语法要求；其包含的丰富科学计算、图像处理模块和 NI Vision 视觉库可以使开发者不必关注底层的实现细节，专注于设计需求本身，进而极大地提高开发效率[6]。本文基于 LabVIEW 开发平台和 NI Vision 机器视觉库，以实现目标跟踪为例，给出一种机器视觉系统的软件设计方案。同时，通过采用成熟的队列消息处理器设计模式，进一步提高系统的扩展性和可维护性。

二、设计方案采用的工具、算法和设计模式

（一）NI Vision 库

NI Vision 视觉库隶属于 NI 公司的视觉开发模块，该模块包含 NI Vision Builder 和 IMAQ Vision

收稿日期：2022-06-06

基金项目：2021 年辽宁省教育厅科学研究经费项目（面上项目）“军训用枪的瞄准镜图像处理系统研究与实现”（项目编号：LJKZ1261）；2021 年辽宁机电职业技术学院科研项目“军训用枪的瞄准镜图像处理系统研究与实现”（项目编号：ky202102）。

作者简介：吴丹阳（1987— ），女，辽宁丹东人，辽宁机电职业技术学院华孚仪表学院讲师，研究方向：虚拟仪器技术、机器视觉；魏元焜（1987— ），男，辽宁丹东人，辽宁机电职业技术学院华孚仪表学院讲师，研究方向：虚拟仪器技术、机器视觉。

两个部分。IMAQ Vision 以 LabVIEW 语言封装了大量的图像处理库函数，包括：图像文件 IO、标定、Overlay、颜色提取、分类、匹配、灰度图像形态学、二进制图像形态学、形状匹配、性状检测、机器学习、模式识别等丰富的功能。而 NI Vision Builder 是为了进一步简化图像处理任务而设计的交互式开发环境，但其对 LabVIEW 平台的依赖较强，不便于独立使用，本文主要使用 IMAQ Vision 库。

（二）MeanShift 算法

Meanshift 聚类算法是一种无参数的聚类算法，能够在根据样本点计算数据概率密度分布区间。该算法已成功应用于图像平滑、图像分割和运动目标跟踪等领域[7]。

设 R^d 为 d 维空间，$x=\{x_i\}(i=1,2,...,n)$ 为离散数据集合，MeanShift 可由式(1)定义。

$$M_h(x)=\frac{1}{k_n}\sum_{x_i\in S_h}(x_i-x) \qquad 式(1)$$

其中，$S_h(x)=\{y:(y-x)^T(y-x)\le h^2\}$ 为以 h 为半径的球体区域。通过引入核函数 $K(x)=o_{k,d}k(\|x\|^2)$ 可得到概率密度函数如式(2)所示。

$$f(x)=\frac{1}{nh^d}\sum_{i=1}^{n}K\left(\frac{x-x_i}{h}\right) \qquad 式(2)$$

其中，o 为正则化系数，使得核函数不定积分为 1。通过对式求偏导求极值，即可得 MeanShift 所指向的最大概率密度梯度方向。MeanShift 算法的实质就是通过不断迭代搜索数据分布概率密度分布梯度峰值，使其满足阈值要求[8]。

（三）队列消息处理器设计模式

在大型的软件设计中，使用适当的设计模式来规范程序设计是提高软件质量的必要前提。LabVIEW 的图形化编程具有先天的并行执行优势，但在使用不当时，也会影响软件设计质量。例如，在一个小型的 LabVIEW 程序中，通常可以使用一个简单的 while 循环来完成任务，而随着软件功能的增加，循环内的工作也会越来越多，致使 while 循环内的程序框图极其臃肿、连线密布交错，严重影响了程序的可读性和可维护性[9]。

本文采用的是队列消息处理器设计模式，这种设计模式的基础是状态机设计模式[9]。状态机将系统的工作流程划分为多个状态，系统功能的状态在程序中以状态迁移的方式实现。在多个状态中，通常需要有一个空闲状态，在空闲状态中主要完成检测用户的界面操作指令和刷新界面数据两项工作。当检测到用户操作指令后，根据设计需求进行状态迁移，实现指定功能和运算，根据运算结果更新内部数据，这些数据最终会在界面刷新中反馈给用户，最终实现整个程序功能的运转。

队列消息处理器在状态机的基础上，加入了消息机制，将状态迁移指令视作状态机自身的消息。在很多情况下，某个状态需要指定后续多个连续执行的状态，这就需要将消息指令进行排列。为实现这一需求，本文以队列数据结构实现了队列化的消息，简称为消息队列。消息队列对消息的处理方式遵循队列数据结构“先入先出”的特性，如图 1 所示。

图 1　消息队列数据结构

三、目标跟踪系统设计

本节将在介绍整体框架的基础上，详细介绍以本文方案实现目标跟踪软件系统的方法和过程，系统整体框图如图 2 所示。

系统运行于 LabVIEW 平台之上。有用于在用户界面发出指令，系统的 UI 事件检测功能检测到用户的指令；程序逻辑需求（状态迁移规则）会根据用户指令生成一个或多个状态迁移指令，并将全部指令作为新消息压入消息队列；状态机作为程序运行引擎，负责程序的运转、指令发送和提取，它根据本次状态的执行结果决定接下来要执行的功能（即接下来要迁移到的状态）并将其压入消息队列，在下次循环中从队首读取一个消息，转入对应状态中实现相应的逻辑功能，将运行结果反馈给用户。基于这一框架，本文依次介绍了状态机、消息队列、目标跟

踪相关逻辑控制模块的实现。

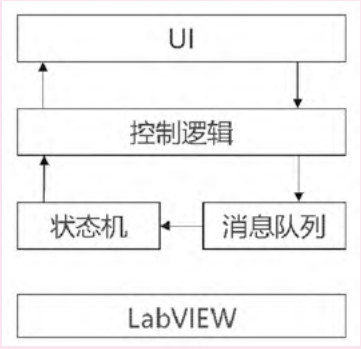
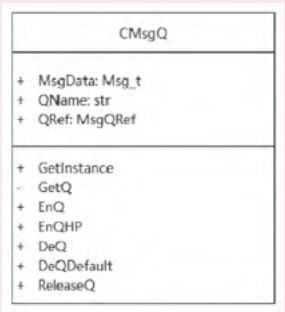


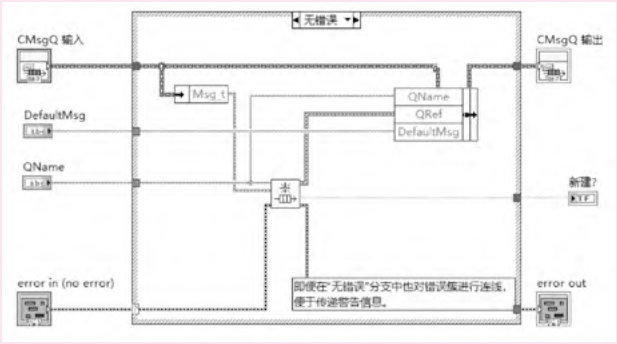
图 2　目标跟踪系统框图

（一）基于面向对象编程的消息队列设计

利用 LabVIEW 面向对象编程[10]方法设计了消息队列类，类如图 3(a)所示。其中，类属性定义了消息数据(MsgData，包含消息名称和参数)、队列名称(QName)以及队列引用(QRef)。类方法定义了获取队列引用(GetInstance)、获取消息队列(GetQ)、消息入队(EnQ)、消息高优先级入队(EnQHP)、阻塞消息出队(DeQ)、超时消息出队(DeQDefault)以及释放队列引用(ReleaseQ)。



消息队列类图



GetQ 方法的实现

图 3　LabVIEW 消息队列类图

图 3 以获取消息队列(GetQ)为例，给出了消息队列类方法的代码实现，类中的其他方法与之类似，都是在无错误的分支下对 LabVIEW 的消息队列函数和相关数据进行了封装。CMsgQ 类的所有方法功能见表 1 所示。

表 1　CMsgQ 类方法说明

方法名称	作用
GetInstance	对 GetQ 的封装，不需要实例化对象直接获取队列引用
GetQ	获取队列引用，但需要先实例化对象，私有方法
EnQ	消息入队
EnQHp	高优先级(队首)消息入队
DeQ	消息出队，阻塞式
DeQDefault	默认消息出队，超时 10ms 的非阻塞式消息出队，超时后返回默认消息(状态机的空闲状态)
ReleaseQ	释放消息队列

（二）消息队列状态机设计

消息队列状态机整体框架如图 4 所示，由队列初始化、状态循环和释放队列几个关键步骤构成。初始化时，由 GetInstance 方法获取队列引用，再以 EnQ 方法指定状态机初态(Init)。状态循环是整个状态机的核心，它由一个 While 循环维持状态机的持续运转，以一个条件结构容纳了各个状态的代码，类似于 C 语言的 switch-case 结构。条件结构的条件接线端连接到 DeQDefault 方法的消息输出端，该方法以 10ms 为超时限制，实现非阻塞的消息出队。如果当前队列中有积压的消息命令，则从队首直接读取一个消息，否则直接该方法直接返回一个默认消息(即 Idel 状态)。条件结构根据 DeQDefault 输出的消息，实现状态迁移，在不同的状态里实现不同的功能。在各个状态中，由控制逻辑决定了次态。次态由条件结构右侧的 EnQ 方法压入消息队列，这一新的消息将在后续的某次 While 循环中被从队列中取出，并迁移到对应状态。由此，实现了状态机的运转。在后期扩展中存在耗时任务时，可将耗时任务从状态机主循环中移除，布置在下方的 VideoGrab.vi。通

过这种多线程并行的机制，可以使状态机不受耗时操作的限制，可以及时响应用户指令、更新界面以及迁移状态。

图 5 给出了目标跟踪系统的状态迁移过程。状态机的初始化状态(Init)由状态机开始前的 EnQ 方法指定，主要负责内部数据和界面控件的初始化。初始化完成后直接迁移到空闲状态(Idel)，这一状态主要负责检测用户发出的指令，并在超时等待后根据控制逻辑要求决定状态机的次态。若用户未发出开始追踪指令，则空闲状态等待超时后，迁移到界面更新状态(UpdateUI)，并再次迁移回空闲状态，循环往复。在此循环中，一旦用户发起开始目标追踪的要求，将会被空闲状态捕捉到，则迁移到设置跟踪目标状态(SetTarget)，此状态主要负责设置关注区(ROI)和跟踪算法的初始化，并设定跟踪开始标志变量，随后则直接迁移到跟踪状态(Trace)实现目标跟踪，并给出跟踪位置，最后将跟踪位置在界面更新状态里显示。由于此时设置了开始跟踪标志位，在空闲状态超时后，将直接迁移到跟踪状态，如此循环往复。在这一循环中，一旦用户取消了跟踪要求，则立即取消开始跟踪标志，在下一个空闲状态完成后，则直接迁移到界面更新状态，而不再进行目标追踪。

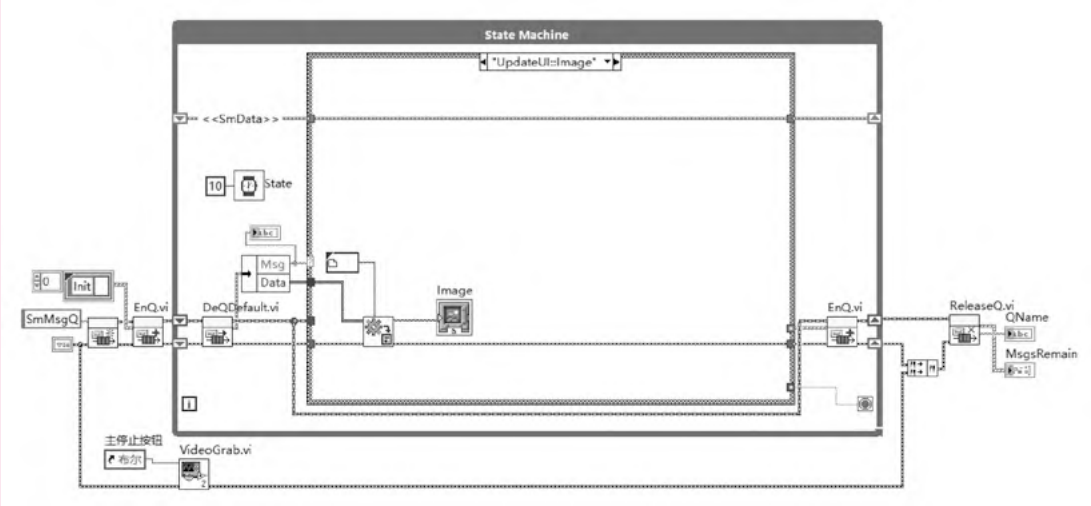


图 4　消息队列状态机设计模式程序框图

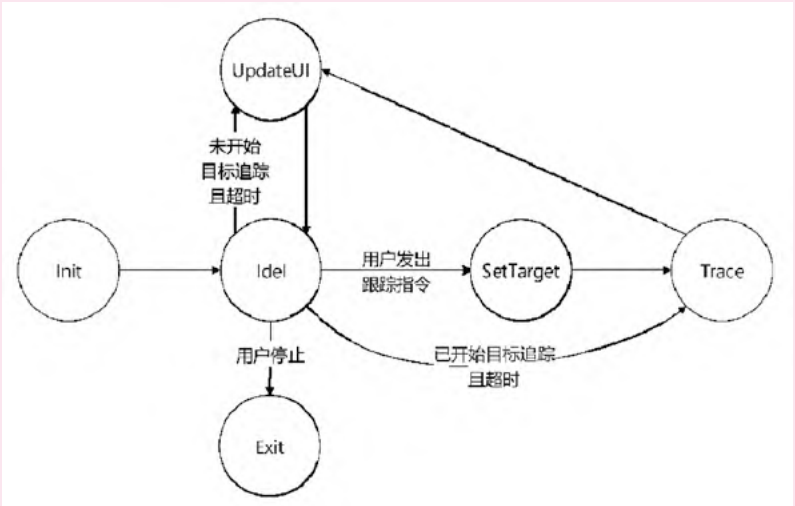


图 5　目标跟踪系统状态迁移图

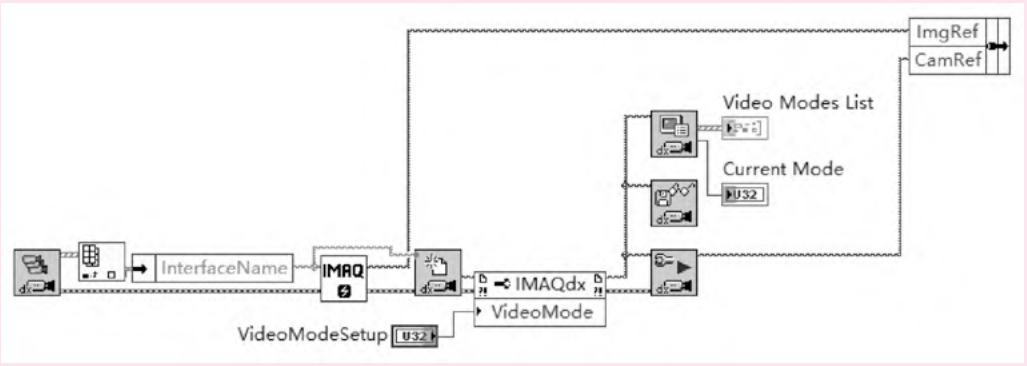
结合图 4 可知，队列消息处理器模式以一个很小规模的程序框图实现了复杂的逻辑控制和状态迁移功能。此外，如果需要添加一个新的功能，仅需将该功能拆解为若干个新的状态，以新的条件结构分支的方式添加到状态机中，通过合理的逻辑控制设计状态迁移规则，在不扩大程序框图规模的情况下，

灵活地实现了系统功能的增加和修改。这极大地提高了本系统的可维护性和程序框图的可读性。

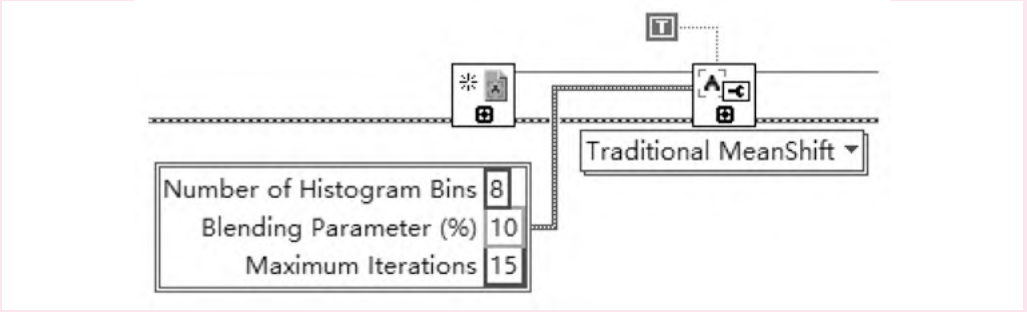
（三）基于 NI Vision 库的目标跟踪设计

基于 NI Vision 库的目标跟踪的 LabVIEW 核心代码如图 6 所示。如果将这些代码完全展开，则会使程序框图占用较大面积，连线凌乱，降低代码可读性和软件的可维护性。因此需要将其整合到图 5 所示的各相关状态中。

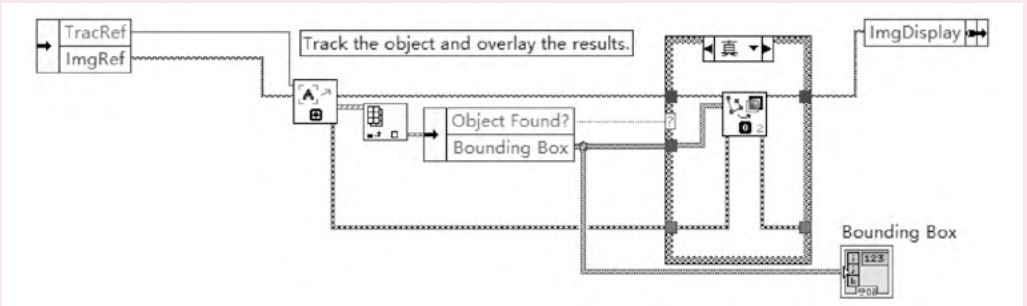
图 6(a) 实现了目标跟踪的初始化工作，在初始化状态(Init) 实现，这部分代码依次实现了摄像头枚举、在内存中为 IMAQ 图片开辟内存空间、打开枚举到的摄像头、显示摄像头基本信息、工作模式以及配置图像采集信息。初始化后的数据信息通过按名称捆绑簇保存到状态机数据中，在其他状态中，即可通过按名称解绑访问这些资源，这种方式也会应用在后续的状态中。图 6(b)给出了跟踪器的初始化过程，创建了跟踪器引用，并配置跟踪参数、选择跟踪算法的跟踪算法。图 6(b) 实现的也是初始化工作，但是如果和图 6(a) 合并在一起会使程序框图占用较大空间，因此可以创建两个初始化类的状态：Init::Cam、Init::Trace。在状态开始运行前，将这两个状态依次压入消息队列，即可在状态机运行时按顺序实现状态迁移。图 6(c) 实施具体的跟踪过程，如果检测到目标，则将检测到的目标位置层叠到抓取到的图像上，否则不对抓取到的图像进行处理。



(a) 初始化摄像头



(b) 初始化跟踪器(采用传统 MeanShift 算法)



(c) 实施跟踪并显示跟踪结果

图 6　目标跟踪核心代码

四、实验验证

在CPU为Intel(R) Core(TM) i5-8500、内存为8GB的计算机上对系统功能进行了验证，以人脸作为跟踪目标，在相对复杂的背景下进行了目标跟踪测试，运行效果如图7所示。结果表明，系统成功实现了指定目标的追踪，可见使用的MeanShift跟踪算法可以实现一般目标的有效追踪，且对相对复杂的背景和存在一定干扰的情况下仍能取得较好的跟踪效果。

图7　程序运行效果验证

五、结论

本文提出了一种高效、灵活的目标追踪系统设计方案，其高效性和灵活性体现在便于功能扩展、集成新的功能以构建更加复杂的机器视觉系统，这一特点得益于本文采用了LabVIEW面向对象编程设计方式以及本文实现的消息队列状态机架构。对于相对耗时的操作，可将其分配置新的消息处理循环，使之独立于UI线程，提高程序运行效率，防止界面的卡顿。系统采用的MeanShift追踪算法由NI Vision库实现，可以对一般目标实施有效的跟踪，对于实现短开发周期的机器视觉系统设计具有实用价值。

参考文献：

[1] 刘忠超，盖晓华.基于机器视觉和PLC的猕猴桃分级控制系统设计[J].中国农机化学报，2020(01):131-135.

[2] 张丽英，马春平，郭翰韬，等.基于OpenCV的LED灯丝上电快速视觉检测系统[J].计算机应用，2021(S2):275-279.

[3] 王晓东，赵志诚，叶泽甫.基于机器视觉的浇口杯定位研究[J].铸造，2021(12):1447-1452.

[4] 李泽平，郭俊先，郭阳，等.基于支持向量机的无核白葡萄串分级系统设计与测试[J].食品与机械，2021(10):106-111.

[5] 李全全.Python OpenCV在智慧党建人脸识别中的应用[J].中国有线电视，2020(2):167-171.

[6] 杨高科.图像处理、分析与机器视觉(基于LabVIEW)[M].北京：清华大学出版社，2018:36-37.

[7] 陈薇，袁文定，方强，等.基于自适应卡尔曼滤波的Meanshift跟踪算法[J].制造业自动化，2021(06):16-20,

[8] 王会.视频监控下的人脸跟踪与识别系统[J].计算机工程与应用,2014(12):175-179.
[9] 杨高科.LabVIEW虚拟仪器项目开发与管理[M].北京:机械工业出版社,2012:82-90.
[10] 陈树学,刘萱.LabVIEW宝典[M].北京:电子工业出版社,2011:342-346.

[责任编辑:胡大威]

Software Design of Target Tracking System Based on LabVIEW

WU Danyang, WEI Yuankun
(Huafu Instrument College,Liaoning Mechatronics College, Dandong,Liaoning,118009,China)

Abstract: In machine vision systems, object tracking is a common functional requirement. In large-scale system design, parallel processing of multiple tasks is often required. This article utilizes the efficient parallel execution feature of LabVIEW to provide a programming solution for an object tracking system. This design scheme uses LabVIEW as the software design platform and is based on the NI Vision machine vision library, which can quickly achieve object tracking function. By utilizing the LabVIEW object-oriented programming method and queue message processor programming pattern, the object tracking function can be efficiently coupled with the system, reducing the dependency on system functions, and significantly improving the scalability and maintainability of the system. Experiment shows that the proposed scheme can quickly achieve the expected functions.

Key words: Machine vision; Object tracing; LabVIEW; NI Vision; Design pattern