学号 20154452

密级\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**东北大学本科毕业论文**

基于强化学习的在轨自适

应光流提取方法设计与实现

学 院 名 称：计算机科学与工程

专 业 名 称 ：计算机科学与技术

学 生 姓 名 ：杨哲

指 导 教 师 ：栗伟 教授

二○一八年六月

**郑 重 声 明**

本人呈交的学位论文，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确的方式标明。本学位论文的知识产权归属于培养单位。

本人签名： 日期：

摘 要

光流是计算机视觉问题中的一个基本要素，通过光流提取可以完成诸多任务，虽然光流法的应用很广泛，但是其面向的环境大都不同，所以针对相异的环境必须使用与其相适应的方法来完成，这就对光流的先验参数有了要求。本文将面向更高效地选择参数的问题，自适应地为稀疏光流提取算法寻找合适的参数，通过遗传算法首先完成对自适应提取的实验方法设计，再引入强化学习方法，对遗传算法的架构进行优化，提升效率。

关键词：先验参数；稀疏光流；自适应；遗传算法；强化学习

**ABSTRACT**

Optical flow is one of the basic element of the Computer Vision problems .Although optical flow is widely used and extracting optical flow can complete couple of mission ,environment it faces always can’t be the same .So aiming at different circumstances ,we must make suitable plans which give the prior parameters a high requirement to solve .This paper will concentrate on how to choose the parameters effectively ,which means searching suitable parameters for sparse optical flow adaptively .First using Genetic Algorithm finishes the adaptive part ,then optimizing this GA part with Reinforcement Learning to make it more effective .

**Key words**: prior parameter ; sparse optical flow ; adaptive ; Genetic Algorithm ;Reinforcement Learning

目录

[1 绪论 7](#_Toc24139)

[1.1 研究背景 7](#_Toc32284)

[1.2 国内外研究现状 7](#_Toc25459)

[1.3 研究内容和技术路线 9](#_Toc26752)

[1.3.1 研究内容 9](#_Toc28627)

[1.3.2 研究方法 9](#_Toc13655)

[1.3.3 预期结果 10](#_Toc13695)

[2 光流法的目标追踪应用研究 11](#_Toc22877)

[2.1 光流法原理 11](#_Toc32109)

[2.1.1 光流的基本含义 11](#_Toc11313)

[2.1.2 Lucas-Kanade算法的工作原理 12](#_Toc12605)

[2.2 基于光流法的目标追踪系统设计 12](#_Toc1922)

[2.2.1 核心函数介绍 12](#_Toc18471)

[2.2.2 系统模块 13](#_Toc8393)

[2.2.3 效果展示 13](#_Toc22620)

[2.3 光流法的先验参数与评价标准研究 13](#_Toc31147)

[2.2.1 先验参数介绍 13](#_Toc22617)

[2.2.2 评价标准研究 14](#_Toc25709)

[2.4 本章小结 14](#_Toc14341)

[3 遗传算法自适应方法设计 16](#_Toc17710)

[3.1 遗传算法的原理 16](#_Toc30610)

[3.1.1 遗传算法的基本含义 16](#_Toc15803)

[3.1.2 遗传算法的一般流程 16](#_Toc21103)

[3.2 基于遗传算法的参数自适应实验 16](#_Toc8507)

[3.2.1 在自适应光流提取上的应用原理 16](#_Toc1746)

[3.2.2 系统模块 17](#_Toc13830)

[3.2.3 实验结果 17](#_Toc18093)

[3.3 基于进化策略的实验与比较 18](#_Toc16226)

[3.3.1 进化策略与遗传算法的异同 18](#_Toc1274)

[3.3.2 系统模块 18](#_Toc22012)

[3.3.3 实验结果 18](#_Toc10954)

[3.4 本章小结 18](#_Toc3413)

[4 强化学习优化遗传算法的研究 19](#_Toc22912)

[4.1 强化学习方法介绍及相关机理 19](#_Toc5509)

[4.4.1 强化学习 19](#_Toc19479)

[4.4.2 Q-learning算法基本原理 19](#_Toc17392)

[4.2 基于RLGA方法自适应光流提取实验 19](#_Toc14183)

[4.2.1 遗传算法与Q-learning强化学习 19](#_Toc750)

[4.2.2 数据集介绍 20](#_Toc18884)

[4.2.3 系统模块 21](#_Toc26486)

[4.2.3 实验结果 21](#_Toc11375)

[4.3 本章小结 21](#_Toc25926)

1 绪论

1.1 研究背景

光流提取有着诸多分支，比如目标检测、超分辨率等。近年来随着卫星软硬件技术的迅速发展，卫星逐渐向小型化、低成本化、快速设计及部署的方向发展，每年有过百颗小卫星成功发射，这些小卫星有着不同的载荷和任务目标，物理参数，轨道等等，而在微小卫星的一些任务过程中，经常会拍摄到一些景象，里面会含有很多物体在移动，将这些物体当作目标进行检测与追踪，而光流很适合来完成这项任务，因此本次研究的光流提取方法，将面向目标检测与追踪领域进行应用的设计与实现。

现在已经有很多成熟的光流算法，但根据NFL定理所述，这些算法并不能适用于所有的应用场景，尤其是卫星上的应用，我们几乎不可能完全掌握星上环境的变化。这时，为了找到一个普适效果更好的方法，提出利用强化学习，自适应地为某种光流提取算法寻找合适的参数，来完成对不同场景，比如飞机、船舶、汽车等通用性较好的光流计算，实现相关任务。这就降低了了对先验参数的选择压力，让算法更高效，具有一定的工程价值。针对无法事先预知拍摄目标类型的卫星环境，如果在不能人工参与处理过程的情况下，也可以有效解决拍摄时的模糊问题，那么就会让最近比较新兴的软件定义卫星上的图像处理应用，变得更加的有效与方便，最终吸引更多的人来设计和投放APP，可以预见具有一定的经济价值。

1.2 国内外研究现状

现在已经有很多成熟的普适光流计算方法，如L-K 稀疏光流、Farneback全局光流法、HS光流法等等，也有很多的应用，林佳等通过Farneback算法计算图像某层的深度和灰度光流，并作为构建深度和灰度金字塔的基础，构建尺度空间。但是由于光流计算的复杂度太高，会使计算的效率降低，所以国内外的学者针对光流法也做了适当的改进。袁国武等提出一种对L-K稀疏光流算法的简化方法，通过选取图像中具有代表性的Harris角点，并只计算这些像素点的光流信息的方法来降低复杂度。还有些改进是与应用场景结合在一起的，比如秦剑等提出了结合云运动的特点和H-S光流方程的两个约束条件，特定的简化光流计算；曾凯在机器人通过光流计算避障问题时，发现了光流场中的矢量差值与机器人避障时TTC的变化关系，以此提升光流计算的效率。

光流法虽然可以解决计算机视觉领域的不少问题，但是并不一定是最优的解决方法，比如闫钧华发现，面对运动背景过于复杂的情况，光流法就不在适用于检测目标，取而代之的是通过数学模型计算并拟合背景的运动规律，预测背景的运动形势的背景建模方法。因此，为了更有效得适用光流计算算法，还需一些其它的方法工具予以辅助。例如，在[4]的研究中，是以光流估计为其中的一部分，通过利用代表背景复杂程度的背景因子，自适应调整分割阈值，来完成对背景因素的抑制，接着再通过基于梯度的光流方法，解决了面对红外背景下的云图像进行光流分析，提取运动云区，检测弱小目标的目标检测问题。

如何利用强化学习自适应地为算法找寻最合适的参数，针对这一点，也有不少学者在相关领域进行了实践，甚至实现了应用。Yu提出，面对图像恢复问题，提出了基于强化学习的方法，与大多数现有的基于深度学习的方法不同，该方法动态地选择一个工具链来逐步恢复被复杂和混合失真破坏的图像。并且在合成图像和真实图像上的大量实验结果验证了该方法的有效性，证明了该框架具有内在的灵活性。基于这个方法，通过学习和适当的转化，可以应用于更具有挑战性的恢复任务或其他低层视觉问题，比如图像的去模糊问题。

许柯等面对卫星技术的飞速发展造成的传统控制算法效率下降问题，对基于深度强化学习的姿态控制算法进行了研究。该算法无需对卫星的物理参数和轨道参数等先验知识，具有较强的适应能力和自主控制能力，可以满足软件定义卫星适应不同硬件环境，进行快速研发和部署的需求。此外算法通过引入目标网络和并行化启发式搜索算法之后，在网络精度和计算速度方面进行了优化，并且通过仿真实验进行了验证。不仅如此，作者发现基于目标网络的动力学模型网络可以明显提升算法的收敛精度，而基于进化策略的启发式搜索并行化算法可以大大提升算法的收敛速度，从而提高强化学习姿态控制系统在轨运行在线计算的能力。也就是说，深度强化学习可以将对先验知识依赖较大的算法，进行比较大程度的提升，不仅可以使得算法效果提升还可以简化系统设计的时间。而光流计算就是此类型之一，因此本次毕设有很多东西可以在这篇文章中学习。

1.3 研究内容和技术路线

**1.3.1** **研究内容**

本次毕业设计主要面向光流提取方法中的目标检测与追踪应用问题，拟选择的应用场景为在轨卫星，面对的对象为所拍摄的视频，但是卫星数据相当的珍贵与稀缺，因此准备选用以无人机拍摄为主的数据集，作为主要的对象。针对这次毕设，可以将研究分为三个阶段，首先利用现有的光流计算方法完成对视频中对象的目标检测与追踪，接着根据光流计算方法的先验参数建立一个评价体系，然后利用遗传算法完成对参数的自适应设定，最后引入强化学习的方法，优化遗传算法的结构，使方法更具有效率。

**1.3.2** **研究方法**

第一个研究目标是完成对视频中对象的检测与跟踪，即先将取得的视频对象进行逐帧处理，先针对第一帧进行处理，圈出要追踪的目标区域；然后进行该图片预处理，找寻图片中的角点，再完成光流计算，找到在下一帧中该角点的所在位置，标注出初始所画区域的移动方式。拟采用的方法主要为哈尔角点检测法，L-K稀疏光流法，图像金字塔方法。

接下来要完成的内容是遗传算法的应用，首先要对光流算法的先验参数进行分析与比较，确定参数的数量和取值范围，然后确定评价标准作为适应度的参考，即需要对光流算法的参数进行选择与筛选。拟采用的方法就是一般遗传算法的流程，产生后代，优胜劣汰，交叉重组，变异；随机生成一组参数，然后构建一个适应度函数来作为更新改进参数的标准，在一定的进化次数内，不断的重组变异，适者生存，在最后的种群找到一个最优的个体作为最优解。

之后的研究将面向强化学习，需要将强化学习的算法利用到遗传算法的进化中来，自适应得寻找最优解。拟采用基于马尔科夫决策过程的数学框架和贪婪算法的思想来实现强化学习，即传统的Q-learning算法；因为遗传算法中的种群数量很大，在运行时的计算量很大，会影响一定的速度，其次，每次实验遗传算法都是从头开始，不能将优秀的模式学习下来，所以准备使用强化学习来实现RLGA方法，优化提升算法的效果。此外，在顺利的实现上述内容后，再针对该流程进行实验分析，即选取合适的数据集，来对算法的性能与效果进行分析，然后与其他方法，比如进化策略，深度强化学习等进行对比。

**1.3.3** **预期结果**

预期的结果是完成基于强化学习的遗传算法模型，即构建出的RLGA算法结构，集合遗传算法和强化学习的优点，通过数据集完成对Q表的更新与训练，使得能够高效地自适应提取出参数，然后使用选择出来的最优参数，对视频中的角点信息和光流信息进行提取，以此来完成对视频对象中的目标检测与追踪。

2 光流法的目标追踪应用研究

2.1 光流法原理

**2.1.1** **光流与跟踪**

光流通常以视频序列为对象，即将当前帧中的某些点作为目标保存下来，如下图1.1左边的绿色标识点；然后将对象面向下一帧，在该帧的图像中找到先前帧中所保存的点，将两点以矢量的方式连接起来，所得的矢量即为光流，如图1.1中右边的红色实线；该矢量的方向与长度可与称作光流的角度与长度。总结来说，光流就是得到某一像素点或区域在相邻帧的运动位移(△x，△y)。



图1.1 光流示意图

光流可以应用于场景中的物体的运动估计，甚至用于相机相对整个场景的自运动估计，即光流有可能是视频中前景目标运动产生的，也有可能是相机的运动产生的，当然也有可能是两者的同时作用。根据光流对像素点的作用范围，可以将光流分为两类，对图像中所有的像素点监控的称作“稠密光流”，对图像中部分像素点监控的称作“稀疏光流”。在运行速度上来说，由于稀疏光流追踪是某些点的子集，它们往往集中在图像中比较容易跟踪的特定点上，比如角点，如下图1.2所示。



图1.2 FAST角点示意图

稠密光流方法包括Horn-Schunck算法，Dual TV-L方法等，稀疏光流方法一般以Lucas-Kanade方法为主，本文也将使用该方法进行实验。该算法可以应用在稀疏场景中，比如卫星拍摄，或是无人机拍摄的遥感图像中，因为只要给予它围绕每个特征点的小窗口导出的局部信息，就可以运行使用。但是如果运动的幅度过大，像素点的运动有概率运动到局部窗口之外，导致追踪失败，这时基于“金字塔”的L-K算法应运而生，按照细节从低到高的顺序，对图像一步一步的挖掘，使得大幅度的运动也可以被局部窗口捕获。

跟踪是随着场景在顺序图像流中的演变而跟踪某个目标的运动的任务，主要分为两大类：静态场景跟踪和跟踪场景本身。静态场景跟踪是寻常意义上的跟踪，即在图像流中的主体背景保持静止的状态，前景的目标会产生移动，发现并记录这一系列运动的过程；而跟踪场景本身的任务，则是以摄像机的运动估计为主体，常被称作“视觉测距”。

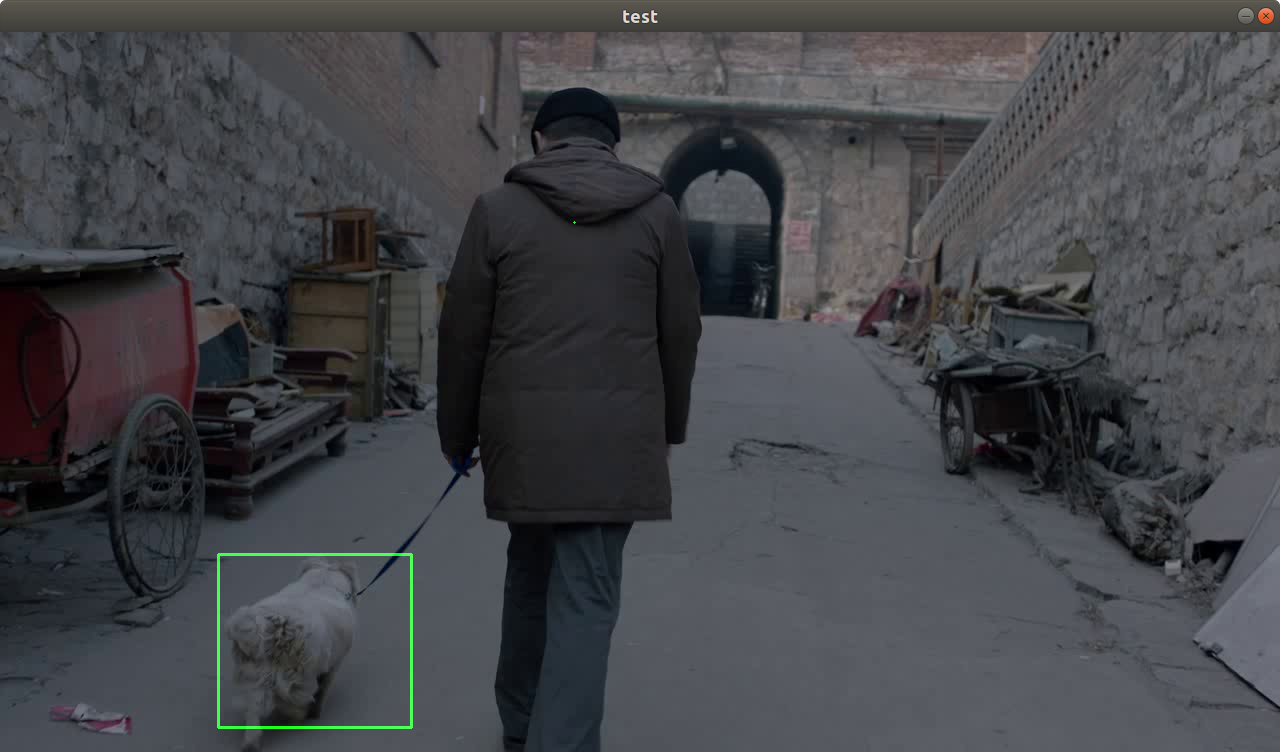


图1.3 光流跟踪示意图

完成跟踪任务的一般步骤为：确定目标，找出角点，找到下一帧中关键点的匹配，如此迭代即可完成目标的跟踪。确定目标的方法不唯一，可以使用一些目标检测算法来计算出目标，也可以通过用户自定义的范围对图像进行操作来完成对目标的提取，如图1.3左侧所示。面对确定好的初始目标，接下来要做的就是检测出目标上的角点，也可以称作特征点，它们一般富含着“有价值”的信息。显然，如果我们在一片草丛中选择一个绿色的点进行追踪，那么在下一帧中再找到这个点就会很困难，所以如果能找到一个独一无二的点，比如草丛上的虫子，那么再次跟踪到这个点的可能性就打打增加。在实际的操作中，选择的特征点应该是唯一的，或几乎是唯一的，并且是可以参数化的，以至于它可以和其他图像中的其他点进行比较，这样的点才能称作有价值。针对这些有价值的点，使用光流方法在下一帧中重新找到这些点，并针对这些重新找到的点再次圈出目标，这就完成了跟踪任务，如图1.3右侧所示。

**2.1.2** **金字塔Lucas-Kanade算法**

LK算法是在一定的空间下，亮度恒定的背景中，对随着时间的迁移而改变的图像上某些点的微小移动进行跟踪的一种算法，该算法需要满足以上的条件，即亮度恒定，时间持续性，空间一致性。

亮度恒定是指在所圈定的场景中，目标的图像的像素点在移动前后没有太大的变化，比如说对于灰度图像来说，代表着像素点的灰度不会随着帧的跟踪而改变。时间持续性是指时间相对于图像运动的尺度来说足够快，这样运动随时间就会变化缓慢，运动幅度也不会太大。空间一致性要求目标区域内属于相同表面的相邻点要具有相似的运动。

对于大多数的视频来说，由于拍摄的原因，经常会有大幅度的、非相干运动出现，如果简单使用L-K算法，就需要一个大的窗口来捕捉这些不规则的运动，但是这通常会打破一致性的运动要求。所以为了解决这个问题，我们可以使用图像金字塔方法，首先跟踪较大的空间尺度，然后通过沿着图像金字塔的水平工作来完善初始运动速度假设，直到跟踪到原始图像像素。即先通过L-K法解决图像顶层问题，然后将最终的运动估计作为下一层的起始点，如此迭代，直到达到最底层。这样就可以在尽可能减少违反运动假设的情况下，尽可能追踪更快更长的运动，这样的方法就是“金字塔L-K光流法”。

2.2 基于光流法的目标追踪系统设计

**2.2.1** **核心函数介绍**

1. goodFeaturesToTrack()

该函数是封装在OpenCV库中，使用了Shi和Tomasi的方法来寻找哈尔角点，便于计算必要的导数运算符，对其进行分析，并返回符合我们定义的适合跟踪的点的列表。

void cv::goodFeaturesToTrack(

cv::InputArray image,

cv::OutputArray corners,

int maxCorners,

double qualityLevel,

double minDistance,

cv::InputArray mask = noArray(),

int blockSize = 3,

)

输入图像可以是任意8位或者32位的单通道图像，也就是说如果是彩色图片需要将其转化成灰度图片后再使用。输出的角点是将包含所有找到的角点的向量或者数组。当选取角点时有区域位置限制时，可以使用mask参数，生成一个与原图像尺寸相同的全零矩阵，然后将想要的区域设置成为非零，这样角点就不会在mask为零的地方生成。blockSize参数表示计算角点时需要考虑的区域大小，默认值为3，如果面对的对象是高分辨率的，可以适当提升。

1. calcOpticalFlowPyrLK()

该函数将图像金字塔与LK稀疏光流提取算法结合的方法，实现了给定特征点的光流信息提取，封装于OpenCV中,利用“良好的跟踪特征”，并返回每个点的跟踪情况。

void cv::calcOpticalFlowPyrLK(

cv::InputArray prevImg,

cv::InputArray nextImg,

cv::InputArray prevPts,

cv::InputOutputArray nextPts,

cv::OutputArray status,

cv::OutputArray err,

cv::Size winSize = Size(15,15),

int maxLevel = 3,

cv::TermCriteria criteria = TermCriteria(

cv::TermCriteria::COUNT | cv::TermCriteria::EPS,

30,

0.01

)

int flags = 0,

double minEigThreshold = 1e-4

)

将前一帧的图像，当前帧图像以及前一帧图像的角点作为输入，可以得到角点在当前帧图像中的位置，也就是新角点的信息，以及每个对应角点追踪是否成功的信息。prevImg和nextImg是初始图像和最终图像，它们的通道和大小应该相同，接下来的prevPts和nextPts分别是初始图像的特征点集合和最终图像中的匹配点集合。数组status和err中含有匹配是否成功的相关信息，status记录是否找到，err表示找到点后的错误度量。

**2.2.2** **系统模块**

1.角点提取

判断该帧图像是否为第一帧，如果是的话，提取该帧图像对应的mask信息，即所选区域信息，然后将图像转成灰度图，调用goodFeaturesToTrack()函数来找到当前区域内所有角点的信息；如果不是第一帧，那么读取上一帧追踪成功的角点信息，作为当前帧的角点。

2.稀疏光流提取

首先得到前一帧的角点信息，前一帧和当前帧的灰度图像，然后调用calcOpticalFlowPyrLK()函数得到新角点信息和追踪状态；之后进行筛检，将所有追踪失败的角点进行标记，在新旧角点集合中都去除掉这些点；然后根据旧角点的区域信息，将所有不在该区域内的追踪后角点去除掉，最终得到有效的新角点集合。通过该集合，可以进行目标区域的重新圈定，也可以作为下一轮迭代的基础。

3.系统流程图

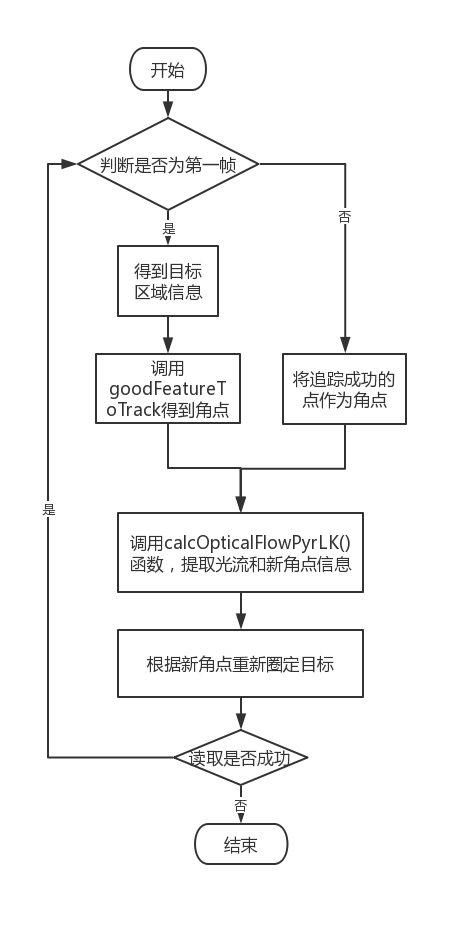


图2.4 光流跟踪模块流程图

**2.2.3** **效果展示**

初始图像的目标区域圈定和开始的角点示意图：

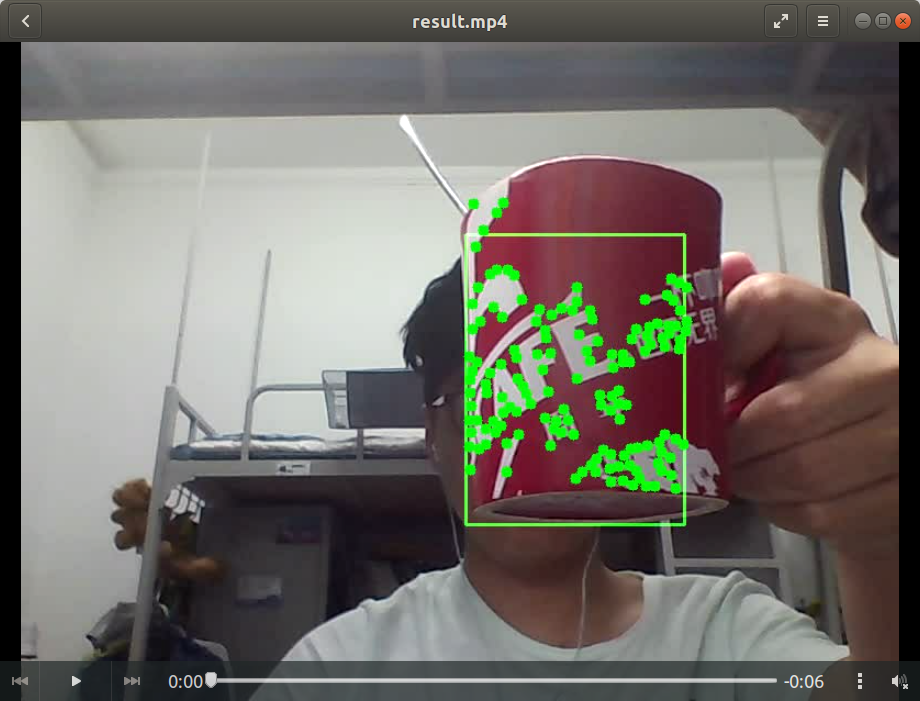


图2.5 开始目标圈定示意图

40帧追踪情况：

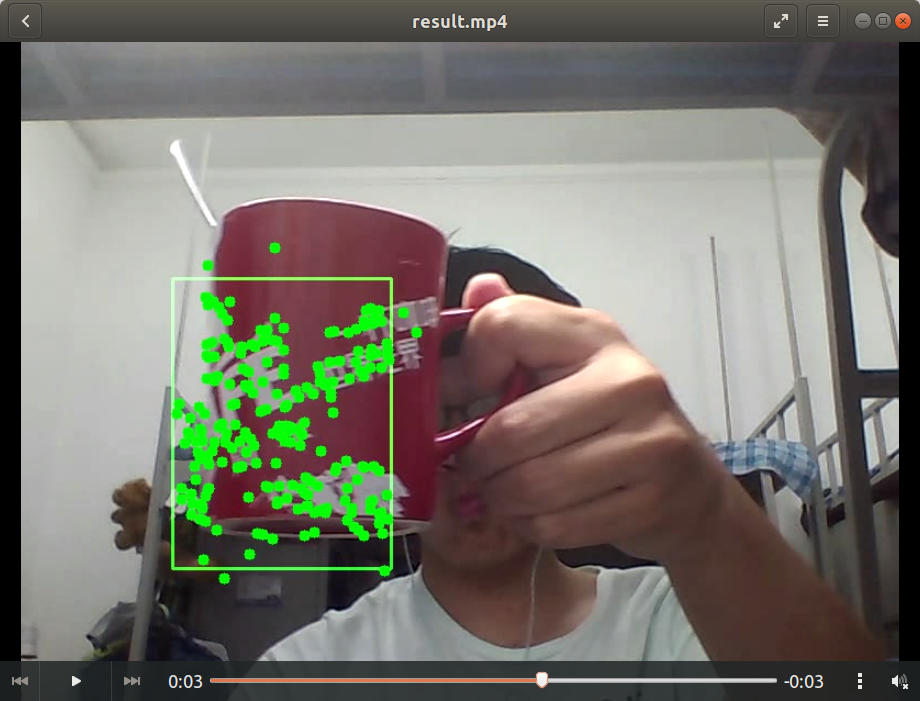


图2.6 中间追踪过程示意图

最终（80帧）追踪情况：

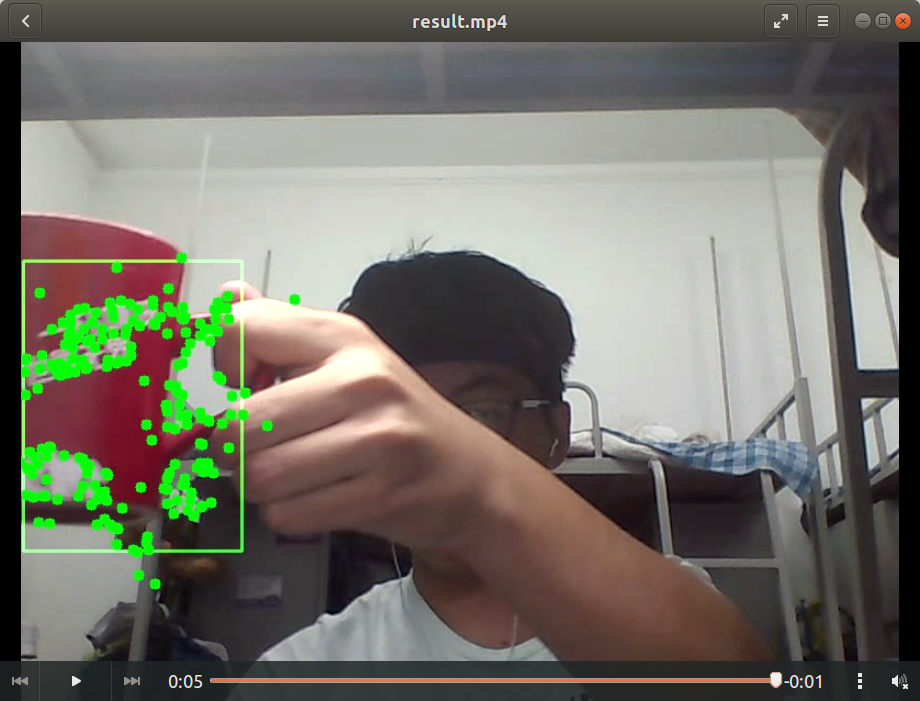


图2.7 最终追踪情况示意图

2.3 光流法的先验参数与评价标准研究

**2.2.1** **先验参数介绍**

本次实验所选择的先验参数是来自goodFeaturesToTrack函数的qualitylevel，minDistance和来自calcOpticalFlowPyrLK函数的winSize，maxLevel以及TermCriteria::COUNT，TermCriteria::EPS。

其中qualitylevel可以决定角点的返回质量，通常介于0.01和0.1之间，绝对不能超过1.0，否则会使得坏特征点增多，导致角点质量的急剧下降；minDistance参数是决定相邻角点之间的最小间距，可以通过调整该参数来决定角点提取的密度。

winSize是size类型的参数，它决定了计算局部相干运动时的窗口大小。maxlevel参数是决定图像金字塔层数的参数，它通过调整金字塔的层数来获得图像堆栈深度的不同信息。最后两个参数是来自与criteria类，它们共同决定算法结束搜索匹配的时间，其中COUNT是代表最大迭代次数，EPS代表每次迭代的最小变化值，次数超过COUNT或者每次的变化不满足EPS的标准都会使搜索停止。

通过改变以上的参数，会使每次追踪的情况都不一样，比如改变角点检测的参数会影响角点的生成质量和数量，改变L-K算法的参数，会使追踪的准确率产生变化，这些参数尤其对在大背景下的小目标运动影响较大，因此将这些参数作为后续实验的参考目标。

**2.2.2** **评价标准研究**

1.MSE：

因为在确定区域后，每次的跟踪的最终结果都会是以块的形式呈现，那么使用MSE来比较两块区域图像的相似度就变得很合理。MSE是指均方误差，在数理统计中均方误差是指参数估计值与参数真值之差平方的期望值，而在图像领域，它可以用来评价当前帧图像与参考图像中某一块的相似程度。MSE数值越小，代表越稳定，即图像的相似度越高。



其中为带匹配块的内部像素点坐标，表示块的相对位移，分别表示当前帧和参考图像的灰度值，M，N分别为匹配块的长和宽，用来决定匹配块的尺寸。

2.founded\_amount:

在每次光流提取后都会对新的角点信息进行筛选，由于在目标追踪的过程要面对很多很多帧的图像，也就是说会进行很多次的光流提取，同理筛选也会进行多次，那么在这么严苛的条件下依然存在的特征点，即为优秀的特征点，那么这些点的数量就被当作了另一个评价标准founded\_amount，数量越多，追踪的成功率越高。

2.4 本章小结

本章完成了使用光流方法完成目标追踪的任务，实现了最终实验中使用的稀疏光流提取部分，使用哈尔角点检测和金字塔L-K算法将视频对象中的特征点和光流信息提取出来，然后基于这些内容完成对目标的追踪。在这之后，分析与实验有关的先验参数，选取了来自两个函数的六个不同参数，分别确认了其取值范围，为后面实验做准备。找到了MSE评价标准，并联合跟踪时的实际情况，将追踪成功的目标点的数量与MSE同时作为评价指标。

3 遗传算法自适应方法设计

3.1 遗传算法的原理

**3.1.1** **遗传算法的基本含义**

遗传算法是使用代码的方式，基于达尔文生物进化理论的自然选择原理和生物学中的遗传进化模型，在面对某些问题时，通过模拟自然界适者生存，不适者被淘汰的生存法则，来不断寻找最优解的方法。

遗传算法的主要特点是能够直接通过选择来达到某种问题所需要的效果，不需要使用求导或者其它复杂的分析过程。通过基于概率的选择方法，在环境不明确，法则不确定的情况下，依然可以通过自身的迭代来自适应地调整，优化搜索空间，以此来找到最优的解。

根据生物学原理，每个人都会有他独一无二的遗传信息，可以以DNA为例，每个人的DNA的都是不同的，各自有各自的特点，有血缘关系的人某些部分是相近的，没有血缘关系的人也有可能有部分相似的可能，但即使有些许的不同也可以让他们的表现型有很大差异。

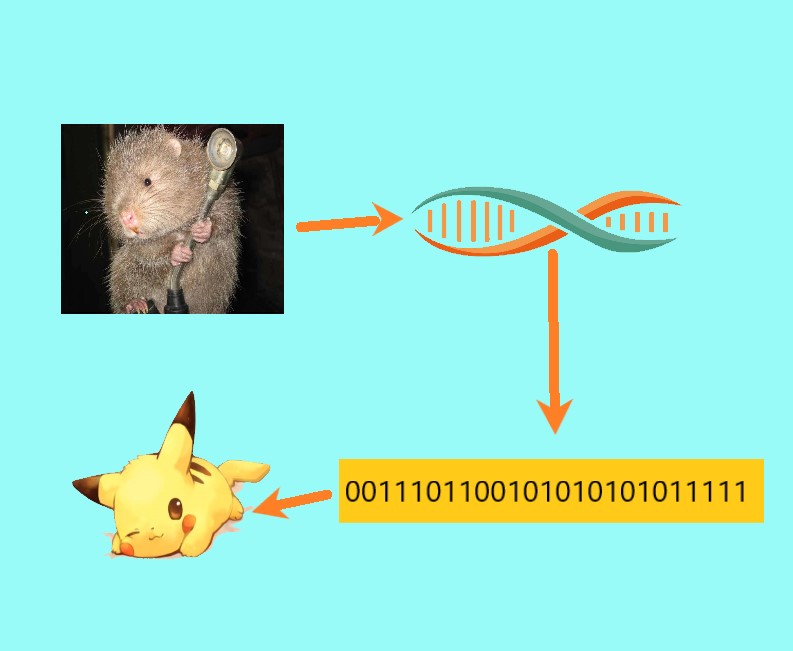


图3.1 DNA在电脑里的转化

如果用遗传算法来模拟，那么这些个体的遗传信息就要进行转化，如上图3.1所示，将信息转变成电脑能读懂的语言，比如二进制编码，再在电脑上复现个体的性状。用0和1来模拟DNA，因为在电脑的世界中，所有的东西都是由0和1构成的，所以可以以0和1为基准将这类信息准确的表示出来，就是遗传算法的基准。

现实中生物的种群会不断的繁衍，进化，在这个过程中又会发生变异，淘汰的事件，在电脑有了自己能理解的DNA形式后，继续模拟上述的过程也就变得不再困难。

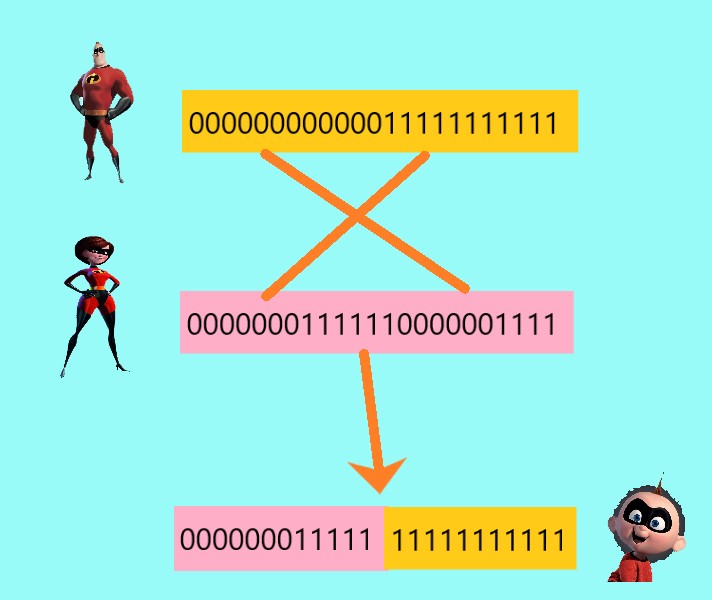


图3.2 交叉重组示意图

如上图3.2所示，在将父亲母亲的DNA以二进制的语言表示之后，将父亲的DNA拿出一半，母亲的DNA拿出一半，重新组合成一组新的DNA，繁衍出了一个新的小宝宝，这就在计算机中完成了对生物交叉重组的模拟。同理如果将某个位置的0置换成1，或者1替换成0，就可以模拟出生物的变异。有了交叉重组和变异，可以说就已经完成了生物学上的遗传，并且在基于这些信息的基础上，依据进化的理论，就可以完整的模拟出来生物圈中种群的进化，也就完成了遗传算法。

**3.1.2** **遗传算法的一般流程**

遗传算法通常分为五大部分，如下图所示。

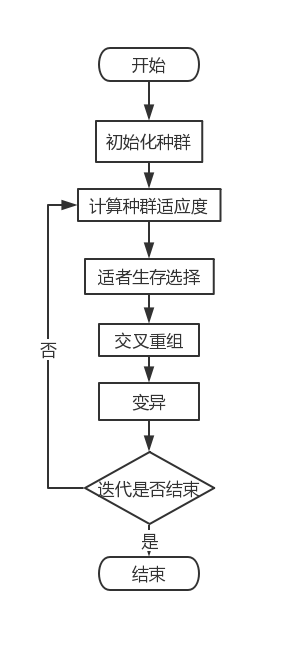


图3.3 遗传算法一般步骤

首先要对种群进行初始化，将所需要的参数DNA化，即需要将问题的已知条件变成DNA，然后每个个体都拥有着这样的DNA，但每个个体又是独一无二的；再批量产生这些的个体，以设定好的DNA格式为标准，通过随机化的方法为每个个体的DNA赋值，得到一个初代种群。

每个问题都有评价标准，过渡到进化问题就是个体的适应度，针对不同的问题得到不同适应度计算函数，这就是第二个步骤计算种群的适应度，适应度高低决定了个体的生存概率；接下来进行自然选择步骤，将适应度低的个体淘汰，保留适应度高的优良个体；然后对选择后的种群进行繁衍，个体之间的交叉重组，产生子代，然后每个个体又可以变异，这就是最后的两个步骤。形成的新的种群，根据迭代是否终止的条件可以决定是进行下一轮的进化，还是成为最终的优秀种群。

3.2 基于遗传算法的参数自适应实验

**3.2.1** **在自适应光流提取上的应用原理**

将遗传算法应用在光流问题上，也就是需要将光流问题构造成一个自然环境，然后模拟在这个环境里面个体进行自然选择的过程。对应遗传算法的一般流程，可以通过以下的对应方式，完成算法在光流问题上的应用。

首先确定DNA的形式，将本文第二章所确定的光流问题的先验参数来模拟每个个体DNA的形式，即DNA的长度为6，然后每个DNA都有其独特的取值范围。针对不同的取值范围，为DNA进行随机化赋值，重复进行，以此生成不同个体的种群。

将光流问题中的评价标准，作为适应度，也就是说用光流提取的过程来模拟适应度函数。自然选择的操作与一般遗传算法大体相同，只是在变异阶段时，DNA中每个基因的范围都是不一样的，因此要对各自的范围进行变异操作。

**3.2.2** **系统模块**

1.确定参数：

遗传算法本身也有自己的参数，比如种群的大小，变异和重组的强度以及生物进化的代数，尤其是变异和重组的强度可以对算法的效果产生很大的影响，本次实验的参数设置如下。

表3.1 遗传算法参数设置

|  |  |
| --- | --- |
| 参数名称 | 参数值 |
| POP\_SIZE | 100 |
| CROSS\_RATE | 0.8 |
| MUTATION\_RATE | 0.1 |
| GENERATIONS | 500 |

如上表3.1所示，POP\_SIZE代表种群的总大小，即个体的总数量；CROSS\_RATE代表交叉重组的概率，MUTATION\_RATE代表变异的概率；GENERATIONS代表总共的迭代次数。

2.产生种群

表3.2 光流先验参数范围

|  |  |
| --- | --- |
| 参数名称 | 参数范围 |
| qualitylevel | [0.01,0.1] |
| minDistance | [0,100] |
| winSize | [3,100] |
| maxLevel | [0,100] |
| TermCriteria::COUNT | [0,100] |
| TermCriteria::EPS | [0,1] |

针对表3.2中每个先验参数不同的范围，随机产生一个DNA，然后根据所给定的种群规模的参数值POP\_SIZE，产生相对应数量的值，最终得到(POP\_SIZE,DNA\_SIZE) 形式的矩阵，该矩阵作为初始的种群。具体的实现流程如下：

*create\_population:*

*1.create a mod for DNA;*

*dna\_mod = np.empty((POP\_SIZE,DNA\_SIZE))*

*2.make values for each person’s DNA in their own range*

*dna\_mod[i][j] = np.random.rand()\*M*

*i-index of population j-index of DNA*

*M-multiples to make an original range*

3.优胜劣汰

通过适应度计算函数，得到种群中所有个体的适应度，然后通过将每个个体适应度与适应度的总和作比较，将这个结果作为概率来进行有放回的随机抽取，进行种群数量次，最终所有抽取出来的个体组成新种群。

本部分实验的适应度是从光流提取功能模块中得到的，首先需要将DNA分配到光流参数的对应位置，组成两个光流函数所需要的参数形式；然后调用函数，对指定帧数区间的图片进行分析提取，得到的评价标准作为适应度返回，具体实现如下：

*get\_fitness:*

*1.make a mod to store results :*

*fitness = np.empty(POP\_SIZE)*

*2.translate the DNA into params:*

*get feature\_params and lk\_params*

*3.get MSE result as fitness*

*fitness[i] = -get\_MSE(feature\_params,lk\_params)*

4.交叉重组

在种群中对每个个体进行判断，如果该个体满足交叉重组的条件，即产生的随机数小于交叉充足的概率，发生了交叉重组的事件，保留该个体，作为“父亲”；再从种群中随机选择一个除了该个体以外的个体，作为母亲；然后对DNA进行二值随机选择，DNA对应位置为1的部分保留父亲对应的基因，为0的部分则是保留母亲对应位置的基因，具体实现如下：

*crossover:*

*1.judge the person whether crossovers or not*

*if np.random.rand() < CROSS\_RATE*

*2. if crossover happens,find another person to make crossover*

*i\_ = np.random.randint(0,POP\_SIZE,1)*

*choice=np.random.randint(0,2,size=DNA\_SIZE).astype(np.bool)*

*mom = pop[i\_].flatten()*

*parent[choice] = mom[choice]*

5.基因变异

对个体中的每个基因进行判断，如果该基因满足变异的条件，即产生的随机数小于个体变异的概率，发生了变异的事件，那么就根据该基因的所在范围，随机的分配给其一个新的数值，产生的新个体代替原来的个体，具体实现如下：

*mutate:*

*1.judge each DNA whether mutates or not:*

*if np.random.rand()<MUTATION\_RATE*

*2.if mutate happens,each DNA mutate in its own way*

*dna[i] = np.random.rand()\*M*

*i-index of DNA M-multiples to make an original range*

**3.2.3** **实验结果**

将种群的适应度表示出来，作为实验结果，因为选取的评价标准是MSE，数值越小效果越好，所以在得到MSE的值后，将其取负值来表示适应度。因为会有追踪失败的情况发生，这时的MSE的值设成-10000。

初始种群的适应度：

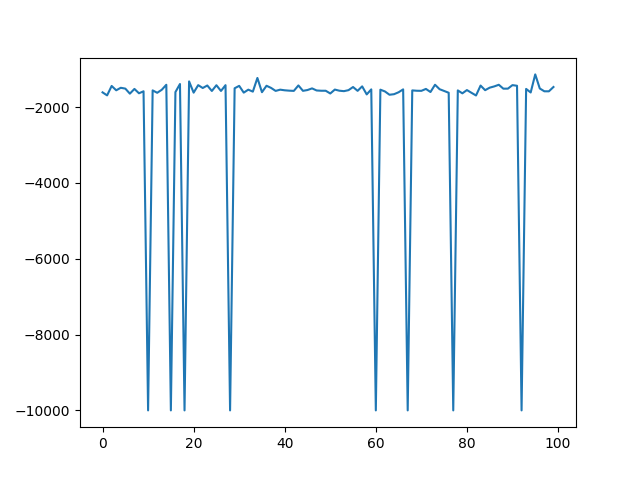


图3.4 初始种群的MSE适应度

从上图3.4可以看出，产生的初始种群适应度普遍一般，甚至还有-10000的情况，即追踪完全失败的情况发生。

第十代：

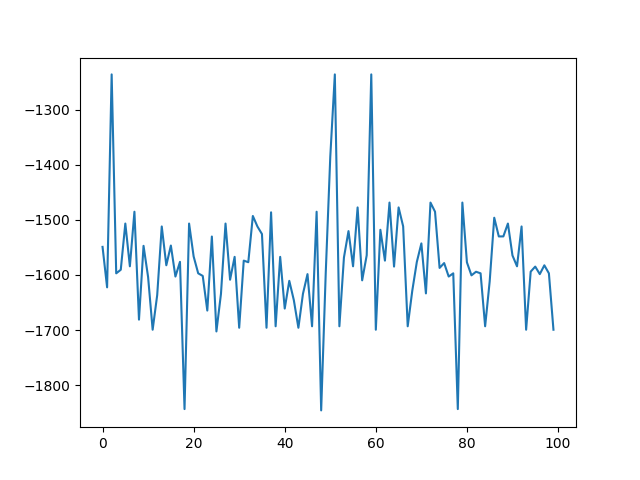


图3.5 第十代种群适应度情况

从上图3.5可以看出，种群的质量大幅度提高，不再有-10000的情况出现，最低的适应度仍然比初始种群中的最高适应度大，因此可以明显看出种群的进化。

第三十代：

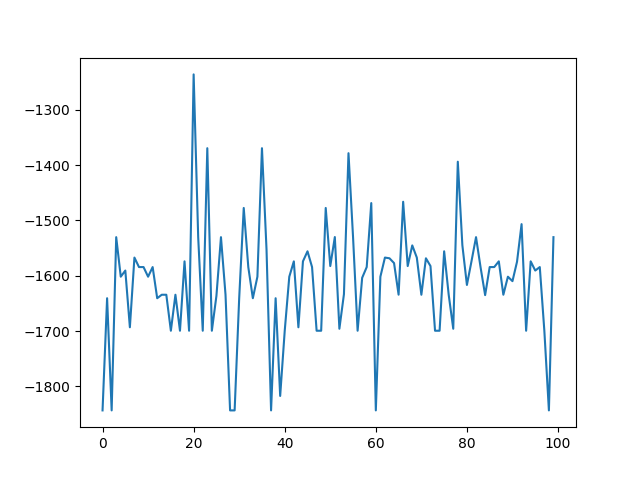


图3.6 第三十代种群适应度展示

第五十代：

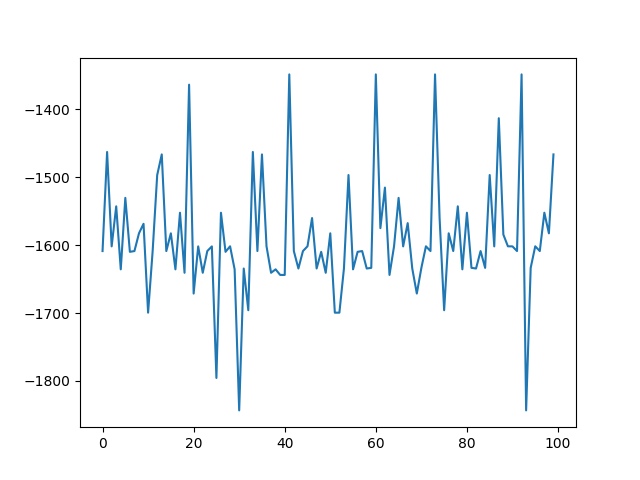


图3.7 第五十代种群适应度展示

由图3.6和图3.7所展示的，种群中“优良个体”的数量在不断的增加，图像的整体趋势在逐渐趋于平稳的同时也在稳步的向上提升，同时有几个极其优秀的个体存在，可以取他们当中适应度最大的所对应的DNA，转化成参数集，当做结果以供测试使用。实验只进行了50代的迭代，就使用了700秒的运行时间，可见遗传算法虽然效果不错，但是运行效率却令人不敢恭维。

3.3 基于进化策略的实验与比较

**3.3.1** **进化策略与遗传算法的异同**

进化策略和遗传算法共享着一些东西，它们都用DNA的形式作为遗传信息的载体，并基于此进行一代代的进化与革新，通过留下适应者，淘汰不适者来完成自然选择的过程。

而进化策略中不同的是，在变异的过程中不再通过统一的标准进行判断，取而代之的是为每个基因都分配了一个变异强度，在传统的交叉重组后，更频繁发生变异。还有更重要的是进化策略中的自然选择过程不再通过随机选择最优个体的方式进行，而是使用确定化的方法，即通过直接选取适应度最高的前n个个体来组成新的种群。

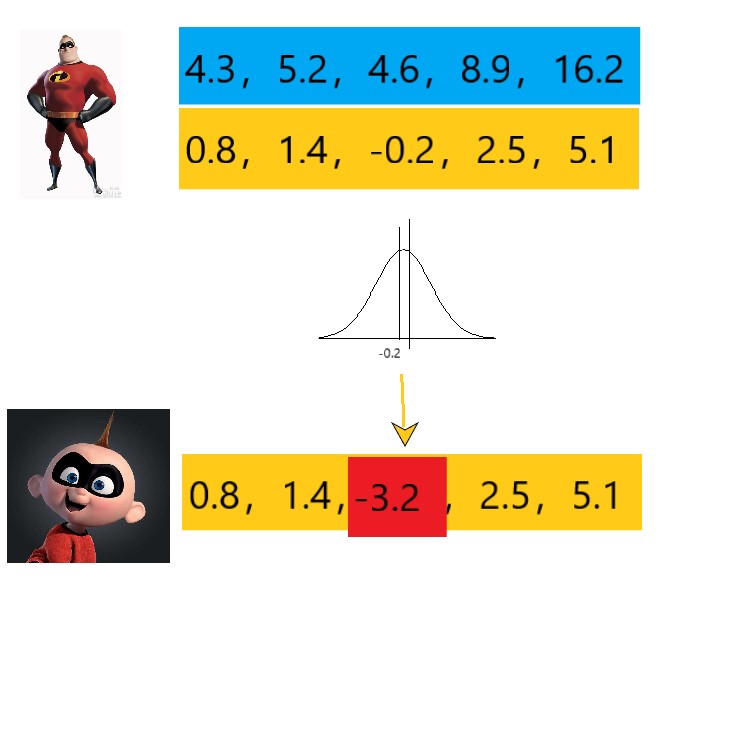


图3.8 进化策略示意图

进化策略的方法其实与正态分布有很大联系，将上一代遗传下来的DNA看做平均值，与DNA对应的变异强度看做标准差，这样就能通过正态分布来产生一个与遗传下来的DNA相近的值了，这样产生的个体的性状就会发生改变，通过这种方式来产生新的一代种群。

进化策略有几种不同的形式，通常是以父代和子代的数量区分的，比如有(1+1)-ES方法，(μ/ρ+λ)-ES。(1+1)-ES即表示只考虑一个父代，然后只生成一个子代，在父代与子代中选取最好的一个个体作为下一轮迭代的父代，以这种方式进行适者生存。这种方式的迭代速度很快，但是不一定能够找到最优值，因为个体只有两个，没有太大的“远见”，很容易在局部最优上停止；而(μ/ρ+λ)-ES方法是指，在数量为μ的种群中，选取ρ个个体作为父代，生成λ数量的子代个体，然后将所有的个体依据适应度排在一起，取前μ个作为新的种群，开始新一轮迭代。这种方式会比(1+1)-ES方法更容易达到问题的最优解，有远见，但是如果种群的数量过大，则运行的效率也会降低，与遗传算法相比，则有着更容易达到最优解的优点。

**3.3.2** **系统模块**

1.产生种群：

在传统产生种群的基础上，在每个个体的DNA都被分配一个属于其自 己的变异强度，通过字典存储这两个信息。

*create\_population:*

*1.create DNA like Genetic Algorithm*

*2.create mutation strength for each DNA and store in dictionary:*

*pop = dict(DNA = dna\_mod,*

*Mut\_strength = np.random.rand(POP\_SIZE,DNA\_SIZE)\*3)*

2.生成子代:

每次随机找到两个父代，然后进行交叉重组，在这个过程中父代的DNA 和变异强度都会遗传给子代；对新产生的子代，对每个DNA根据变异强度 进行变异，也就是在正态分布抽取一个样本，作为新的DNA，这样子代就 产生了，将父代和子代结合到一起，成为目前的新种群。

*make\_kids:*

*1.randomly choose two people as parents,then crossover to make kids*

*2.let the mut\_strength mutate*

*strength[:] = np.maximun(strength+*

*(np.random.rand(\*strength.shape)-0.5),0)*

*3.mutate according to DNA and mut\_strength:*

*dna += strength \* np.random.randn(\*dna.shape)*

*4.turn DNA into their own bound*

*dna[i] = np.clip(dna[i],\*DNA\_BOUND[i])*

*i-the index of DNA*

3.优胜劣汰:

对种群计算适应度，针对适应度由大到小排序，取前原种群大小数量的 个体作为新的种群，作为下一代进化的基础。

*kill\_bad:*

*1.calculate the population’s fitness*

*2.sort the population by their fitness, and keep top POP\_SIZE*

*good\_idx = idx[fitness.argsort()][-POP\_SIZE]*

*idx - the index of old population*

*pop = pop[good\_idx]*

**3.3.3** **实验结果**

每次迭代的结果都是一个递增的曲线，是因为在kill\_bad的模块里面将种群中所有的个体按照适应度高低顺序排列好了，因此判断种群是否进化的标准就可以以曲线的上下界数值为依据。

第一次和第十次迭代后的种群：

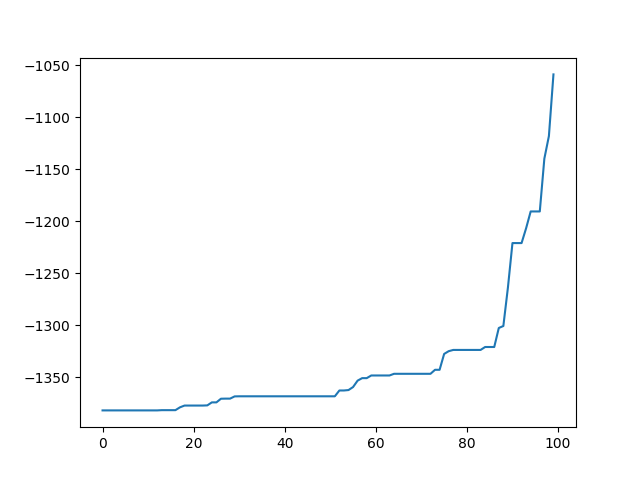
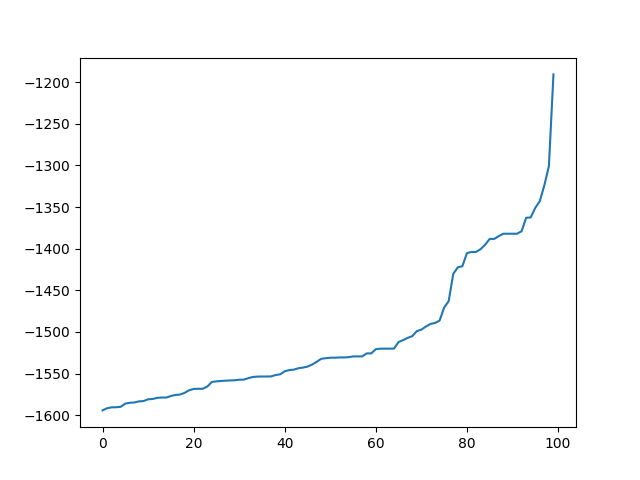


图3.9 第一次和第十代迭代种群适应度

从上图3.9中可以看出，右图种群的适应度与左图的相比有很明显的提升，第一代中的最优在第十代中已经处于靠近末端的位置，显而易见，种群的质量在初始阶段有很大幅度的提升。

第三十、五十代结果：

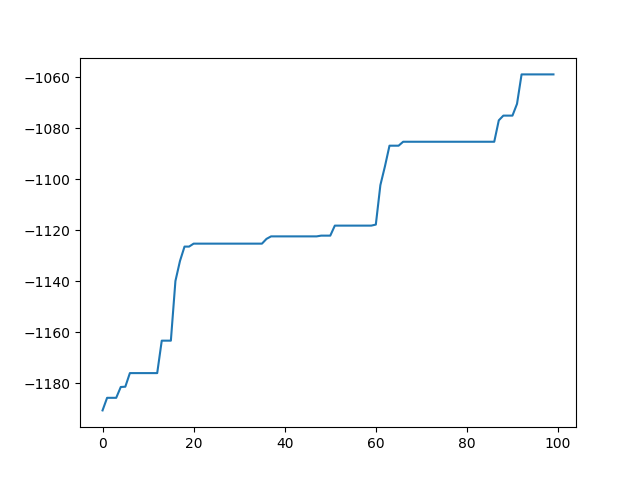
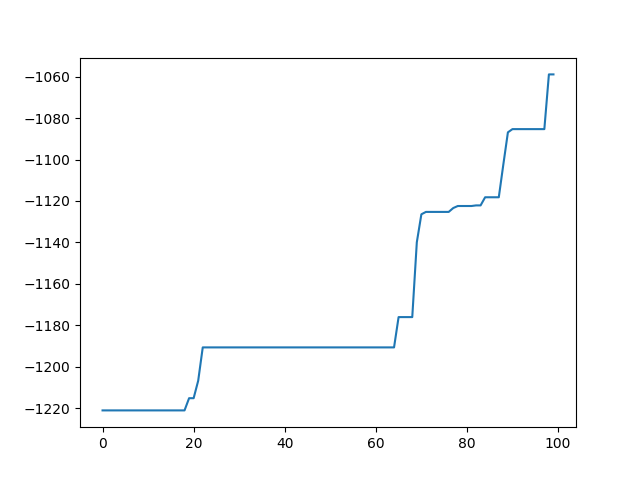


图3.11 第三十次、五十次迭代种群适应度

从上图3.11可以看出，随着代数的增加，种群的整体质量在不断提高，而且高适应度种群的数量也在稳步抬升，可以在高适应度种群中提取出一些个体，将其DNA作为优质参数提供给追踪模块，完成自适应提取任务。

使用进化策略方法的实验与遗传算法相比，有了更多的优质个体，种群的质量从第一次迭代开始就一直保持着高质量，不会有突然的坏点，或者个体质量波动的情况，这点比遗传算法更加稳定；而且进化策略方法对种群适应度的提升程度也是比遗传算法大的，最终代的下界和上界都有提升，平均提升两百左右的数值。但是进化策略由于产生了更多的个体，导致适应度计算时调用光流提取函数的次数增加，使得运行的速度下降，进行五十代迭代需要1453秒的时间，比遗传算法更加费时。所以进化策略虽然可以大幅提高种群的质量，但是运行的效率下降，并不是自适应提取的最佳算法选择。

3.4 本章小结

本章以遗传算法为核心，设计了遗传算法在自适应光流提取问题上的应用方法。以确定好的先验参数和评价标准为基准，在构建完成的遗传算法的架构下，进行产生种群，适者生存，交叉重组，基因变异的迭代过程，不断生成、筛选参数，提高参数的质量，最终完成了光流的自适应提取。之后选用与遗传算法相近的进化策略方式，通过产生子代，最适生存的过程，也完成了自适应提取。

对比两次实验的结果发现：两种方式虽然都能完成实验，但是进化策略比遗传算法的提升更加明显，不过两种方式的运行时间很长，进化策略的运行时间是大约遗传算法的两倍，因此虽然进化策略提升了自适应效果，但是却丧失了效率，不能算得上是优化，这就需要一个更好的方式来提升遗传算法。

、

4 强化学习优化遗传算法的研究

4.1 强化学习方法介绍及相关机理

**4.4.1** **强化学习**

强化学习是机器学习中的一个重要部分，它是一种在特定环境中令计算机不断尝试，通过得到的反馈来进行学习训练的方法，在这个过程中没有指导，没有提示，只是让计算机自我尝试，然后对尝试的结果打分，计算机会将高打分的行为当作“好”的行为，即奖励；同时将低分或者负分的行为当作“不好”的行为，即惩罚。在不断的尝试中积累经验，趋利避害，最终得到一个“成熟”的计算机，这就是强化学习的作用。

强化学习的理论基础是马尔可夫决策过程（MDP），一个马尔可夫决策过程由一个四元组构成 [1] R.Sutton et al. Reinforcement learning: An introduction , 1998

:表示所有状态的集合，对，表示第i步的状态。

:表示动作的集合，对a∈A，表示第i步的动作。

: 表示当前状态转移到下一状态的概率，表示的是在当前状态下，选择了动作集合中某个的，然后达到其他状态的概率。

，是回报函数。在状态下，选择的动作会得到一个回报。

MDP的决策过程如下：选取一个智能体，设定它的初始状态为，在动作集合  中挑选一个动作执行，智能体以概率转移到下一个状态，把该状态设为。然后重复以上的过程，就可以完成状态的不断转移，直到达到目的，具体的过程由下图4.1所示。

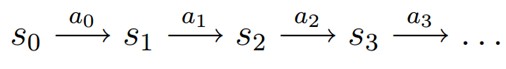


图4.1 *MDP*状态转移示意图

因为每一步转移都会产生回报，而回报r是由状态s和动作a共同得到的，也就是说在状态转移图的基础上可以加入回报机制，如下图4.2所示，使得MDP更加合理。



图4.2 带有回报机制的*MDP*示意图

基于马尔可夫模型，强化学习继承了很多的环节，比如奖励、行动等的定义，类比MDP，可以将强化学习流程以下图的方式概括。

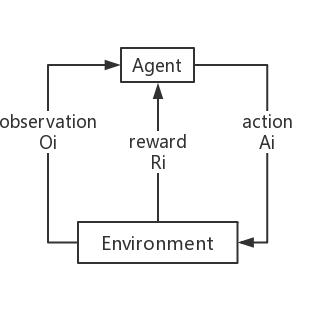


图4.3 强化学习流程示意图

如上图4.3所示，一个Agent在某个位置时根据某些标准做出抉择，选出一个action 实施，到达一个新的状态observation，这个过程会对环境造成影响，环境将这个影响以reward的形式反馈给agent。根据reward的正负，Agent会调整自己选择时的策略，向可以获得更多的奖励方向发展，变得更加“聪慧”，这就是一次强化学习的广义流程。

在上述流程中，Agent在做出选择时根据的策略是强化学习的核心，有它才能有效地确定行为。在一些情况下，策略可以时简单的函数或者查找表，而在另外一些情况下，它也可能涉及到广泛的计算，比如说一个搜索过程。根据策略的不同可以将强化学习进行分类，但分类的标准不止这一种。

如果将Agent能否理解环境作为标准，可以将强化学习分为不理解环境方法和理解环境的方法，分别将其称作Model-free和Model-based。具体如下表所示。

表4.1 基于能否理解环境的分类

|  |  |  |
| --- | --- | --- |
|  | Model-free | Model-based |
| 含义 | 对环境没有事先的了解，所作的行动都是基于环境的反馈，一步一步地从中学习 | 对环境有事先的了解，能够使用一个模型来模拟环境，会在模拟环境的行动进行预测 |
| 方法 | Q learning，Sarsa，Policy Gradients | Q learning，Sarsa，Policy Gradients |

因为能理解环境的方法只是多出对环境事先分析的部分，其余部分依然是需要从环境中得到反馈，继而学习，所以两种分类的常用方法是相同的。除了这种方法，还有根据是基于概率的(Policy-Based)还是基于价值的(Value-Based)分类。

表4.2 基于概率和价值的分类

|  |  |  |
| --- | --- | --- |
|  | Policy-Based | Value-Based |
| 含义 | 对周围的环境进行分析，得到做出各种动作的概率，然后基于概率选择，每种动作都有可能被选中。 | 与环境交互后，得到的是每种动作对应的价值，根据最高价值来选择动作。 |
| 方法 | Policy Gradients | Q learning，Sarsa |

强化学习还有其他的分类方式，基于回合更新还是单步更新标准，回合更新就是要等待一个完整的过程结束后，然后根据这一回合中的转折来更新选择策略；而单步更新就是在整个的过程中，每一步都会产生反馈，都在更新，不用等待结束，可以边实行边学习。但是对比之下，单步更新更加具有效率，所以现在大部分主流方法都是基于单步更新的。

最后一种分类方法是将强化学习分为了在线学习(On-Policy)和离线学习(Off-Policy),具体的区别如下表4.3所示。

表4.3 基于在线和离线的分类

|  |  |  |
| --- | --- | --- |
|  | On-Policy | Off-Policy |
| 含义 | 在线学习就是指本体Agent必须参与，只能根据自己边与环境交互边学习的方式来提升自己。 | 离线学习则是除了自己学习外，也可以参考过往的经验进行更新，而且也不用边训练边学习，可以先进行训练，再统一学习。 |
| 方法 | Sarsa | Q learning，Deep-Q-Network |

**4.4.2** **Q learning算法基本原理**

Q learning算法是强化学习中的一种，它是基于Q表的方法，会通过Q表来选取行为来对环境实行，获得反馈，然后依据这个奖励或者惩罚性质的反馈来更新Q表的值，以便下一次的选择更具智慧。Q learning基于价值做选择，这就使得其更加果断，也更容易做出最好的决断，而其离线学习的性质可以让它有继承之前经验的优点，会让训练变得更加有效，因此本次实验将选用这种方法。

表4.4 Q表示意图

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Up | Down | Left | Right |
| S1 | (s1,a1) | (s1,a2) | (s1,a3) | (s1,a4) |
| S2 | (s2,a1) | (s2,a2) | (s2,a3) | (s2,a4) |

如上表4.4所示，Q表的行索引代表可以选择的行为种类，列索引代表所在的状态，表中的内容(s1,a1)是指在s1状态时，选择a1动作到达s2状态的决策的价值。选择Q(si,ai)的值中最大的一个，作为在该状态下所选的工作，不断进行选择，直到终点状态，然后根据得到的反馈回去对Q表进行更新，整个过程如下图4.4所示。

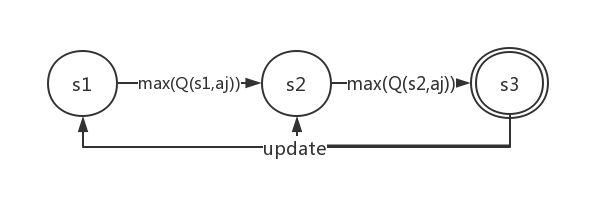


图4.4 Q learning一次学习过程

每次选择后都会得到一个奖励，然后根据这个奖励对Q表进行更新，当然不能直接将其填进Q表进行更新，因为如果这样会造成在选择的时候变成贪婪算法选择，因此要对得到的奖励进行一些操作。标准的Q learning算法如下，里面写明了如何进行更新，也对选择进行了进一步规定。

*Initialize Q(s,a) arbitrarily*

*Repeat(for each episode):*

*Initial s*

*Repeat(for each step of episode):*

*Choose a from s using policy derived from Q(e.g.,Ɛ-greedy)*

*Take action a, observe r, s’*

**

**

*Until s is terminal*

对以上算法中的公式进行说明：在更新时，首先要在下一个状态中选取价值最大的一步，然后乘以衰减值，加上到达时所获得的奖励，所得到的最终结果作为的现实值；然后将本身原来的值作为对这一步的估计值；用现实值减去估计值再乘以学习效率，之后加上原来的值变成新的值，完成更新。这里虽然使用到了，但并未对状态做任何决策，只是对其Q值做了最大的一次估计，将引入状态的计算，使每一步都真正的相互联系到一起。

在公式中有很多参数，它们都拥有着很重要的意义。是使用在选择决策上的一种策略，比如该值为0.9的时候，在动作决策时会有90%的概率按照最大价值来选择，剩余10%概率会使决策随机的选择行为。是学习效率，决定现实与估计的误差值有多少要被学习，它应该小于1；是指对未来奖励值的衰减度，范围[0,1]，对于它的改变可以决定Agent是否具有远见。当=1时，Agent会变得能够“预测未来”，可以看到后面所有行动的价值，这样就会缺乏改变的动力，而未来也不一定是正确的，活在未来的Agent不能称为最好；当=0时，Agent只能看到眼前的利益，就又变成了贪婪算法；所以要适当选值，让机器人既现实又有远见。

4.2 基于RLGA方法自适应光流提取实验

**4.2.1** **遗传算法与Q-learning强化学习**

遗传算法和Q learning强化学习方法都可以做为自适应实验的实现方法，但是其各有长短，针对本次实验，遗传算法可以完成自适应提取，但是每次迭代都涉及种群中所有个体的操作，比如算法每次都要得到所有个体的适应度，而调用适应度计算函数时都会对目标进行一次至少10帧的跟踪，种群中所有个体的时间累加起来，会使运行效率大大降低。

而Q learning强化学习是基于Q表的创建，它虽然可以通过一次又一次的尝试来更新Q表，单步的尝试让调用跟踪函数的次数降低，也就相应的提升了效率。但是针对与本次实验来说，六个先验参数的选择范围都比较广，平均至少有100区段。如果用传统方法构建Q表，会导致Q表的列索引种类特别多，同时假设每一个状态选择的动作作为当时参数的值，那么也就至少需要6步才能走到终点，也就意味着Q表的行索引个数会有6个。最终会有至少600个表格需要填写，因为不可能选择到全部动作，所以要全部更新这些表格的内容几乎是不可能的。

为了更好的解决问题，决定采用Q-learning方法来优化遗传算法，那么不仅可以解决Q表更新困难的问题，也可以将遗传算法的效率问题解决，强化学习和遗传算法也经常被人一起提及，在考量了诸多方法后，最终决定使用RLGA方法来完成本次实验。

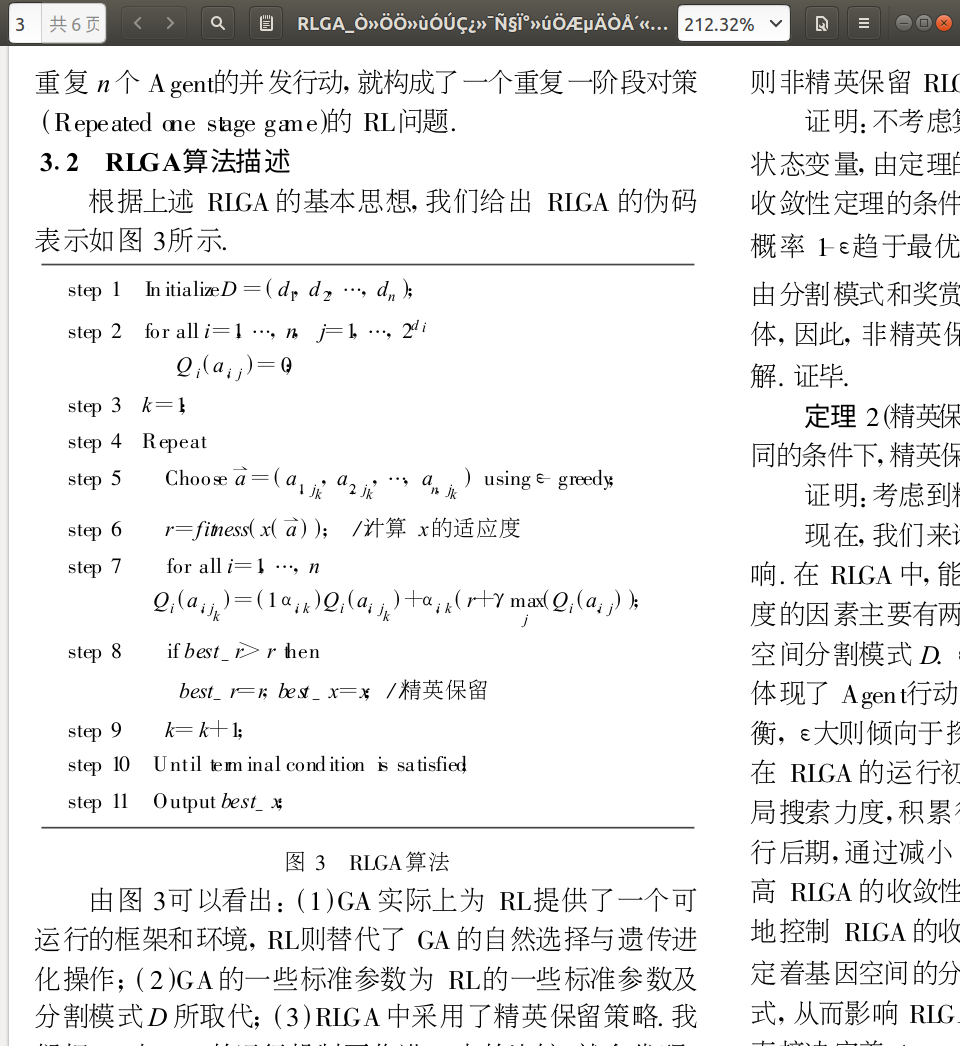
**4.2.2** **RLGA方法原理**

RLGA方法可以概括成以下几个流程：DNA空间分割，多Agent构建，并发行动，联合更新以及最优替代。

首先是基于GA方法中的DNA构建，对构造成功的原始DNA的空间进行分割，将分割后的每个小部分空间都视作强化学习部分中的环境。比如给定的DNA空间按照某种分割模式被分成了n部分，将分割后的部分记为A=（A1，A2，...，An)。

为了对所有分割后的部分都使用到强化学习方法，这就需要不止一个Agent来进行调控，由于使用的方法是Q learning算法，这也就意味着需要构建多个Q表来满足要求。不仅是构建，决策的动作选择，也需要多个Agent共同行动，在各自所属的环境选择出各自最优秀的部分，然后将这些部分按照分割的顺序结合到一起，就形成了一个被选择出来的完整DNA。而这样的多Agent的并发行动，就构成了一个可解的重复一阶对策的强化学习问题。

将重组好的DNA放入适应度计算函数，得到其相应的适应度，然后经过一些符合问题要求的转化，将其变成环境的反馈，也就是转变成奖励或惩罚。将这个反馈平等返回到所有Agent中，各Agent完成对各自Q表的更新，因为分割的空间各异，所以每个Agent逐渐将对自己部分的最优值的价值提高。将RLGA的思想转化为算法，具体的流程如下：

**

在该方法中，GA的参数被RL的参数取代，GA中的自然选择和交叉变异部分被RL中的动作选择和多Agent并发贪婪行动选择所替代，而GA则提供给了RL一个可以运行并更新的环境，两种方法的融合，使得该方法可以更有效率地并且更准确的完成自适应提取任务。

**4.2.3** **数据集介绍**



图4.5 egtest数据集示意图

egtest03数据集是DARPA计划期间在埃格林空军基地收集的数据，在空军基地的环境下，由飞机拍摄的图像，图像中共有三辆军车在前进，追踪的目标是处于中间部分的车。由于卫星所拍摄的数据通常有密级的限制，不容易获得，因此正好可以使用飞机拍摄的遥感图像来模拟卫星拍摄的遥感图像。数据集中一共2570帧图像，还配套一个mask.txt文件，文件中每条内容记录了每10帧图像的mask的变化情况。

**4.2.3** **系统模块**

1.DNA空间与分割

通过遗传算法的方式，确定DNA形式，完成种群的构建，并将该种群作为强化学习的环境。因为一共六个先验参数，所以DNA的长度为六，本次实验选择的分割方式是六等分，如下表所示，即每个Agent只面对一个参数。

表4.5 DNA分割情况

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Agent0 | Agent1 | Agent2 | Agent3 | Agent4 | Agent5 |
| qualitylevel | minDistance | winSize | maxLevel | COUNT | EPS |
| dna[0] | dna[1] | dna[2] | dna[3] | dna[4] | dna[5] |

2.多Agent的Q-learning方法

针对分组后的DNA，分别构建相同组数的Agent，即构建相同数量的Q表，各个Agent行为的选择相互独立，但是只有当所有的行为都并行地选择完毕后，将各个动作联合起来组成原格式的DNA，即选择好的角点选取和光流提取参数，将这个参数传入光流跟踪模块，得到MSE评价标准，将其转化成奖励后同时反馈给所有的Agents，让它们更新各自的Q表，具体的流程如下图所示。

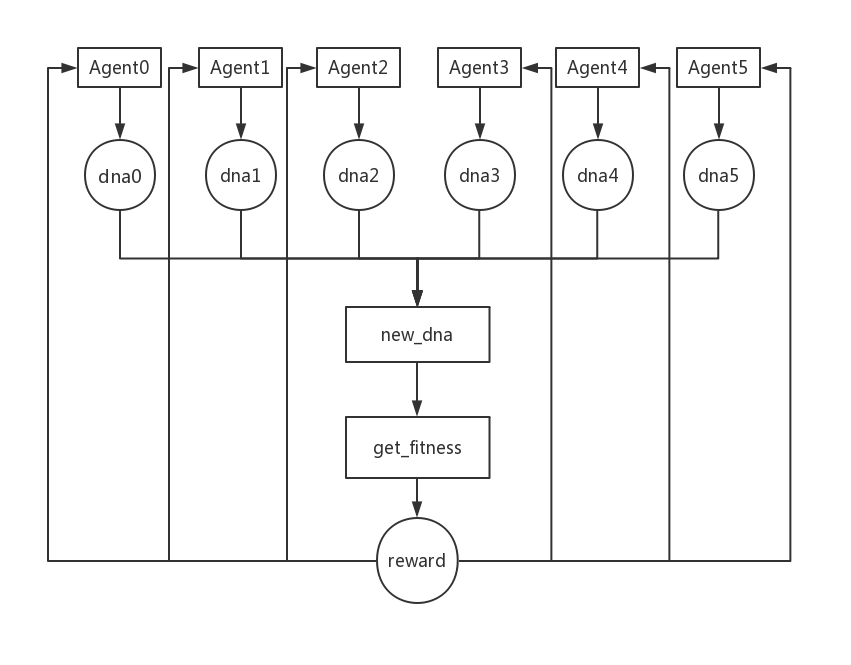


图4.6 多Agent更新Q表过程

3.Q表和种群数据的保存与读取

利用Q learning方法的离线特性，在每一轮的训练结束之后，将所有Agent的Q表保存下来，同时将此次的种群数据也保留下来。在训练的时候区分首次和非首次训练，首次训练时重新创建全为0的Q表和新的种群数据，非首次训练时将要读取上次迭代保存下来的数据，每次读取、更新和保存的过程视为一次迭代。在这个过程中记录最优适应度的个体DNA情况，并且在每次迭代的最后，按照更新好的Q表选择出一个DNA，通过比较这类个体的DNA在测试追踪下的效果，观察强化学习的效果。

4.训练模块

在遗传算法和进化策略的实验中，面对的只是10帧的图像，但是这次的训练集设置成了前100帧图像，每10帧图像进行20次mse的计算，然后对所有的Q表进行更新。由于mse是越小越好，所以在适应度计算函数中得出的mse应该取负，判断是否是第一次训练，如果是，奖励为0，把第一次的负值mse作为当前最大值；如果不是，则将当前负值mse与记录的最大值作差，所得到的值作为反馈。每次训练结束后，将Q表，种群和最大值信息保存下来，以便下次训练时使用。

5.测试模块

测试集选用数据集的第500-1000帧，将最终强化学习所选出的top5参数分别传入测试函数，每10帧比较一次mse的情况，将所有的mse绘制成图，观察图的变化情况，并与图像中物体的跟踪情况进行对比。

**4.2.4** **实验结果**

在100次迭代对前100帧的训练后，根据更新后的Q表，每个Agent选出Top5的值，然后相对应的组成参数，结果如下：

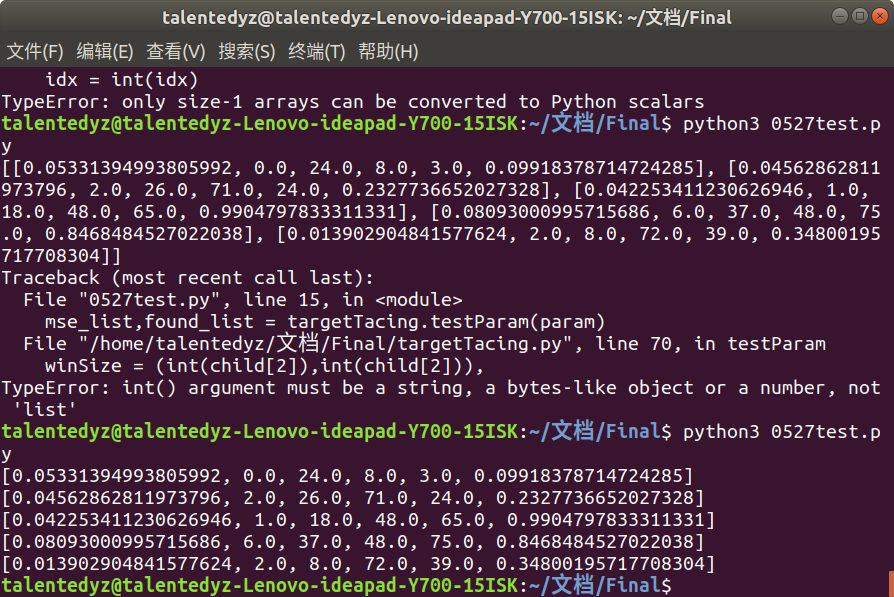


图4.7 强化学习选出的top5参数

可以看出，经过强化学习训练后的参数明显地避开了参数选择范围中不合理的部分，比如说第二个参数minDistance，如果超越10效果就会变得不好，如果超越20在当前数据集合下，会让跟踪的目标失效。但是也不是所有的结果都完全合理，因为强化学习的贪婪选择特性，几个坏的选择也是有可能出现的，但占比会很小。比如在选出的top5参数中，第4行参数中的winSize选择成了37，这会造成跟踪时找寻目标的窗口过大，导致在目标追寻的过程中很容易会出现偏移，如下图4.8所示，在追踪到中后部分时，红色的跟踪框已经明显的脱离被跟汽车的主体。



图4.8 追踪失败示意图

在该参数下进行计算的mse变化图如下图4.9所示。

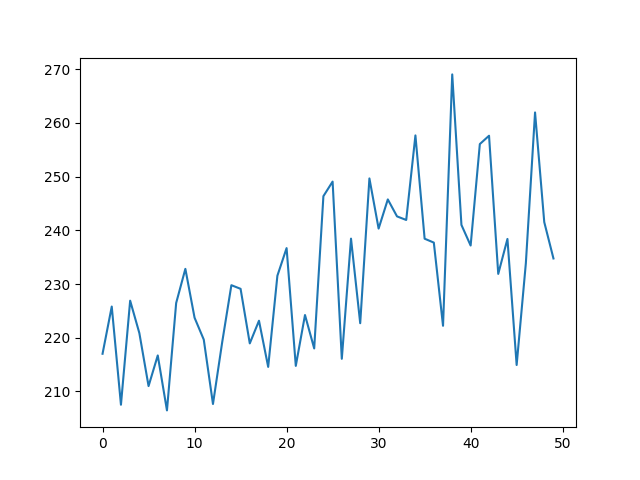


图4.9 第四次参数的mse结果

根据追踪时呈现出来的效果和在相对时间节点处的mse做比较，不难发现，在开始阶段的追踪是成功的，mse的标准维持在230左右，但随着汽车的推进，目标渐渐偏移，mse的主体渐渐提高到260左右，也就是说如果折线在这附近波动，可以认定追踪失败。

之后将其余参数代入测试函数，分别得到其对应的mse曲线：

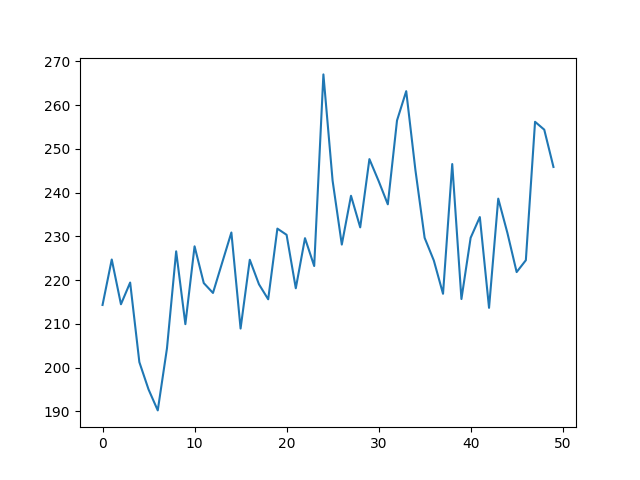
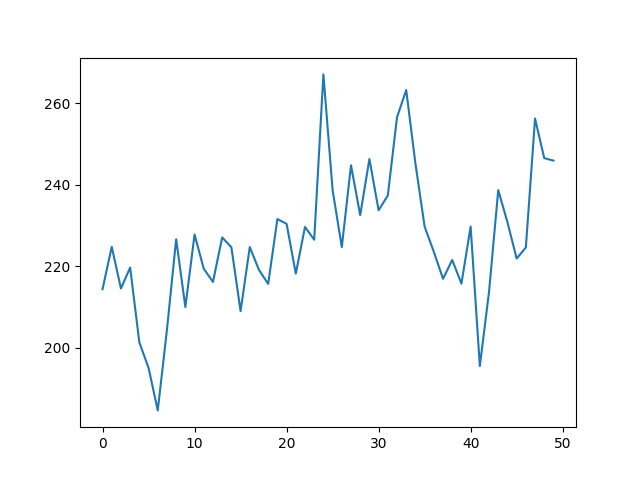


图4.10 rank1（左）和rank2（右）的mse结果图

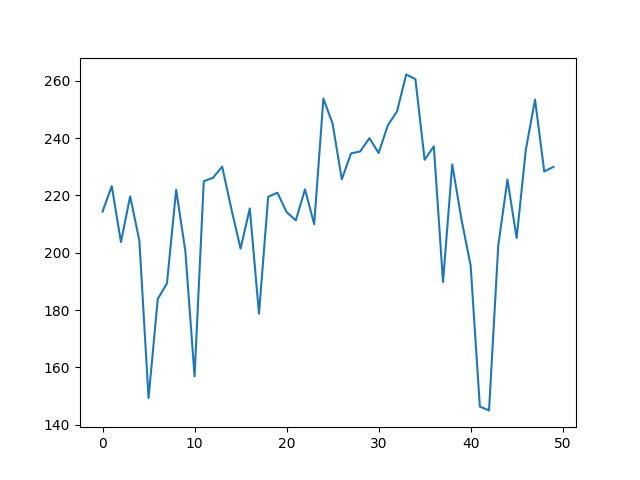
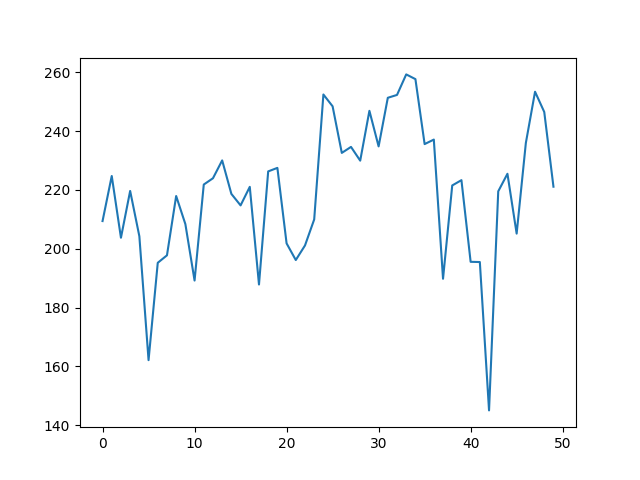


图4.11 rank3（左）和rank5（右）的mse结果图

明显看到，这些折线一般在260以下的范围内变化，因此可以在top5中将5种参数所得到的mse平均值作为评价标准，选取其中最低的一个参数作为最优参数，本次实验的最优参数值选取过程如下：

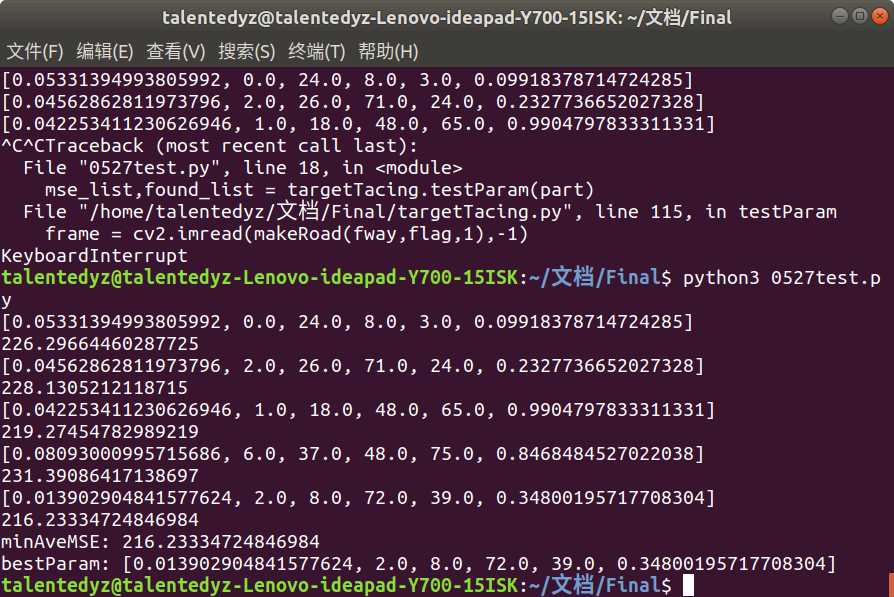


图4.12 最优参数选取过程

将最优参数传入光流跟踪函数，实验效果如下：

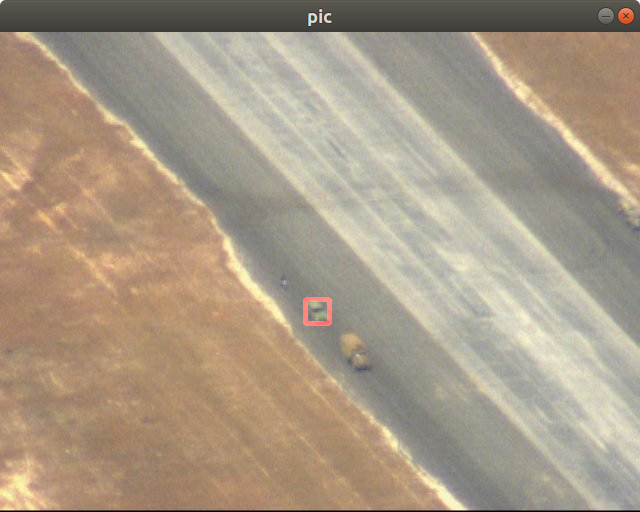


图4.13 1000帧（左）和1250帧（右）追踪情况



图4.14 1500帧跟踪（左）和1500帧实际（右）

如上图 4.14所示，在1500帧的跟踪下，所生成的红色目标框（图4.14左）已经几乎和数据集所给出的绿色目标框重合（图4.14右）。最终得出结果，可以通过强化学习的方法，在避免不必要的时间浪费的同时，完成对光流参数的自适应选择，并取得不错的应用效果。

4.3 本章小结

本章首先通过对遗传算法与强化学习的比较，分析了两种方法的优势和不足，为了更好的完成任务，选择了将两种方法的优势集中的RLGA算法，并将该算法的应用环境移到了光流提取上来。

然后针对egtest03数据集，将其拆分成训练集和测试集，针对前100帧的图像反复调用强化学习方法进行训练更新，以mse为评价标准计算图像间的相似度，并转化为环境的反馈，通过不断试错，使得智能体变得更加聪慧。并利用经过多次训练后的智能体，输出最有可能选取的5组参数，选取其中mse平均值最低的一组作为所选取的最终参数，然后利用该参数完成了追踪的任务。