

可视化使用说明书

目录

1. 环境要求.....	2
1.1. Easyx 配置介绍.....	2
2. 使用步骤.....	3
2.1. 获取数据.....	3
2.2. 进行可视化仿真.....	3
2.3. 使用 Python 脚本自动获取数据(额外功能).....	4
2.4. 使用自动修改 Seed 和 Scenario 脚本.....	5
2.5. 使用自动修改修改 DecisionMaker 脚本.....	5
3. 目前功能以及待开发功能清单.....	6
4. 各个获取数据的接口说明.....	7
4.1. AGV 运行过程以及其携带的 Container.....	7
4.2. 计算 AGV 等待 QC 装货的数量.....	7
4.3. 计算 AGV 等待 YC 操作的列表以及每个堆场目前堆存 Container 的数量.....	7
4.4. 计算 AGV 空闲的数量, Delayed 以及 Waiting 船舶数量.....	7
4.5. 计算在堆场中每条船待装载的 Container 的数量.....	8
4.6. 计算当前泊位停靠的船.....	8
5. FQA.....	8

1. 环境要求

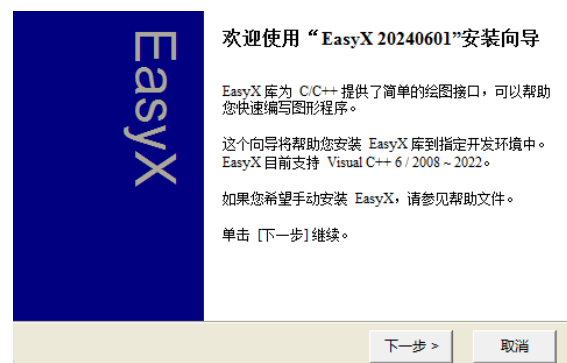
Windows 系统, .NET 7.0, C++11 及以上, Easyx 库, Visual Studio 2022

1.1. Easyx 配置介绍

打开 Easyx 官网 <https://easyx.cn/>



打开下载好的文件, 点击下一步



选择当前电脑 Visual Studio 适配的版本, 等待安装完成即可。

执行安装
请针对您所用的开发平台, 选择安装或卸载。



2. 使用步骤

2.1. 获取数据

打开从 GitHub 上下载的文件

名称	修改日期	类型
Calculated_Data	2024/10/9 16:24	文件夹
Calculation_Program	2024/10/9 15:13	文件夹
Image_Materials	2024/10/9 16:14	文件夹
visualization_program	2024/10/9 15:10	文件夹

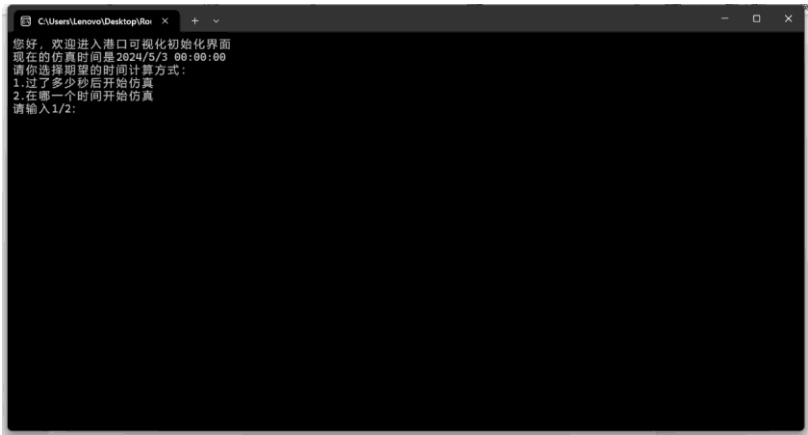
点击 Calculation_Program

AGV运行过程以及其携带的Container	2024/10/9 15:26	文件夹
计算AGV等待QC装货的数量	2024/10/9 15:28	文件夹
计算AGV等待YC操作的列表以及每个堆场目前...	2024/10/9 15:30	文件夹
计算AGV空闲的数量	2024/10/9 15:35	文件夹
计算Container堆存时待运输的Vesse的数量	2024/10/9 15:36	文件夹
计算当前泊位停靠的船	2024/10/9 15:38	文件夹

分别点击文件夹进去，打开 Visual Studio 分别运行 6 次，获取完成后会自动中断。

2.2. 进行可视化仿真








点击 Visualization_program, 打开 Visual Studio 开始运行





成功打开这个界面就说明环境以及配置完毕，可以开始使用了

2.3. 使用 Python 脚本自动获取数据(额外功能)

点击 Python Script

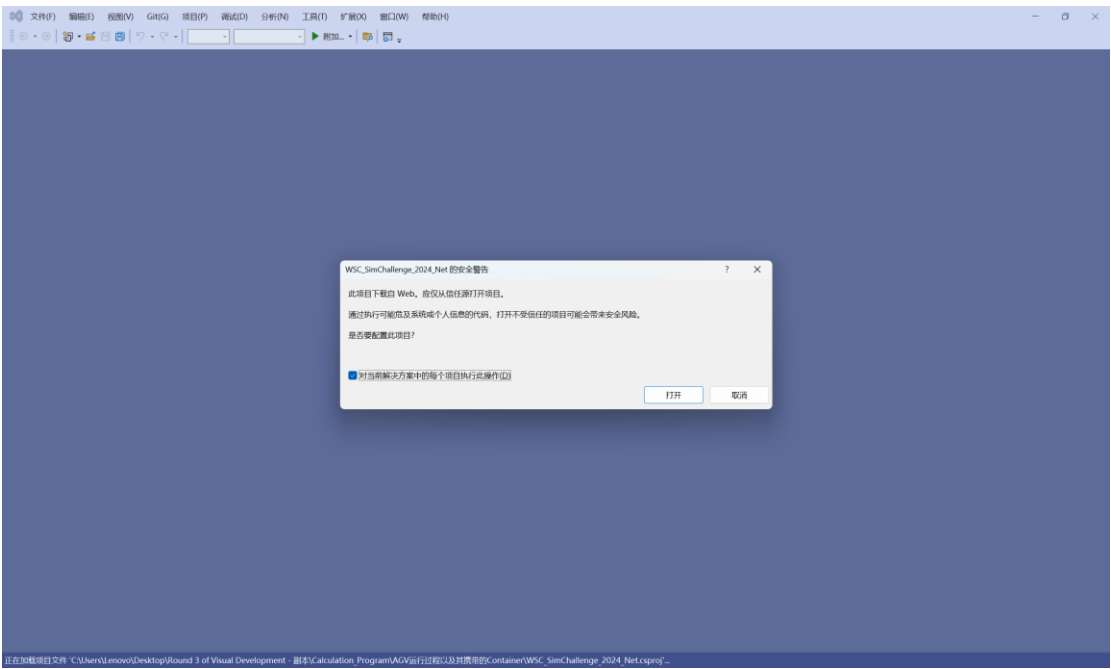
 Calculated_Data	2024/10/10 10:54	文件夹	
 Calculation_Program	2024/10/14 14:06	文件夹	
 Image_Materials	2024/10/14 19:32	文件夹	
 Python Script	2024/10/15 19:04	文件夹	
 Visualization_program	2024/10/9 15:10	文件夹	
 README.md	2024/10/10 11:21	Markdown 源文件	1 KB
 可视化开发使用说明书.pdf	2024/10/14 14:26	Microsoft Edge PD...	285 KB

点击应用程序即可开始获取数据并开启仿真（通常需要等待 10-20min）

 一键开启数据采集and可视化仿真.exe	2024/10/15 18:48	应用程序	15,384 KB
 一键开启数据采集and可视化仿真.py	2024/10/15 17:02	Python 源文件	6 KB

使用此脚本时，需把每个 sln 文件打开过一次，确保不会出现上面这个弹窗!!!

使用此脚本时获取数据时，不能切屏，最好连鼠标也不要移动!!!





2.4. 使用自动修改 Seed 和 Scenario 脚本

点击 Python Script








 Calculated_Data	2024/10/10 10:54	文件夹	
 Calculation_Program	2024/10/14 14:06	文件夹	
 Image_Materials	2024/10/14 19:32	文件夹	
 Python Script	2024/10/15 19:04	文件夹	
 Visualization_program	2024/10/9 15:10	文件夹	
 README.md	2024/10/10 11:21	Markdown 源文件	1 KB
 可视化开发使用说明书.pdf	2024/10/14 14:26	Microsoft Edge PD...	285 KB

点击应用程序开始使用

 修改Seed和Scenario.exe	2024/10/17 16:30	应用程序	6,830 KB
 修改Seed和Scenario.py	2024/10/17 16:30	Python 源文件	4 KB

2.5. 使用自动修改修改 DecisionMaker 脚本








点击 Python Script Materials

 Calculated_Data	2024/10/17 16:54	文件夹	
 Calculation_Program	2024/10/17 16:30	文件夹	
 Image_Materials	2024/10/17 16:30	文件夹	
 Python Script	2024/10/18 13:16	文件夹	
 Python Script Materials	2024/10/18 13:14	文件夹	
 Visualization_program	2024/10/17 16:30	文件夹	
 README.md	2024/10/17 16:30	Markdown 源文件	1 KB



再点击修改 DecisionMaker，把修改好的文件放进此文件夹中

 修改DecisionMaker	2024/10/18 12:59	文件夹	
 修改Seed和Scenario	2024/10/17 16:30	文件夹	

完成后回到最开始的文件夹，点击 Python Script

 Calculated_Data	2024/10/10 10:54	文件夹	
 Calculation_Program	2024/10/14 14:06	文件夹	
 Image_Materials	2024/10/14 19:32	文件夹	
 Python Script	2024/10/15 19:04	文件夹	
 Visualization_program	2024/10/9 15:10	文件夹	
 README.md	2024/10/10 11:21	Markdown 源文件	1 KB
 可视化开发使用说明书.pdf	2024/10/14 14:26	Microsoft Edge PD...	285 KB

点击应用程序开始使用

 修改DecisionMaker.exe	2024/10/18 13:15	应用程序	6,830 KB
 修改DecisionMaker.py	2024/10/18 13:14	Python 源文件	3 KB

3. 目前功能以及待开发功能清单

- (1) AGV 整个运行过程的模拟
- (2) Container 在 Yard Block 堆存数量
- (3) AGV 在 YC 等待 Stacking 的队列
- (4) AGV 在 YC 等待 Unstacking 的队列
- (5) AGV 在 QC 等待 Loading 的队列
- (6) 仿真时间显示
- (7) 控制开始仿真时间功能
- (8) 数据加上时间
- (9) 显示堆场中 Container 运输到的分别所属 Loading 的 Vessel
- (10) AGV 装载集装箱显示去哪条船
- (11) AGV 卸货显示来自那条船
- (12) 空闲 AGV 总数量
- (13) 用颜色区分 AGV 中 Stack 和 Unstack
- (14) 加快渲染速度
- (15) 目前 Berth 停的 Vessel
- (16) 快速修改 seed 还有情景
- (17) 一次性覆盖 DecisionMaker
- (18) QC 运行过程显示
- (19) YC 运行过程显示
- (20) Vessel 等待队列
- (21) Vessel 超时队列
- (22) 更新图片素材
- (23) 更换图像库

4. 各个获取数据的接口说明

4.1. AGV 运行过程以及其携带的 Container

每一行的第一列是时间，第二，三列是当前 AGV 的 X,Y 坐标，第四列是 0 代表了 AGV 非 Delivering 状态，1 代表了 DeliveringToYard 状态，2 代表了 DeliveringtoQuaySide 状态，第五列输出的是 AGV 当前装载的 Container 所属船的 ID，如果第四列是 1,则输出 Container 卸货船的 ID,如果第四列是 2，则输出 Container 装载船的 ID。

12 行一次循环。

X,Y 坐标变化设置：当 AGV.Picking.Start, AGV.DeliveringToYard.Start, AGV.DeliveringtoQuaySide.Start 时，开始以 AGV.speed 的值每秒变化坐标。

第四列 flag 变化设置：开始默认为 0，在 AGV.DeliveringToYard.Start 设置成 1, AGV.DeliveringtoQuaySide.Start 设置成 2，在 AGV.Picking.Start 设置成 0, AGV.HoldingatYard.AttemptToFinish, AGV.HoldingatYard.TryFinish 设置成 0, AGV.HoldingatQuaySide.AttemptToFinish, AGV.HoldingatQuaySide.TryFinish 设置成 0。

4.2. 计算 AGV 等待 QC 装货的数量

输出 QC 当前 AGV 等待 QC 来装货的列表。

12 行一次循环。

在 QC 类中新增列表 agvs, AGV.HoldingatQuaySide.Start 将当前 agv 导入所属 QC 的 agvs 列表, AGV.HoldingatQuaySide.AttemptToFinish, AGV.HoldingatQuaySide.TryFinish 在当前 agv 所属 QC 的 agvs 列表删除 agv。

4.3. 计算 AGV 等待 YC 操作的列表以及每个堆场目前堆存 Container 的数量

每一行的第一列是时间，第二列是当前堆场堆存集装箱的数量，第三列是当前堆场 AGV 等待 YC 操作的列表，第四列中 1 是 Stack, 2 是 UnStack。(第五，六……列依此类推)

16 行一个循环。

当前堆场堆存集装箱的数量设置：Container.BeingStacked.Depart 增加, Container.BeingUnstacked.TryStart 减少。

4.4. 计算 AGV 空闲的数量, Delayed 以及 Waiting 船舶数量

每一行第一列是时间，第二列 AGV 空闲的数量，第三列是 Delayed 船舶数量，第四行是 Waiting 船舶数量。

1 行一个循环。

AGV 空闲数量设置：AGV.BeingIdle.Start 增加, AGV.BeingIdle.AttemptToFinish, AGV.BeingIdle.TryFinish 减少。

Delayed 船舶设置: Vessel.Waiting.Start 添加进列表, Vessel.Berthing.Start 从列表移除, 如果在列表存在超过 2h(7200s), delayed_num 增加, 并从列表移除。
Waiting 船舶设置: Vessel.Waiting.Start 增加, Vessel.Berthing.Start 减少。

4.5. 计算在堆场中每条船待装载的 Container 的数量

每一行第一列是时间, 后面 30 列分别对应了每条船运输过来堆场 Container 的数量。

16 行一次循环。

每条船待装载的 Container 的数量设置: 在 YardBlock 类中增加

Dictionary<string, int> YB_Vessel, 在 Container.BeingStacked.Depart 增加, Container.BeingUnstacked.TryStart 减少。

4.6. 计算当前泊位停靠的船

输出当前泊位所停靠的船

4 行一次循环。

5. FQA