目录

[启动程序： 2](#_Toc458783484)

[运行模块（不分先后）： 2](#_Toc458783485)

[启动程序（不分先后）： 2](#_Toc458783486)

[下单 2](#_Toc458783487)

[小车取货架 4](#_Toc458783488)

[运货架到拣货台 4](#_Toc458783489)

[拣货 6](#_Toc458783490)

[拣货员拿到商品 6](#_Toc458783491)

[商品放入订单箱 6](#_Toc458783492)

[完成单个商品拣货 6](#_Toc458783493)

[货架完成拣货 9](#_Toc458783494)

[送货架回仓储区 9](#_Toc458783495)

[订单打包 10](#_Toc458783496)

[其他说明： 11](#_Toc458783497)

[关闭和启用路径 11](#_Toc458783498)

[单向路 12](#_Toc458783499)

[仓库显示设置 12](#_Toc458783500)

[增加路径 13](#_Toc458783501)

[增加节点 13](#_Toc458783502)

# 启动程序：

## 运行模块（不分先后）：

服务器（ViewServer），拣货台（ViewStation），小车（ViewDevice）

## 启动程序（不分先后）：

拣货台（开始），小车（发心跳），结果如图1：系统初始状态没有订单，拣货台初始会提示“暂无新订单” 。

模块之间通过TCP/IP通信，多线程收发和处理，所以多台小车和多个拣货台都可以放不同电脑上运行，同一个局域网内不需要做任何额外的配置，直接启动即可。

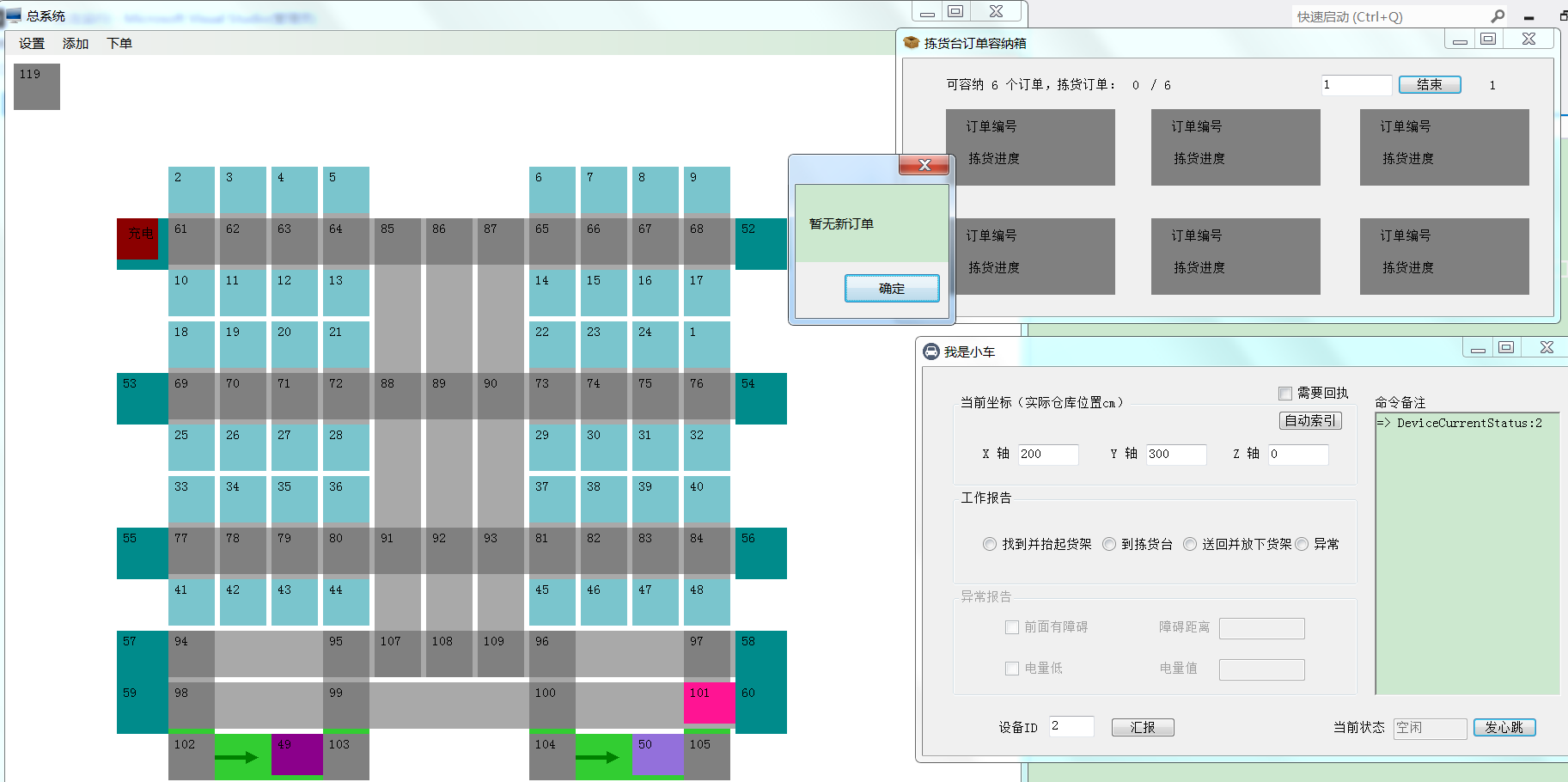


图1：程序启动后演示

# 下单

服务器端：订单应该是外界直接导入或提供接口给外部系统，所以这里通过服务器端展示独立的下单窗体，这里做了个简单的Demo来调用下单接口（如图2），另外，系统处于任何运行的状态，都可以通过订单窗体进行下单，其他系统也可以直接发包过来进行下单.

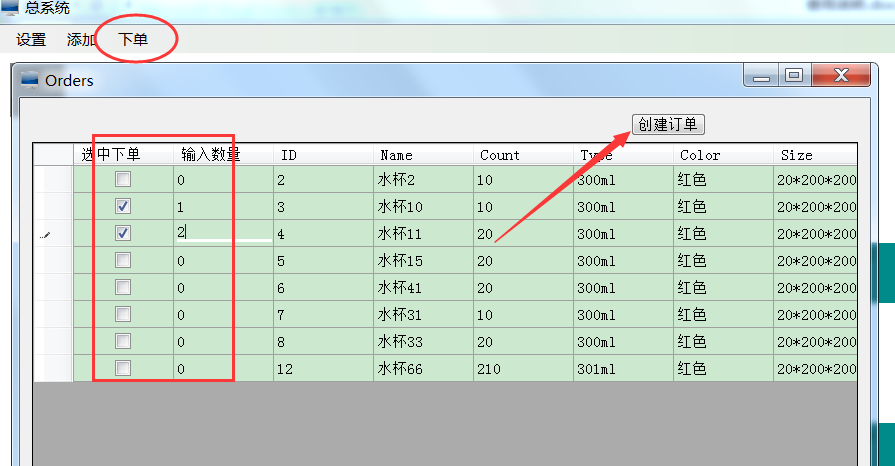


图2：下单窗体

服务器：（下单成功后）如图3

向拣货台发送订单信息：订单编号，商品数量

小车发送去取货架命令

图3：下单后小车和拣货台收到消息后的窗体

# 小车取货架

小车：收到去取货架的命令，包括了对应行走路线（仅发转弯时的节点）

边走边发心跳给系统，服务器会实时看到小车的位置

此处的“自动索引”按钮只将系统发来的转弯节点，依次作为当前坐标发送心跳给服务器，服务器则可以看到小车行走的路线，最终小车到货架下方（如图4）.

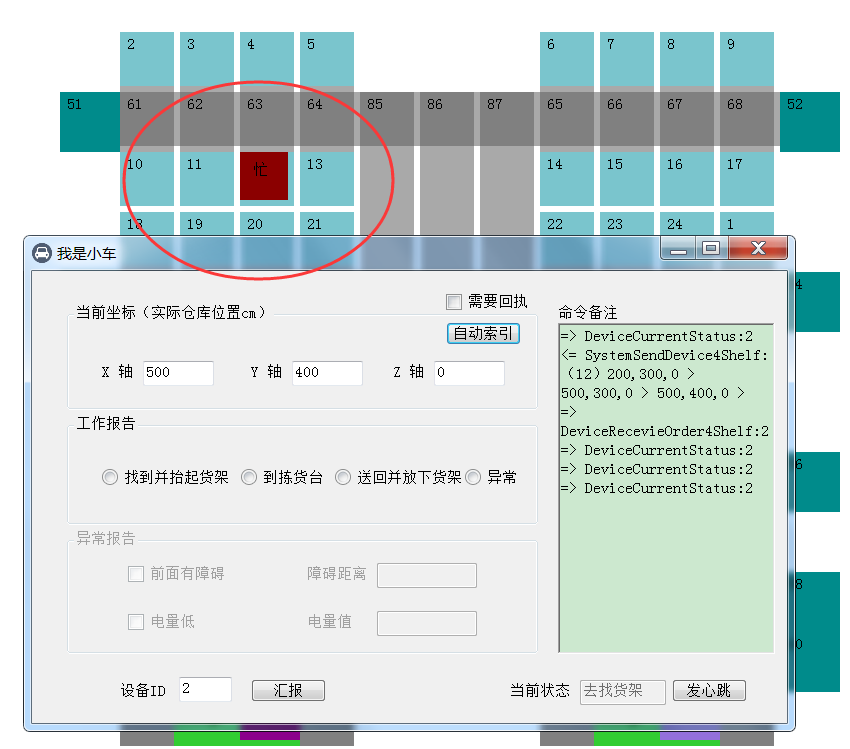


图4：小车找货架

# 运货架到拣货台

小车：到达货架位置

发消息给服务器告知（已经到达货架，并抬起货架）

服务器：收到抬起命令（如图5）

删除货架

修改小车标志为已有货架（屎黄色）

发送“去拣货台“命令给小车

小车：导航到拣货台（不断发心跳包）

发消息给服务器告知（已经到达拣货台）（如图6）



图5：小车抬起货架

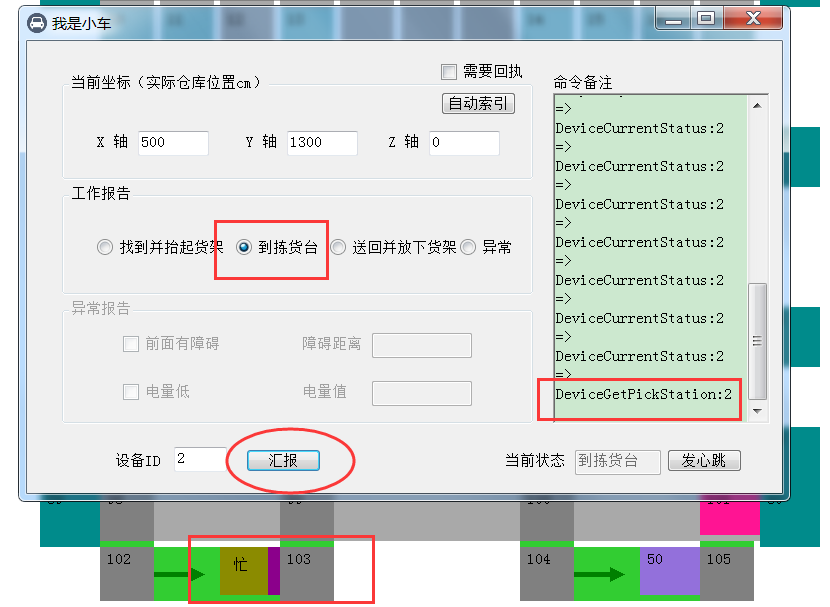


图6：小车到达拣货台

# 拣货

## 拣货员拿到商品

服务器：收到小车到拣货台命令

通知拣货台：商品名称和对应位置

拣货台：收到商品信息

显示：货架库位状态和商品名称

拣货员：看到商品信息

扫码：去货架将对应库位将商品取下，并扫码（手动输入条码，点击“拣货”）

另外：如果扫错商品不会有任何数据修改，窗体会得到提醒（如图7）

## 商品放入订单箱

服务器：收到商品条码信息

通知拣货台：商品对应订单位置

货架有其他待拣商品

有：若有不发任何命令

无：通知小车去别的地方，或回仓储区

拣货台：收到订单编号（如图8）

对应订单箱亮灯（不同颜色标注）

拣货员：看到不同颜色

将商品放入订单箱，并关闭订单灯（单击订单箱）

## 完成单个商品拣货

拣货台：拣货员关闭订单灯

订单箱内商品增加（如图9），并恢复待拣货状态

货架有其他待拣商品

有：显示下一个商品信息

无：提示换货架等待

另外：任何时候，可以随时下单，系统会将订单发给有空位的拣货台（如图10）

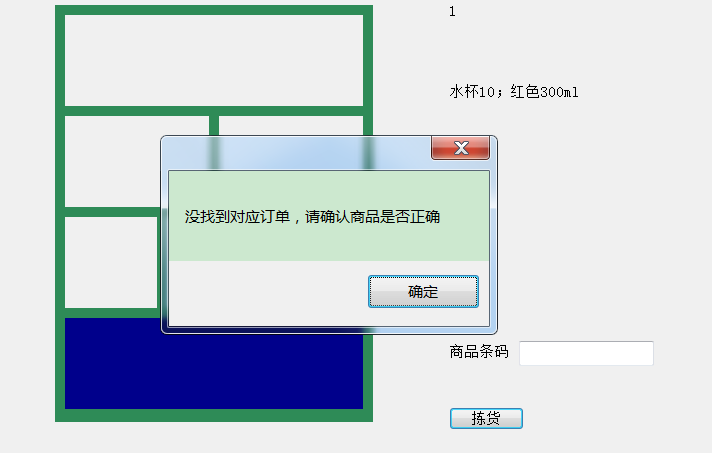


图7：商品扫码错误（输入错误的商品条码）



图8：订单箱点亮

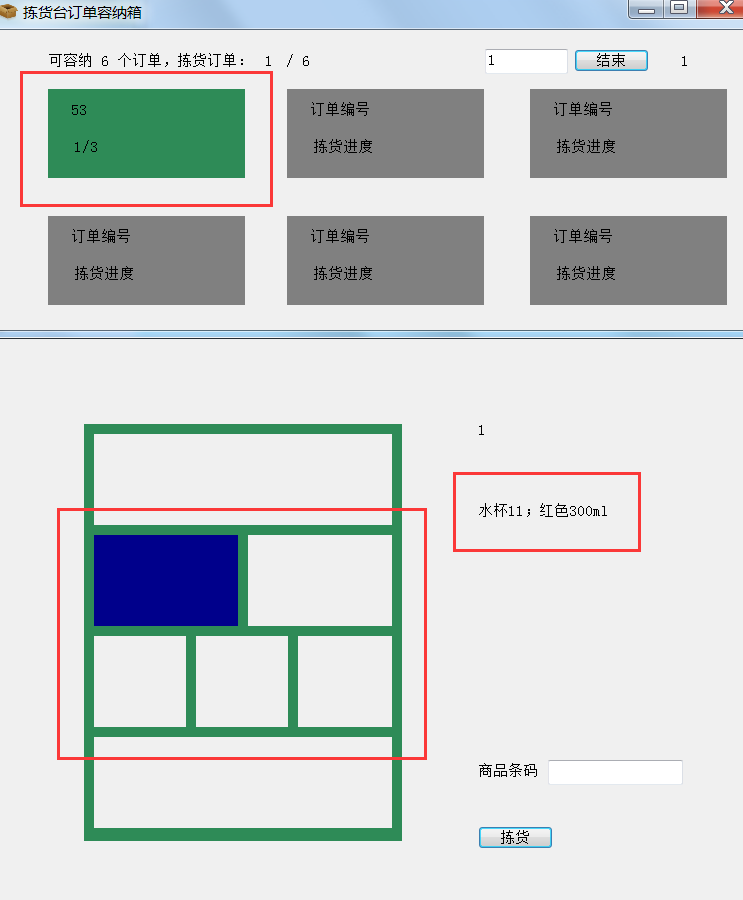


图9：拣货员将商品放入订单箱，并关灯（单击订单箱）

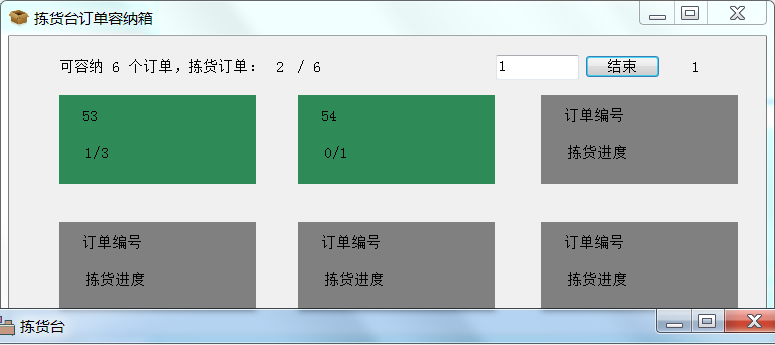


图10：新下单

# 货架完成拣货

当前货架在当前拣货台的最后一个商品被拣货台扫码后，小车收到送货架去别地方的命令（如图11）

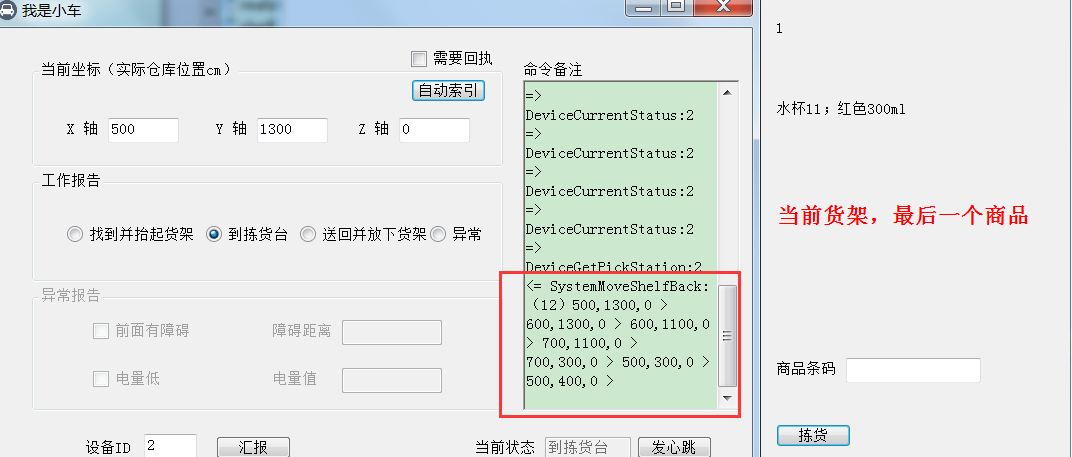


图11：当前货架拣货完成

# 送货架回仓储区

小车：接到运货架回仓储区命令（如图13）

按对应路径运回，并放下货架后，告知系统

系统：接到货架被运回消息

将货架显示

将小车设为可用（深红色）

如果有新的任务，则分配给小车

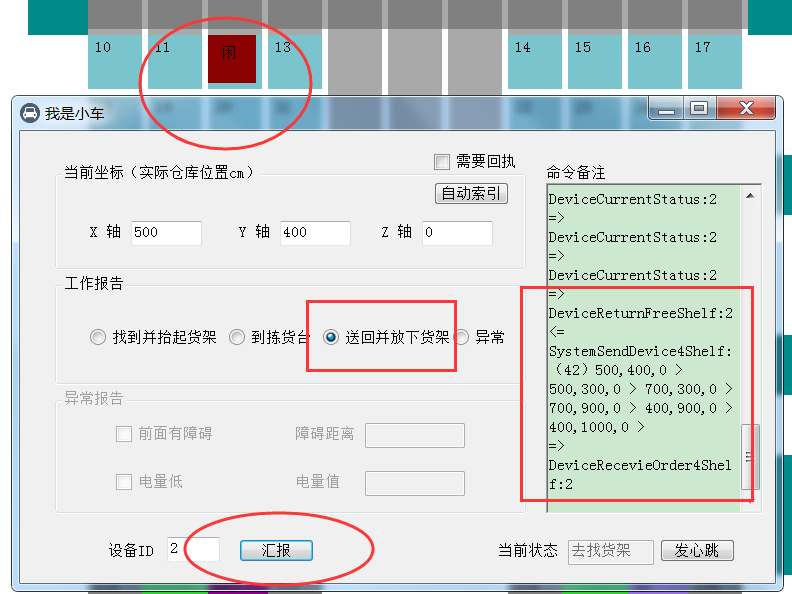


图13：小车送货架回仓储区

# 订单打包

订单商品拣货完成后，订单箱会用其他标志告知打包员（红色，如图12），打包员拿走订单箱后（双击），订单箱恢复待拣货状态。

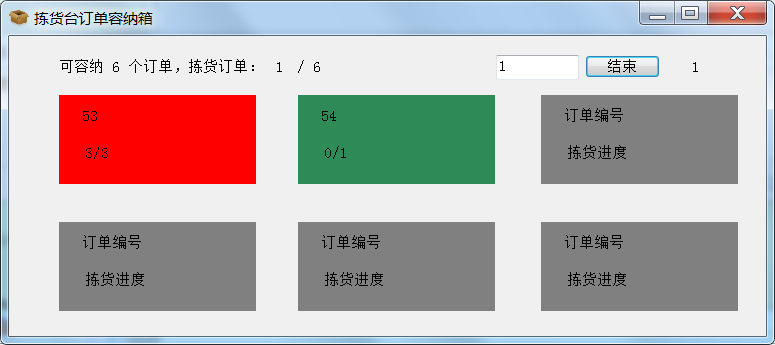


图12：订单完成拣货

# 小车充电

小车运货架回仓储区后，没有任务的情况下，会找到最近闲置的充电桩去充电。

# 其他说明：

## 关闭和启用路径

选中对应路径点击右键，关闭后红色标注，启用后恢复原来颜色（如图14 – 图16），关闭后路径规划时会避开，如果当前两节点没有路径可以到达，则会提示“不可达”。另外每次改后系统都会进行记录，下次启动仍然有效。



图14：关闭路径

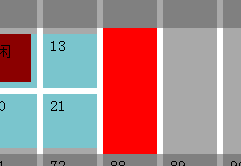


图15：路径关闭状态

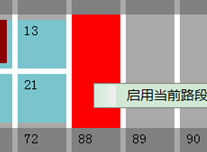


图16：启用路径

## 单向路

系统支持单向路，为了防止阻塞在拣货台，所以拣货台用了单向路设计，用带箭头绿色路径标识（如图17）

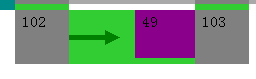


图17：拣货台单向路

## 仓库显示设置

可以对于地图显示的颜色和尺寸进行动态设置，并立即生效（如图18）。所有相关设置都是保存在配置文件，所以可以直接修改配置文件(Config.ini)

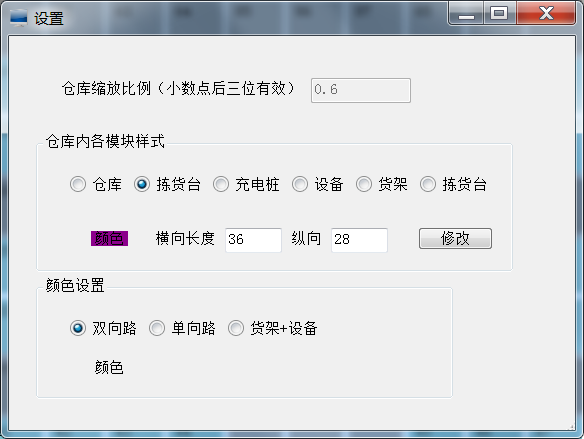


图18：仓库显示设置

## 增加路径

直接选中节点，然后不停的选下一个跟当前节点相连的节点组成路径，如果一条路上有多个节点包括直角转弯，则可以从头一直点到尾，设置窗体如图19

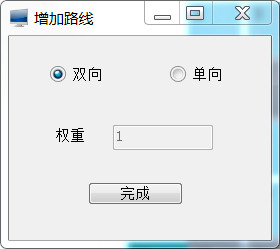


图19：增加路线

## 增加节点

仓库内的路线转弯都是以节点为基础的，节点也可以动态添加，如图20中左上角编号为119的节点即为动态添加的节点

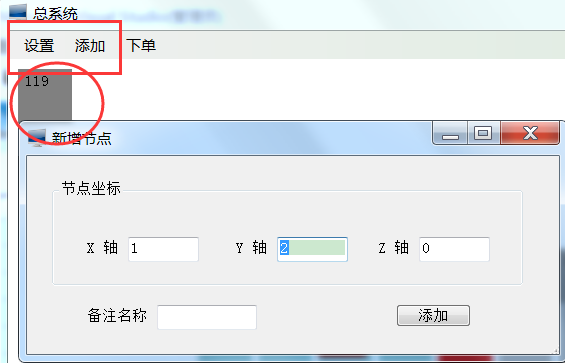


图20：添加节点