MS001-A.03.001DD.1.0

MS-001

规划软件详细设计说明书

编制/日期：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

审核/日期：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

批准/日期：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

杭州三坛医疗科技有限公司

文档更改履历

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 发布/实施日期 | 更改内容概述 | 更改者 |
| V1.0 |  | 文件新编 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**目录**

[1. 引言 1](#_Toc1374)

[1.1. 编写目的 1](#_Toc15508)

[1.2. 项目背景 1](#_Toc9729)

[1.3. 术语及缩写词 1](#_Toc7102)

[1.4. 参考资料 2](#_Toc18150)

[2. 总体设计 2](#_Toc30979)

[2.1. 需求概述 2](#_Toc32223)

[2.2. 软件结构 3](#_Toc15334)

[3. 软件注册功能设计说明 4](#_Toc368)

[3.1. 功能描述 4](#_Toc16800)

[3.2. 功能 4](#_Toc24467)

[3.3. 性能 4](#_Toc30567)

[3.4. 输入项目 4](#_Toc29528)

[3.5. 输出项目 5](#_Toc11243)

[3.6. 算法 5](#_Toc22767)

[3.7. 程序逻辑 6](#_Toc31195)

[3.8. 接口 8](#_Toc23199)

[3.9. 存储分配 9](#_Toc31597)

[3.10. 限制条件 9](#_Toc26279)

[3.11. 测试要点 10](#_Toc14137)

[4. 登录功能设计说明 10](#_Toc23858)

[4.1. 功能描述 10](#_Toc30957)

[4.2. 功能 10](#_Toc16052)

[4.3. 性能 10](#_Toc7334)

[4.4. 输入项目 11](#_Toc18031)

[4.5. 输出项目 11](#_Toc18919)

[4.6. 算法 11](#_Toc6647)

[4.7. 程序逻辑 11](#_Toc17294)

[4.8. 接口 13](#_Toc23610)

[4.9. 存储分配 14](#_Toc16671)

[4.10. 限制条件 14](#_Toc23284)

[4.11. 测试要点 14](#_Toc23387)

[5. 网络功能设计说明 15](#_Toc14881)

[5.1. 功能描述 15](#_Toc24780)

[5.2. 功能 15](#_Toc15110)

[5.3. 性能 15](#_Toc15380)

[5.4. 输入项目 15](#_Toc23140)

[5.5. 输出项目 15](#_Toc26868)

[5.6. 算法 16](#_Toc22261)

[5.7. 程序逻辑 16](#_Toc32082)

[5.8. 接口 16](#_Toc259)

[5.9. 存储分配 18](#_Toc16409)

[5.10. 限制条件 18](#_Toc740)

[5.11. 测试要点 18](#_Toc11854)

[6. 选择功能设计说明 19](#_Toc1229)

[6.1. 功能描述 19](#_Toc9241)

[6.2. 功能 19](#_Toc23596)

[6.3. 性能 19](#_Toc6271)

[6.4. 输入项目 19](#_Toc661)

[6.5. 输出项目 19](#_Toc16665)

[6.6. 算法 19](#_Toc16743)

[6.7. 程序逻辑 20](#_Toc13915)

[6.8. 接口 21](#_Toc14743)

[6.9. 存储分配 21](#_Toc32395)

[6.10. 限制条件 21](#_Toc9451)

[6.11. 测试要点 21](#_Toc28293)

[7. 数据管理功能设计说明 21](#_Toc17974)

[7.1. 功能描述 21](#_Toc8598)

[7.2. 功能 22](#_Toc10317)

[7.3. 性能 22](#_Toc18702)

[7.4. 输入项目 22](#_Toc2510)

[7.5. 输出项目 22](#_Toc22903)

[7.6. 算法 23](#_Toc1007)

[7.7. 程序逻辑 24](#_Toc2721)

[7.8. 接口 25](#_Toc2267)

[7.9. 存储分配 28](#_Toc15057)

[7.10. 限制条件 28](#_Toc26891)

[7.11. 测试要点 29](#_Toc4537)

[8. 切割复位功能设计说明 30](#_Toc16012)

[8.1. 功能描述 30](#_Toc16554)

[8.2. 功能 30](#_Toc26874)

[8.3. 性能 31](#_Toc19132)

[8.4. 输入项目 31](#_Toc8528)

[8.5. 输出项目 32](#_Toc17976)

[8.6. 算法 32](#_Toc18693)

[8.7. 程序逻辑 33](#_Toc27073)

[8.8. 接口 35](#_Toc21263)

[8.9. 存储分配 41](#_Toc8148)

[8.10. 限制条件 42](#_Toc7661)

[8.11. 测试要点 42](#_Toc15125)

[9. CT分区功能设计说明 42](#_Toc3891)

[9.1. 功能描述 42](#_Toc31305)

[9.2. 功能 43](#_Toc1059)

[9.3. 性能 44](#_Toc16628)

[9.4. 输入项目 44](#_Toc2132)

[9.5. 输出项目 44](#_Toc20966)

[9.6. 算法 44](#_Toc2698)

[9.7. 程序逻辑 45](#_Toc836)

[9.8. 接口 46](#_Toc18709)

[9.9. 存储分配 52](#_Toc24530)

[9.10. 限制条件 52](#_Toc14973)

[9.11. 测试要点 53](#_Toc1924)

[10. 规划功能设计说明 53](#_Toc7241)

[10.1. 功能描述 53](#_Toc29088)

[10.2. 功能 53](#_Toc24852)

[10.3. 性能 54](#_Toc1279)

[10.4. 输入项目 54](#_Toc16475)

[10.5. 输出项目 55](#_Toc24355)

[10.6. 算法 55](#_Toc32163)

[10.7. 程序逻辑 56](#_Toc10565)

[10.8. 接口 57](#_Toc27116)

[10.9. 存储分配 61](#_Toc4740)

[10.10. 限制条件 62](#_Toc31108)

[10.11. 测试要点 62](#_Toc219)

[11. X-Ray分区功能设计说明 63](#_Toc22973)

[11.1. 功能描述 63](#_Toc22161)

[11.2. 功能 63](#_Toc3351)

[11.3. 性能 64](#_Toc22578)

[11.4. 输入项目 64](#_Toc2512)

[11.5. 输出项目 64](#_Toc24408)

[11.6. 算法 64](#_Toc30214)

[11.7. 程序逻辑 65](#_Toc10454)

[11.8. 接口 67](#_Toc13770)

[11.9. 存储分配 70](#_Toc25294)

[11.10. 限制条件 70](#_Toc26344)

[11.11. 测试要点 70](#_Toc17052)

[12. 配准功能设计说明 71](#_Toc18600)

[12.1. 功能描述 71](#_Toc17102)

[12.2. 功能 71](#_Toc18786)

[12.3. 性能 72](#_Toc27691)

[12.4. 输入项目 72](#_Toc4018)

[12.5. 输出项目 72](#_Toc13097)

[12.6. 算法 73](#_Toc18827)

[12.7. 程序逻辑 73](#_Toc17218)

[12.8. 接口 75](#_Toc8693)

[12.9. 存储分配 77](#_Toc23303)

[12.10. 限制条件 77](#_Toc17979)

[12.11. 测试要点 77](#_Toc16275)

[13. 定位功能设计说明 78](#_Toc26659)

[13.1. 功能描述 78](#_Toc8283)

[13.2. 功能 78](#_Toc21229)

[13.3. 性能 79](#_Toc5816)

[13.4. 输入项目 79](#_Toc12606)

[13.5. 输出项目 79](#_Toc1706)

[13.6. 算法 79](#_Toc21341)

[13.7. 程序逻辑 80](#_Toc4755)

[13.8. 接口 81](#_Toc113)

[13.9. 存储分配 82](#_Toc8710)

[13.10. 限制条件 82](#_Toc29349)

[13.11. 测试要点 82](#_Toc15329)

[14. 软件信息功能设计说明 83](#_Toc23890)

[14.1. 功能描述 83](#_Toc13888)

[14.2. 功能 83](#_Toc5542)

[14.3. 性能 83](#_Toc4882)

[14.4. 输入项目 83](#_Toc31076)

[14.5. 输出项目 83](#_Toc22106)

[14.6. 算法 83](#_Toc19625)

[14.7. 程序逻辑 83](#_Toc20192)

[14.8. 接口 84](#_Toc6203)

[14.9. 存储分配 85](#_Toc2118)

[14.10. 限制条件 85](#_Toc7349)

[14.11. 测试要点 85](#_Toc12454)

[15. 配置功能设计说明 85](#_Toc5510)

[15.1. 功能描述 85](#_Toc31740)

[15.2. 功能 85](#_Toc25549)

[15.3. 性能 85](#_Toc9540)

[15.4. 输入项目 86](#_Toc23496)

[15.5. 输出项目 86](#_Toc18834)

[15.6. 算法 86](#_Toc13282)

[15.7. 程序逻辑 86](#_Toc25305)

[15.8. 接口 87](#_Toc12655)

[15.9. 存储分配 88](#_Toc945)

[15.10. 限制条件 88](#_Toc8591)

[15.11. 测试要点 89](#_Toc24056)

[16. 日志记录功能设计说明 89](#_Toc10980)

[16.1. 功能描述 89](#_Toc12573)

[16.2. 功能 89](#_Toc15655)

[16.3. 性能 89](#_Toc19414)

[16.4. 输入项目 89](#_Toc6807)

[16.5. 输出项目 90](#_Toc17496)

[16.6. 算法 90](#_Toc12732)

[16.7. 程序逻辑 90](#_Toc25486)

[16.8. 接口 90](#_Toc29257)

[16.9. 存储分配 91](#_Toc25326)

[16.10. 限制条件 92](#_Toc24113)

[16.11. 测试要点 92](#_Toc22981)

[17. 日志导出功能设计说明 92](#_Toc24607)

[17.1. 功能描述 92](#_Toc24130)

[17.2. 功能 92](#_Toc31475)

[17.3. 性能 92](#_Toc16265)

[17.4. 输入项目 92](#_Toc16886)

[17.5. 输出项目 93](#_Toc10636)

[17.6. 算法 93](#_Toc18404)

[17.7. 程序逻辑 93](#_Toc18559)

[17.8. 接口 94](#_Toc25335)

[17.9. 存储分配 95](#_Toc28963)

[17.10. 限制条件 95](#_Toc22168)

[17.11. 测试要点 95](#_Toc15422)

[18. 系统退出功能设计说明 95](#_Toc14187)

[18.1. 功能描述 95](#_Toc14072)

[18.2. 功能 95](#_Toc17604)

[18.3. 性能 95](#_Toc5878)

[18.4. 输入项目 96](#_Toc2768)

[18.5. 输出项目 96](#_Toc6243)

[18.6. 算法 96](#_Toc7509)

[18.7. 程序逻辑 96](#_Toc21548)

[18.8. 接口 97](#_Toc8483)

[18.9. 存储分配 98](#_Toc17275)

[18.10. 限制条件 98](#_Toc26539)

[18.11. 测试要点 98](#_Toc9377)

# 引言

## 编写目的

本详细设计说明书是为了规范及保证项目工作合理有序的开展，经概要设计评审后进一步细化，分别对每一功能进行详细细化包括解决方案、接口等方面的设计，明确描述所有输入输出参数、类型逻辑算法以及调用关系。作为开发人员和测试人员进一步变成和编写测试用例依据。

本项目详细设计说明书用于从具体上指导模块化手术导引系统规划软件的开发，并面向项目组全体成员。

## 项目背景

传统的骨科手术非常依赖有经验的医生，手术过程也较复杂且漫长。为了能更加精准的定位病灶，缩短手术的时间，并尽可能减轻患者的痛苦，现开发规划软件，作为模块化手术导引系统的重要组成部分，能将术前CT影像数据与C形臂X光机影像数据融合，指导外科手术。

## 术语及缩写词

* 冠状位（coronal plane或frontal plane）：为左,右方向将人体纵切为前后（腹背）两部分的断面；
* 矢状位(sagittal plane)：将躯体纵断为左右两部分的解剖平面；
* 横断位(horizontal plane 或transaxial plane)：将躯体纵断为上下两部分的解剖平面；
* AP（Anteroposterior）：正位，从胸前指向背部的方向，Anterior为前，Posterior为后，因此PA表示反方向；
* LAT（lateral）：侧位，左和右；
* 锥形投影：以点光源发射的锥形光束生成的投影图像，类似小孔成像原理；
* 平行投影：以平行光穿透组织产生的投影图像。
* DICOM：一种医疗图像格式；
* DRR：利用术前CT影像，产生的仿真锥形投影图像；
* 手术规划：基于术前CT的基础上，规划手术路径，包括设置、调整钢针的大小、长度、颜色等；
* 术中配准：匹配术前CT仿真透视图像（如DRR）与术中透视图，获取当前病灶（路径）的真实位姿；
* C臂校准：通过术前大量采集透视图像（已知布局的棋盘格）来计算C臂内参及畸变参数，用于C臂图像的校正及透视变换；
* 定位：根据软件规划通道信息，下发的定位目标位置；
* MPS：指重建速度，平均每秒能处理DICOM数据单位为MB对应的数据量。

## 参考资料

1. 《医疗器械软件注册技术审查指导原则》
2. 《医疗器械网络安全注册技术审查指导原则》
3. YY/T 0664-2008 《医疗器械软件软件生存周期过程》
4. YY/T 1406.1-2016 《医疗器械软件 第1部分：YY/T 0316 应用于医疗器械软件的指南》
5. 《MS-001技术需求说明书》
6. 《模块化手术导引系统-规划软件概要设计说明书》
7. GB/T 8567-2006《计算机软件文档编制规范》

# 总体设计

## 需求概述

模块化手术导引系统规划软件包括登录功能、模块选择功能、手术方案功能、手术外设功能、辅助设备功能、帮助功能六大功能以及软件注册子功能、登录子功能、网络子功能、选择子功能、数据管理子功能、切割复位子功能、CT分区子功能、规划子功能、X-Ray分区子功能、配准子功能、定位子功能、软件信息子功能、配置子功能、日志记录子功能、日志导出子功能、系统退出子功能。

## 软件结构

模块化手术导引系统

-

规划软件

登录功能

手术方案功能

手术外设功能

软件注册子功能

帮助功能

CT分区子功能

规划子功能

X-Ray分区子功能

配准子功能

登录子功能

数据管理子功能

模块选择功能

选择子功能

切割复位子功能

定位子功能

网络子功能

辅助设备功能

配置子功能

日志导出子功能

系统退出子功能

日志记录子功能

软件信息子功能

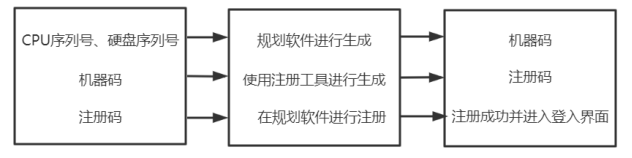
# 软件注册功能设计说明

## 功能描述

1. 为了防止软件被盗用，需要对软件进行保护；
2. 当系统软件在一台主机上安装后，会根据主机的硬件信息生成一串机器码；
3. 需要一个专用工具。针对机器码及设定的密钥，生成注册码；
4. 将注册码录入，注册成功后方能使用，此时也将注册码写入到配置文件中。

## 功能

1. 启动软件，读取配置文件中的注册码，验证是否正确；
2. 验证通过后，正常进入登录界面；
3. 首次安装等情况下，验证不成功，弹出注册对话框；其中机器码根据CPU系列号及硬盘序列号通过内部算法自动生成一串机器码；
4. 使用注册工具根据机器码，通过内部算法对应的注册码，将注册码输入注册界面进行注册，注册成功后将注册码写入配置文件中并进入登录界面。



软件注册功能功能IPO图

## 性能

无。

## 输入项目

1.硬件信息：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| 1 | CPU序列号 | cpuId | QByteArray | 16~32位 | 自动获取 | / |
| 2 | 硬盘序列号 | diskId | QByteArray | 8~32位 | 自动获取 | / |

2.机器码：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| 1 | 机器码 | systemCode | QByteArray | 32位 | 手动输入 | / |

## 输出项目

1.机器码：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输出形式** | **备注** |
| 1 | 机器码 | systemCode | QByteArray | 32位 | 字符串 | / |

2.注册码：

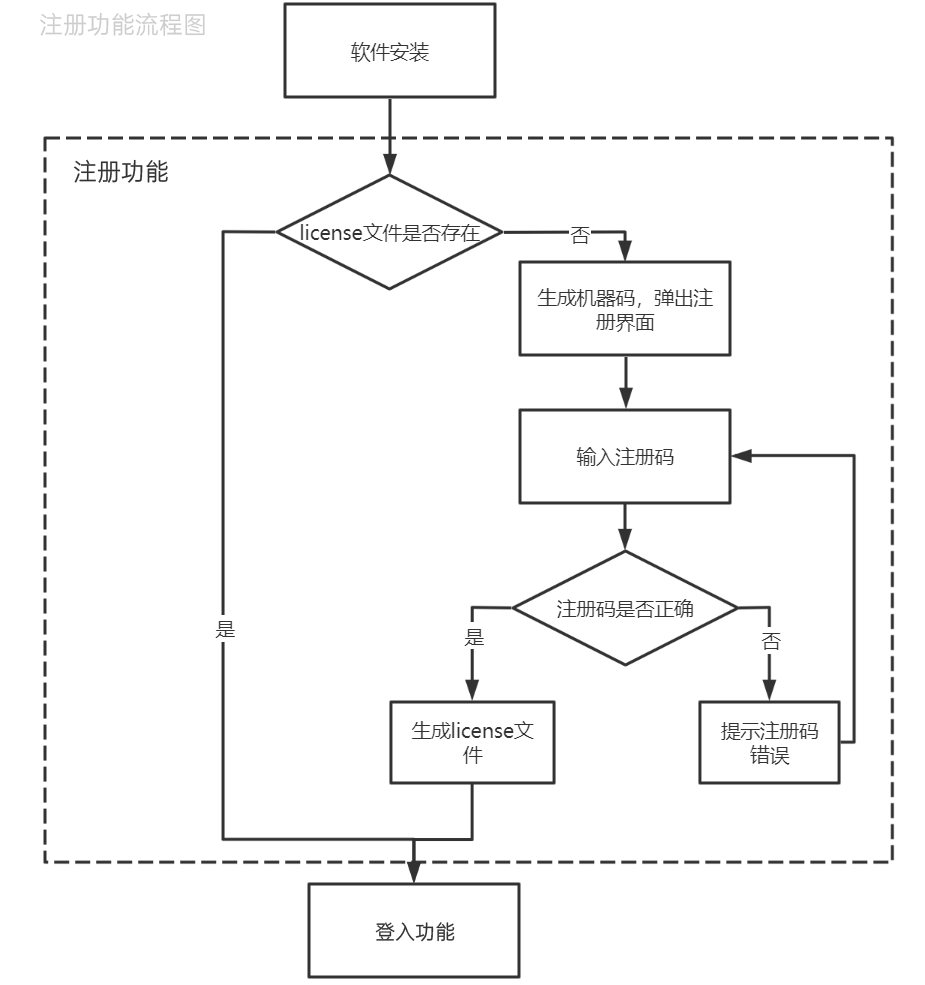
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输出形式** | **备注** |
| 1 | 注册码 | licensegenerated | QByteArray | 32位 | 字符串 | / |

## 算法

3.6.1. 注册算法  
1.算法目的：  
为了防止软件被盗用，需要对软件进行保护。  
2.算法流程：  
1）规划软件启动后会检查配置文件license.ini中是否有32位的注册码，如果不存或注册码无效则被判断为未注册。  
2）如果规划软件未注册，则程序会取出本机cpu序列号和硬盘序列号拼接，然后经过MD5函数生成MD5码即机器码提供给用户，同时生成随机密钥。  
3）在注册工具中输入机器码，注册工具会将机器码和密钥字符串经过BKDR Hash生成32位的注册码提供给用户。  
4）将注册码输入到软件后，软件程序也会将机器码和密钥字符串经过BKDR Hash生成32位的注册码，然后与用户输入的注册码进行比对，如果相同，则注册成功。  
5）软件注册成功，程序同时也会将注册码写入到license.ini，这样下次登入程序时，执行步骤 1）判断为已注册，程序直接进入登入界面。

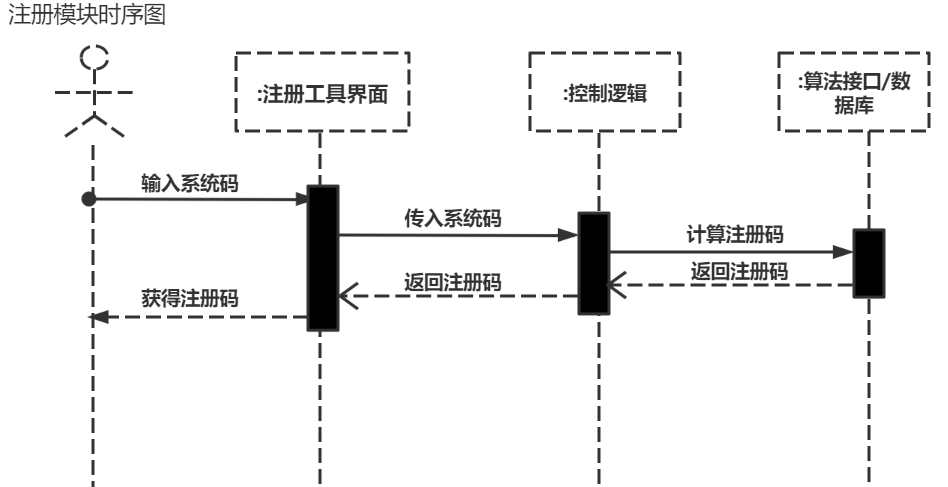
## 程序逻辑

使用流程图来展示：

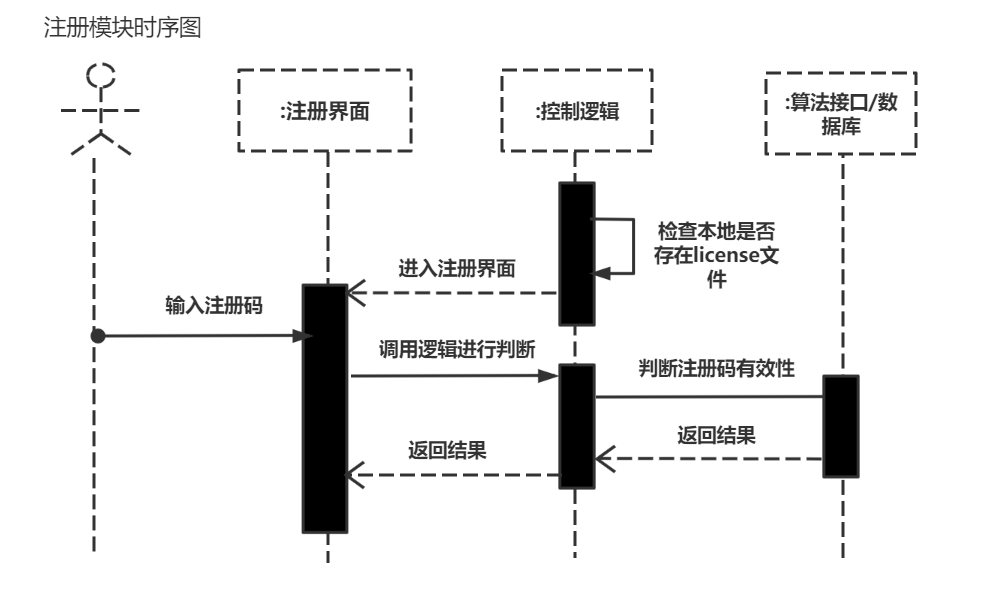


注册功能在软件内的流程图

使用时序图来展示：



注册功能获取注册码时序图



注册功能使用注册码注册时序图

## 接口

注册功能包含SystemInfo、SoftLicense、Register三个主要类。其中重要接口详细说明如下：

**3.8.1. SystemInfo**

/\*\*

\* @brief The SystemInfo class

\*

/\*\*

/\*\*

\* @brief diskId

\* 获取硬盘序列号

\*/

void diskId()

/\*\*

\* @brief cpuId

\* 获取CPU序列号

\*/

void cpuId()

**3.8.2. SoftLicense**

/\*\*

\* @brief The SoftLicense class

\*

/\*\*

/\*\*

\* @brief systemCode

\* 是否生成机器码测试

\*/

void systemCode()

/\*\*

\* @brief isValid

\* 判断注册码的有效性测试

\*/

void isValid()

/\*\*

\* @brief generateLicense

\* 是否生成注册码测试

\*/

void generateLicense()

**3.8.3. Register**

/\*\*

\* @brief The Register class

\*

/\*\*

/\*\*

\* @brief registerLicense

\* 是否生成机器码测试

\*/

void registerLicense()

/\*\*

\* @brief cancelClicked

\* 点击取消注册测试

\*/

void cancelClicked()

/\*\*

\* @brief shutdownClicked

\* 点击关机测试

\*/

void shutdownClicked()

## 存储分配

注册功能本地文件存储内容：保存在当前软件根目录下“./ license.ini”文件。

## 限制条件

无。

## 测试要点

1. 硬件正常，机器码可以生成；
2. 硬件去除，机器码生成失败；
3. 注册码正确，注册成功；
4. 注册码错误，注册失败；
5. 注册码格式正确，非本机生成，注册失败。

# 登录功能设计说明

## 功能描述

1. 登录功能是用户进入系统的唯一入口，保证了软件的安全；登录功能布局设计、风格合理；
2. 登录功能在软件中属于常驻内存、不可重入，属于顺序处理且代码没有覆盖要求。本功能与其他功能耦合较小：登录成功可进入数据管理功能；
3. 用户信息以特定的内部加密方式保存在数据库中，防止数据泄露。
4. 用户登录进入软件之后30分钟内不操作会自动进入锁屏界面，解锁需要输入账户对应密码，保证软件具有网络安全的自动注销能力。
5. 不同账户拥有不同权限，保证软件网络安全的授权能力。
6. 用户拥有唯一账户，需要身份验证才能登陆，保证了软件网络安全的人员鉴别能力。

## 功能

1. 系统启动后自动开启软件，登录功能包含用户登录接口：输入正确的账号密码后可以成功离开登录功能进入数据管理功能；
2. 账号密码错误时，软件提示对应消息；
3. 退出登录则系统自动关机。

## 性能

无。

## 输入项目

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | 账号 | user\_id | varchar | 满足规定正则 | 手动输入 | 无 |
| **2** | 密码 | user\_pwd | varchar | 满足规定正则 | 手动输入 | 无 |

## 输出项目

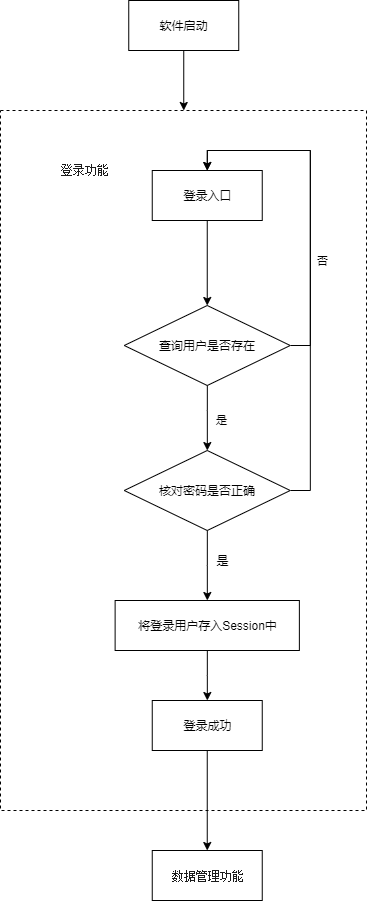
数据库中读取的数据或错误警告信息。

## 算法

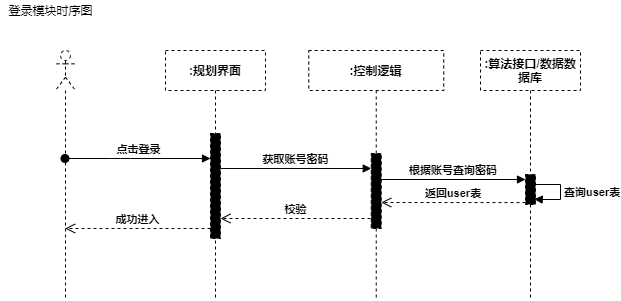
无。

## 程序逻辑

登录功能独立于其他功能，随软件启动而启动。



登录功能流程图



登录功能时序图

## 接口

登录功能主要包含LoginWidget。其中重要接口详细说明如下：

/\*\*

\* @brief The LoginWidget class

\* 登录功能ui和逻辑

\*/

*class* LoginWidget：

/\*\*

\* @brief on\_pButton\_login\_clicked

\*/

void on\_pButton\_login\_clicked();

/\*\*

\* @brief isValidLicense 软件是否注册

\* @return

\*/

bool isValidLicense();

/\*\*

\* @brief initLoginUI 登录界面初始化

\*/

void initLoginUI();

/\*\*

\* @brief verifyCrc 验证Crc是否相等

\* @param crc 输入1

\* @param crc1 输入2

\*/

void verifyCrc(uint32\_t crc, uint32\_t crc1);

/\*\*

\* @brief closeEvent 关闭登录界面

\* @param event

\*/

void closeEvent(QCloseEvent \*event);

/\*\*

\* @brief keyPressEvent 登录界面键盘相应

\* @param event

\*/

void keyPressEvent(QKeyEvent \*event);

/\*\*

\* @brief shutdownClicked 关机

\*/

void shutdownClicked();

## 存储分配

登录功能信息存在本地数据库中：软件所在目录下“./dbvrs”文件。

## 限制条件

1. 账号格式必须是标准字符串且长度1-19位；
2. 密码格式必须是标准字符串且长度6-19位。

## 测试要点

1. 用户名密码正确，登录；
2. 用户名正确，密码错误；
3. 用户名不存在；
4. 用户名1-19位有效字符验证，无效字符长度验证；
5. 密码6-19位有效字符验证，无效字符长度验证。

# 网络功能设计说明

## 功能描述

1. 网络通讯功能是本软件重要组成部分，主要用于和导引软件通讯。网络通讯功能基于自定义的协议实现了一套内部的客户端；
2. 网络通讯功能在软件中属于常驻内存、不可重入，不属于顺序处理且代码没有覆盖要求。本功能与其他功能耦合较小：收到导引软件的正侧位数据和将数据解码并发送给X-Ray分区功能，将定位信息加密发送给导引软件；
3. 网络通讯功能遵循内部加密协议，保证了软件网络安全的传输保密性和完整性能力。

## 功能

网络通讯功能独立于软件顺序流程，登录成功后客户端后台静默运行。

网络通讯功能详细功能：

1. 支持本机客户都开放端口实时调整；
2. 软件成功登录后，网络通讯功能自动启动，并在主界面标题栏增加链接状态提示；
3. 支持收到导引软件的影像信息（内部协议）；
4. 支持发送规划螺钉信息（内部协议）到导引软件。

## 性能

1. 收到并解析数据响应时间在1s内；
2. 加密并发送数据响应时间在1s内。

## 输入项目

网络通讯功能输入包括客户端端口参数和规划螺钉信息。 相应内容详见定位功能和配置功能具体说明。

## 输出项目

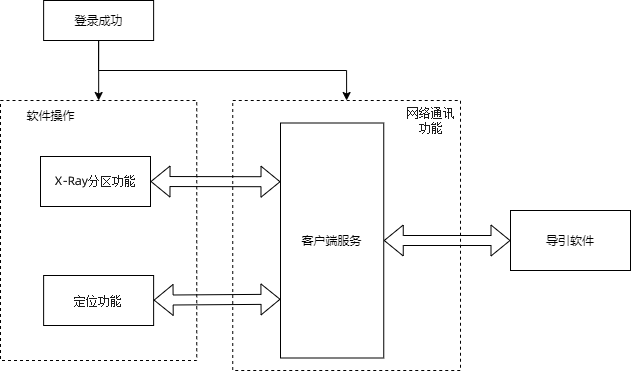
网络通讯功能输出包括X-Ray影像和影像对应相机信息以及影像间空间信息。详细内容详见X-Ray分区功能和配准功能。

## 算法

无。

## 程序逻辑

软件登录成功后网络通讯功能自动启动，根据配置在后台静默运行。



网络通讯功能流程图

## 接口

网络通讯功能主要包含ClientSocket，其中重要接口详细说明如下：

/\*\*

\* @brief The ClientSocket class

\* 网络功能管理类

\*/

*class* ClientSocket; :

/\*\*

\* @brief disconn 断开链接

\* @return

\*/

int disconn();

/\*\*

\* @brief connected

\* 保持链接

\*/

void connected();

/\*\*

\* @brief received

\* 收到数据

\*/

void received();

/\*\*

\* @brief recvTcpRead

\* 解析tcp包头

\*/

void recvTcpRead();

/\*\*

\* @brief recvTcpData

\* 解析tcp数据

\*/

void recvTcpData();

/\*\*

\* @brief onDisconn

\* 断开链接

\*/

void onDisconn();

/\*\*

\* @brief init 功能初始化

\* @param hostname

\* @param port

\*/

void init(*const* QString& hostname, *const* quint16& port);

/\*\*

\* @brief sendData 发送数据

\* @param arr

\*/

void sendData(QByteArray\* arr);

/\*\*

\* @brief receivedData 收到数据

\* @param byteArray

\*/

void receivedData(*const* QByteArray& byteArray);

/\*\*

\* @brief connectState 链接状态

\* @param state

\*/

void connectState(*const* int& state);

/\*\*

\* @brief sendDataResult 发送收到结果

\* @param result

\*/

void sendDataResult(*const* bool& result);

## 存储分配

网络通讯功能无本地存储，收到数据解密成功会发数据发送给X-Ray分区功能处理、解密失败将直接丢弃。

## 限制条件

1. 开放端口不能被占用；
2. 与导引软件必须处于同一网段下；
3. 发送和收到的数据必须满足内部加密要求。

## 测试要点

1. 网络通讯长时间连接稳定性测试；
2. 网络断开自动重连测试；
3. Ip与导引软件不在同一网段测试；
4. 端口被占用后，无法正常通讯测试；
5. 传输数据加密测试。

# 选择功能设计说明

## 功能描述

1. 作为功能化导引系统重要组成，功能选择是其主要体现。凭账号、密码成功登录界面后进入该功能。机械臂作为基础功能是必选项，其他功能作为扩展功能预留，目前置灰，不可操作；
2. 本功能是常驻内存、不可重入，属于顺序处理方式。

## 功能

1. 机械臂作为基础功能，必选；默认选中显示；
2. 其他功能预留，目前置灰显示，不可选；
3. 下一步操作，进入下一数据管理功能。

## 性能

无。

## 输入项目

无。

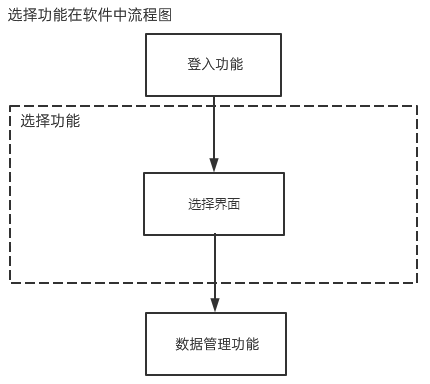
## 输出项目

无。

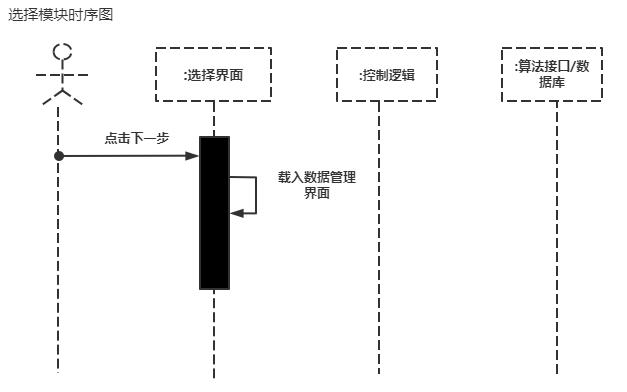
## 算法

无。

## 程序逻辑



选择功能流程图



选择功能时序图

## 接口

选择功能主要类为OptionWidget。其中重要接口详细说明如下：

**6.8.1. OptionWidget**

/\*\*

\*@brief The OptionWidget class

\*选择功能主窗口

/\*\*

/\*\*

\*@brief initUI 初始UI界面

\*/

void initUI();

## 存储分配

无。

## 限制条件

1. 机械臂作为基础功能，必选。

## 测试要点

1. 机械臂默认被选中测试；
2. 选择其他功能无效测试；
3. 选择机械臂，下一步按钮测试；
4. 机械臂功能无法取消选中测试。

# 数据管理功能设计说明

## 功能描述

1. 该功能是获得患者数据资料信息的来源；
2. CT数据以及规划数据只能从U盘导入；
3. 数据列表字段显示分别为：序号、CT号、姓名、性别、年龄、导入时间；
4. 多序列缩略图列表显示内容：序列缩略图、姓名、出生日期、扫描时间、序列、层数；
5. 功能内部代码增加边缘性检测，详见接口部分。

## 功能

1. 进入该功能，底下任务栏对应功能字体高亮显示；
2. 点击导入按钮，识别U盘，确保网络安全的健康数据的完整性与真实性的能力并加载显示数据列表；
3. 可选中只导入CT数据；或选中导入CT数据及规划数据；或选中导入规划数据；
4. 导入的CT数据会根据CT号在数据库里查重。有重复则弹出框提示；
5. 数据列表保留最近200条，按导入时间降序排列；
6. 数据列表中选中行，将被高亮显示；
7. 被选中行双击，或者点击加载按钮，可对数据实现加载；
8. 搜索框内输入内容，将激活显示清除和搜索按钮；
9. 根据输入内容，匹配CT号和姓名模糊查询，并将查询结果显示在列表中；
10. 点击清除按钮则清除输入内容，隐藏清除、搜索按钮；并将数据列表还原；
11. 导入时进度条显示导入过程，完成后进度条隐藏；
12. 导入完成，自动填充显示多序列缩略图，且默认选中第一列；
13. 2D断层预览图显示被选中的CT序列，默认中间位置的断层片；
14. 序列缩略图列表中存在多序列的可以选择切换，对应2D断层预览图同步切换；
15. 2D断层预览图可以通过鼠标滚轮切换显示，也可以拖动滚动条切换显示；
16. 2D断层预览图可以通过鼠标右键实现窗宽/窗位调节；
17. 2D断层预览图可以通过鼠标中键实现平移；
18. 2D断层预览图可以通过鼠标左键+ctrl键实现旋转；
19. 2D断层预览图可以通过鼠标右键+ctrl键实现缩放；
20. 点上一步按钮，回到上一选择功能；
21. 点下一步按钮，首次导入的数据将被写入数据库，同时更新到数据列表首行，保证软件网络安全的数据备份与灾难恢复能力；并进入切割复位功能。

## 性能

1. 数据列表查询检索返回在1s内。

## 输入项目

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| 1 | ct序列 | files | 文件 | 符合DICOM3.0格式 | 弹窗选择 | 选择病例 |
| 2 | 规划数据 | files | 文件 | 符合软件内部定义规范 | 弹窗选择 | 选择已规划数据 |

## 输出项目

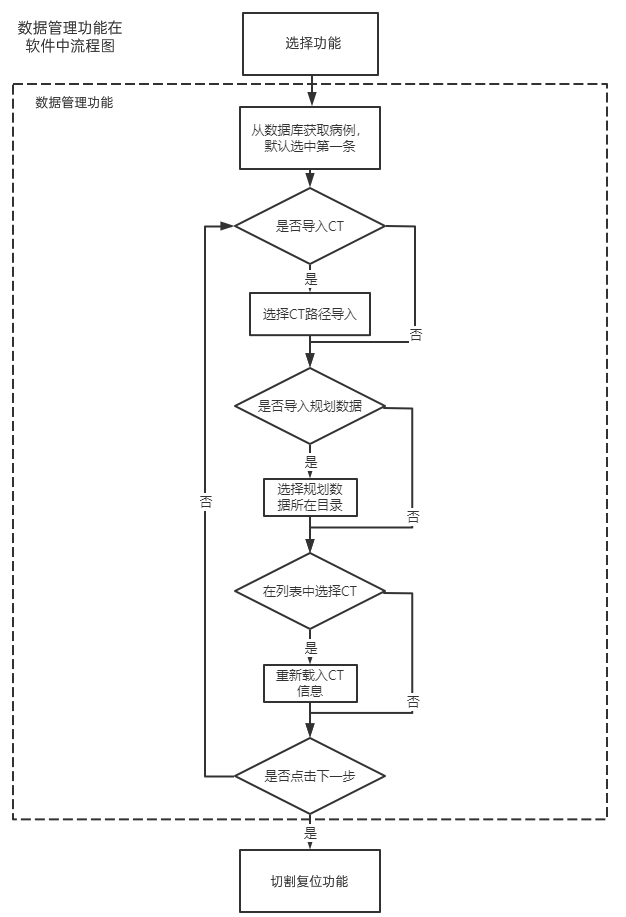
a. 患者信息

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | **标识** | **数据类型** | **有效范围** |
| 姓名 | Name | wchar\_t | 1~19宽字符 |
| 出生日期 | Birthday | char | 20字符 |
| CT序列编号 | CT\_Series | int | 0~100 |
| CT创建时间 | CT\_ Date | char | 20字符 |
| CT对应序列的切片数 | CT\_Slices | int | 1~5000 |

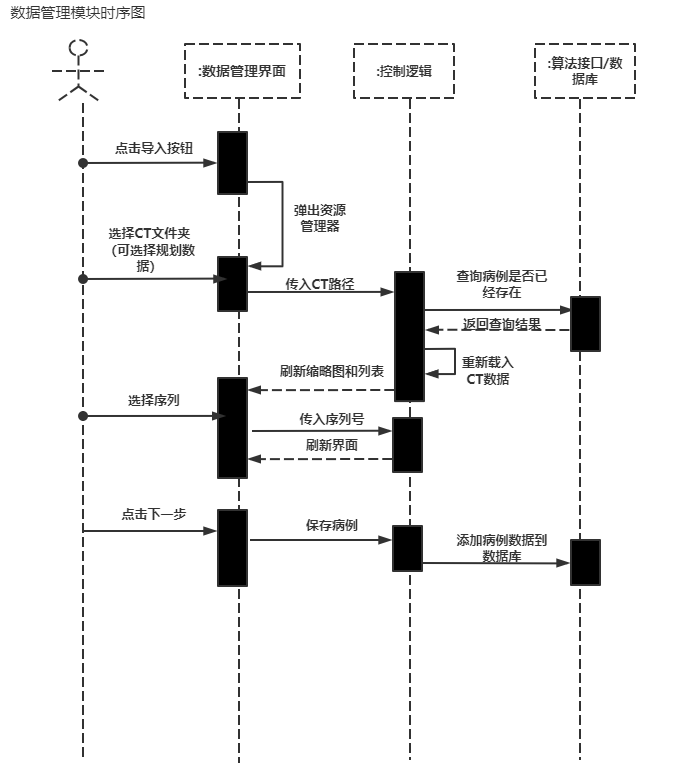
## 算法

无。

## 程序逻辑



数据管理功能流程图



数据管理功能时序图

## 接口

数据管理功能包含ImportPreviewWidget、CtManagerWidget、DicomBackgroundImporter、SeriesView四个主要类。其中重要接口详细说明如下：

**7.8.1. ImportPreviewWidget**

/\*\*

\*@brief The ImportPreviewWidget class

\*数据管理主窗口

/\*\*

/\*\*

\*@brief initPatientList 初始病人列表

\*/

void initPatientList();

/\*\*

\*@brief updatePatientList 初始病人列表

\*/

void updatePatientList();

/\*\*

\*@brief updatePatientList 更新病人列表

\*/

void updatePatientList();

/\*\*

\*@brief StartImport 开始导入数据

\*@param path CT数据路径

\*/

void StartImport(const QString& path = QString());

**7.8.2. CtManagerWidget**

/\*\*

\*@brief The CtManagerWidget class

\*数据列表信息窗口

/\*\*

/\*\*

\*@brief initUI 初始UI界面

\*/

void initUI();

/\*\*

\*@brief initPatientList 初始病人列表

\*/

void initPatientList();

/\*\*

\*@brief updatePatientList 更新病人列表

\*/

void updatePatientList();

/\*\*

\*@brief receiveSearchClicked 收到查询按钮点击

\*@param text 查询内容

\*/

void receiveSearchClicked(const QString& text);

/\*\*

\*@brief receiveSearchClicked 收到清除按钮点击

\*/

void receiveClearClicked();

**7.8.3. DicomBackgroundImporter**

/\*\*

\*@brief The DicomBackgroundImporter class

\*DICOM数据导入

/\*\*

/\*\*

\*@brief GetSeries 获取所有CT序列信息

\*@return 所有CT序列信息

\*初始病人列表

\*/

std::vector<SeriesInfo> GetSeries();

/\*\*

\*@brief GetPreview 获取当前预览数据

\*@param i 序列号

\*@return 当前预览数据

\*/

PreviewData GetPreview(int i);

/\*\*

\*@brief GetLoadInformation获取加载信息

\*@return 加载信息

\*/

QString GetLoadInformation();

**7.8.4. SeriesView**

/\*\*

\*@brief The SeriesView class

\*多序列显示列表

/\*\*

/\*\*

\*@brief clearData 清空列表数据

\*/

void clearData();

/\*\*

\*@brief ReadingFinished 读取CT数据结束

\*@param infoList 序列信息列表

\*/

void ReadingFinished(const std::vector<SeriesInfo> &infoList);

/\*\*

\*@brief ReadingFinished 读取CT数据结束

\*@param lpSeries\_sn 切片SN号

\*@param lpSlices\_num 切片序列号

\*@param lpSeriesId 系列ID号

\*/

void SetDefaultSelectItem(int \*lpSeries\_sn, int \*lpSlices\_num, QString \*lpSeriesId);

## 存储分配

1. 存储病人信息队列： std::deque<PATIENT\_INFO> m\_dequePatientInfo；
2. 存储查询得到的病人信息队列： std::deque<PATIENT\_INFO> m\_dequeSearchPatientInfo。

## 限制条件

1. 只导入规划数据时，必须是对应的CT数据已经被成功导入；
2. 进入下一功能，必须选取CT序列切片数≥3；
3. 查询时根据姓名与CT号模糊检索。
4. 导入CT前检测CT图像清晰度。

## 测试要点

1. 进入该功能，底下任务栏对应功能字体是否高亮显示；
2. 点击导入按钮，是否正常识别U盘；
3. 点击导入按钮，U盘是否正常加载显示数据列表；
4. 选中只导入CT数据是否正常；
5. 选中导入CT数据及规划数据是否正常；
6. 选中导入规划数据是否正常；
7. 导入重复的CT，是否有弹出框提示；
8. 数据列表是否保留最近200条数据；
9. 数据列表是否按导入时间降序排列；
10. 数据列表中选中行，是否被高亮显示；
11. 双击选中行，数据是否正常加载；
12. 选中行，点击加载按钮，数据是否正常加载；
13. 搜索框内输入内容，是否激活显示清除和搜索按钮；
14. 点搜索按钮，是否将查询结果显示在列表中；
15. 点击清除按钮是否清除输入内容；
16. 点击清除按钮是否隐藏清除、搜索按钮；
17. 点击清除按钮是否将数据列表还原；
18. 导入过程是否进度条显示；
19. 导入完成进度条是否隐藏；
20. 导入完成，序列缩略图是否填充，且默认选中第一列；
21. 导入完成，患者信息是否正常读取显示；
22. 对应2D断层预览图是否正常显示；
23. 序列缩略图列表中存在多序列选择切换是否正常，对应2D断层预览图是否同步切换；
24. 2D断层预览图是否可以通过鼠标滚轮切换显示，拖动滚动条是否正常切换显示；
25. 2D图鼠标键交互测试，左键图层查看，中键图层切换，中键长按平移，右键调整窗宽/窗位，ctrl+右键放大缩小；
26. 3D图鼠标键交互测试，左键旋转，滚轮放大缩小，中键长按平移，右键调整窗宽/窗位，ctrl+右键放大缩小；
27. 点上一步按钮，是否回到上一选择功能；
28. 点下一步按钮，首次导入的数据将被写入数据库，同时是否更新到数据列表首行；
29. 点下一步按钮，是否正常进入切割复位功能。

# 切割复位功能设计说明

## 功能描述

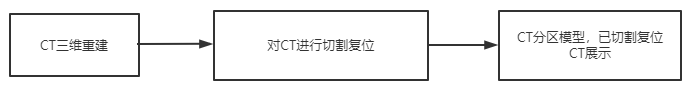
1. 切割复位功能主要分为切割和复位两大功能。切割功能主要是为了去除CT中的干扰，将CT切割为n（n≤10）个部分为复位提供基础。复位功能就是为骨骼（CT）做虚拟复位，将n（n≤10）个骨块通过手动移动位置重新拼接实现复位效果，然后将复位后的CT保存提供给其它功能（如规划功能）使用；
2. 切割复位功能在软件中属于常驻内存、不可重入，属于顺序处理且代码没有覆盖要求。本功能与其他功能耦合较小：利用全局静态指针获取病例功能三维重建的数据；通过写出到本地文件的方式将自身切割或复位后的结果提供给CT分区功能；
3. 功能内部代码增加边缘性检测，详见接口部分。

## 功能

软件处于数据管理功能且任务栏中点击【下一步】即可进入切割复位功能，或软件处于CT分区功能且在任务栏中点击【上一步】即可返回切割复位功能。

切割复位功能包含CT正位视图显示区、CT侧位视图显示区、骨块列表区、切割复位功能操作区、微调操作区。切割复位功能详细功能：

1. CT正位视图显示区显示CT数据的正位视图；
2. CT侧位视图显示区显示CT数据的侧位视图；
3. CT正侧位视图可以调节CT的窗宽/窗位、重置窗宽/窗位；
4. 切割骨块列表涵盖骨块的序号、骨块名称、是否选择复位的复选框，其中序号按顺序自动递增，骨块默认名由“骨块\_”加切割序号组成，列表支持对骨块重命名，选择复选框用于选择复位的骨块；
5. 进入界面后默认切换到切割功能，列表区默认切换到切割骨块列表。正侧位CT视图显示区显示在数据管理功能选择的CT（三维重建后的）；
6. 点击切割按钮激活切割状态在视图显示区进行切割区域画定，然后在弹出框中选择切除内部/外部，显示区自动更新为切割后的骨块（CT），点击保存按钮将保存骨块；
7. 点击重置按钮，CT显示区的骨块将会被重置到原CT。在切割骨块列表中选中骨块CT显示区会加载对应骨块的CT进行显示；
8. 点击镜像按钮将会对当前骨块(CT)进行水平镜像操作，正侧位CT显示区会更新为镜像后的骨块CT；
9. 在切割骨块列表中勾选（在对应的复选框里勾选）将要进行复位的骨块，然后点击“复位”，切割功能将切换为复位功能，原切割骨块列表将切换为复位骨块列表；
10. 复位骨块列表包含序号，骨块名称，是否显示/隐藏骨块复选框，列表支持多选，选中高亮。正侧位CT视图显示区显示复位效果图；
11. 在复位骨块列表选中多个骨块然后点击微调按钮，骨块将一起进行位置调整，默认情况下复位骨块列表里显示/隐藏复选框全部为选中状态即显示状态，勾选掉即为隐藏状态对应骨块将不显示在CT视图显示区里；
12. 软件处于切割复位功能，在任务栏中点击【上一步】，会自动放弃当前操作数据，软件进入数据管理功能；
13. 软件处于切割复位功能，在任务栏中点击【下一步】，会自动保存当前界面状态、更新CT数据，软件进入CT分区功能。



切割复位功能功能IPO图

## 性能

1. 三维数据重建速度≥15MPS；
2. 切割数量≤10。

## 输入项目

切割复位功能-全功能输入项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | CT重构结果 | 无 | VolumeObject | 不为空 | 自动输入 | 病例载入功能重构结果 |

切割复位功能-切割骨块列表相关输入项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | 选择骨块 | 无 | int | 0-9 | 鼠标选择 | 修改椎体部位名称 |
| **2** | 骨块名称 | 无 | string | 0-9个字符 | 键盘输入 | 修改椎体标记 |

切割复位功能-复位骨块列表相关输入项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | 选择骨块 | 无 | Std::vector<int> | Length<=10 | 鼠标选择 | 选中骨块调整位置 |

## 输出项目

切割复位功能-全功能输出项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输出方式** | **备注** |
| **1** | 界面信息 | 无 | xml | 不为空 | 自动保存 | 界面上的信息，如窗宽/窗位等 |
| **2** | CT切割结果 | 无 | mhd | 不为空 | 点击保存按钮 | CT分区空间结果 |
| **3** | CT复位结果 | 无 | mhd | 不为空 | 自动保存 | 点击下一步自动保存CT复位结果 |

## 算法

**8.6.1. CT切割算法**

算法目的：

将CT中感兴趣区域单独保存为一个CT。

算法流程：

1. 在空间中将目标CT调整好位置；
2. 在渲染窗口画定区域，保存区域边缘在窗口坐标系下的坐标；
3. 将画定区域换算成世界坐标系下的坐标。然后根据焦距计算投影到CT模型上的点的坐标，这些点坐标的集合即为目标切割区域；
4. 遍历CT上的体素，将在投影区域范围内的体素擦除；
5. 将遍历后的CT即为切割结果。

**8.6.2. CT复位算法**

算法目的：

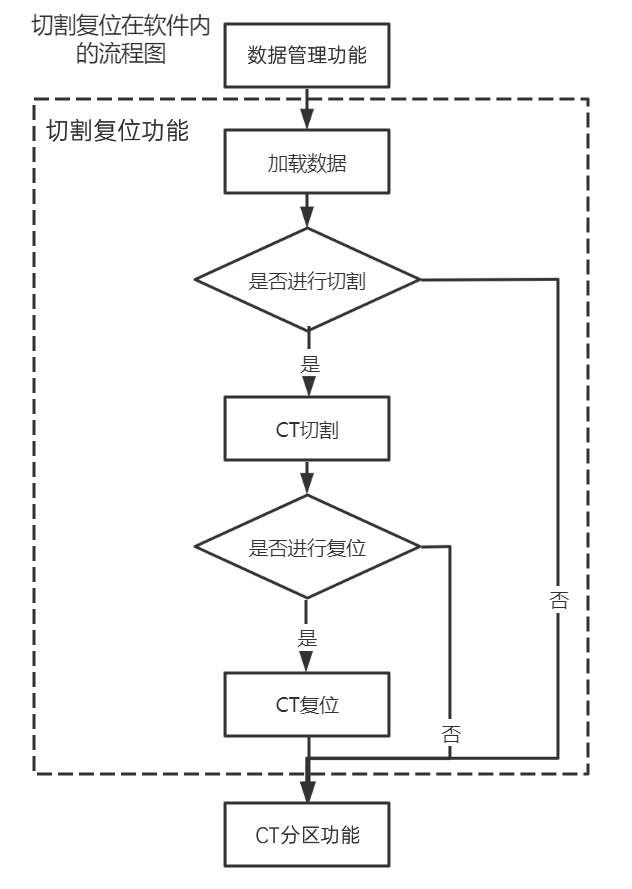
将多个已经切割好的骨块CT，调整好位置以后合并成一个CT即复位结果。

算法流程：

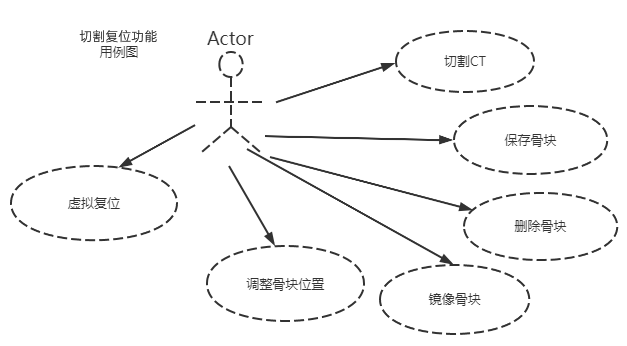
1. 选中骨块在空间中调整好位置；
2. 计算出所有骨块在空间中的最小包围区域并生成同等大小的CT；
3. 将所有骨块中的可视像素点对应填充到复位结果的CT中；
4. 将填充好的复位结果CT保存成mhd格式。

## 程序逻辑

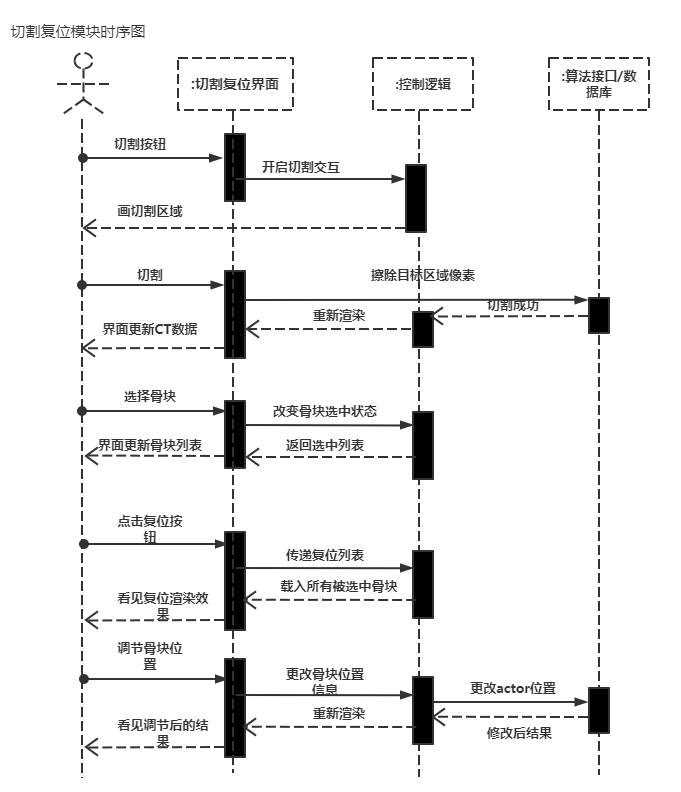
切割复位功能是本软件顺序流程的一个步骤，上一步为数据管理功能，下一步为CT分区功能。分别用切割复位功能在软件内的流程图、CT分区功能系统用例图、CT分区功能时序图来展示本功能的程序逻辑。



切割复位功能在软件内的流程图



CT切割复位功能用例图



切割复位功能时序图

## 接口

切割复位块包含CutRepositionCtWidget、CuttingRenderWidget、两个主要类。其中重要接口详细说明如下：

**8.8.1. CutRepositionCtWidget**

/\*\*

\* @brief The CutRepositionCtWidget class

\* 虚拟复位分区功能主窗口

/\*\*

/\*\*

\* @brief initWidget 界面初始化

\* @param seriesId CT的序列id

\*/

void initWidget(*const* std::string &seriesId)

/\*\*

\* @brief resetModel

\* 重置ct模型

\*/

void resetModel()

/\*\*

\* @brief saveModel

\* 保存ct模型

\*/

void saveModel()

/\*\*

\* @brief deleteModel

\* 删除ct模型

\*/

void deleteModel()

/\*\*

\* @brief virtualReset

\* 虚拟复位

\*/

void virtualReset()

/\*\*

\* @brief switch2Cutting

\* 切换切割状态

\*/

void switch2Cutting()

/\*\*

\* @brief addNewItem2List

\* 添加新item到切割列表

\*/

void addNewItem2List()

/\*\*

\* @brief rename骨块重命名

\* @param name输入名称

\*/

void rename(QString &name)

/\*\*

\* @brief addItem2List2 添加item到虚拟复位列表

\* @param namelist

\*/

void addItem2List2(QStringList &namelist)

/\*\*

\* @brief getNameListChecked

\* 获取所有checkbox被选中的item的名字

\*/

QStringList getNameListChecked()

///////////////////////////////////////

/\*\*

\* @brief getSelectedItems获取所有被选中的虚拟复位item

\* @param v 由被选中的item的index组成的vector

\* \*/

void getSelectedItems(std::vector<int> \*v)

/\*\*

\* @brief selectLastItem 选中列表最后一项

\* @param currentIndex 当前被选中的item的index

\*/

void selectLastItem(int currentIndex)

/\*\*

\* @brief updateName

\* 更新骨块名

\*/

void updateName(QString &file, short \*data)

/\*\*

\* @brief addMirrorItem2List

\* 添加镜像item到列表

\*/

void addMirrorItem2List()

/\*\*

\* @brief saveData

\* 保存数据

\*/

void saveData()

/\*\*

\* @brief selectItem

\* 选中item

\*/

void selectItem(int currentIndex)

/\*\*

\* @brief resetCuttingStatus

\* 重置切割标志位

\*/

void resetCuttingStatus()

/\*\*

\* @brief readCutResult读取切割结果

\* @param path 读取路径

\* @param itkImage 读取结果

\*/

void readCutResult(QString &path, itk::Image<short, 3>::Pointer &itkImage)

/\*\*

\* @brief isFileExists判断文件是否存在

\* @param fileName文件路径

\*/

bool isFileExists(QString &fileName)

**8.8.2. CuttingRenderWidget**

/\*\*

\* @brief The CuttingRenderWidget class

\* CT渲染窗口

/\*\*

/\*\*

\* @brief saveOnePoint 保存屏幕上的点

\* @param x 点的横坐标

\* @param y 点纵坐标

\*/

void saveOnePoint(int x, int y)

/\*\*

\* @brief clearAllSavedPoints

\* 清空所有已保存的点

\*/

void clearAllSavedPoints()

/\*\*

\* @brief loadSingleModel

\* 载入单个骨块

\*/

void loadSingleModel()

/\*\*

\* @brief reloadSingleModelByIndex根据索引载入对应骨块

\* @param i 骨块的index

\*/

void reloadSingleModelByIndex(int i)

/\*\*

\* @brief reloadSingleModel根据骨块路径载入对应骨块

\* @param fullpath 骨块mhd全路径

\*/

void reloadSingleModel(QString & fullpath)

/\*\*

\* @brief reloadModels

\* 虚拟复位载入所有选中骨块

\*/

void reloadModels()

/\*\*

\* @brief setMyCameraViewAngle设置相机张角

\* @param camera相机指针

\*/

void setMyCameraViewAngle(vtkCamera \* camera)

/\*\*

\* @brief doCutting切割骨块

\* @param pattern 画定区域图案

\* @param cuttingFlag 切割内部还是外部的标志

\*/

void doCutting(*const* cv::Mat & pattern, int cuttingFlag)

/\*\*

\* @brief getVtkTransformMatrix获取vtkCamera的位置变换矩阵

\* @param posePara1 6自由度vector

\*/

void getVtkTransformMatrix(std::vector<double>& posePara1)

/\*\*

\* @brief initVolumeProperty

\* 初始化volume property

\*/

void initVolumeProperty()

/\*\*

\* @brief initVolume初始化volume

\* @param volume 目标volume

\* @param fullpath ct全路径

\*/

void initVolume(vtkSmartPointer<vtkVolume>& volume, QString & fullpath)

/\*\*

\* @brief initItkData初始化itkData

\* @param itkData

\* @param fullpath 骨块全路径

\*/

void initItkData(itk::Image<short, 3>::Pointer & itkData, QString & fullpath)

/\*\*

\* @brief pushVolumeInVector 将文件载入到内存vector

\* @param fullpath 骨块全路径

\*/

void pushVolumeInVector(QString & fullpath)

/\*\*

\* @brief pushNameInVector将骨块名载入到内存vector

\* @param fullpath 将骨块全路径

\*/

void pushNameInVector(QString & fullpath)

/\*\*

\* @brief pushVolumes 根据列表将volume载入到内存

\* @param list volume列表

\*/

void pushVolumes(QStringList &list)

/\*\*

\* @brief eraseVolumeByIndex根据索引擦除volume

\* @param i目标volume的index

\*/

void eraseVolumeByIndex(int i)

/\*\*

\* @brief resetCTRenderCamera

\* 重置camera

\*/

void resetCTRenderCamera()

/\*\*

\* @brief saveVirtualRestRenderer

\* 保存虚拟复位

\*/

void saveVirtualRestRenderer()

## 存储分配

切割复位功能本地文件存储内容：

1. 骨块的mhd和raw保存在当前病例目录："./cutting/result/"文件夹下；
2. CT颜色窗宽/窗位信息保存在当前病例目录下："./cutting/opacityInfo.xml.xml"文件；
3. CT切割复位信息保存在当前病例目录下："./cutting/resultInfo.xml"文件。

切割复位功能内存存储内容：

1. CT影像信息（全局唯一）： g\_src\_obj。

## 限制条件

1. 切割骨块列表长度为10；
2. 2≤复位骨块数≤10；
3. 骨块命名由下划线和字母数字汉字组成。

## 测试要点

1. 正侧位视图是否正常显示；
2. 切割骨块保存；
3. 切割骨块列表最多显示12个骨块；
4. 3D视图鼠标键交互测试；
5. 骨块删除测试；
6. 骨块镜像测试；
7. 切割后，CT重置测试；
8. 磁盘空间不足，切割骨块是否提示；
9. 切割骨块切除内部与外部测试；
10. 骨块镜像后定位测试；
11. CT微调按钮测试；
12. 选择一个骨块进入复位；
13. 选择两个骨块进入复位；
14. 选择12个骨块进入复位；
15. 选中一个骨块，微调复位；
16. 选中两个骨块，微调复位；
17. 鼠标交互操作复位骨块；
18. 下一步数据保存完整性测试；
19. 上一步按钮返回数据管理测试。

# CT分区功能设计说明

## 功能描述

1. CT分区功能主要用来对CT影像进行分区，辅助配准功能和规划功能，是软件的重要功能。将CT镜像转换为正侧位图像后，对椎体按照一定要求进行分区并标记椎体名称；
2. CT分区功能在软件中属于常驻内存、不可重入，属于顺序处理且代码没有覆盖要求。本功能与其他功能耦合较小：利用全局静态指针获取病例功能三维重建的数据；通过写出到本地文件的方式将自身CT分区结果发送给定位功能；
3. 离开功能时会弹窗提示用户再次确认分区的规范性，详见本章第10节限制条件；
4. 功能内部代码增加边缘性检测，详见接口部分。

## 功能

CT分区功能包含CT正位视图显示区、CT侧位视图显示区、分区列表区、分区功能操作区。CT分区功能详细功能：

1. 软件处于切割功能，且在任务栏中点击【下一步】或软件处于规划功能，且在任务栏中点击【上一步】可实现进入CT分区功能；
2. CT分区功能包含CT正位视图显示区、CT侧位视图显示区、分区列表区、分区功能操作区。CT分区功能详细功能；
3. CT正位视图显示区显示CT数据的正位视图；
4. CT侧位视图显示区显示CT数据的侧位视图；
5. CT正侧位视图可以调节CT的窗宽/窗位、重置窗宽/窗位；
6. 分区列表涵盖椎体的编号、椎体部位、椎体标记，其中编号按顺序自动递增，椎体部位通过下拉菜单定义椎体名称（颈椎、胸椎、腰椎、骶骨），椎体标记包括如下列表（C1-C7、T1-T12、L1-L5、S1-S2）；
7. 分区列表数量最多为12；
8. 进入界面后自动在CT正侧位视图区划分区域；
9. 分区操作应实现分区线增加功能、分区线删除功能；
10. 可以对CT显示的图像进行上下翻转；
11. 可以对CT显示的图像进行左右翻转；
12. 软件处于CT分区功能，在任务栏中点击【上一步】，会自动放弃当前操作数据，软件进入切割功能；
13. 软件处于CT分区功能，在任务栏中点击【下一步】，会自动校验当前操作数据（CT分区是否正确）、自动保存当前操作数据（将分区内容保存到本地数据文件中），软件进入规划功能；
14. 软件处于CT分区功能，在任务栏中点击【上一步】，会自动放弃当前操作数据，软件进入切割功能；
15. 软件处于CT分区功能，在任务栏中点击【下一步】，会自动校验当前操作数据（CT分区是否正确）、自动保存当前操作数据（将分区内容保存到本地数据文件中），软件进入规划功能。

****

CT分区功能功能IPO图

## 性能

1. CT分区数量≤12。

## 输入项目

CT分区功能-全功能输入项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | CT重构结果 | 无 | VolumeObject | 不为空 | 自动输入 | 病例载入功能重构结果 |

CT分区功能-分区列表相关输入项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | 椎体部位 | 部位 | int | 0-3 | 鼠标选择 | 修改椎体部位名称 |
| **2** | 椎体标记 | 标记 | int | 0-21 | 鼠标选择 | 修改椎体标记 |

## 输出项目

CT分区功能-全功能输出项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输出方式** | **备注** |
| **1** | CT分区边框信息 | 无 | xml | 不为空 | 自动保存 | 界面上边框位置信息 |
| **2** | CT分区结果 | 无 | xml | 不为空 | 自动保存 | CT分区空间结果 |

## 算法

**9.6.1. CT分区算法**

算法目的：

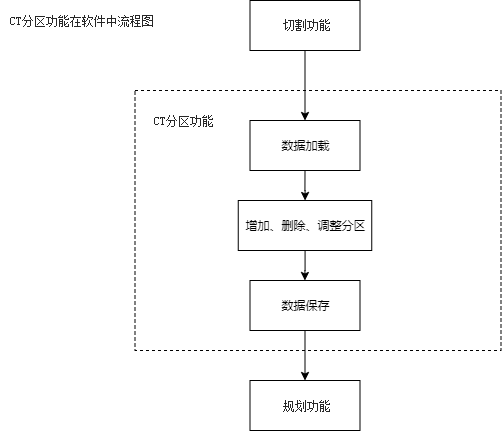
建立CT所选感兴趣区域的姿态坐标矩阵。

算法流程：

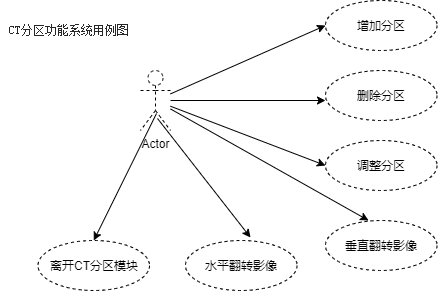
1. 对CT数据进行投影，获取正侧位DRR图像；
2. 通过包围盒的形式手动分割选取感兴趣区域；
3. 以所选区域的中心点作为原点，正位方向为Z轴，区域上平面为Y轴建立区域姿态坐标矩阵。

## 程序逻辑

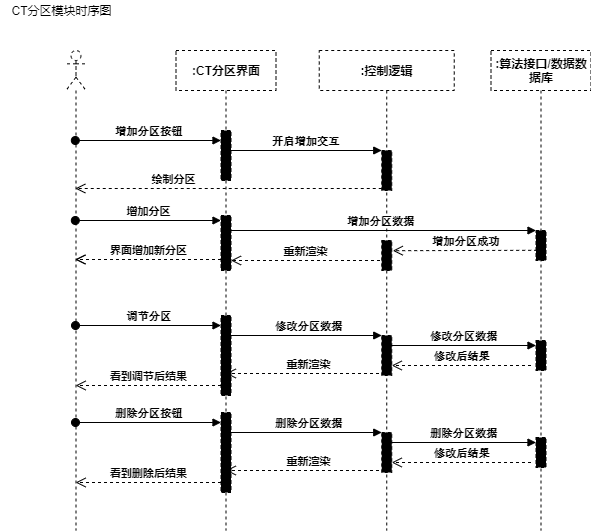
CT分区功能是本软件顺序流程的一个步骤，上一步为切割功能，下一步为规划功能。分别用CT分区功能在软件内的流程图、CT分区功能系统用例图、CT分区功能时序图来展示本功能的程序逻辑。



CT分区功能在软件内的流程图



CT分区功能系统用例图



CT分区功能时序图

## 接口

CT分区功能包含SpineSegCtWidget、VtkSplitCtWidget、spine\_openGL\_widget、segm\_data四个主要类。其中重要接口详细说明如下：

**9.8.1. SpineSegCtWidget**

/\*\*

\* @brief The SpineSegCtWidget class

\* CT分区功能主窗口

/\*\*

/\*\*

\* @brief RestInterface 进入界面，ui全部复位

\* @return 是否成功进入界面，ui全部复位

\*/

bool RestInterface() *override*;

/\*\*

\* @brief VerificationSelfData 离开（下一步），验证数据完整

\* @return 是否成功

\*/

bool VerificationSelfData() *override*;

/\*\*

\* @brief WriteSelfModularData 离开（上、下一步），保存数据

\* @return 是否成功离开（上、下一步），保存数据

\*/

bool WriteSelfModularData() *override*;

/\*\*

\* @brief ReadSelfModularData 进入界面，读取本功能保存数据

\* @return 是否成功进入界面，读取本功能保存数据

\*/

bool ReadSelfModularData() *override*;

/\*\*

\* @brief ReadOtherModularData 进入界面，读取其他功能保存数据

\* @return 是否成功进入界面，读取其他功能保存数据

\*/

bool ReadOtherModularData() *override*;

**9.8.2. VtkSplitCtWidget**

/\*\*

\* @brief The VtkSplitCtWidget class

\* CT 分区vtk渲染窗口，正侧位窗口均在内部

\*/

*class* VtkSplitCtWidget;

/\*\*

\* @brief AddNewBox

\* 增加新的分区线

\*/

void AddNewBox();

/\*\*

\* @brief DeleteBox

\* 删除当前选中分区线

\*/

void DeleteBox();

/\*\*

\* @brief FlipV

\* CT影像垂直翻转

\*/

void FlipV();

/\*\*

\* @brief FlipH

\* CT影像水平翻转

\*/

void FlipH();

/\*\*

\* @brief ResetWindowLevel

\* 复位窗宽/窗位

\*/

void ResetWindowLevel();

/\*\*

\* @brief ClearView

\* 清空当前窗口分区线

\*/

void ClearView();

/\*\*

\* @brief ResetWidget

\* 复位当前窗口

\*/

void ResetWidget();

/\*\*

\* @brief SetTips

\* 设置分区标识符

\* @param strlist 标识符列表

\*/

void SetTips(*const* QStringList &strlist);

/\*\*

\* @brief SaveBoxData 保存分区线位置

\* @param file\_name 保存文件名称

\* @return 保存结果

\*/

bool SaveBoxData(*const* QString &file\_name);

/\*\*

\* @brief ReflashLabelActor

\* 手动刷新窗口

\*/

void ReflashLabelActor();

/\*\*

\* @brief GetBroundPos

\* 获取分区边框位置

\*/

void GetBroundPos();

**9.8.3. spine\_openGL\_widget**

/\*\*

\* @brief The spine\_openGL\_widget class

\* VTK 渲染窗口

\*/

*class* spine\_openGL\_widget;

/\*\*

\* @brief EventCallBack 鼠标交互回调函数

\* @param object 窗口类

\* @param event 交互类型

\* @param clientdata 窗口发送数据

\* @param calldata 窗口接收数据

\*/

*static* void EventCallBack(vtkObject\* object, unsigned long event, void\* clientdata, void\* calldata);

/\*\*

\* @brief \_leftMousePress

\* 鼠标左键按下事件

\*/

void \_leftMousePress();

/\*\*

\* @brief \_rightMousePress

\* 鼠标右键按下事件

\*/

void \_rightMousePress();

/\*\*

\* @brief \_mouseMove

\* 鼠标移动事件

\*/

void \_mouseMove();

/\*\*

\* @brief \_leftMouseRelease

\* 鼠标左键抬起事件

\*/

void \_leftMouseRelease();

/\*\*

\* @brief \_rightMouseRelease

\* 鼠标右键抬起事件

\*/

void \_rightMouseRelease();

**9.8.4. segm\_data**

/\*\*

\* @brief The segm\_data class

\* 分区配准信息存储类

\*/

*class* segm\_data;

/\*\*

\* @brief flipData 翻转影像

\* @param data 要反转的图片

\* @param aix 翻转轴

\* @return

\*/

vtkImageData \*flipData(vtkImageData \*data, int aix);

/\*\*

\* @brief setImageData 设置当前图片

\* @param data 突变

\*/

void setImageData(vtkImageData \*data);

/\*\*

\* @brief DRRInit

\* DRR渲染初始化

\*/

void DRRInit();

/\*\*

\* @brief readConfig

\* 读取配置文件

\* @param file 文件路径

\*/

void readConfig(*const* QString &file);

/\*\*

\* @brief readCtConfig

\* 读取CT分区配置文件

\* @param file 文件路径

\*/

void readCtConfig(*const* QString &file);

/\*\*

\* @brief readRXayConfig

\* 读取X-Ray分区配置文件

\* @param file 文件路径

\*/

void readRXayConfig(*const* QString &file);

/\*\*

\* @brief initCamMat

\* 初始化相机矩阵

\*/

void initCamMat();

/\*\*

\* @brief readShowImage 读取图片

\* @param file 图片路径

\* @param data 读取结果

\*/

void readShowImage(*const* QString &file, vtkImageData \*data);

/\*\*

\* @brief readShowRemoveBallImage 读取去小球图片

\* @param file 图片路径

\* @param data 读取结果

\*/

void readShowRemoveBallImage(*const* QString &file, vtkImageData \*data);

/\*\*

\* @brief readDataImage 以字符串形式读取图片

\* @param file 图片路径

\* @param data 读取结果

\*/

void readDataImage(*const* QString &file, short \*data);

/\*\*

\* @brief simpleBlob 图像预处理

\* @param m 输入图像

\* @return 返回图像

\*/

cv::Mat simpleBlob(cv::Mat m);

## 存储分配

CT分区功能本地文件存储内容：

1. CT分区边框信息保存在当前病例目录下："./SpineSegCt/CTSpineSegBox.xml"文件；

2. CT分区结果保存在当前病例目录下："./SpineSegCt/ CTSpineSegResult.xml"文件。

CT分区功能内存存储内容：

1. CT影像和分区信息（全局唯一） segm\_data::instance()。

## 限制条件

1. 确保患者CT的头部方向向上；
2. 分区红点位于椎间盘的中心；
3. 分锥线与相邻椎体上平板平行；
4. 分区线无法相交；
5. 分区线不能超出图像边缘；
6. 分区线位于分区框内部；
7. 椎体列表始终按照顺序排布；
8. 增加提示条：关于CT分区的操作注意事项；
9. 使用成熟可靠的第三方重建功能。

## 测试要点

1. 进入CT分区，整个椎体被默认框选；
2. 分区线最多画11条；
3. 分区线只能单个删除，无法批量删除；
4. 选中椎体无法删除测试；
5. 分区线只能在上下分区线间调整角度；
6. CT正侧位右键窗宽窗位调整；
7. CT窗宽窗位重置；
8. 椎体部位椎体标记修改；
9. 查看椎体部位分类与椎体标记数据是否完整并且对应；
10. 点击分区只能画一次线；
11. CT上下翻转、左右翻转测试；
12. CT分区是否高亮；
13. 点击下一步，查看数据保存的完整性；
14. 点击上一步，返回切割复位页面，数据未保存。

# 规划功能设计说明

## 功能描述

1. 规划功能是本软件实现术前规划功能的重要途经，是整个软件的核心功能。用于术前路径的规划，在术前CT或MR数据三维重建结果上自定义手术中放置螺钉的位置、直径、长度等；
2. 规划功能在软件中属于常驻内存、不可重入，属于顺序处理且代码没有覆盖要求。本功能与其他功能耦合较小：利用全局静态指针获取病例功能三维重建的数据；自身实现重建数据的浏览、螺钉的规划以及测量功能；和定位功能复用同一个螺钉数据模型实现规划功能和定位功能螺钉数据的同步；
3. 功能内部代码增加边缘性检测，详见接口部分。

## 功能

软件处于CT分区功能，且在任务栏中点击【下一步】或软件处于X-Ray分区功能，且在任务栏中点击【上一步】可实现进入规划功能。

规划功能分为三维数据展示区、三维数据操作区、CT分区结果预览区、规划螺钉列表区、器官分割列表区、规划功能按键区。

规划功能详细功能：

1. 三维数据展示区实现医学影像四视图的常用功能和视图；
2. 三维数据操作区和三维数据展示区只能同时存在一种2D或3D视图；
3. CT分区结果预览区实现CT分区结果的三维预览，可以快速定位三维数据操作区中2D影像的位置；
4. 规划螺钉列表区展示当前已规划螺钉的详细信息（序号、名称、直径、长度、水平夹角、类型、颜色、显示状态）；
5. 器官分割列表区实现CT数据的血管、骨骼、脏器分割功能；
6. 规划功能按键区包括螺钉的手动添加、删除、自动添加、长度测量、角度测量、长度测量清除、角度测量清除、保存（规划数据保存在本地）、导出（规划数据、CT分区信息）功能；
7. 软件处于规划功能，在任务栏中点击【上一步】，会自动放弃当前操作数据，软件进入CT分区功能；
8. 软件处于规划功能，在任务栏中点击【下一步】，会自动校验当前操作数据（是否已经规划了螺钉）、自动保存当前操作数据（将已规划螺钉保存到本地数据文件中），软件进入X-Ray分区功能。

****

规划功能功能IPO图

## 性能

1. 规划螺钉数量≤50。

## 输入项目

规划功能-全功能输入项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | CT重构结果 | 无 | VolumeObject | 不为空 | 自动输入 | 病例载入功能重构结果 |
| **2** | 规划螺钉文件 | 无 | dat | 不为空 | 自动输入 | 本病例已规划过数据 |

规划功能-规划螺钉相关输入项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | 螺钉名称 | 名称 | QString | 满足规定正则 | 键盘输入 | 螺钉名称修改 |
| **2** | 螺钉直径 | 无直径 | double | 不为空 | 键盘输入 | 螺钉直径修改 |
| **3** | 螺钉颜色 | 颜色 | QColor | 满足颜色格式 | 鼠标点击 | 螺钉颜色修改 |
| **4** | 是否显示 | 显示 | bool | 1或0 | 鼠标点击 | 螺钉显示状态修改 |

## 输出项目

规划功能-全功能输出项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输出方式** | **备注** |
| **1** | 规划螺钉文件 | 无 | dat | 不为空 | 自动保存 | 将已规划螺钉保存为本地数据 |
| **2** | 规划结果 | 无 | ScrewModel | 数量不为0 | 自动同步 | 将已规划数据同步到定位功能 |

规划功能-功能按键输出项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输出方式** | **备注** |
| **1** | 【导出】 | 规划螺钉文件 | dat | 不为空 | 自动保存 | 将已规划螺钉导出到U盘 |
| **2** | 【保存】 | 规划螺钉文件 | dat | 数量不为0 | 自动同步 | 将已规划螺钉保存为本地数据 |

## 算法

规划功能包含算法：螺钉自动增加、器官分割。

10.6.1. 自动添加螺钉算法

算法目的：

在分节的脊柱CT数据中寻找两端椎弓根的部位，并以此部位设置螺钉的长度，方向和位置，从而实现椎弓根螺钉的自动添加。

算法流程：

1. 首先通过手动分锥步骤，得到单节椎体；
2. 利用阈值分割获得骨性结构，剔除软组织；
3. 在单节椎体中根据椎体的结构特征找到椎体的关键点；
4. 根据上述关键点确定椎弓根的区域；
5. 计算椎弓根螺钉区域的中轴线；
6. 在中轴线上寻找两个点作为椎弓根螺钉的两个端点，其中一个点在骨性结构内距离椎体前缘10mm，另一个点在骨性结构外距离人字脊10mm。

10.6.2. 自动器官分割算法

算法目的：

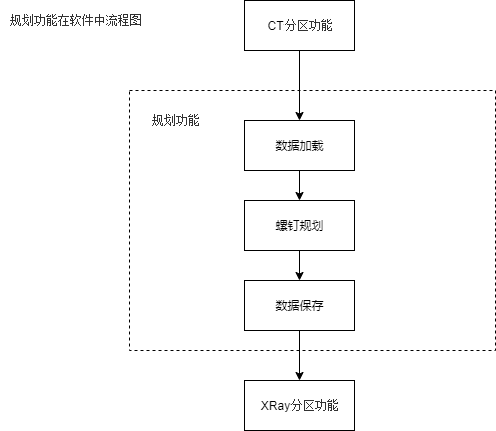
将病人的CT数据中的骨骼和软组织区分开，用于单独显示和单独运算。

算法流程：

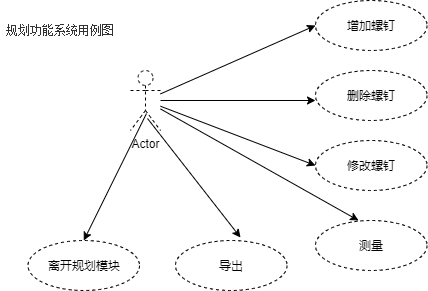
1. 导入CT数据，对所有体素进行遍历；
2. 所有大于阈值的体素标记为1，小于阈值的体素标记为0，获得骨骼蒙版；
3. 再将骨骼蒙版中所有1变成0，0变成1，获得软组织蒙版；
4. 骨骼蒙版和CT数据对应体素相乘，得到骨骼数据；
5. 软组织蒙版和CT数据对应体素相乘，得到软组织数据。

## 程序逻辑

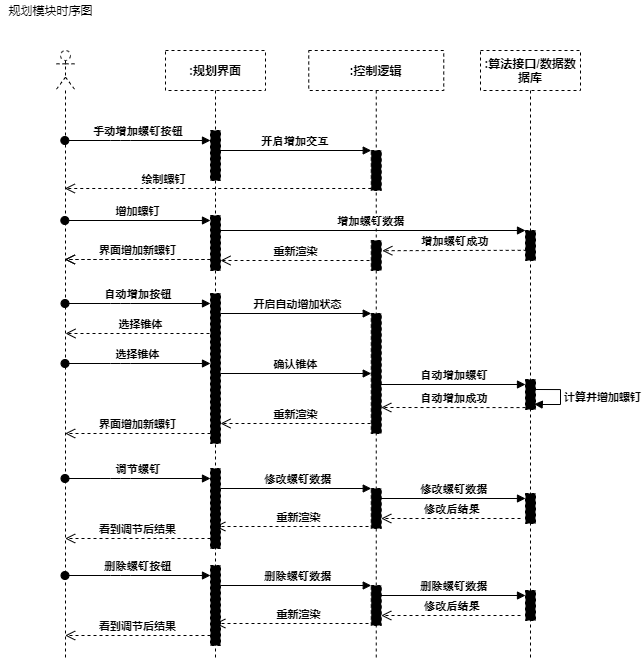
规划功能是本软件顺序流程的一个步骤，上一步为CT分区，下一步为X-Ray分区。分别用规划功能在软件内的流程图、规划功能系统用例图、规划功能时序图来展示本功能的程序逻辑。



规划功能在软件内的流程图



规划功能系统用例图



规划功能时序图

## 接口

规划功能包含PlanWidget、FourViewManager、LineManager、PlanMeasureManager四个主要类。其中重要接口详细说明如下：

**10.8.1. PlanWidget**

/\*\*

\* @brief The PlanWidget class

\* 规划功能主窗口

/\*\*

/\*\*

\* @brief RestInterface 进入界面，ui全部复位

\* @return 是否成功进入界面，ui全部复位

\*/

bool RestInterface() *override*;

/\*\*

\* @brief VerificationSelfData 离开（下一步），验证数据完整

\* @return 是否成功

\*/

bool VerificationSelfData() *override*;

/\*\*

\* @brief WriteSelfModularData 离开（上、下一步），保存数据

\* @return 是否成功离开（上、下一步），保存数据

\*/

bool WriteSelfModularData() *override*;

/\*\*

\* @brief ReadSelfModularData 进入界面，读取本功能保存数据

\* @return 是否成功进入界面，读取本功能保存数据

\*/

bool ReadSelfModularData() *override*;

/\*\*

\* @brief ReadOtherModularData 进入界面，读取其他功能保存数据

\* @return 是否成功进入界面，读取其他功能保存数据

\*/

bool ReadOtherModularData() *override*;

**10.8.2. FourViewManager**

/\*\*

\* @brief The FourViewWidget class

\* 医学四视图

\*/

/\*\*

\* @brief Change3dColor 修改3d视图vr渲染颜色

\* @param color 被修改颜色

\*/

void Change3dColor(*const* QColor &color);

/\*\*

\* @brief SelectAndChange3dColor 弹窗选择颜色

\*/

void SelectAndChange3dColor();

/\*\*

\* @brief Reduction2dWindowLevel 复位窗宽/窗位

\*/

void Reduction2dWindowLevel();

/\*\*

\* @brief Set2dViewLocked 锁定2d视图

\* @param locked 是否锁住

\*/

void Set2dViewLocked(*const* bool &locked);

/\*\*

\* @brief FourViewRender 窗口刷新

\*/

void FourViewRender();

/\*\*

\* @brief SetWidgetSize 设置窗口尺寸

\* @param size 尺寸大小

\*/

void SetWidgetSize(*const* QSize &size);

/\*\*

\* @brief SetType 设置窗口类型

\* @param type 具体类型

\* @param parent 窗口父类

\*/

void SetType(*const* type &type, QWidget \*parent);

**10.8.3. LineManager**

/\*\*

\* @brief The LineManager class

\* 螺钉管理类

\*/

/\*\*

\* @brief RemoveLine 删除螺钉

\* @param idx 要删除路螺钉的id

\*/

void RemoveLine(int idx);

/\*\*

\* @brief ChangeDiameter 修改螺钉直径

\* @param diameter 直径

\* @param idx 被修改螺钉的id

\*/

void ChangeDiameter(double diameter, int idx);

/\*\*

\* @brief ChangeColor 修改螺钉颜色

\* @param idx 被修改螺钉id

\* @param color 目标颜色

\*/

void ChangeColor(int idx, QColor color);

/\*\*

\* @brief ShoworHideLine 修改螺钉可视化状态

\* @param idx 目标螺钉id

\* @param show 目标螺钉状态

\*/

void ShoworHideLine(int idx, bool show);

/\*\*

\* @brief SetScrewType 设定螺钉类型

\* @param idx 目标螺钉id

\* @param type 目标螺钉状态

\*/

void SetScrewType(int idx, int type);

**10.8.4. PlanMeasureManager**

/\*\*

\* @brief The PlanMeasureManager class

\* 测量管理类

\*/

/\*\*

\* @brief The MeasureState enum

\* 测量类型

\*/

*enum* MeasureState

{

*Empty*, // 空闲状态

*Distance*, // 长度测量

*Angle*, // 角度测量

};

/\*\*

\* @brief EmptyMeasureWidget 清空测量

\* @param state 清空类型

\*/

void EmptyMeasureWidget(*const* MeasureState &state = *Empty*);

/\*\*

\* @brief AssignmentMeasureWidget 分配测量

\* @param state 分配类型

\*/

void AssignmentMeasureWidget(*const* MeasureState &state = *Empty*);

/\*\*

\* @brief BeginMeasure 开始测量

\* @param state 测量类型

\*/

void BeginMeasure(*const* MeasureState &state);

/\*\*

\* @brief OffAllMeasure 关闭测量

\*/

void OffAllMeasure();

## 存储分配

规划功能本地文件存储内容：

1. 已规划的数据保存在当前病例目录下： "./Planning/SaveScrewData.dat"文件。

规划功能内存存储内容：

1. 已规划螺钉数据（全局唯一）: ScrewTabelModel ::Instance()；
2. 分区数据在规划功能体现: QStringList PlanWidget::cone\_name\_list\_；
3. 器官分割数据在规划功能体现:

QList<vtkVolume \*> VtkOrganSegmentationWidget:: volums\_。

## 限制条件

1. 点击【下一步】离开功能，需确保至少规划一颗螺钉；
2. 进入功能需确保病例功能已经加载并重构影像；
3. 进入功能需确保CT分区功能至少分割一块分区；
4. 螺钉只能在2D视图上进行规划；
5. 长度测量只能在2D视图上进行；
6. 角度测量只能在2D视图上进行；
7. 四视图无法无限制缩放；
8. 使用成熟可靠的第三方测量功能。

## 测试要点

1. 查看2D视图，3D视图是否正常显示；
2. 2D视图鼠标交互测试（左键中键查看图层，中键长按平移，右键窗宽窗位调整，ctrl+右键放大缩小），窗宽窗位重置；
3. 3D视图鼠标交互测试（左键旋转，中键长按平移，右键窗宽窗位调整，ctrl+右键及滚轮放大缩小），窗宽窗位重置；
4. 3D颜色修改；
5. 3D渲染方式修改；
6. 点击2D图，最大化在中间显示；
7. 查看示意图椎体标记是否正确；
8. 手动添加螺钉测试；
9. 自动添加螺钉，选中椎体，查看螺钉是否显示；
10. 螺钉删除测试；
11. 2D图角度测量及清除、长度测量及清除测试；
12. 数据保存及导出完整性测试，保存位置正确性测试；
13. 磁盘空间不足，保存数据测试；
14. U盘空间不足，导出数据测试；
15. 螺钉名称、颜色、直径、类型、显示与隐藏修改测试；
16. 血管、组织、骨骼分割测试；
17. 点击下一步，数据保存正确性及完整性测试；
18. 点击上一步，返回CT分区页面。

# X-Ray分区功能设计说明

## 功能描述

1. X-Ray分区功能主要用来对X-Ray影像进行分区，辅助配准功能，是软件的重要功能。收到X-Ray正侧位图像后，对椎体按照操作规范进行分区并标记椎体名称；
2. X-Ray分区功能在软件中属于常驻内存、不可重入，属于顺序处理且代码没有覆盖要求。本功能与其他功能耦合较小：借助通讯功能收到X-Ray正侧位影像和拍摄时参数；通过写出到本地文件的方式将自身X-Ray分区结果发送给定位功能；
3. 离开功能时会弹窗提示用户再次确认分区的规范性，详见本章第10节限制条件；
4. 功能内部代码增加边缘性检测，详见接口部分。

## 功能

软件处于规划功能，且在任务栏中点击【下一步】或软件处于配准功能，且在任务栏中点击【上一步】可实现进入X-Ray分区功能。

X-Ray分区功能包含X-Ray正位视图显示区、X-Ray侧位视图显示区、分区列表区、分区功能操作区。X-Ray分区功能详细功能：

1. X-Ray正位视图显示区可以接收并显示X-Ray影像，若还未收到图像区域显示黑色底图；
2. X-Ray侧位视图显示区可以接收并显示X-Ray影像，若还未收到图像区域显示黑色底图；
3. X-Ray正侧位视图可以调节CT的窗宽/窗位、重置窗宽/窗位；
4. Ray正侧位视图可以进行图像预处理（滤波、去小球）；
5. 分区列表涵盖椎体的编号、椎体部位、椎体标记，其中编号按顺序自动递增，椎体部位通过下拉区框定义椎体名称，椎体标记包括如下列表；
6. 分区列表数量最多为12；
7. 进入界面后自动在X-Ray正侧位视图区划分区域；
8. 分区操作应实现分区线增加功能、分区线删除功能；
9. 可以对X-Ray显示的图像进行旋转（顺时针、逆时针）；
10. 软件处于X-Ray分区功能，在任务栏中点击【上一步】，会自动放弃当前操作数据，软件进入规划功能；
11. 软件处于X-Ray分区功能，在任务栏中点击【下一步】，会自动校验当前操作数据（X-Ray分区是否正确，X-Ray分区和CT分区是否存在交集）、自动保存当前操作数据（将分区内容保存到本地数据文件中），软件进入配准功能。

lALPDgQ9zGZ-CkI-zQI7_571_62

X-Ray分区功能功能IPO图

## 性能

1. X-Ray分区数量≤12。

## 输入项目

X-Ray分区功能-全功能输入项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | Xray正位图片 | 无 | QImage | 不为空 | 通讯功能传入 | 正位图片 |
| **2** | Xray侧位图片 | 无 | QImage | 不为空 | 通讯功能传入 | 侧位图片 |
| **3** | Xray内外参 | 无 | QString | 不为空 | 通讯功能传入 | 内外参数 |

X-Ray分区功能-分区列表相关输入项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | 椎体部位 | 部位 | int | 0-3 | 鼠标选择 | 修改椎体部位名称 |
| **2** | 椎体标记 | 标记 | int | 0-21 | 鼠标选择 | 修改椎体标记 |

## 输出项目

X-Ray分区功能-全功能输出项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输出方式** | **备注** |
| **1** | XRay分区边框信息 | 无 | xml | 不为空 | 自动保存 | 界面上边框位置信息 |
| **2** | XRay分区结果 | 无 | xml | 不为空 | 自动保存 | XRay分区空间结果 |
| **3** | Xray正位图片 | 无 | bmp | 不为空 | 自动保存 | 正位图片 |
| **4** | Xray侧位图片 | 无 | bmp | 不为空 | 自动保存 | 侧位图片 |
| **5** | Xray内外参 | 无 | ini | 不为空 | 自动保存 | 拍摄时内外参数 |

## 算法

11.6.1 XRay分割算法

算法目的：

建立X光所选感兴趣区域的空间姿态坐标矩阵。

算法流程：

1. 通过已有数据确定正侧位X光图像空间关系；
2. 以包围盒的形式手动分割选取感兴趣区域；
3. 以所选区域的中心点作为原点，正位方向为Z轴，区域上平面为Y轴建立区域姿态坐标矩阵。

11.6.2. Xray消除标记点（去小球）算法

算法目的：

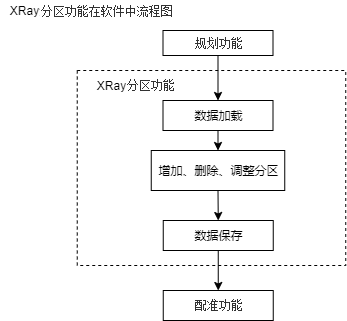
消除图像中的黑色标记点，还原标记点处应该具有的像素值。

算法流程：

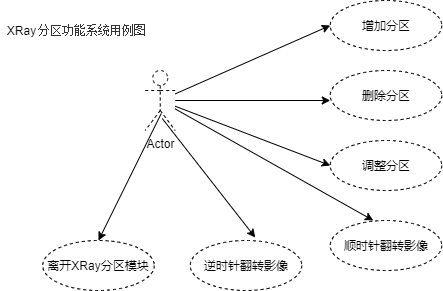
1. 识别图像中的圆形标记球，计算出每个标记球的圆心坐标和半径；
2. 根据每个小球的坐标和半径，从圆外的相邻像素进行差值沿半径方向填充到圆内的像素。

## 程序逻辑

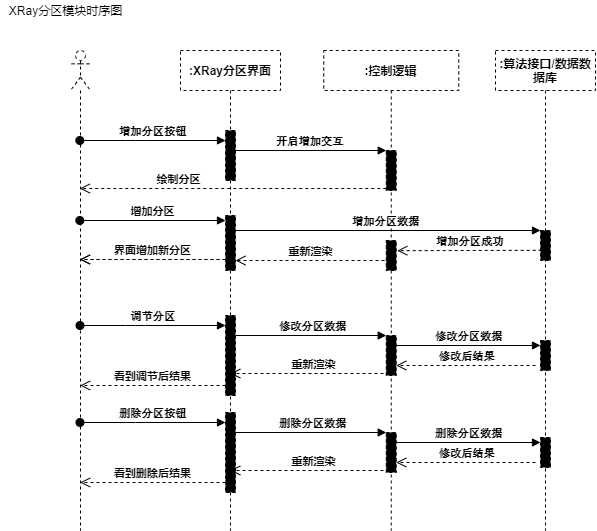
X-Ray分区功能是本软件顺序流程的一个步骤，上一步为规划功能，下一步为配置功能。分别用X-Ray分区功能在软件内的流程图、X-Ray分区功能系统用例图、X-Ray分区功能时序图来展示本功能的程序逻辑。



X-Ray分区功能在软件内的流程图



X-Ray分区功能系统用例图



X-Ray分区功能时序图

## 接口

XRay分区功能包含SpineSegXRayWidget、VtkSplitXRayWidget、spine\_openGL\_widget、segm\_data四个主要类。其中重要接口详细说明如下：

**11.8.1. SpineSegXRayWidget**

/\*\*

\* @brief The SpineSegXRayWidget class

\* XRay分区功能主窗口

/\*\*

/\*\*

\* @brief RestInterface 进入界面，ui全部复位

\* @return 是否成功进入界面，ui全部复位

\*/

bool RestInterface() *override*;

/\*\*

\* @brief VerificationSelfData 离开（下一步），验证数据完整

\* @return 是否成功

\*/

bool VerificationSelfData() *override*;

/\*\*

\* @brief WriteSelfModularData 离开（上、下一步），保存数据

\* @return 是否成功离开（上、下一步），保存数据

\*/

bool WriteSelfModularData() *override*;

/\*\*

\* @brief ReadSelfModularData 进入界面，读取本功能保存数据

\* @return 是否成功进入界面，读取本功能保存数据

\*/

bool ReadSelfModularData() *override*;

/\*\*

\* @brief ReadOtherModularData 进入界面，读取其他功能保存数据

\* @return 是否成功进入界面，读取其他功能保存数据

\*/

bool ReadOtherModularData() *override*;

**11.8.2. VtkSplitXRayWidget**

/\*\*

\* @brief The VtkSplitXRayWidget class

\* XRay分区vtk渲染窗口，正侧位窗口均在内部

\*/

*class* VtkSplitCtWidget;

/\*\*

\* @brief AddNewBox

\* 增加新的分区线

\*/

void AddNewBox();

/\*\*

\* @brief DeleteBox

\* 删除当前选中分区线

\*/

void DeleteBox();

/\*\*

\* @brief FlipV

\* CT影像垂直翻转

\*/

void FlipV();

/\*\*

\* @brief FlipH

\* CT影像水平翻转

\*/

void FlipH();

/\*\*

\* @brief ResetWindowLevel

\* 复位窗宽/窗位

\*/

void ResetWindowLevel();

/\*\*

\* @brief ClearView

\* 清空当前窗口分区线

\*/

void ClearView();

/\*\*

\* @brief ResetWidget

\* 复位当前窗口

\*/

void ResetWidget();

/\*\*

\* @brief SetTips

\* 设置分区标识符

\* @param strlist 标识符列表

\*/

void SetTips(*const* QStringList &strlist);

/\*\*

\* @brief SaveBoxData 保存分区线位置

\* @param file\_name 保存文件名称

\* @return 保存结果

\*/

bool SaveBoxData(*const* QString &file\_name);

/\*\*

\* @brief ReflashLabelActor

\* 手动刷新窗口

\*/

void ReflashLabelActor();

/\*\*

\* @brief GetBroundPos

\* 获取分区边框位置

\*/

void GetBroundPos();

**11.8.3. spine\_openGL\_widget**

详见CT分区第8节。

**11.8.4. segm\_data**

详见CT分区功能第8节。

## 存储分配

XRay分区功能本地文件存储内容：

1. XRay分区边框信息保存在当前病例目录下：

"./XRaySpineSegbox/XRaySpineSegbox.xml"文件；

1. XRay分区结果保存在当前病例目录下：

"./XRaySpineSegbox/ XRaySpineSegResult.xml"文件；

1. XRay正位影像存储位置在当前病例目录下：

"./XRaySpineSegbox/ frontal\_XRay. bmp"文件；

1. XRay侧位影像存储位置在当前病例目录下：

"./XRaySpineSegbox/ lateral\_XRay. bmp"文件；

1. XRay拍摄时内外参数存储位置在当前病例目录下：

"./XRaySpineSegbox/ XRaySetting. ini"文件；

1. XRay拍摄时内外参数缓存存储位置在当前病例目录下：

"./XRaySpineSegbox/ XRaySetting\_cache. ini"文件。

XRay分区功能内存存储内容：

1. XRay影像和分区信息（全局唯一）： segm\_data::instance()。

## 限制条件

1. 确保患者X-Ray影像的头部方向向上；
2. 分区红点位于椎间盘的中心；
3. 分锥线与相邻椎体上平板平行；
4. 分区线无法相交；
5. 分区线不能超出图像边缘；
6. 分区线位于分区框内部；
7. 椎体列表始终按照顺序排布。

## 测试要点

1. 进入X-Ray分区，整个椎体被默认框选；
2. 分区线最多画11条；
3. 分区线只能单个删除，无法批量删除；
4. 选中椎体无法删除测试；
5. 分区线只能在上下分区线间调整角度；
6. X-Ray正侧位右键窗宽窗位调整；
7. X-Ray窗宽窗位重置，去小球及隐藏小球测试，滤波开启与关闭测试；
8. 椎体部位椎体标记修改；
9. 查看椎体部位分类与椎体标记数据是否完整并且对应；
10. 点击分区只能画一次线；
11. X光片顺时针、逆时针测试；
12. X-Ray分区是否高亮；
13. 点击下一步，查看数据保存的完整性；
14. 点击上一步，返回规划页面，数据未保存；
15. 增加提示条：关于X-RAY分区的操作注意事项。

# 配准功能设计说明

## 功能描述

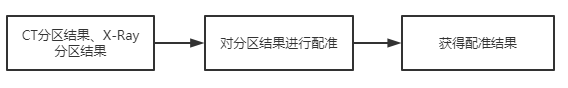
1. 配准功能是本软件实现定位功能的重要途径，是软件的重要功能。目的是使CT分区的结果和X-Ray分区结果位置重合，从而达到统一坐标系的效果；
2. 配准功能在软件中属于常驻内存、不可重入，属于顺序处理且代码没有覆盖要求。本功能与其他功能耦合较小：利用单例的segm\_data静态指针获取CT分区结果和X-Ray结果，将CT分区与X-Ray对应，同自动配准或手动配准将CT与X-Ray的位置对应起来。
3. 功能内部代码增加边缘性检测，详见接口部分。

## 功能

配准功能分为正位渲染窗口、渲染窗口、分区列表和CT操作区。

配准功能详细功能：

1. 正位渲染窗宽显示正位CT分区数据和对应的正位X-Ray分区数据，窗口右上角有窗宽/窗位重置按钮，点击将重置窗口中CT的窗宽/窗位，重置窗宽/窗位按钮旁边是去小球按钮，点击去小球按钮X-Ray图像中的mark球会被去掉减少配准干扰，去小球按钮旁边是滤波按钮，点击滤波按钮X-Ray图像会显示滤波后的效果。窗口底部有对比度滑条，滑动滑条改变CT与X-Ray的对比度；
2. 侧位渲染窗口显示侧位CT分区数据和对应的侧位X-Ray分区数据，区域内的按钮和滑条位置与正位渲染窗口位置一致功能相同；
3. 分区列表显示的是CT分区结果和X-Ray分区结果的交集，列表的每一行都包含分区的序号，椎体标记和相似度；
4. CT操作区包括微调面板、自动配准按钮和确认按钮，点击微调按钮改变CT位置可实现手动配准，点击确认按钮获得相似度同时也是确认一次手动配准完成，点击自动配准按钮CT会自动向着与X-Ray重合的位置进行平移旋转，自动配准完成将弹出确认提示框，点击确的则获取相似度（即配准完成）；
5. 软件处于配准功能，在任务栏中点击【上一步】，会自动放弃当前操作数据，软件进入X-Ray分区功能；
6. 软件处于配准功能，在任务栏中点击【下一步】，会自动校验当前操作数据（是否已经有配准数据）、自动保存当前操作数据（将下发矩阵保存到本地数据文件中）。如果当前用户没有进行配准就点击【下一步】，将弹出提示“未配准不能进入定位界面”，不允许用户进入定位功能；如果用户点击【下一步】且已经有配准数据（做过配准操作），将弹出提示“当前的配准图像是否符合您的手术预期”，用户点击确定才能进入定位功能降低了误操作带来的风险。



配准功能功能IPO图

## 性能

1. 三维数据重建速度≥15MPS。

## 输入项目

配准功能-全功能输入项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | segm\_data | 无 | 指针 | 不为空 | 自动输入 | CT分区和X-Ray分区信息 |

## 输出项目

配准功能-全功能输出项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输出方式** | **备注** |
| **1** | 相似度 | 无 | double | 0.00-1.00 | 界面显示 | 配准完成后CT与X-Ray的相似度 |
| **2** | 下发矩阵 | 无 | xml | 不为空 | 自动保存 | 将配准矩阵转换位定位功能使用的下发矩阵 |

## 算法

**12.6.1. 配准算法**

1.算法目的：

将三维CT图像与二维X光图象配准，通过对CT三维数据DRR重建后的二维图像与X光二维图像求图像相关性，搜索相关性最高的结果作为配准结果。

2.算法流程：

1）确定搜索范围T，最大步长S=T，确定收敛步长S<0.1；

2）以步长S向五个自由度正反方向搜索（正位图像对前后方向不敏感），计算正位相关性；

3）计算搜索过程中相关性最大的值与当前位置相关性进行比较；

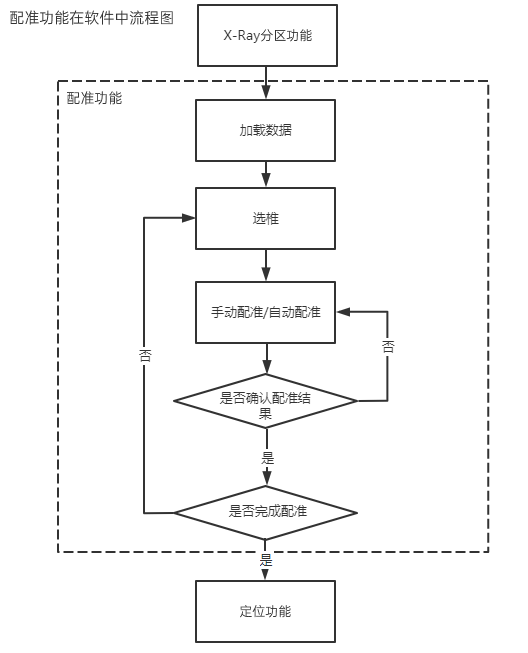
4）如果存在相似度比当前位置相似度高，更新当前位置。如果不存在令S=S/2。重复2，3，4；

5）当S<0.1,令S=T。以步长S向前后方向搜索，计算侧位相关性；

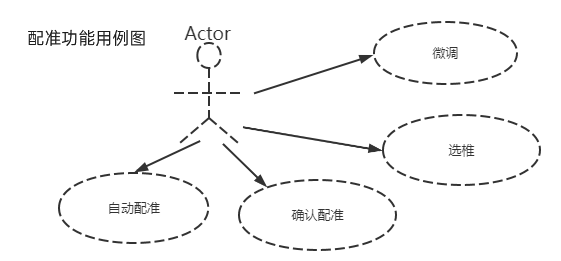
6）与3、4类似，直到S<0.1跳出搜索。结果即作为配准结果。

## 程序逻辑

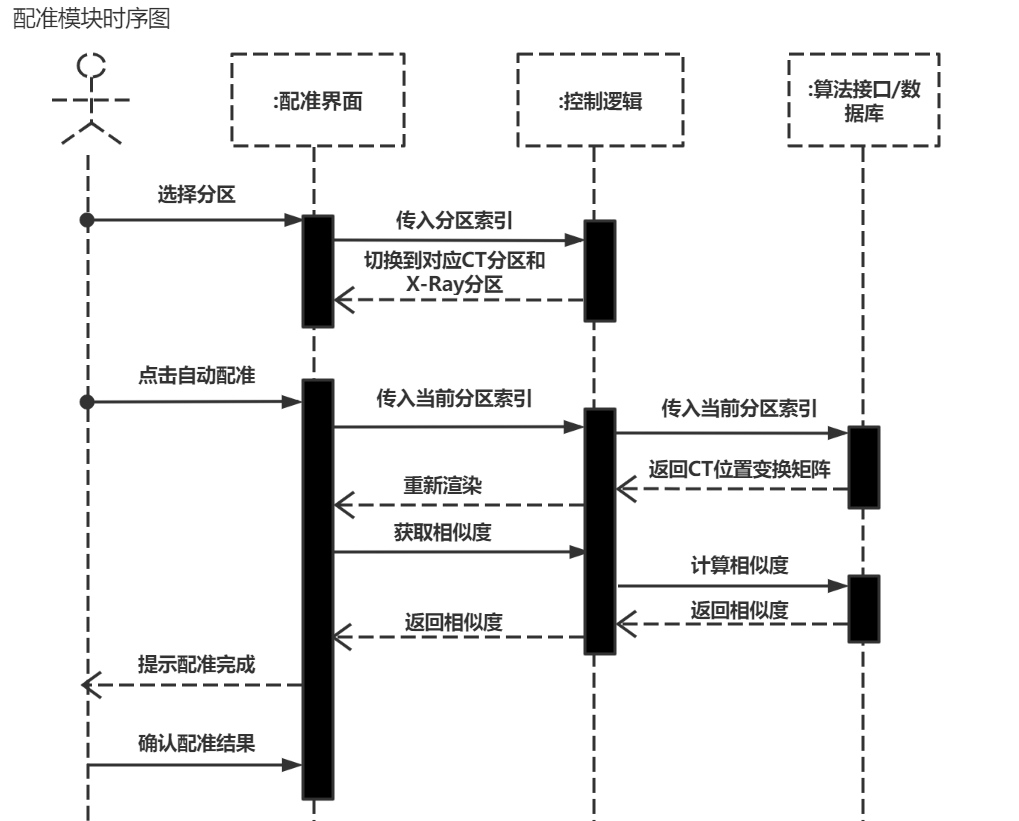
配准功能是本软件顺序流程的一个步骤，上一步为X-Ray分区，下一步为定位功能。分别用配准功能在软件内的流程图、配准功能系统用例图和时序图来展示本功能的程序逻辑。



规划功能在软件内的流程图



配准功能系统用例图



配准功能时序图

## 接口

配准功能包含CtRegistrationWidget、spine\_openGL\_widget两个主要类。其中重要接口详细说明如下：

**12.8.1. CtRegistrationWidget**

/\*\*

\* @brief The CtRegistrationWidget class

\* 规划功能主窗口

/\*\*

/\*\*

\* @brief initUI

\* 初始化UI

\*/

void initUI();

/\*\*

\* @brief updateWinLevel

\* 更新窗宽/窗位

\*/

void updateWinLevel(int c\_x, int c\_y)

/\*\*

\* @brief SelectAndChange3dColor

\* 判断是否已配准

\*/

void SelectAndChange3dColor();

/\*\*

\* @brief resetColor

\* 改变ct颜色

\*/

void resetColor();

/\*\*

\* @brief resetSlider

\* 重置ct颜色

\*/

void resetSlider();

/\*\*

\* @brief getXRayStr

\* 重置滑条位置

\*/

QString getXRayStr();

**12.8.2. spine\_openGL\_widget**

/\*\*

\* @brief The spine\_openGL\_widget class

\* 规划功能主窗口

/\*\*

/\*\*

\* @brief RestInterface

\* 事件回调

\*/

static void EventCallBack(vtkObject\* object, unsigned long event, void\* clientdata, void\* calldata);

/\*\*

\* @brief leftMousePress

\* 鼠标左键按下响应

\*/

void \_leftMousePress();

/\*\*

\* @brief leftMousePress

\* 鼠标右键按下响应

\*/

void \_leftMousePress();

/\*\*

\* @brief RestInterface 鼠标移动响应

\* @return 是否成功进入界面，ui全部复位

\*/

void \_rightMousePress();

/\*\*

\* @brief leftMouseRelease

\* 鼠标左键释放响应

\*/

void \_leftMouseRelease();

/\*\*

\* @brief rightMouseRelease

\* 鼠标右键释放响应

\*/

void \_rightMouseRelease();

## 存储分配

配准功能本地文件存储内容：

1. 配准的列表对应项和对应的相似度值数据保存在当前病例目录下 ："./registration/RegistrationAList.xml"文件；
2. 配准的所有的MatchMat数据保存在当前病例目录下 ："./registration/RegistartionAMatchMat.xml"文件；
3. 配准的正侧位X光的值数据保存在当前病例目录下 ："./registration/RegistartionAXRayValue.xml"文件。

## 限制条件

1. 至少有一节椎体被配准；

## 测试要点

1. 选中椎体自动配准；
2. 配准后还原，查看是否返回原始姿态；
3. 多次配准查看相似度是否变化；
4. CT颜色修改，窗宽窗位调整及重置测试；
5. X光片小球显示与隐藏、滤波开启与关闭测试；
6. CT与X光对比度调节；
7. 正位或侧位分别选中微调；
8. 直接点下一步，有提示；
9. 自动配准后，点下一步，数据保存正确性验证；
10. 自动配准后，手动微调，点下一步，有提示；
11. 配准菜单高亮显示；
12. 仅手动调整，点下一步，有提示；
13. 手动调整，点确定，点下一步。

# 定位功能设计说明

## 功能描述

1. 定位功能是本软件最后一个步骤，是软件最终输出。定位功能将直接发送数据到导引软件，是软件的重要功能。定位功能将规划功能的规划数据和配准功能的配准结果进行整合，把目标螺钉的实际位置信息发送出去；
2. 定位功能在软件中属于常驻内存、不可重入，属于顺序处理且代码没有覆盖要求。本功能与其他功能耦合性一般：和规划功能复用影像的预览和螺钉数据、通过本地文件的形式获取配准功能的配准结果、利用信号槽的方式和通讯功能数据对接；
3. 再下发通道前，软件会先初步校验配准结果的准确性。
4. 功能内部代码增加边缘性检测，详见接口部分。

## 功能

软件处于配准功能，且在任务栏中点击【下一步】软件将进入定位功能。

定位功能分为三维数据展示区、规划螺钉列表区、定位功能按键区。

定位功能详细功能：

1. 软件处于配准功能，且在任务栏中点击【下一步】，软件将进入定位功能；
2. 定位功能分为三维数据展示区、定位螺钉列表区、定位功能按键区；
3. 三维数据展示区实现医学影像四视图的常用功能和视图；
4. 规划螺钉列表区展示当前已规划螺钉的详细信息（序号、名称、直径、长度、水平夹角、类型、颜色、显示状态）；
5. 软件处于定位功能，在任务栏中点击【上一步】，软件将进入配准功能。

****

定位功能功能IPO图

## 性能

无。

## 输入项目

定位功能螺钉信息和影像信息跟规划功能复用，其可视化无额外输入。其余输入如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | 配准结果文件 | 无 | dat | 不为空 | 自动输入 | 配准功能保存的数据 |

## 输出项目

定位功能-全功能输出项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输出方式** | **备注** |
| **1** | 螺钉信息 | 无 | QBarry | 不为空 | 信号槽发送 | 点击发送按钮和，推送螺钉信息 |

## 算法

**13.6.1. 计算螺钉术中位置**

算法目的：

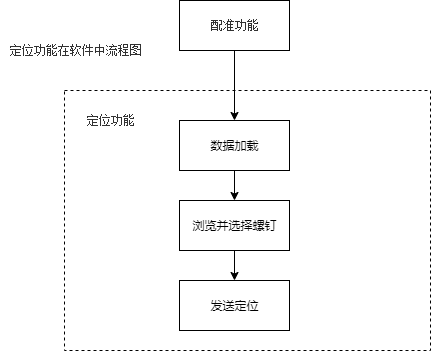
将规划功能中的术前规划数据变为术中规划数据。

算法流程：

1. 得到选择螺钉所在椎体的配准结果；
2. 得到选择螺钉所在椎体的旋转、镜像信息；
3. 得到选择螺钉的术前位置；
4. 计算螺钉的术中位置。

## 程序逻辑

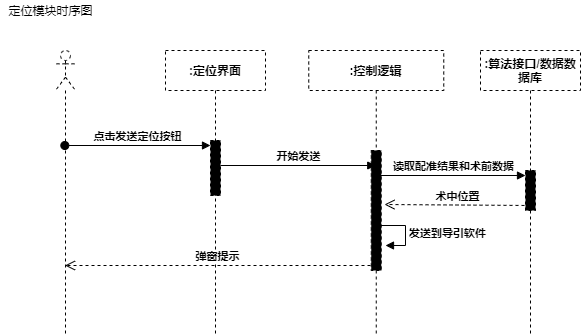
定位功能是本软件顺序流程的最后步骤，上一步为配准功能。分别用定位功能在软件内的流程图、定位功能系统用例图、定位功能时序图来展示本功能的程序逻辑。



定位功能在软件内的流程图



定位功能系统用例图



定位功能时序图

## 接口

规划功能包含ChannelPositioningWidget、FourViewManager、LineManager三个主要类。其中重要接口详细说明如下：

**13.8.1. ChannelPositioningWidget**

/\*\*

\* @brief The ChannelPositioningWidget class

\* 定位功能主窗口

/\*\*

/\*\*

\* @brief RestInterface 进入界面，ui全部复位

\* @return 是否成功进入界面，ui全部复位

\*/

bool RestInterface() *override*;

/\*\*

\* @brief VerificationSelfData 离开（下一步），验证数据完整

\* @return 是否成功

\*/

bool VerificationSelfData() *override*;

/\*\*

\* @brief WriteSelfModularData 离开（上、下一步），保存数据

\* @return 是否成功离开（上、下一步），保存数据

\*/

bool WriteSelfModularData() *override*;

/\*\*

\* @brief ReadSelfModularData 进入界面，读取本功能保存数据

\* @return 是否成功进入界面，读取本功能保存数据

\*/

bool ReadSelfModularData() *override*;

/\*\*

\* @brief ReadOtherModularData 进入界面，读取其他功能保存数据

\*/

bool ReadOtherModularData() *override*;

**13.8.2. FourViewManager**

详见规划功能第8节。

**13.8.3. LineManager**

详见规划功能第8节。

## 存储分配

定位无本地文件存储内容。定位功能内存存储内容：

1. 已规划螺钉数据（全局唯一）: ScrewTabelModel ::Instance()。

## 限制条件

1. 进入功能需确保至少有一节椎体完成配准；
2. 只能修改螺钉显示状态；
3. 软件若未和导引软件通讯，则无法发送螺钉信息。

## 测试要点

1. 2D图与3D图显示是否正常；
2. 2D图鼠标交互操作，左键滚轮查看图层、中键长按平移、右键窗宽窗位调整、ctrl+右键放大缩小；
3. 3D图鼠标交互操作，左键旋转、中键长按平移、右键窗宽窗位调整、ctrl+右键及滚轮放大缩小；
4. 3D图窗宽窗位重置，颜色修改，渲染方式修改；
5. 螺钉信息查看是否正确；
6. 螺钉的显示与隐藏，批量显示与隐藏；
7. 定位发送验证；
8. 定位菜单高亮显示；
9. 点击上一步，返回配准页面。

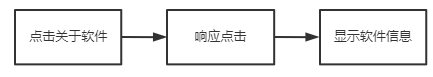
# 软件信息功能设计说明

## 功能描述

1. 软件信息功能管理软件基本信息，与其它功能耦合度较小。

## 功能

1. 点击软件顶部工具栏的关于按钮，弹出消息框显示软件基本信息（软件名称、软件版本、完整版本、注册证书编号、说明书编制日期）和软件厂商信息（公司名称、邮箱、生产许可证号、生产地址、注册地址）。



软件信息功能功能IPO图

## 性能

无。

## 输入项目

无。

## 输出项目

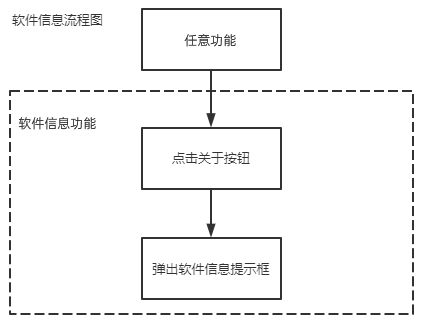
无。

## 算法

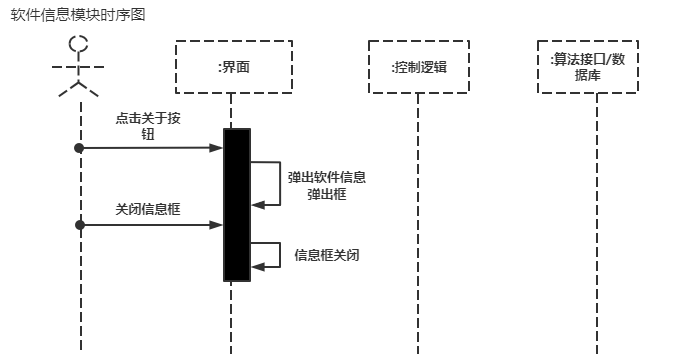
无。

## 程序逻辑

软件信息功能用流程图来展示本功能的程序逻辑。



软件信息功能在软件内的流程图



软件信息功能时序图

## 接口

规划功能包含AboutDialog这个类。其中重要接口详细说明如下：

**14.8.1. AboutDialog**

/\*\*

\* @brief The AboutDialog class

\*

/\*\*

/\*\*

\* @brief AboutDialog

\* 初始化界面并加载软件信息

\*/

void AboutDialog()

## 存储分配

无。

## 限制条件

无。

## 测试要点

1. 关于页面正常显示与关闭；
2. 关于页面信息是否完整，版本、公司信息、软件名称等。

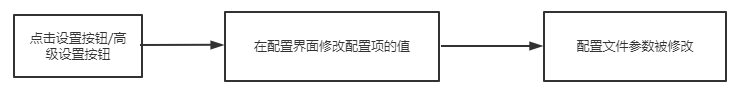
# 配置功能设计说明

## 功能描述

1. 配置功能通过管理软件配置文件实现对软件配置参数的管理与其它功能耦合度较小。

## 功能

1. 软件处于运行状态，点击软件顶部工具栏的设置/高级设置按钮，系统会弹出设置界面。修改设置界面上设置项对应的值，然后点击确定，配置文件会相应更新。



配置功能功能IPO图

## 性能

无。

## 输入项目

配置功能输入项目为系统配置文件和用户修改值。

## 输出项目

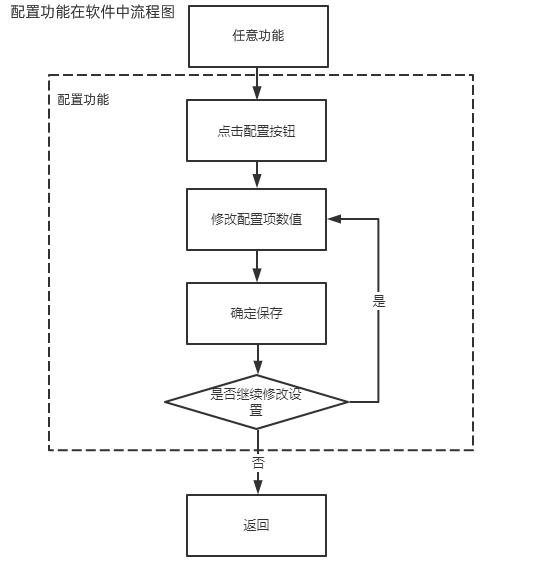
配置功能相关输出为对系统配置文件参数的修改。

## 算法

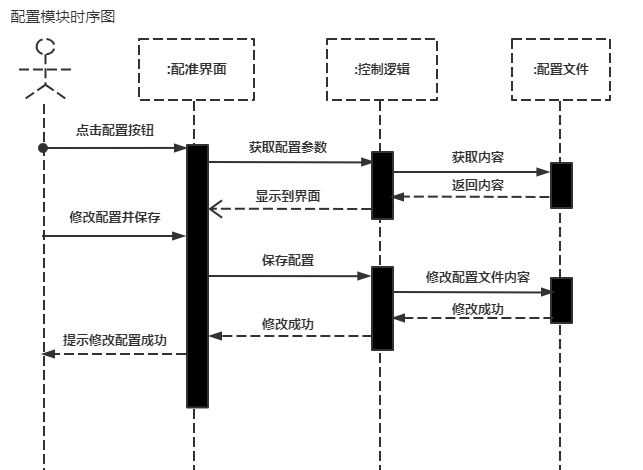
无。

## 程序逻辑

配置功能流程图来展示本功能的程序逻辑。



配置功能在软件内的流程图



配置功能时序图

## 接口

配置功能包含SetDialog、OtherSetDialog两个主要类。其中重要接口详细说明如下：

**15.8.1. SetDialog**

/\*\*

\* @brief The SetDialog class

\*

/\*\*

/\*\*

\* @brief setDialog

\* 构造时加载config.ini

\*/

void setDialog()

/\*\*

\* @brief init

\* 初始化UI

\*/

void init()

/\*\*

\* @brief okClicked

\* 将改变项目保存到config.ini

\*/

void okClicked()

**15.8.2. OtherSetDialog**

/\*\*

\* @brief The OtherSetDialog class

\*

/\*\*

/\*\*

\* @brief setDialog

\* 构造时加载config.ini

\*/

void setDialog()

/\*\*

\* @brief init

\* 初始化UI

\*/

void init()

/\*\*

\* @brief okClicked

\* 将改变项目保存到config.ini

\*/

void okClicked()

## 存储分配

配置功能本地文件存储内容：

配置文件路径："./config.ini"文件。

## 限制条件

1. 对参数数值范围有限制；
2. 对设置的参数值类型有限制；

## 测试要点

1. 参数配置页面正常打开与关闭；
2. 所有可修改参数有效验证无效验证；
3. 相机参数是否会自动更新。

# 日志记录功能设计说明

## 功能描述

1. 日志记录功能用来记录关键操作及关键结果，通过记录异常和警告来分析处理不可预见的错误，保证了软件网络安全的审核和网络节点鉴别能力；
2. 日志记录功能在软件中属于常驻内存、不可重入，不属于顺序处理且代码有覆盖要求。本功能与其他功能耦合性较高，软件所有功能均与日志功能有数据交换。

## 功能

1. 软件开启后记录关键操作步骤和异常信息；
2. 软件关闭时最后进入日志记录功能，保存关闭信息。

****

日志记录功能功能IPO图

## 性能

日志记录功能所有操作在后台进行，及时相应时间在毫秒级。

## 输入项目

日志记录功能输入项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | 操作流水 | 无 | string | 不为空 | 自动输入 | 软件关键操作步骤 |
| **2** | 警告、错误信息 | 无 | string | 不为空 | 自动输入 | 软件警告、错误信息 |

## 输出项目

日志记录功能输出项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输出方式** | **备注** |
| **1** | 日志条目 | 无 | log | 不为空 | 自动保存 | 软件运行后，日志实时更新 |

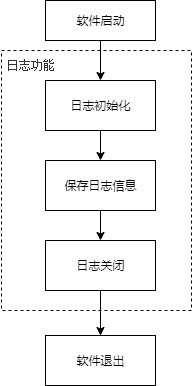
## 算法

无。

## 程序逻辑

日志记录功能独立于其他所有功能随软件启动而开启，随软件关闭而结束。

用日志记录功能在软件内的流程图来展示本功能的程序逻辑。



日志记录功能在软件内的流程图

## 接口

日志记录功能包含TextDecorator、Logger两个主要类。其中重要接口详细说明如下：

**16.8.1. TextDecorator**

// ============================================================

// time in 24 hours hh:mm:ss format

// ============================================================

*static* std::string TimeStamp();

// ============================================================

// date YYYY:MM:DD format

// ============================================================

*static* std::string DateStamp();

// ============================================================

// Here is an example of a simple log decorator, you can define your own decorator

// ============================================================

*class* TextDecorator;

**16.8.2. Logger**

// ============================================================

// New Logger with a new log file and new log title

// ============================================================

*class* Logger;

/\*\*

\* @brief Log 增加日志

\* @param p\_entry 日志内容

\*/

void Log(*const* std::string& p\_entry);

## 存储分配

日志记录功能无内存存储内容，日志记录功能本地文件存储内容：

1. 日志文件保存在当前软件根目录下：“./ 3dsoft.log”文件。

## 限制条件

无。

## 测试要点

1. 是否正常生成日志文件；
2. 是否正常输出日志文件；
3. 日志文件是否存在乱码；
4. 日志文件是否记录了用户所有操作记录；
5. 日志记录时间及内容准确性验证；
6. 日志记录用户行为可追溯性验证。

# 日志导出功能设计说明

## 功能描述

1. 日志导出功能用来将记录的日志文件压缩并加密导出到U盘，保证了软件网络安全的健康数据存储保密性能力；
2. 日志导出功能在软件中属于常驻内存、可重入，不属于顺序处理且代码有覆盖要求。本功能与其他无耦合。

## 功能

1. 软件成功登录后提供导出日志功能按钮；
2. 日志导出功能可以判断U盘是否存在，用进度条展示导出进度。对导出到U盘的日志压缩包进行加密处理，导出过程中无法关闭导出窗口；
3. 软件未登录时无法提供导出功能。

****

日志导出功能功能IPO图

## 性能

无。

## 输入项目

日志导出功能输入项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输入方式** | **备注** |
| **1** | 本地日志 | 无 | .dat | 不为空 | 自动输入 | 本地已保存日志 |
| **2** | U盘名称 | 无 | string | 不为空 | 鼠标选择 | 选择导出的U盘 |

## 输出项目

日志导出功能输出项目如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **名称** | **标识** | **类型** | **有效范围** | **输出方式** | **备注** |
| **1** | 加密日志文件 | 无 | Zip | 不为空 | 自动保存 | 导出到U盘的加密日志文件 |

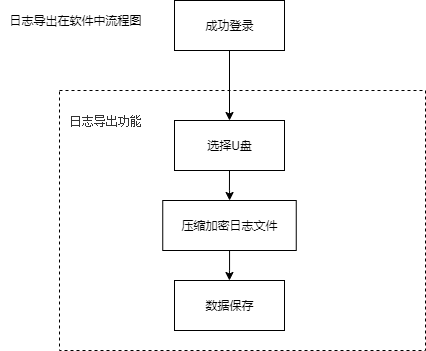
## 算法

无。

## 程序逻辑

日志导出功能独立于其他所有功能随软件登录而开启，随软件关闭而结束。

用日志导出功能在软件内的流程图来展示本功能的程序逻辑。



日志导出功能在软件内的流程图

## 接口

日志导出功能包含ExportDialog主要类。其中重要接口详细说明如下：

**17.8.1. ExportDialog**

/\*\*

\* @brief The ExportDialog class

\* 日志导出功能主要类

\*/

*class* ExportDialog;

/\*\*

\* @brief closeClicked

\* 关闭日志导出弹窗

\*/

void *closeClicked*() *override*;

/\*\*

\* @brief okClicked

\* 开始导出日志

\*/

void *okClicked*() *override*;

/\*\*

\* @brief cancelClicked

\* 开启界面点击

\*/

void *cancelClicked*() *override*;

/\*\*

\* @brief accept

\* 成功导出日志

\*/

void *accept*() *override*;

/\*\*

\* @brief reject

\* 取消导出日志

\*/

void *reject*() *override*;

## 存储分配

日志导出功能无内存存储内容，日志导出功能本地文件存储内容：

1. 日志导出文件保存在选择U盘根目录下：“./ log.zip”文件。

## 限制条件

1. 随软件登录而开启，随软件关闭而结束；
2. 本地必须插入并手动选择U盘后才能开始导出。

## 测试要点

1. 是否正常识别单个或多个U盘；
2. 开始导出后进度条是否正常；
3. 导出过程中是否可以操作软件；
4. 是否可以正常导出；
5. 导出日志文件是否加密；
6. 是否可以退出导出。

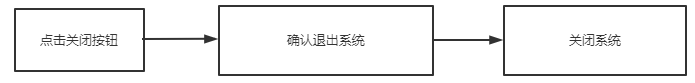
# 系统退出功能设计说明

## 功能描述

1. 系统退出功能用来退出系统并关机；
2. 系统退出功能在软件中属于常驻内存、可重入，不属于顺序处理且代码有覆盖要求。本功能与其他无耦合。

## 功能

1. 点击顶部工具栏的退出按钮，系统会弹出提示框提示是否退出系统，点击确认则系统关闭，点击取消按钮则可取消退出系统操作。



系统推出功能功能IPO图

## 性能

无。

## 输入项目

无。

## 输出项目

无。

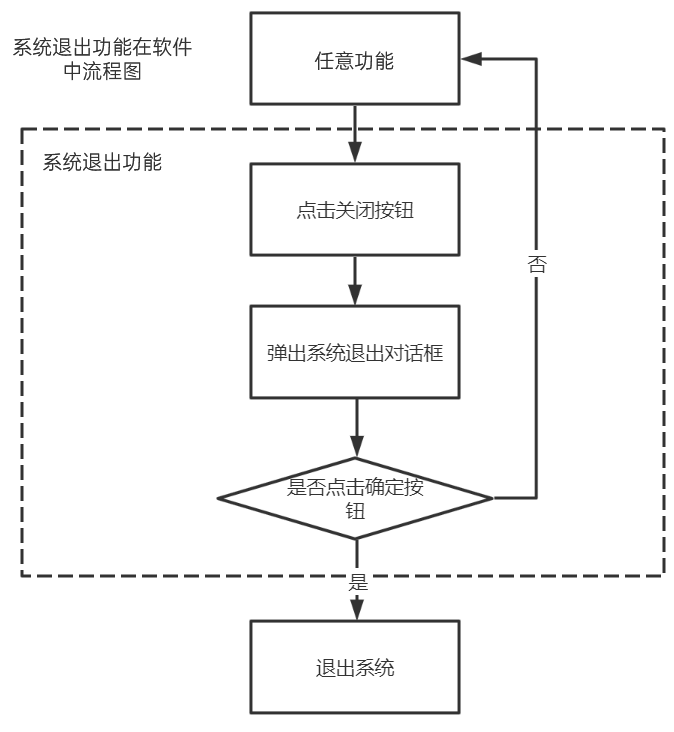
## 算法

无。

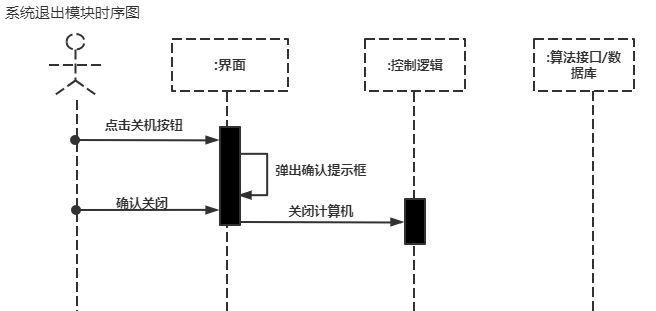
## 程序逻辑

系统退出功能独立于其他所有功能随软件登录而开启，随软件关闭而结束。

用系统退出功能在软件内的流程图来展示本功能的程序逻辑。



系统退出功能在软件内的流程图



系统退出功能时序图

## 接口

系统退出功能包含TitleBarWidget主要类。其中重要接口详细说明如下：

**18.8.1. TitleBarWidget**

/\*\*

\* @brief The TitleBarWidget class

\*

\*/

class TitleBarWidget;

/\*\*

\* @brief on\_pButton\_help\_clicked

\* 帮助按钮响应

\*/

void on\_pButton\_help\_clicked()

/\*\*

\* @brief on\_pButton\_log\_clicked

\* 日志按键响应

\*/

void on\_pButton\_log\_clicked()

/\*\*

\* @brief on\_pButton\_system\_clicked

\* 配准按键响应

\*/

void on\_pButton\_system\_clicked()

/\*\*

\* @brief on\_pButton\_other\_system\_clicked

\* 高级配准按键响应

\*/

void on\_pButton\_other\_system\_clicked()

/\*\*

\* @brief on\_pButton\_shutdown\_clicked

\* 退出按键响应

\*/

void on\_pButton\_shutdown\_clicked()

/\*\*

\* @brief on\_pButton\_arm\_clicked

\* 机械臂连接响应

\*/

void on\_pButton\_arm\_clicked()

## 存储分配

无。

## 限制条件

无。

## 测试要点

1. 关机按钮是否有效；
2. 是否关机页面内容是否合理；
3. 是否关机页面，点确认，系统关机；
4. 是否关机页面，点取消，返回原来页面。