基于Python的程序管制冲突检测与调配辅助教学系统设计与开发

——以武汉南湖机场进近空域为例

廖 勇，赵世昌

（中国民用航空飞行学院空中交通管理学院，四川 广汉 618307）

摘要：程序管制作为雷达管制的备份手段在雷达失效时发挥着重要作用，掌握程序管制技能对管制员至关重要。为此，设计并实现一种基于Python的程序管制冲突检测与调配辅助教学系统。首先设计7层架构，涵盖用户层、表示层、业务层、模型层、数据层、操作系统层和硬件层。随后构建5个核心模块，包括交互界面模块、数据预处理模块、冲突检测模块、动态冲突展示模块和冲突调配模块，并对每个模块的功能和实现技术进行详细设计。最后，以实际教学场景武汉南湖机场进近管制空域为例，使用Python对系统进行模拟验证，学生可以使用该系统自动检测冲突类型，并提出冲突解决方案，从而更好地掌握冲突调配技能。

关键词：程序管制；冲突检测；冲突调配；教学系统

中图分类号：[U8] **文献标志码：**A **文章编号：**1671－1807(2025)05－0121－09

对于国内许多中小运输机场来说，雷达系统价格昂贵，加之每日航班量较少，这些现实条件导致机场放弃了雷达管制，依然依赖程序管制来保障航班安全有序运行[1]。因此，掌握程序管制技能对管制员而言依然至关重要。对于航空管制专业的学生来说则需要在学习阶段打下扎实的程序管制基础。然而，目前针对程序管制的教学研究主要体现以下方面，如管制员模拟训练[2-5]、管制员理论教学[6-8]、管制员培训评估[9-10]。通过研究发现，目前缺乏动态可视化冲突检测和冲突调配的自动生成工具，使得学生在学习过程中无法直观地感受到飞行冲突的产生和调配过程。因此，开发和引入这些动态可视化工具对于提高教学效果、帮助学生掌握复杂的冲突调配技能具有重要意义。

基于此，本文提出一种基于Python的程序管制冲突检测与调配系统，该系统可以根据输入的飞行计划自动检测两架航空器是否有飞行冲突并动态可视化展示冲突点，并能够根据冲突类型自动生成并输出冲突调配方案。通过系统的辅助，学生能更好地掌握复杂的冲突调配技能。

1 需求分析

1.1 用户需求

本系统的用户是学生。在学生学习程序管制时，由于缺乏飞行冲突类型的判断能力和冲突解决方法的经验，导致冲突判断不准、冲突解决方法使用不对等问题。不但会影响理论环节的学习效果，还会导致实践环节管制方案制定错误，最终导致课程目标不能达成。因此，从用户需求角度出发，系统能根据飞行计划自动检测出飞行冲突的类型，并根据飞行冲突生成解决冲突的方法。

1.2 功能需求

根据用户需求，本系统的主要功能是根据飞行计划自动判断是否存在飞行冲突，并输出冲突产生的原因及相应的冲突调配方案。此外，系统还具备一系列辅助功能，如动态演示冲突过程。具体为系统能根据飞行计划仿真航空器的进离场过程，在仿真过程中自动检测航空器是否存在飞行冲突，检测到冲突后，提供冲突的详细信息，自动生成冲突调

收稿日期：2024-10-03

基金项目：四川省科技计划(23ZDYF0586)；中国民用航空飞行学院博士创新能力提升计划(PHD2023-038)

作者简介：廖勇(1983—)，男，四川资阳人，博士，教授，研究方向为空中交通管理；赵世昌(1999—)，男，陕西渭南人，硕士研究生，研究方向为程序管制。

配方案，并通过动态展示功能实时呈现航空器的飞行过程，帮助学生更好地掌握冲突调配方法。

1.3　系统界面需求

为了更好地配合教学，降低系统使用门槛，用户界面的设计需要简洁友好，易于操作，以便用户能够轻松访问系统的各项功能。界面需要清晰地展示航空器的冲突点和调配方案，确保系统操作流程符合用户习惯，提升教学效率。

2　系统构建

2.1　系统整体架构

仿真平台的架构如图 1 所示，整个系统架构分为硬件层、操作系统层、数据层、模型层、业务层、表示层和用户层 7 层。

硬件层：为了方便用户使用，系统能运行于个人电脑上，利用显示器或投影仪展示仿真内容，包括显示器、投影仪和个人电脑。

运行环境：系统能跨平台运行在 Windows、Linux、macOS 系统上，用于实现系统功能。

数据层：用来存储机场进离场航线数据、飞行计划数据和调配方案数据。

模型层：主要包括 3 个部分，分别是用来解析和处理飞行计划的数据处理和解析模型、冲突检测模型和负责冲突调配的逻辑推理模型。

业务层：系统的核心主要包括 5 个模块，分别是交互界面模块、数据预处理模块、冲突检测模块、动态冲突展示模块、冲突调配模块，使用这些模块可以实现用户的交互、数据的预处理、冲突检测、调配方案生成及其可视化展示。

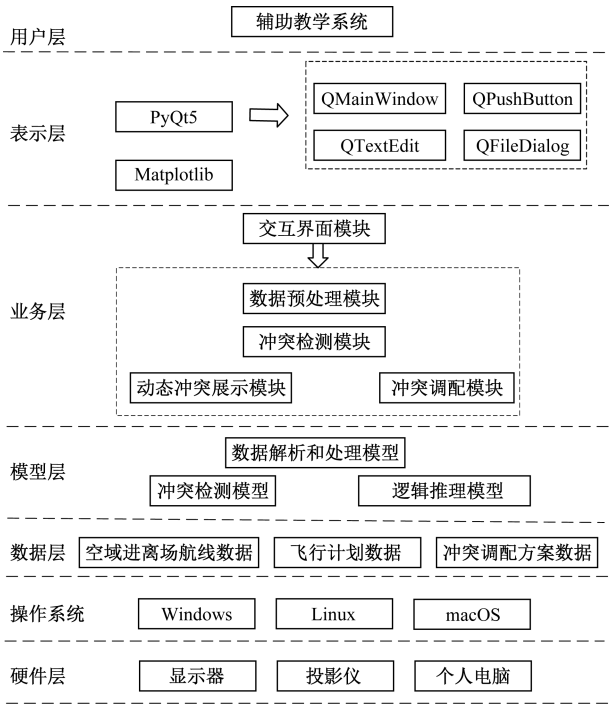


图 1　系统架构

表示层：系统实现主要的技术工具，通过 PyQt5 工具来实现包括主窗口（Mainwindow）、对话框（QFileDialog）、文本输入框（QTextEdit）、按钮（QPushButton）的实现，使用动态冲突展示工具（Matplotlib）展示动态冲突路线。

用户层：用户主要包括学生和老师，可以利用这个系统实现辅助教学。

2.2　功能模块设计

2.2.1　交互界面模块

交互界面模块主要分为 5 个部分：主界面、数据输入区、冲突检测区、动态冲突展示区和冲突调配区（图 2）。主界面采用 PyQt5 5 进行事件驱动和算法布局，PyQt5 是一个功能强大的应用程序设计工具[11]，具有可扩展性稿、易操作等特点。利用 PyQt5 5 的 QMainWindow 和 QValidator 实现数据输入区通过数据解析算法处理并存储飞行计划和尾流数据，并利用异常处理机制提示输入数据的标准格式，通过加载按钮加载进离场航线和冲突调配方案库。冲突调配方案库是指根据进离场冲突发生原因收集的解决冲突方案集合，借助 QTextEdit 和 QValidator 实现交互。冲突检测区则使用事件驱动算法和反馈生成机制，使用 QHBoxLayout 结合 QPushButton 和 QTextEdit 设计冲突检测按钮和结果输出框，确保用户能够直观地检测冲突并获得反馈信息。动态冲突展示区采用动画展示算法和用户输入处理算法，通过 Matplotlib、FuncAnimation 实现动态展示飞行路径、冲突点和冲突路线，并通过 QPushButton 控制动画的播放速度。最后，冲突调配区使用事件驱动算法和规则匹配算法，接收用户输入的飞行计划数据，通过 QPushButton、QTextEdit 和 QHBoxLayout 设计按钮和结果输出框，输出冲突原因及调配方案。

2.2.2　数据预处理模块

数据预处理模块输入部分航空器的飞行计划数据飞行冲突的调配方案库（图 3）。输出部分首先是针对飞行计划数据和飞机尾流等级数据错误格式的提示，如航空器的飞行计划数据，系统会显示“输入格式错误，请重新输入”。当数据格式正确，预处理模块会将输入的数据进行提取算法解析并形成飞行计划列表。经过加载后的调配方案通过字典储存到系统中方便冲突调配时使用。字典

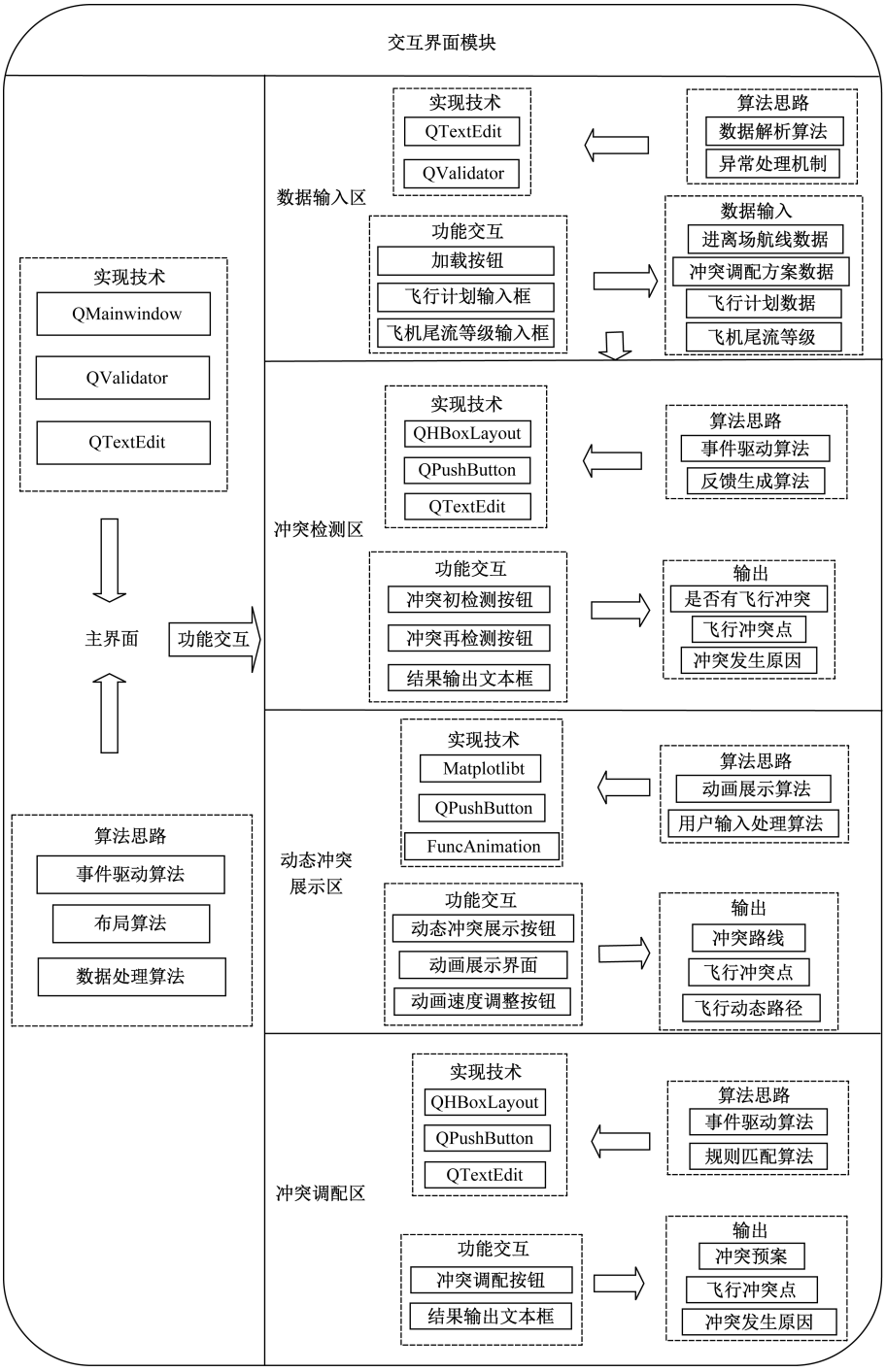


图 2　交互界面模块构建

(dictionary)是一种内置数据结构在冲突调配方案中,字典用于存储预案信息,其中每个冲突原因(键)对应一个或多个解决方案(值)。这样,程序可以快速查找和匹配相应的冲突解决方案。

算法涉及数据验证和数据解析两个方面。数据验证是通对用户输入的飞行计划、飞机尾流等级、冲突调配方案进行格式和内容的验证。这一步通过一系列规则检查数据的有效性,如确保航路点不为空、字符串合法性以及飞机类型的有效性。一旦数据验证通过,系统将提取输入数据中的关键信息进行数据解析。对于飞行计划,提取航路点及其时间信息,并将解析后的数据以飞行计划的形式和冲突调配方案分别储存到字典中,以便于后续操作和快速访问。

实现技术包括使用 Python 语言及其相关库。数据验证利用 try-except 结构,捕获文件读取和格

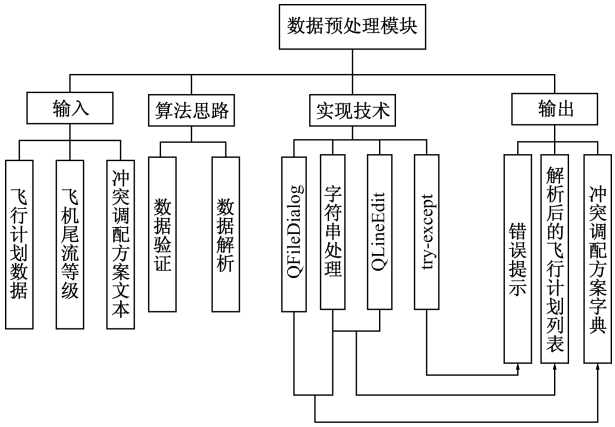


图 3　数据预处理模块构建

式错误等异常。数据解析则通过字符串处理和 QLineEdit，结合 QFileDialog 组件允许用户选择文件，随后提取航迹点及时间信息。最后，采用字典数据结构存储解析后的数据，以实现高效查找和访问，为后续模块提供可靠数据支持。

2.2.3　冲突检测模块

冲突检测模块由初检测子模块和再检测子模块组成(图 4)。冲突初检测模块输入部分是经过数据预处理的飞机飞行计划列表，经过航迹交叉检查和走廊口时间检查算法进行检查，对于航路有交叉且能直接确实是否存在冲突的情况输出对应的结果，对于存在航路交叉但是不能立即判断是否存在冲突的情况，输出潜在冲突，并记录交叉点为潜在冲突点。随后将其作为冲突再检测的输入部分进行再验证。冲突再检测则是通过冲突网络图判定算法、For 循环和数据结构判定潜在冲突点是否有冲突，最后输出是否有飞行冲突、冲突点和冲突发生原因。

冲突初检测算法包括两个主要步骤：航迹交叉检查和导航台时间检查。首先，在航迹交叉检查中，系统通过循环遍历和集合交集操作，提取航空器飞行计划中的航路点，寻找它们是否在同一航路点或内部存在重叠点，若发现重叠则记录为交叉点。同时，时间检查会提取两架飞机的走廊口过台时间，利用条件判断来检测时间差是否在可接受范围内。若时间差满足冲突条件，则标记为有走廊口飞行冲突；如果仅存在重叠但不符合冲突条件，则标记为潜在冲突，并输出内部的潜在冲突点。冲突再检测是在初检测的基础上，进一步确认潜在冲突点是否确实会引发冲突，并识别冲突的原因。首先，通过构建冲突树状图来分析航班经过的顺序和时间，以识别潜在冲突的源头。随后，利用时间序列匹配算法比较两个航班在潜在冲突点的通过时间，确保这些时间差符合冲突定义的条件。最后，通过数据结构记录冲突原因及相关信息。冲突树状图指的是形成飞行冲突的一系列因素所构成的树状图，根据树状图可以对冲突的发生原因进行详细解释。

冲突检测中关键技术主要包括时间处理、重叠点检查、导航台时间检查和冲突判定。航迹检查中的时间处理采用 Python 的时间序列分析来解析和管理航班的轨迹信息。航迹点的重叠检查则通过 Python 的 set 和集合交集来实现。此外，导航台时间检查依赖于条件判断语句来对时间差进行检查。再检测部分，冲突网络图判定通过 IF 循环语句来判断航班经过的顺序和时间冲突，而时间序列匹配算法通过 For 循环和索引访问实现潜在冲突点的比较。最终，所有冲突信息通过 Python 字典数据结

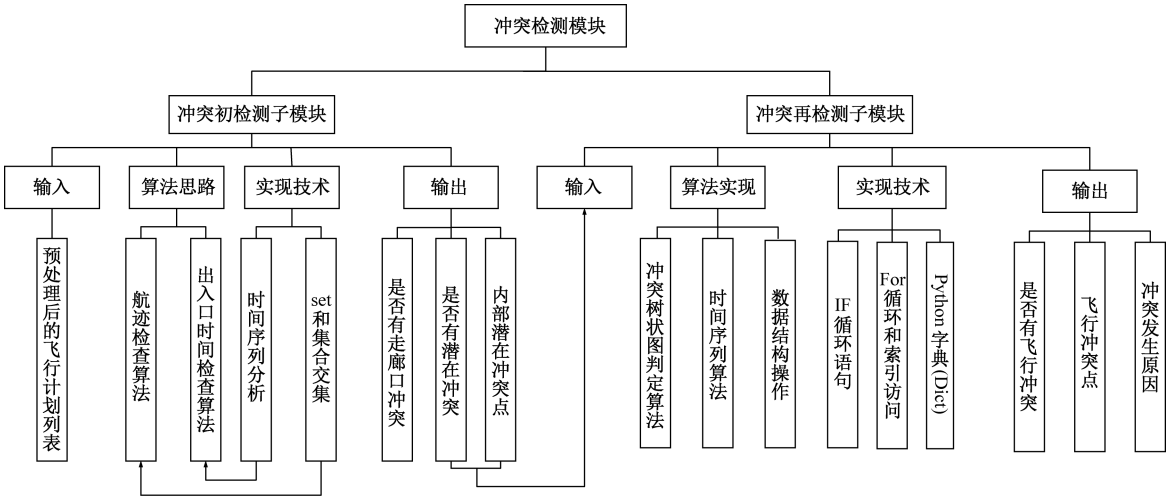


图 4　冲突检测模块构建

构进存储和管理,以便快速查找和匹配相关信息。

2.2.4　动态冲突展示模块

动态冲突展示模块的输入信息包括经过冲突检测后产生的飞行冲突点数据、飞行冲突的原因以及两架航空器的原始飞行计划数据(图 5)。飞行计划数据包含航空器的航路点及其对应的时间信息,而冲突点数据则标记了两架航空器在飞行过程中可能产生冲突的具体时间和位置。系统将这些数据整合后,以动画的形式将冲突过程可视化展示,便于学生直观了解冲突的发生过程和原因,并用于教学和演示。

算法思路主要由 3 个部分构成:①坐标映射算法通过将航点名称转换为具体的二维或三维空间坐标,实现从抽象飞行数据到可视化地图的转换。②动画更新算法根据时间轴的推进实时更新飞机的位置,模拟飞行过程中飞机在航线上移动的状态,通过逐帧更新显示动态效果。③速度控制算法则根据飞行速度和动画播放帧速率之间的关系,调整飞机移动的频率,从而实现不同的飞行速度展示效果,这可以用来模拟航空器飞行速度的变化或进行快进、慢放的教学演示。

实现技术方面,Matplotlib 库在 Python 中是使用最多的绘图工具[12],模块主要依赖 Matplotlib 库实现数据的可视化和图形绘制。本文的动态具体实现中,首先使用 plt.subplots 创建绘制区域,并通过 ax.plot 将进场和离场航线、飞机的飞行路径以线条的形式在图中绘制出来。之后,通过在冲突点的位置进行标记,显示出冲突点及其周边区域。FuncAnimation 模块用于生成动态效果,按照时间序列更新飞机的坐标,使得飞机在绘图中动态移动。为了实现不同的速度效果,系统通过调整 FuncAnimation 的更新频率,控制飞机在图形中的移动速度,最终达到流畅、逼真的动态飞行演示效果。

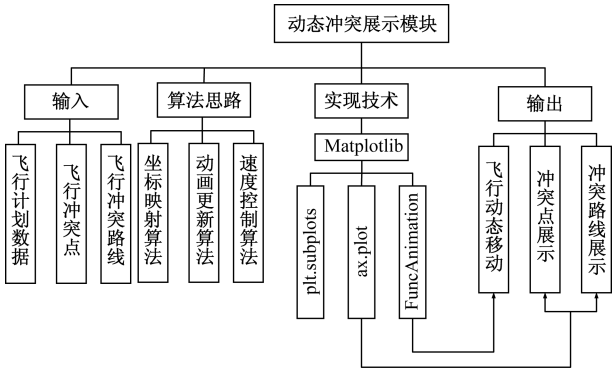


图 5　动态冲突展示模块构建

2.2.5　冲突调配模块

冲突调配模块的输入部分是经过冲突检测后生成的冲突点信息和冲突发生的具体原因,输出部分则是根据匹配的冲突调配方案生成的具体解决方案(图 6)。

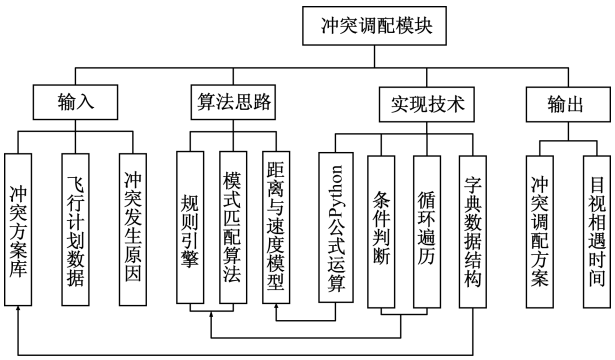


图 6　冲突调配模块构建

规则引擎是一种将规则和数据推行推理的推理引擎[13],通过其可以匹配方案库中的方案。算法思路上,首先通过已经加载的冲突调配方案库,系统采用规则引擎和模式匹配算法,来选择与当前冲突情况最为匹配的解决方案。针对目视相遇策略,算法会根据飞行器的速度和当前距离进行计算,确定相遇的最佳时间点。通过距离与速度模型,算法能精确地估算出两个航空器在空间中相遇的时间,确保在飞行调配过程中能够准确避让潜在的风险。距离与速度模型的计算流程如下。

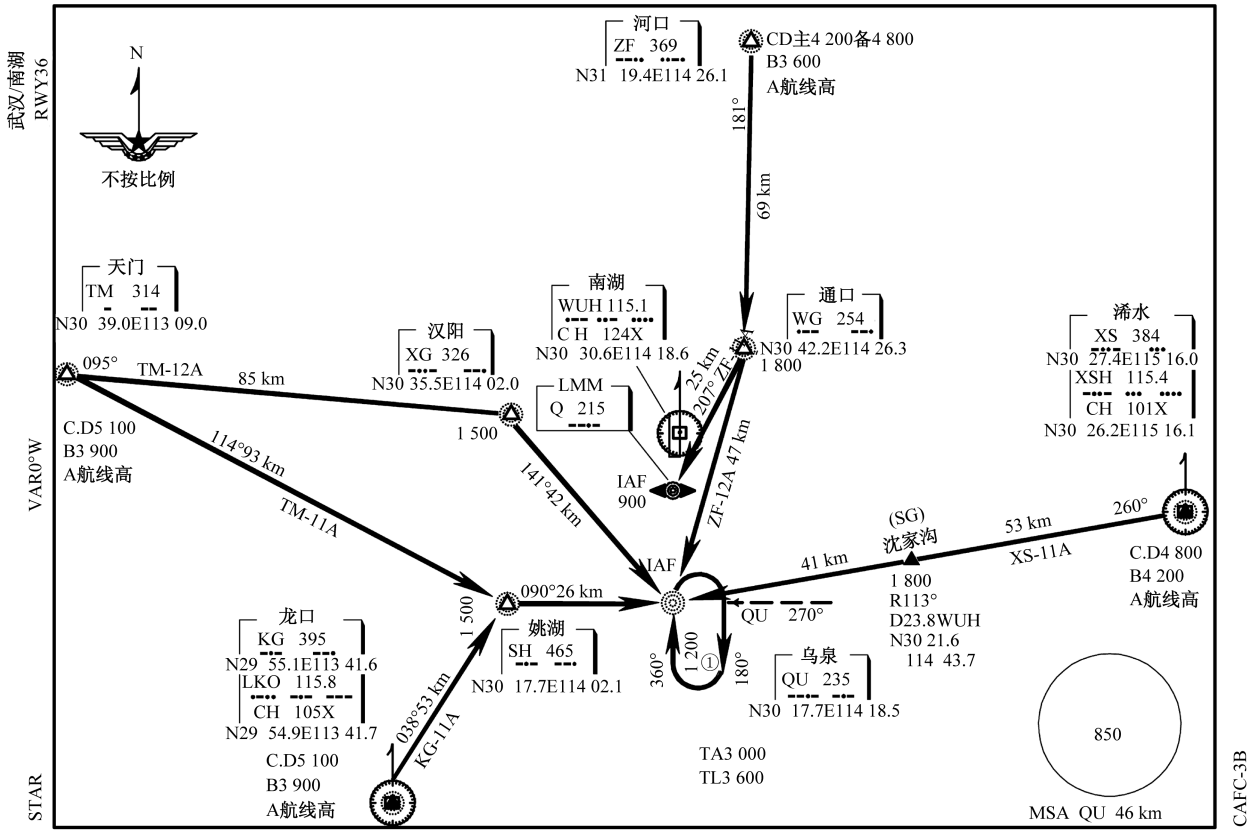
同航段逆向飞行两机相遇时刻的估计,优先选择预计过同一报告点时间差最小的台为基准进行计算,这样相对比较精确,设速度 v_1 飞机,过台时刻为 t_1 ,速度 v_2 飞机,过台时刻为 t_2 ,相遇时刻为 t_e ,则:

$$\begin{cases} t_e = \dfrac{t_1 + t_2}{2}, & v_1 = v_2 \\ t_e = t_1 + \dfrac{t_2 - t_1}{1 + \dfrac{v_1}{v_2}}, & v_1 < v_2 \\ t_e = t_1 + \dfrac{t_2 - t_1}{1 + \dfrac{v_2}{v_1}}, & v_1 > v_2 \end{cases} \tag{1}$$

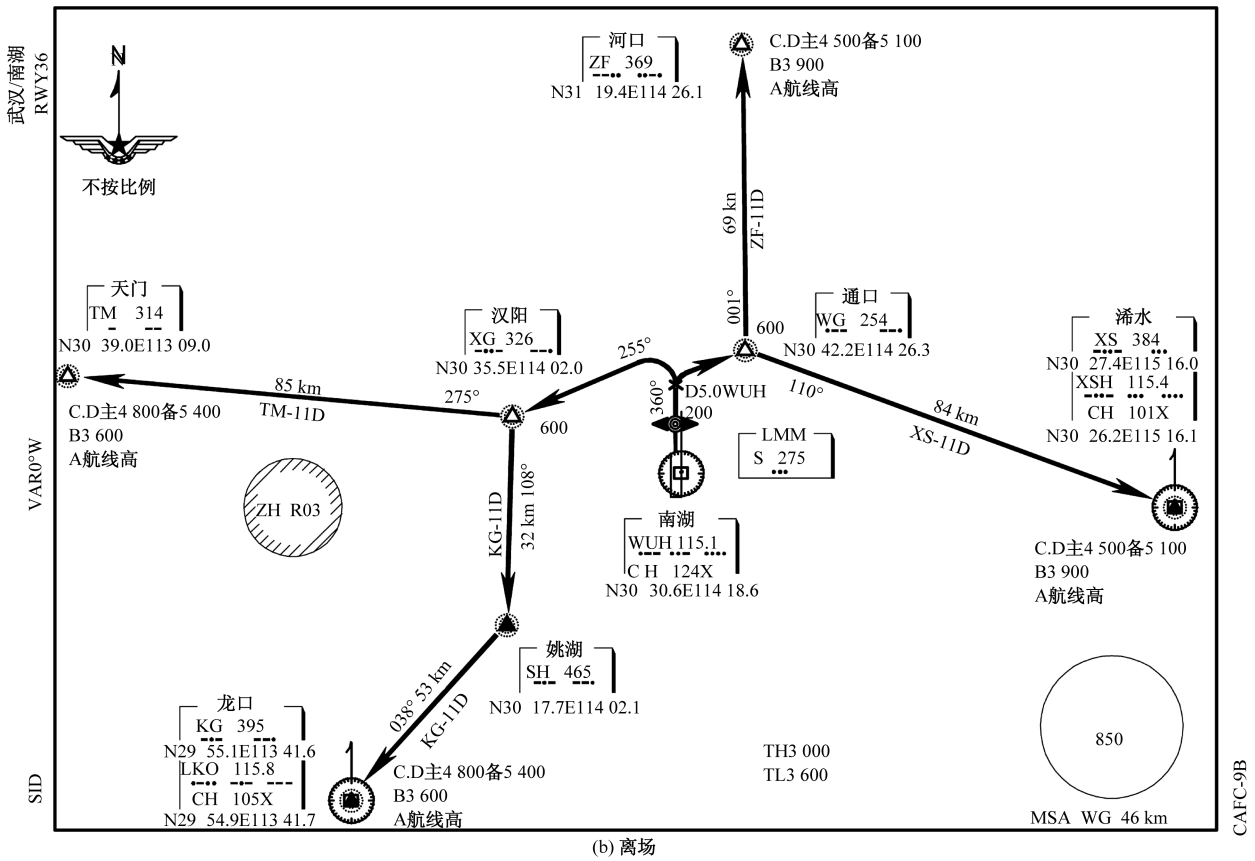
在实现技术方面,规则引擎和模式匹配算法通过条件判断和循环遍历的方式实现。条件判断负责根据冲突原因筛选合适的解决方案。循环遍历可以实现对不同文本的遍历[14],从而用于遍历冲突调配方案库中的不同方案。

3　软件实例分析

以实际教学场景武汉南湖机场进近管制空域为例,武汉南湖机场的空域图如图 7 所示。武汉南



(a) 进场



(b) 离场

图 7　武汉南湖模拟机场进离场航线图

湖机场是一个常用的教学实例[3],其涵盖了 4 个主要走廊口(天门 TM、河口 ZF、浠水 XS、龙口 KG)和多个内部导航台(汉阳 XG、姚湖 SH、乌泉 QV、沈家沟 SG),起始进近定位点为乌泉。

3.1　主界面实现

本系统主界面的设计主要采用 PyQt5 工具,可以实现布局简洁且功能齐全,分为多个主要区域,以提升用户体验和学习效果。

如图 8 所示,右侧展示了武汉南湖机场的进离场程序图。左上方是功能区,用户可以通过“加载”按钮选择不同的冲突调配方案及进离场程序图。左侧的操作区域包括数据输入区、冲突检测区、冲突调配区和动态冲突区的交互界面。用户可以点击“冲突初检测”“冲突再检测”“冲突调配”“动态冲突展示模块”按钮,实现相应模块的交互操作。最终的冲突检测和调配结果将统一显示在输出区中。

3.2　冲突检测模块实现

为了进一步展示各模块的实现过程,选取两架航空器进行模拟验证,分别为进场航空器 A(飞行计划数据:ZF12WG17QU21,尾流等级:H)和离场航空器 B(飞行计划数据:机场 02WG08ZF13,尾流等级:H)。

首先,对两架航空器进行冲突初步检测。将相关数据填入数据输入区后,系统会先进行数据验证格式是否正确。通过验证后输入数据预处理块中进行预处理成两架航空器的飞行计划列表。随后学生可以点击“冲突初检测”按钮。经过初检测模块航迹交叉检查算法判定两架航空器存在交叉点为走廊口 ZF 和内部 WG,并通过时间序列分析两架航空器在 ZF 走廊口的时间不满足冲突情况,因此判定 WG 为潜在冲突点并进行输出。最后结果如图 9 所示。两架航空器存在潜在冲突,潜在冲突点位于 WG。重叠的走廊口为 ZF。如果航空器之间不存在潜在冲突,系统将显示“没有飞行冲突”,并且无法点击“冲突再检测”按钮。

接下来,为进一步确认是否存在飞行冲突,点击“冲突再检测”按钮。经过时间序列分析等算法对 WG 进行判定后最后输出结果如图 10 所示。两架航空器确实存在飞行冲突,冲突点位于 WG,且冲突原因是进场航空器与离场航空器在 ZF-WG 区间相遇。如果再检测结果显示没有飞行冲突,则系统会提示“没有飞行冲突”。

3.3　动态冲突展示模块实现

针对上一步检测的飞行冲突结果采用 Matplotlib 的动画功能展示两架航空器的飞行路径和冲突过程。如图 11 所示,两架航空器分别以不同颜

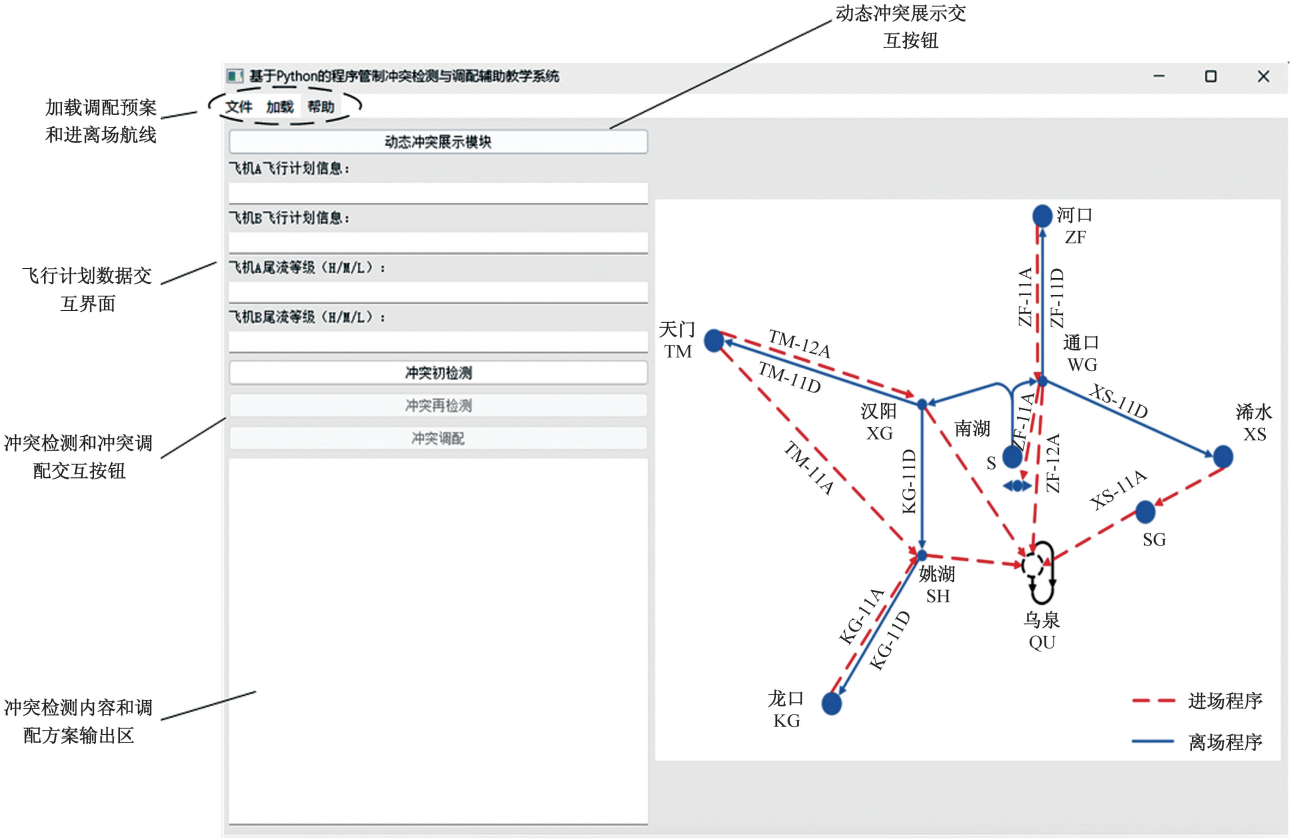


图 8　系统主界面

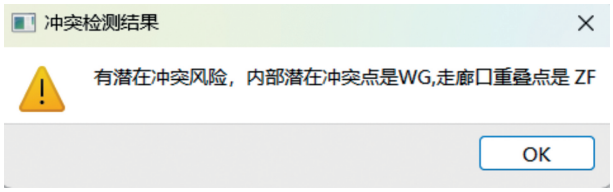


图 9　冲突初检测结果

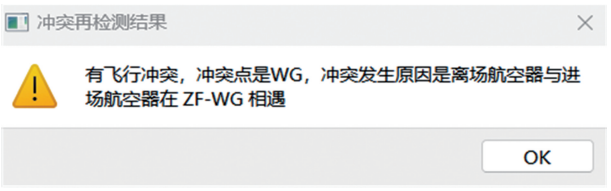


图 10　冲突再检测结果

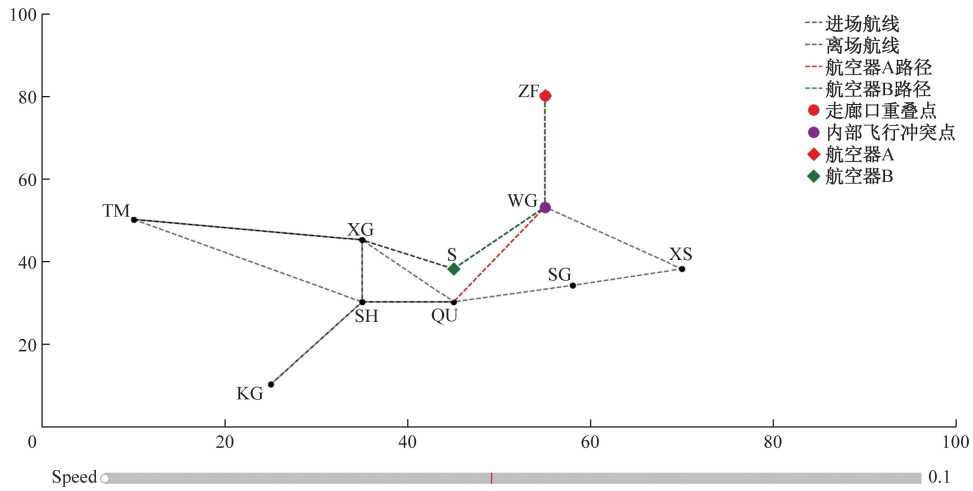


图 11　冲突动态展示界面

色的菱形标记：红色代表航空器 A，绿色代表航空器 B。航空器 A 的飞行路径用红色线条显示，航空器 B 的路径则用绿色线条显示。走廊口重叠点“ZF”用紫色圆形标出，内部飞行冲突点“WG”用红线条动态突出显示。动画过程中，两架航空器将沿着各自的路径动态移动。学生们可以通过界面底部的“速度”按钮加速或者减速动态展示过程，以便详细观察航空器的移动细节及飞行冲突的发生经过，从而更好地理解如何通过调整飞行轨迹来避免冲突。

3.4　冲突调配模块实现

根据上一步得到的冲突检测结果，点击“冲突调配按钮”，系统根据相应的冲突原因进行规则匹配，最终输出的解决方案为采用目视相遇策略，并使用速度和距离模型对两机的相遇时间进行计算得到时间为 12 min，最后将所有的结果进行汇总并进行输出，如图 12 所示：两架飞机采用目视相遇策略，并且两机的相遇时间为 12 min，此时离场飞机更靠近 WG，建议两者选择低高度进行目视相遇。

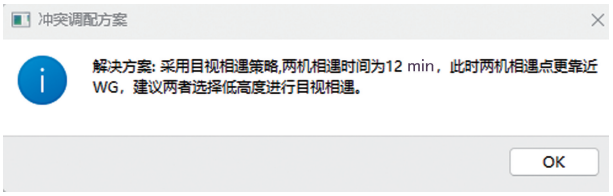


图 12　冲突调配结果展示

4　结语

本文提出的基于 Python 的程序管制冲突检测与调配辅助教学系统，通过自动检测潜在冲突、动态可视化飞行路线和生成调配方案，可以显著提升教学的互动性和实用性。系统的冲突检测模块利用算法实时分析航空器之间的位置关系并能识别冲突情况，使学生能够更好地理解冲突的成因，从而增强了学生对飞行冲突的认识。动态可视化功能通过图形界面展示航空器的实时动态，使学生能够直观地观察航班的变化及冲突演变的过程。冲突调配方案的输出为学生提供了正确答案，有助于增强学生对冲突调配的理解。最后，通过有效的冲突检测与输出冲突解决方案，该系统提升了程序管制教学过程中的直观性，降低了飞行冲突的风险，这种创新的教学辅助工具有比较重要的实际应用价值，可以辅助学生更好地应对挑战，从而提高其职业素养和冲突调配能力。

参考文献

[1]　刘林. 空中交通现代化安全管制办法介绍及分析[J]. 电子测试，2016(19)：124-169.

[2] 尚玉磊. 程序管制模拟机教学建议探讨[J]. 科技视界，2014(22)：347.
[3] 廖勇，方娜. 程序管制教学仿真系统设计与实现[J]. 科技和产业，2022，22(2)：264-269.
[4] 王思明. 新时期“程序管制”模拟机教学工作[J]. 湖北农机化，2020(16)：89-90.
[5] 白鹏. 机场塔台和程序管制模拟实践教学课程改革与创新创业研究[J]. 当代教育实践与教学研究，2017(9)：142-144.
[6] 程韬. 基于 CDIO 教育理念的程序管制课程联动性教学研究[J]. 中国民航飞行学院学报，2016，27(6)：73-75.
[7] 周建，王同乐，刘昕，等. 基于微格教学法的管制教员培训模式研究[J]. 中国民航飞行学院学报，2015，26(1)：73-77.
[8] 卞晓峰. 夯实管制培训基础促进培训中心发展[J]. 民航管理，2014(1)：56-58.
[9] 潘卫军，况金宏，王文博，等. 程序管制下考虑高度层穿越的航路容量评估模型[J]. 科学技术与工程，2015，15(1)：320-324.
[10] 杜实，王磊. 空管程序管制模拟培训效度的评估研究[J]. 中国民航飞行学院学报，2011，22(6)：30-33.
[11] 谭斌，王婷. YOLOv5 与视差计算算法的目标检测与测距系统设计[J]. 科学技术与工程，2024，24(21)：9015-9024.
[12] 孙振林，柳飞，陶水忠，等. 基于 Python 的房屋安全健康监测数据处理与预测分析[J]. 科学技术与工程，2024，24(6)：2469-2479.
[13] 刘宝林，莫海峰，冯磊，等. 规则引擎驱动的配电网问题诊断[J]. 电力系统及其自动化学报，2023，35(7)：151-158.
[14] 刘秀丽. 基于 Python 语言的好友管理系统的设计[J]. 现代信息科技，2022，6(15)：6-10.

Program Control Conflict Detection and Deployment of Auxiliary Teaching System Design and Development Based on Python：Taking Wuhan Nanhu Airport Approach Airspace as an Example

LIAO Yong，ZHAO Shichang

(School of Air Traffic Management，China Civil Aviation Flight University，Guanghan 618307，Sichuan，China)

Abstract：As a backup means of radar control，procedural control plays an important role in case of radar failure. It is very important for controllers to master procedural control skills. To this end，a Python-based procedural control conflict detection and deployment auxiliary teaching system was designed and implemented. Firstly，a seven-layer architecture is designed，covering the user layer，presentation layer，business layer，model layer，data layer，operating system layer and hardware layer. Then，five core modules are constructed：interactive interface module，data preprocessing module，conflict detection module，dynamic conflict display module and conflict deployment module，and the functions and implementation technologies of each module are designed in detail. Finally，taking the actual teaching scenario of Wuhan Nanhu Airport approach control airspace as an example，Python was used to simulate and verify the system. Students can use the system to automatically detect conflict types and propose conflict solutions，so as to better master conflict deployment skills.

Keywords：program control；conflict detection；conflict resolution；teaching system