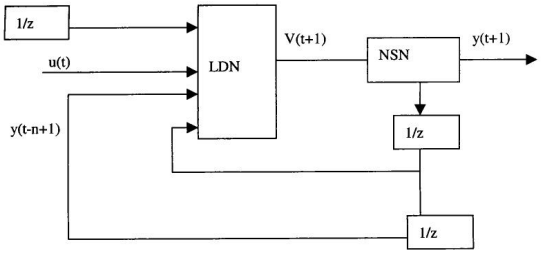

图 1　NDNN 模型结构图（u 表示系统的输入，y 表示系统的输出）

对于 LDN，U^T（t+1）=$[V_1(t+1), V_2(t+1), \ldots V_r(t+1)]$

其中，$V_k(t+1)=\sum_{i=1}^{m} a_{ki}\, u(t-i+1)+\sum_{j=1}^{m} b_{kj}\, y(t-j+1)$，（k=1，2，…r）

对于 NSN，y（t+1）=f$[V_1(t+1), V_2(t+1), \ldots V_r(t+1)]$

根据图一所示的非线性动态的神经网络线性动态网络（LDN）和线性静态网络（NSN）的模型，可以看出根据两者的不同特点对系统起到不同的作用。其中要对系统的动态特征加以学习，对非线性静态网络进行优化，可以对系统的非线性关系加以映射。如此，两者结合起来能够用简单的网络结果和较少的计算量、训练量，使得未知结构的非线性动态特征达到最佳状态。

（2）网络的学习过程

网络的学习过程，其核心内容就是对网络不断进行训练，通过改正其连接权值，保证网络的实际输出能够和理想的输出精度要求相符合。此外，在神经网络插补控制器进行学习训练的时候，可以利用 fANN 模板和插补控制器 CANN 连接的形式，实现对网络的学习过程。通过此种形式，能够很好省去学习过程中产生的中间数据，这样就极大提高了网络学习的质量和效率。

4　构建数控插补的神经网络控制模型

在对曲线模型进行插补的过程中，利用刀具在进行插补的时候，其速度方向也在不断发生变化。然而，由于刀具加工的速度不会无限增大，所以我们需要分别得出 x、y 轴的分速度变化曲线。然后对 x、y 两轴的速度进行计算，以便利用两轴联动来带动刀具不断朝着不同的方向进行插补。

其中，在每一段的插补内，x、y 轴的运动速度计算公式为：

$$v_{(x_i)}=\frac{f(x_{i+1})-f(x_i)}{\sqrt{((x_{i+1})-x_{i)}{}^2+((y_{i+1})-y_{i)}{}^2}}$$

$$v_{(y)}=\frac{f(y_{i+1})-f(y_i)}{\sqrt{((x_{i+1})-x_{i)}{}^2+((y_{i+1})-y_{i)}{}^2}}$$

其中，f（）表示刀具的进给速度，可以按照具体的加工情况进行精准地确定。

（1）网络结构分析

利用神经网络 fANN 对自由曲线进行构建分析后，在对其进行插补的过程中，能够利用 fANN 模块在不同点处获得的数据信息，并且得到在不同插补段内 X 轴与 Y 轴的运动速度，实现对两轴的驱动，以便完成两轴之间的联动，这样就可以推动刀具朝着不同的方向进行插补，从而确定刀具运动到的每一个定位点，其中，定位点的坐标需要变为直角坐标的形式。

通过对网络结构的分析，其输入层里面存在两个神经元，分别与 x 轴、y 轴的数值相对应，输出层存在四个神经元，分别与 x 轴、y 轴的数值及其插补后的 x 轴、y 轴的速度相对应，这样就得到了一个单隐层的网络结构模型，其隐层的单元数设置为七。神经网络的插补控制器可以用 CANN 来表示。

（2）网络的数控插补原理

在周期插补的过程中，需要将未知表达式曲线上的点全部输入到 fANN 里面，这样就能够在 fANN 输入端获取相应的坐标值，将其作为插补控制器的输入数据。把数据输入到 CANN 里面的时候，可以得到实际的输出结果，也就是下一个插补点的坐标值及其相关的运动速度，这样就能够获取插补周期的控制命令。把此控制命令传输到机床的 CNC 控制系统里面，完成周期的插补操作过程。最后，把插补周期过程中得到的输出端的数据值进行反馈，将此输出值作为插补器的输入端，进行重复训练，从而实现完整的插补运算过程。

5　进行仿真实验和结果分析

为了进一步证明构建的神经网络插补模型的科学性以及准确性，可以利用 MATLAB 仿真软件进行测试。依据仿真的结果可以得出，利用神经网络的非线性逼近及其自主学习的能力，通过对未知方程式曲线的辨识，能够实现对未知表达式曲线的插补过程。

目前，一般都应用神经网络的插补器进行相关运算，极大改善了传统插补方式计算量繁多复杂、出错率较高的不足，这样的神经网络数控系统插补器在工作的时候只需要对下一个插补点进行插补运算，也就是说当构建好网络，在输入端位置配置好前一个插补点的坐标值，就能够在输出端得到下一个插补点的坐标值，以及下一个插补点两轴所对应的运动速度等。最后把得到的数据传输到控制系统里面，以便完成系统的自动运行。

6　结语

本文主要利用神经网络插补模型对自由曲线以及未知表达式曲线进行插补，在这一过程中，只需根据自由曲线中的部分型值点就可以完成此操作，而且不需要其他的数学计算过程。此种算法原理简单、计算量较小，并且插补精度也相比于传统的插补方式高出许多，插补速度也得到了极大的提升。

因此，我们利用神经网络插补器来对未知曲线表达式进行插补加工，这样极大提高了插补的速度以及准确率，同时也减轻了计算负担。此外，如果想要进一步的提升神经网络的逼近水平，减少加工的误差，可以对相关的学习算法进行优化与改进。

参考文献：

[1]邬再新，李华兵.神经网络在自由曲线插补中的应用研究[J].组合机床与自动化加工技术，2019（2）：49-52.

[2]李华兵.神经网络在自由曲线插补中的应用研究[D].甘肃：兰州理工大学，2019.

[3]刘少贞.基于 RBF 神经网络的四轴机器臂轨迹规划研究[D].陕西：西安工业大学，2016.

[4]夏炎.多关节机械臂轨迹规划和轨迹跟踪控制研究[D].黑龙江：哈尔滨工业大学，2017.

[5]徐扣.六自由度机械臂的逆运动学求解与轨迹规划研究[D].广东：广东工业大学，2016.

网络安全机制研究与 Python 实现相关性分析

◆杨虹

（辽宁警察学院　辽宁　116036）

摘要：随着大数据的发展，数据与网站之间的共享也越来越密切，因此网络的安全性也被引起广泛重视。现阶段，网络泄密问题频发，常有信息被人随意窃取以及利用的情况发生，引发不良影响。为推进网络安全机制的建立，本文从 Python 实现角度展开探讨，就网络安全机制概念、Python 特征、以及两者之间的相关性加以分析，以为相关研究提供依据。

关键词：网络安全机制；Python；相关性；技术应用

基金项目：辽宁省教育厅本科教学改革教研项目，课题名称：基于 CBE 模式“公安视听技术”专业课程体系建设研究与实践

我们要想知道网络安全机制与 Python 实现相关性之间的关系，就必须先对网络安全机制和 Python 概念和所涉内容进行了解，这样才能更好理解两人者之间的关系，明确如何将 Python 在网络安全机制中运用，并对网络安全机制发展予以促进，现将相关在内容做如下探讨。

1　网络安全机制是什么

网络安全主要是指网络硬件基础设施的安全和网络访问的安全，避免黑客对网站进行攻击，通过获取网络用户信息，引发违法犯罪事件。目前市场上有许多产品用于解决网络安全的问题，我们常见的例如：防火墙系统、入侵检测安全技术等。

由于互联网的开放性、连通性和自由性，网络用户在享受各类共有信息资源的同时，也存在着自己的信息被别人窃取和恶意破坏的可能。因此网络安全的目标就是保护网络用户在有可能被窃取信息和篡改的情况下不被非法操控者所控制。而具体的要求则要达到保密性、完整性、可控性等，以保障用户的信息安全[1]。

2　网络安全机制主体内容

（1）加密机制

加密机制主要用于对加密技术可靠性的衡量，主要用于掌握解密过程所呈现出的难度性，而难度又来自于密钥的长度和算法。而加密机制又可按对称密钥与非对称密钥予以划分，两个不同种类的密钥所具备的长度和算法也不一样，这也就导致了密钥在解密过程中所体现出来的优缺点不一样。对称密钥的优点：加密处理简单，解密速度快；缺点：密钥在管理过程中有困难。非对称密钥又细化为公钥和私钥系统，优点：解决了密钥管理问题，加密强度增强，也增加了密钥安全度。缺点也与对称密钥相反，加密、解密的速度变慢[2]。

（2）安全认证机制

安全认证在电子商务活动中尤为重要，为了能够确保商务、交易及支付活动的真实可靠，需用一种途径来确保活动中的对象真实身份。安全认证是一种对电子商务活动的保护，它涉及安全管理、加密处理、PKI 及认证管理等重要问题。并且在安全认证机制中还具有一系列完备的法律来进行相应的管理，例如信用立法、电子签名法、电子交易法、认证管理法律等。

（3）访问控制策略

访问控制策略也分为三种，例如：入网访问控制、网络的权限控制和目录级安全控制。其中入网控制是网络访问的第一层控制，它主要是筛选了哪些用户能够访问哪些服务器，以及用户的入网时间和入网地点，只有通过各道关卡，该用户才能顺利入网。

3　Python 的概述

Python 属计算机程序范畴内的一种设计语言，为面向对象设计的具有动态特征的语言类型，最初被设计用于编写自动化脚本（shell），随着版本的不断更新和语言新功能的添加，越来越多被用于独立的、大型项目的开发。

Python 应用于图形处理，能方便进行图形处理；数学处理，提供大量与许多标准数学库的接口；文本处理，提供的 re 模块能支持正则表达式，还提供 SGML，以及 XML 分析模块等，大部分程序员善于应用 python 展开针对 XML 程序的深入开发[3]。

另外，Python 也属一种编程语言，因其具简单易学特征，故较易于被开发者接受。而 Python 的应用范围也是非常广泛，几乎所有大型互联网的程序运用中都有它的身影，都在通过 Python 完成各种各样的任务，例如国外应用较为广泛的 Google、YouTube，以及国内的美团、百度、知乎、新浪等，涉及生活中的多个方面，在我们应用的每个环节，都有通过 Python 的一个编程运用。而它主要的应用领域在以下几个，Web 应用开发、自动化运维、人工智能领域、网络爬虫、科学计算和游戏开发。其中就 Web 应用开发而言，程序员可更为有序且轻松地对复杂的 Web 程序进行分析和管理，使程序员工作负担明显减轻。例如我们经常用来搜索的 Google，和国内集音乐、电影等搜索于一体的豆瓣都是通过利用 Python 的语言来实现的。另外针对自动化运维而言，在常规情况下，对于 Python 参与编写的系统管理脚本来讲，不管是可读性，以及性能，或是代码重用度方面，同时还包括扩展性方面，均较普通 shell 脚本更占优势。在现今社会，人工智能发展的越来越迅速，而人工智能最多的涉及关于情绪的设定，在许多的人工智能设定中很多神经网的设置都是通过 Python 的语言来进行设置的。Python 在人工智能领域内深度学习、机器学习等，均属主流编程语言系统。

在网络爬虫、科学计算、游戏开发等领域，由于编程语言的特点，Python 在这些领域都有很大的发展，对这些领域的扩展起到了很大的作用。而在这些领域中，很大一部分的使用都需要进行注册，而注册这些网络软件、游戏等工具时，都是通过用户自己的实名制信息，而在这个网络访问中有很多的信息安全就需要被保护，因此网络安全和 Python 之间的相关性联系就显得很重要。

4　网络安全机制与 Python 的实现相关性

在我们现在的社会中，我们有很多信息不再是用过去的那种方式，用纸进行记录、统计和计算等，而是通过用计算机来进行操作。自从 1971 年后，微型计算机出现后，计算机的发展越来越迅速，而在随着计算机越来越轻薄的同时，很多软件也在逐渐丰富，在计算机中被输入的程序也越来越多。而从台式到笔记本电脑，从小灵通到智能手机，以及十多年前的 2G 网到现在的 5G 迅速发展，我们跟网络、跟各种程序的联系也越来越密切，而在这些接触的过程中，我们在享受着数据共享的便捷时，也必然承受着共享网络数据带给我们的危险。然而，在计算机网络应用以及普及阶段，其虽然会取得相对良好的效果，但随之衍生出来的问题也逐渐增多，包括：数据盗窃以及系统破坏等，如果不加以重视，那么就会对相关用户造成非常严重的影响。

现在我们经常都把自己的个人信息上传到网上，就以我们目前很火的支付宝、微信、淘宝等软件来说，这些软件几乎覆盖了我们生活的每个角落。在我们进行这些平时用的软件时，都要求实名制，以及可以读取自己的储存器、手机联系人等权限，当我们在使用的时候，软件的后台可以通过我们的 IP 地址找到我们在哪里，找到我们的信息。这些情况在正常范围内对我们对的生活很有帮助。而在这些软件中，很多都是有用到 Python 的计算机语言。

5　Python 的技术应用

Python 具有七大特点：简单易学、速度快、免费开源、可移植性、具可解释性、具精准的面向对象语言、库较为丰富。这是一种超高级语言。

Python 有一个交互式的开发环境，因为 Python 采用的是解释运行，因此编译时间会降低。Python 的语法简单，在的内部设置有几种高级数据结构，比如列表。Python 具有其面向性，可在 MS-DOS、Windows、Windows NT、Linux、soloris 等多种 OS 上运行。

Python 可被用来做批处理语言，写一些简单工具进行数据处理。Python 还可运用在函数语言中，推进人工智能科技的开发，还具备 lisp 语言的大多数功能。Python 在过程语言中也有参与，在我们常见的程序开发中出现。Python 具有面向对象语言的特征，经常作为大型软件的开发原型，再用 C++改写，还有的直接用 Python 进行开发。关于图像、音视频、动画等工作的处理，可以通过 Python 中的 PIL、Piddle、ReportLab 等模块来实现。另外在制作动态图表、统计分析图表时，也可以用 Python 实现。在更复杂的三维场景建立，也可以

用 PyOpenGl 模块来进行操作。

Python 在科学计算领域有其独特的地位。有许多工具模块可以提高人员在大量的计算、矢量分析、神经网络等工作时的效率。尤其是在教育科研方面，可以发挥出独特的优势。

Python 为网络编程工作提供了很多的解决方案和模块，提高工作效率，做出自己的服务器软件，无论是 c/s，还是 b/s 模式。Python 早就作为游戏编程的辅助工具。在《星球大战》中扮演了重要的角色。在“阿贝斯（Abyss）”、“星球之旅（Star Trek）”、“Indiana Jones”超级大片中担当特技和动画制作的工业光魔公司（Industrial Light）就是利用的 Python 语言。现今，利用 Python 可以写出很棒的游戏程序。目前，Python 已经进入到很多企业级应用和政务应用中，在全世界，有很多公司的软件开发或者应用都是用的 Python 语言，例如：ERP、CRM。

6　结语

网络安全是时代热点和急需解决的问题，在这个时候建立网络安全机制也是对用户的一种保护，同时又降低了商家关于在软件被恶意使用、破坏等情况下产生的损失。而 Python 在这样的环境下也会随着问题的出现而开发解决办法，促进 Python 的成长。逐渐的网络安全机制与 Python 的相关性也会越来越密切。

参考文献：

1]聂晶. Python 在大数据挖掘和分析中的应用优势[J]. 广西民族大学学报（自然科学版），2018，24（01）：76-79.

[2]陆树芬. 基于 Python 对网络爬虫系统的设计与实现[J]. 电脑编程技巧与维护，2019（02）：26-27+51.

[3]蔡敏. Python 语言的 Web 开发应用分析[J]. 无线互联科技，2019，16（04）：27-28.

一种虚拟信标节点机制的 DV-Hop 定位改进算法

◆胡平霞　龚静　丁锋　张玉平

（湖南环境生物职业技术学院　湖南　421001）

摘要：本文为提高无线传感器网络中非测距定位算法 DV-Hop 的定位精度，特提出一种通过信标节点移动形成虚拟信标节点的 VADV-Hop 改进算法，优化信标节点部署位置，使用网络平均跳距校正信标节点平均跳距。仿真实验表明 VADV-Hop 算法在不增加信标节点数量上表现出较稳定的定位精度。

关键词：无线传感器网络；定位；平均跳距；虚拟信标节点

基金项目：衡阳市科技计划项目（2016KJ20）、湖南省教育厅科学研究项目（16C0565）、衡阳市社会科学联合会（2017D107）、湖南环境生物职业技术学院支柱工程项目（湘环院教字〔2017〕46 号）。

无线传感器网络（Wireless Sensor Network， WSN）是由静止或移动的传感器节点以自组织和多跳的方法组成的无线网络，是一种集感知数据、采集数据、处理数据、传输数据等技术实现物理世界与信息世界数据交互的一种分布式网络系统。传感器节点返回的数据包含位置信息，没有位置的数据实用价值很低，因此传感器节点定位技术是 WSN 的关键技术之一，其定位精度直接影响 WSN 的发展和应用。

根据定位过程中是否要测量未知节点和信标节点间的距离，把定位方法划分为基于测距定位（Range-based）方法和免测距定位（Range-free）方法[1]。其中免测距定位方法由于其低成本、易实现等特点得到很多研究者关注。常用免测距定位方法有 DV-Hop 算法[2]、质心算法[3]、APIT（Approximate Point-in-Triangulation Test）算法[4]、凸规划（Convex）算法[5]等，它们都是通过已知节点位置信息实现对未知节点的定位。

1　DV-Hop 算法

美国 Lutegesi 大学的 Dragons Niculescu 等人在 2003 年提出 DV-hop（Distance Vector-Hop）算法采用距离矢量路由机制定位，是一种基于多跳测距的免距离测距方法[2]。DV-Hop 算法是目前研究较多的免测距算法，该算法通过信标节点位置估算未知节点位置，然后信标节点的数量在实际应用中受到成本限制，因此本文在 DV-Hop 算法思想上提出一种基于虚拟信标节点技术的改进算法。

1.1 定位过程

DV-Hop 算法节点定位过程[6]如下：

（1）确定信标节位置及最小跳数

通过信标节点广播的数据包，信标节点确定其他信标节点坐标及最小跳数，未知节点确定信标节点坐标及最小跳数。

（2）确定平均跳据，估算距离

使用信标节点的位置坐标及信标节点间的最小跳数，通过公式 1 来计算信标节点的平均跳距，使用平均跳距结合公式 2 计算未知节点到信标节点的距离。

$$HopSize = \frac{\sum_{i\neq j}\sqrt{(x_i-x_j)^2+(y_i-y_j)^2}}{\sum_{i\neq j} h_{ij}} \quad （公式 1）$$

节点 i 的坐标表示为（x_i，y_i），节点的坐标表示为（x_j，y_j），i、j 均为网络中的信标节点。信标 h_{ij} 表示二节点 i、j 间的最小跳数，$HopSize$ 表示 i、j 间的跳距平均值。

$$d_{uv} = HopSize_i \times hop_{uv} \quad （公式 2）$$

$HopSize_i$ 是未知节点 u 选用最邻近信标节点 i 的跳距平均值，hop_{uv} 是未知节点 u 到信标节点 v 的最小跳数值。

（3）确定未知节点位置

未知节点在获得 2 个以上跟信标节点的参考距离后，使用相应信标节点坐标（x_u，y_u）及 d_{uv}，转换成线性方程，通过数学方法（最小二乘法）计算得到未知节点位置坐标。

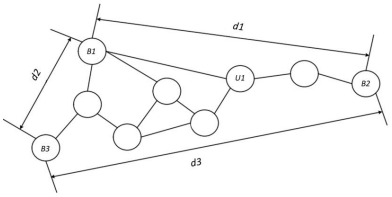

图 1　DV-Hop 估算示意图

图 1 中 U_1 为需要定位的未知节点，通过已知节点即信标节点 B_1、B_2、B_3 的位置信息来测算 U_1 的位置。首先 B_1、B_2、B_3 通过洪泛算法进行广播，各节点保存最小跳数信息。其次 B_1、B_2、B_3 节点计算自身平均跳距，U_1 使用最邻近的节点 B_1 平均跳距作为自身跳距平均值分别计算到信标节点的距离。最后建立位置矩阵，通过最小二乘法计算 U_1 的坐标。

1.2 算法分析

1.2.1 客观条件分析