

ADAS Mini Car 组件包使用指南_含web版

小车使用指南

当前版本同时支持命令行模式和web模式，推荐使用命令行模式。

基础配置

小车的用户名：mousika

小车的密码：mousika

路由器SSID: AIFORGOOD

路由器密码: 123456789

路由器IP: 192.168.50.1

小车上用到的python库及版本

chardet==3.0.4

paramiko==2.4.2

requests>=2.20.0

psutil==5.6.6

Flask==1.0.2

opencv-contrib-python==4.1.1

opencv-python==4.1.1

Cython==0.29.22

future==0.18.2

futures==3.1.1

gunicorn==20.1.0

h5py==2.10.0

keras==2.3.0

keras-applications==1.0.8

keras-preprocessing==1.1.1

numpy==1.19.2

onnx==1.4.1

pandas==0.22.0

pillow==8.2.0

protobuf==3.8.0

pybind11==2.6.2

pycuda==2019.1.2

pyserial==3.5
rplidar-roboticia==0.9.5
scipy==0.19.1
tensorflow==1.15.2
tensorrt==7.1.3.0
testresources==2.0.1
tornad==6.1
uff==0.6.9
gunicorn
openni
redis

服务器上用到的库及版本

keras==2.3.0
numpy==1.19.5
opencv-python==4.5.1.48
pandas==1.1.5
pillow==8.1.0
rich==10.0.1
scipy==1.5.4
tensorflow-gpu==1.15.2

命令行模式操作（推荐）

连接wifi

- 1. 小车通过网线连接到路由器。
- 2. 登录到路由器查看小车的IP。
- 3. 使用ssh 登陆小车
- 4. 输入以下命令连接wifi（SSID和Password根据路由器的环境修改）

```
1 | sudo nmcli dev wifi connect <SSID> password <Password>  
2 | sudo nmcli con modify <SSID> connection.permissions ''
```

连接手柄

- 1. 使用ssh登陆小车
- 2. 输入以下命令

```
1 | sudo bluetoothctl
```

- ▶ 3. 同时按住手柄的SHARE键和图标键，当手柄前面的指示灯闪烁时松开按键
- ▶ 4. 继续输入命令

```
1 | scan on
```

- ▶ 5.找到wireless controller结尾的设备

```
[NEW] Device A4:AE:12:39:84:D8 Wireless Controller
```

- ▶ 6.继续输入命令

```
1 | scan off
2 | trust <mac address>
3 | connect <mac address>
4 | quit
```

- ▶ 7.当connect <mac address>失败时，重复connect <mac address>操作
- ▶ 8.连接手柄需保持手柄处于配对模式（按住SHARE键+图标键会保持配对模式60s），连接成功后手柄指示灯会常亮

小车运行

驾驶程序的目录结构

```
|— code_template.py---->>新增模块的模板
|— config.py---->>配置文件
|— convert-to-uff.py---->>tensorflow模型转tensorrt模型
|— dellcar
| |— config.py---->>读取配置文件的方法
| |— __init__.py---->>库的连接
| |— log.py---->>日志
| |— memory.py---->>处理输入输出
| |— parts---->>存放模块
| | |— actuator.py---->>电机控制模块
| | |— camera.py---->>相机调用模块
| | |— controller.py---->>手柄映射模块
| | |— datastore.py---->>数据处理
| | |— __init__.py
```

- | | | └─ keras.py----->>tensorflow模型的推演
- | | | └─ lidar.py----->>激光雷达的处理
- | | | └─ sign_detect----->>目标检测算法
- | | | | └─ camera.py----->>调试用的相机调用
- | | | | └─ display.py----->>调试用的显示结果
- | | | | └─ libyolo_layer.so----->>动态链接库
- | | | | └─ Makefile
- | | | | └─ sign_detect.py----->>交通标识检测模块
- | | | | └─ trt_yolo.py----->>调试用的目标检测入口
- | | | | └─ visualization.py----->>画框
- | | | | └─ yolo_classes.py----->>处理类别
- | | | | └─ yolo_layer.cu
- | | | | | └─ yolo_layer.h
- | | | | | └─ yolo_layer.o
- | | | | └─ yolo_with_plugins.py----->>推演方法
- | | | └─ tensorrt.py----->>tensorrt模型的推演
- | | └─ transform.py----->>函数转类
- | └─ vehicle.py----->>模块处理
- └─ drive.py----->>小车启动入口
- └─ __init__.py
- └─ requirement.txt----->>用到的python库及其版本
- └─ sample.py----->>模块调用的示例
- └─ sign_models----->>目标检测训练好的模型
- | └─ yolov3-tiny-416.cfg
- | | └─ yolov3-tiny-416.trt
- └─ unittest----->>模块测试
- └─ camera_test.py----->>相机测试
- └─ control_test.py----->>电机测试
- └─ static----->>测试用的静态文件
- └─ tensorflow_test.py----->>推演测试
- └─ tensorrt_test.py----->>

修改配置文件

```

1  import os
2
3  # PATHS
4  CAR_PATH = PACKAGE_PATH = os.path.dirname(os.path.realpath(__file__))
5
6  # VEHICLE
7  DRIVE_LOOP_HZ = 20
8  MAX_LOOPS = None
9
10 # CAMERA
11 CAMERA_RESOLUTION = (360, 640) # (height, width)

```

```
12 FINAL_RESOLUTION = (270, 480) # 如果不是使用的单摄，最后经过拼接后分辨率就会不一样，
13 CAMERA_FRAMERATE = DRIVE_LOOP_HZ
14
15 # JOYSTICK
16 AUTO_RECORD_ON_THROTTLE = True
17
18 TUB_PATH = os.path.join(CAR_PATH, 'tub')
19 SMOOTH = False # Change it to True if you want to have the smooth behavior
```

配置参数说明

DRIVE_LOOP_HZ: 主循环中每秒执行的次数

MAX_LOOPS: 最大执行周期

CAMERA_RESOLUTION: RGB相机的分辨率，支持16:9和4:3

FINAL_RESOLUTION: 拼接后的分辨率，用tensorrt时需要提前注册分辨率，所以写在了配置文件里

CAMERA_FRAMERATE: 相机帧率

AUTO_RECORD_ON_THROTTLE: 在手柄驾驶时，是否启用速度不为0时自动记录数据

TUB_PATH: 数据存放目录

SMOOTH: 是否使用横移功能

启动命令

小车的启动入口是~/mycar/drive.py

- 1)手柄启动 (要先成功连接手柄)

```
1 | cd ~/mycar
2 | sudo python3 drive.py --model js
```

或者是下面这个命令

```
1 | cd ~/mycar
2 | sudo python3 drive.py
```

- 2)模型启动 (要先将训练好的模型放在小车上)

```
1 | cd ~/mycar
2 | sudo python3 drive.py --model <模型的路径>
3 | # 举例说明
4 | # sudo python3 drive.py --model models/mymodel
```

手柄按键映射

默认手柄已经完成映射，如果需要修改映射方法可以到~/mycar/dellcar/parts/controller.py 246-248行进行修改

左边摇杆：控制左转右转，往左摇是左转，往右摇是右转，只有在速度不为0或使用横移功能时才会有效果

右边摇杆：控制前进后退，往上摇是前进，往下摇是后退，当速度不为0时默认会记录数据

R2按钮：控制横移，使用横移功能之前需要在配置文件内将SMOOTH的值改为True，之后按下R2按钮，再使用左边摇杆进行左转右转时就会变为左横移右横移。

数据传输

当需要将小车收集的数据传输到服务器进行训练或者将服务器训练好的模型传输到小车进行推演时，需要进行数据的传输，传输的方法有很多种，下面是一个简单的示例

```
1 | rsync -a ~/mycar/tub/ rui@192.168.1.188:mycar_server/tub_rui
2 | rsync -a rui@192.168.1.188:mycar_server/models/mymodel ~/mycar/models/
```

rsync: 一个开源的数据传输工具

-a: 以递归压缩的方式传输

~/mycar/tub/: 要传输的目录，tub后面加上"/"说明要传输的是tub文件夹内的文件，不包括tub文件夹，如果想传输包含这个文件夹可以将后面的"/"去掉

rui: 目标服务器的用户名，根据实际情况修改

192.168.1.188: 目标服务器的ip，根据实际情况修改

mycar_server/tub_rui: 目标路径，实际上是/home/rui/mycar_server/tub_rui的简写，rsync会自动创建tub_rui目录

其他的参数和使用方法请自行搜索rsync文档

WEB页面功能说明

基础操作

- 1. 使用web页面时，如果修改了代码(包括config文件)，需要重启服务或重启小车才能生效。
重启服务命令：

```
1 | sudo systemctl restart gunicorn.service
```



- 2. 当程序出现异常，例如web无法正常运行或是小车启动时小车状态变成红色需要debug时，以下命令查看服务状态：

```
1 | systemctl status gunicorn.service
```

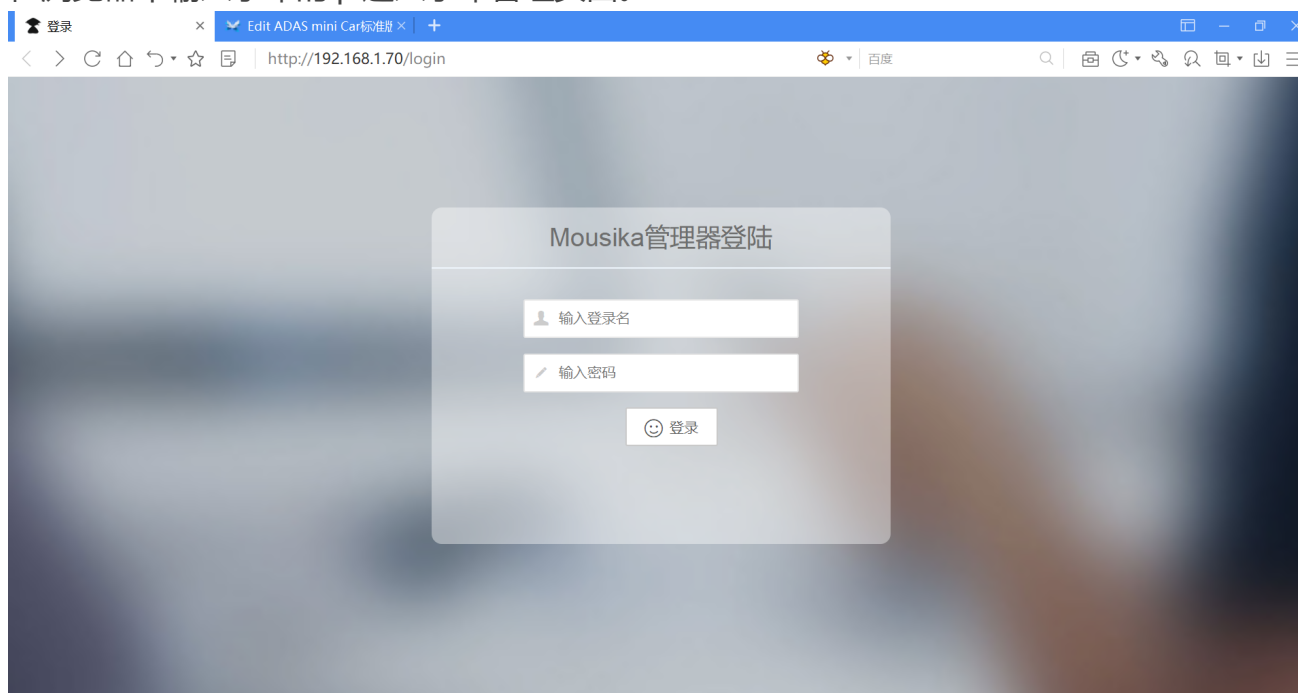
登陆 & 修改密码

- ▶ 1. 小车通过网线连接到路由器。
- ▶ 2. 登录到路由器查看小车的IP。

▶ 所有名单

互联网	图标	客户端名称	客户端 IP 地址	客户端 MAC 地址
		mousika-desktop	192.168.1.71	DHCP 00:28:F8:C2:ED:3B

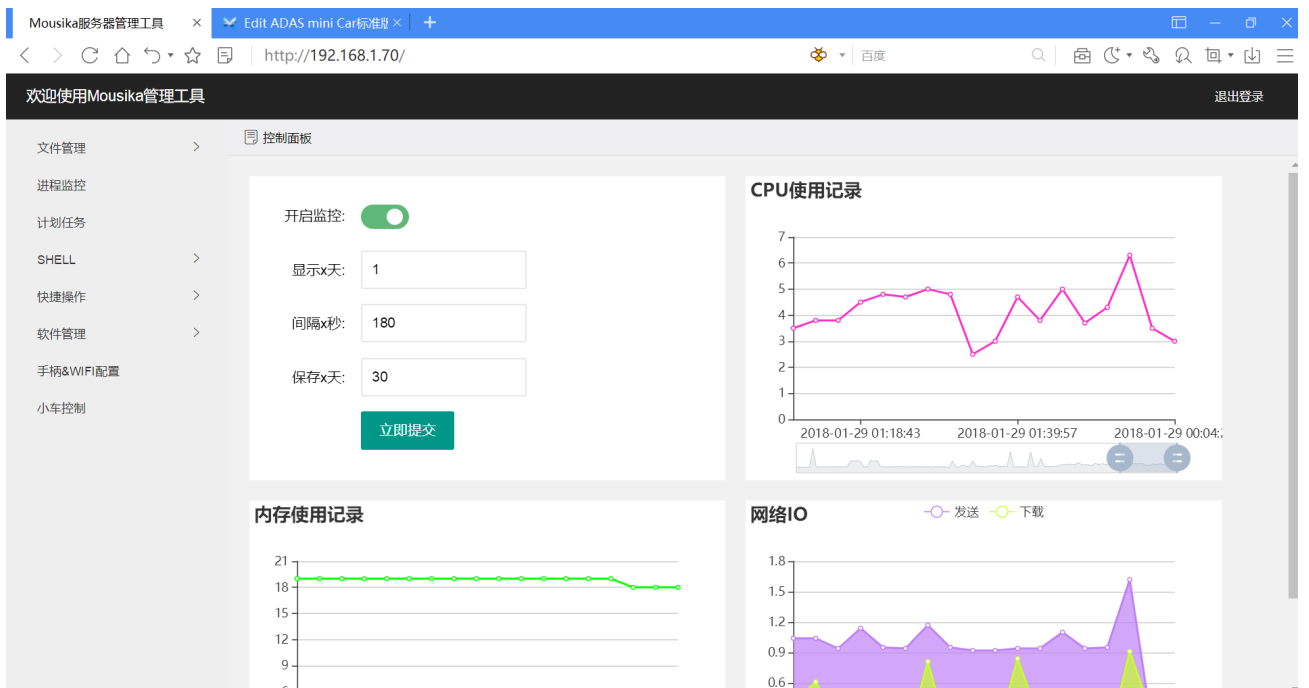
- ▶ 3. 用一台同一网段的电脑，在**Chrome**浏览器输入小车的IP。
- ▶ 4. 在浏览器中输入小车的ip 进入小车管理页面。



- ▶ 5.. 在“Mousika管理器登陆”中输入用户名：mousika，以及密码：mousika。

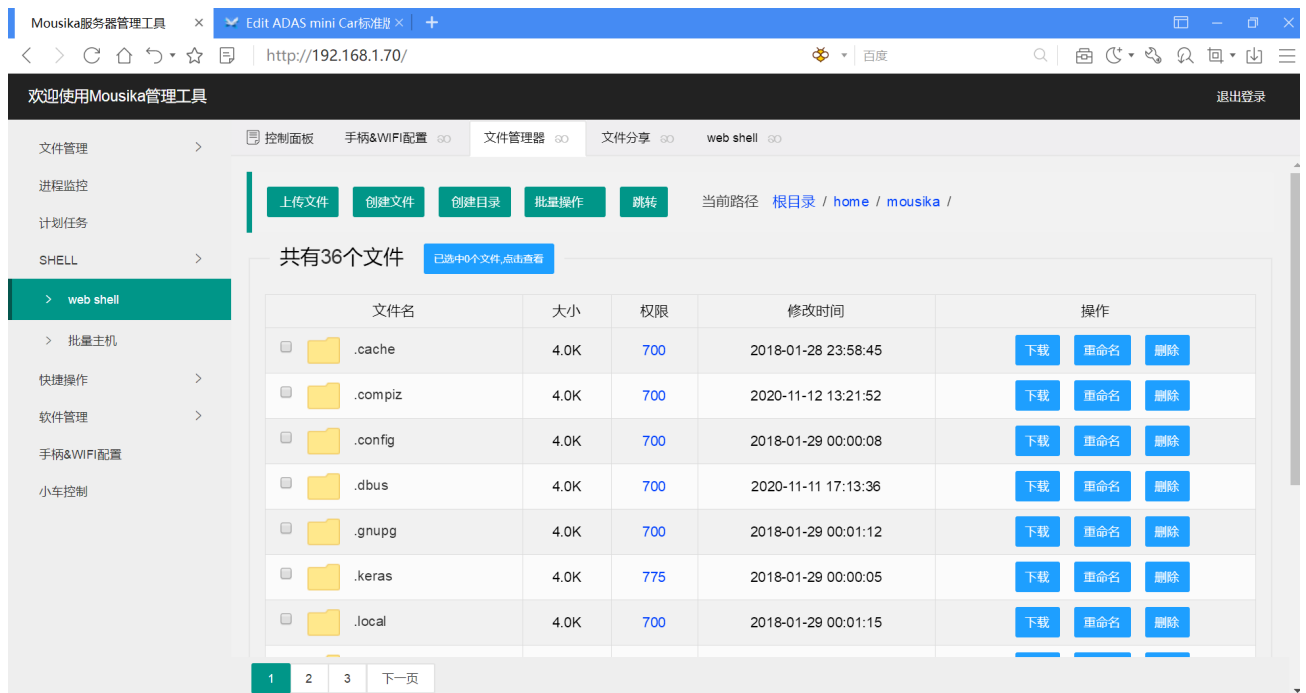


- ▶ 6. 成功登录页面。

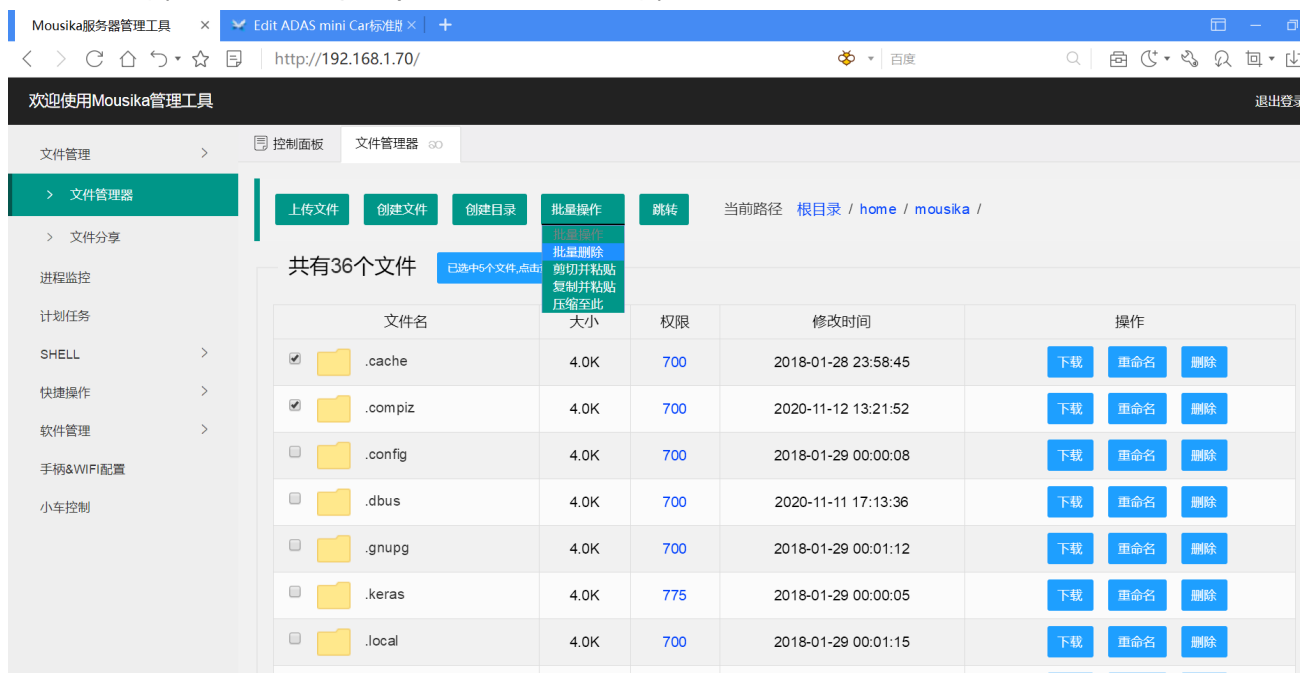


文件管理

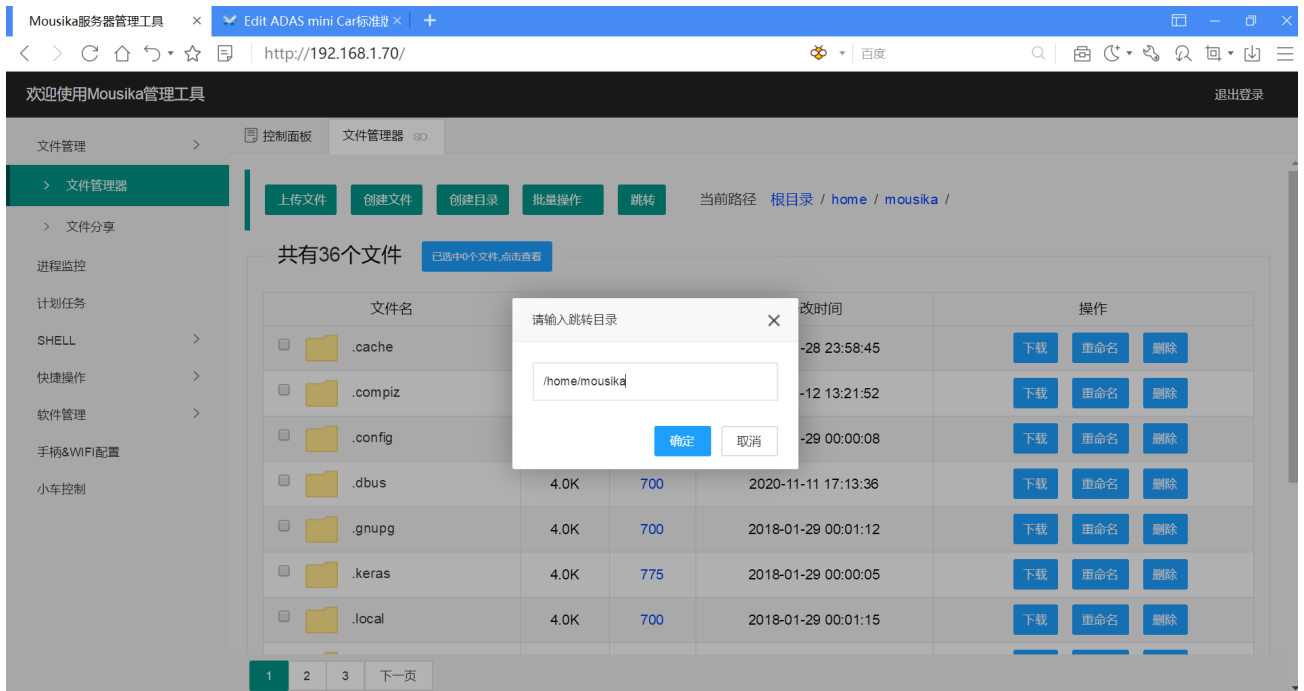
- ▶ 1. 点击“文件管理器”进入文件管理页面，可对文件进行上传、下载、删除、创建、重命名等操作。



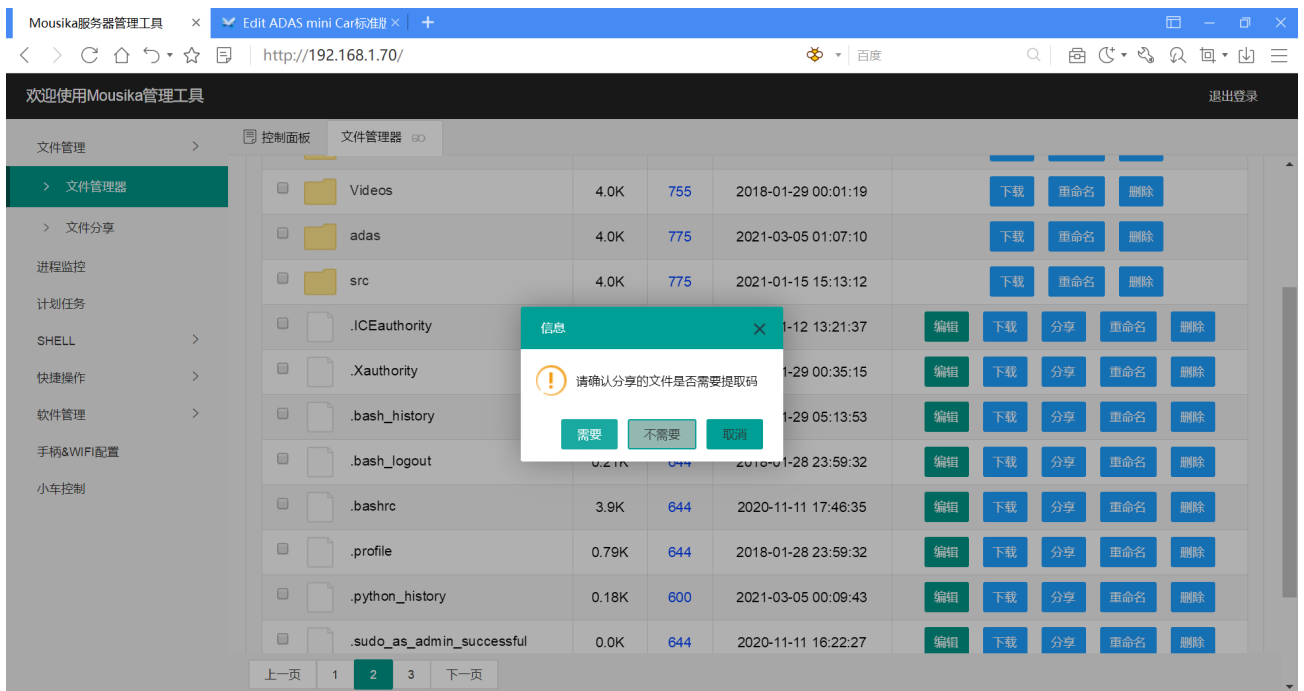
2. 点击“批量操作”按钮可对选中的文件进行批量操作。



3. 点击“跳转”按钮可以直接跳转到指定的目录。



4. 点击“分享”按钮，可将文件分享到“文件分享”页面。

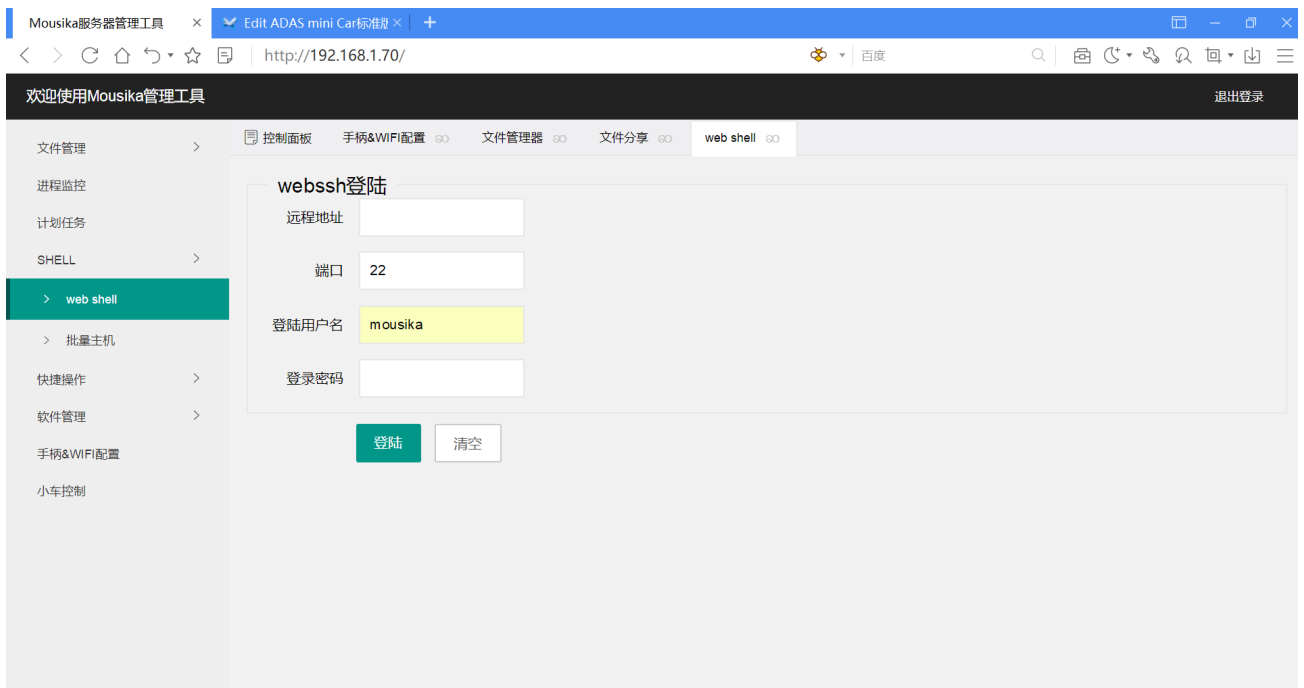


5. 点击“文件分享”进入文件分享页面。此页面提供一个类似于网盘的文件分享的功能,让你或其他人快捷下载文件。在文件管理器中点击“分享”按钮，将文件分享到这里。

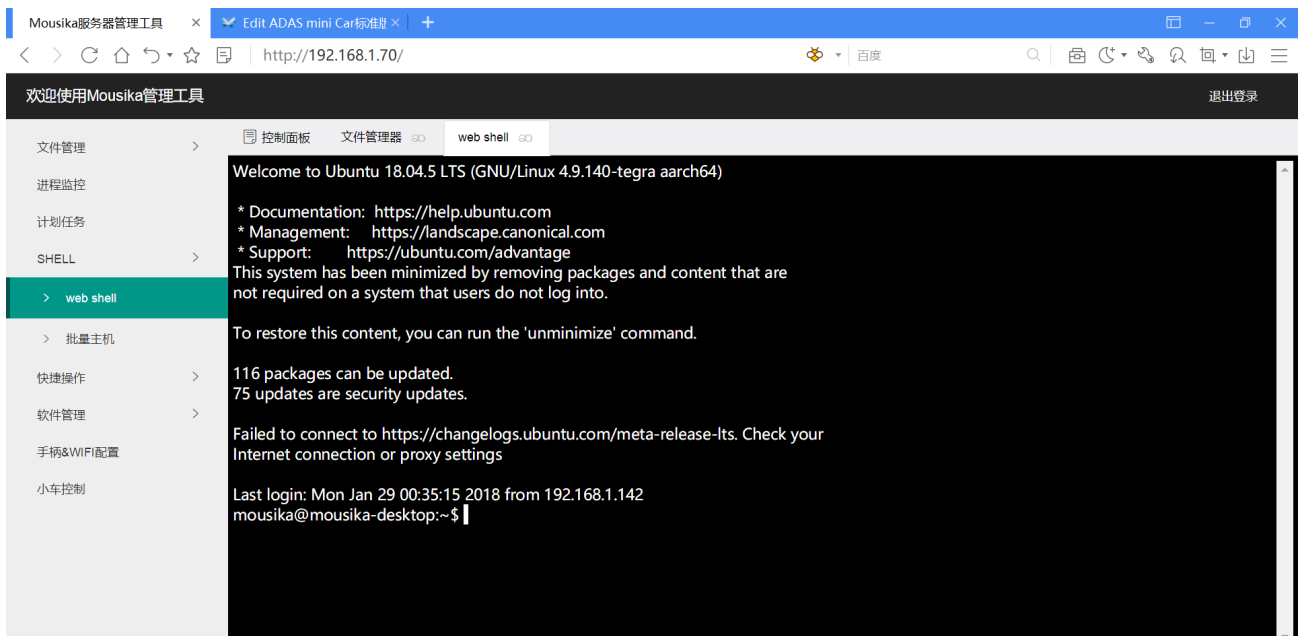
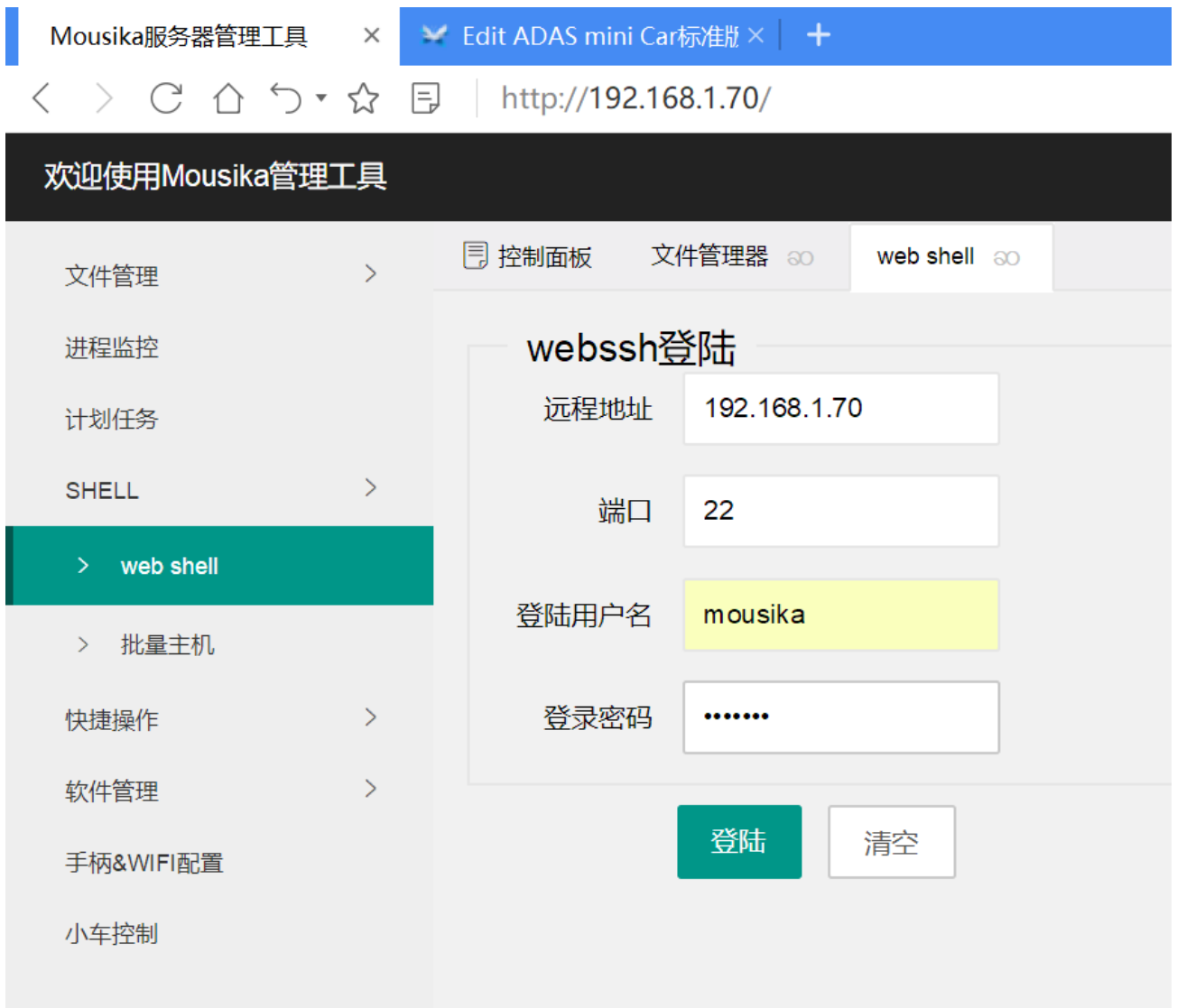


WEBShell

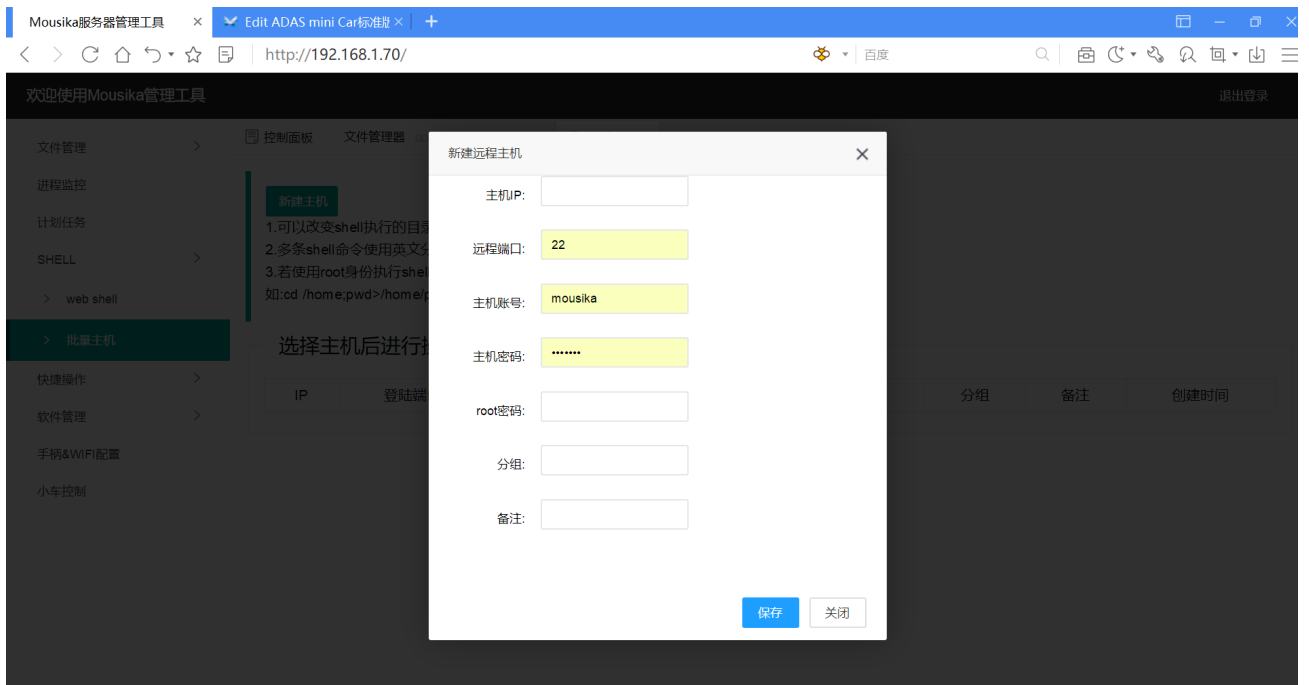
1. 点击“SHELL”目录下的“web shell”进入web shell页面，可进行webssh登录。



如果是本地登录，远程地址为：127.0.0.1（如果不是本地登录，远程地址输入该机的地址即可），端口为：22，用户名和密码都为：mousika。点击“登陆”后，进入ssh页面。



2. “批量主机”页面。点击“新建主机”按钮可新建主机。若使用root身份执行shell,请在shell最后追加#root。如:cd /home;pwd>/home/pwd.txt #root



手柄 & WIFI配置

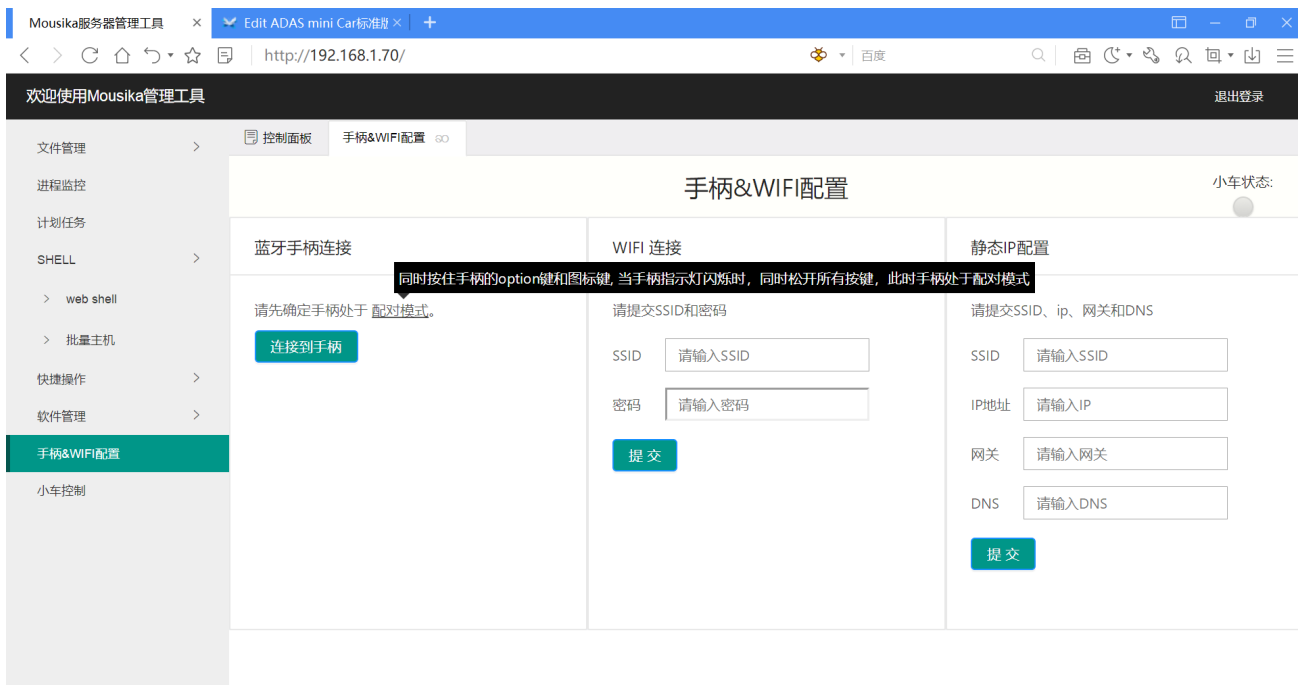
蓝牙手柄连接

1. 当需要手动操控小车或者收集训练数据时，先连接蓝牙。点击目录中的“手柄&WIFI配置”。首先请先确定手柄处于“配对模式”，即同时按住手柄的Share键和图标键，当手柄前面的指示

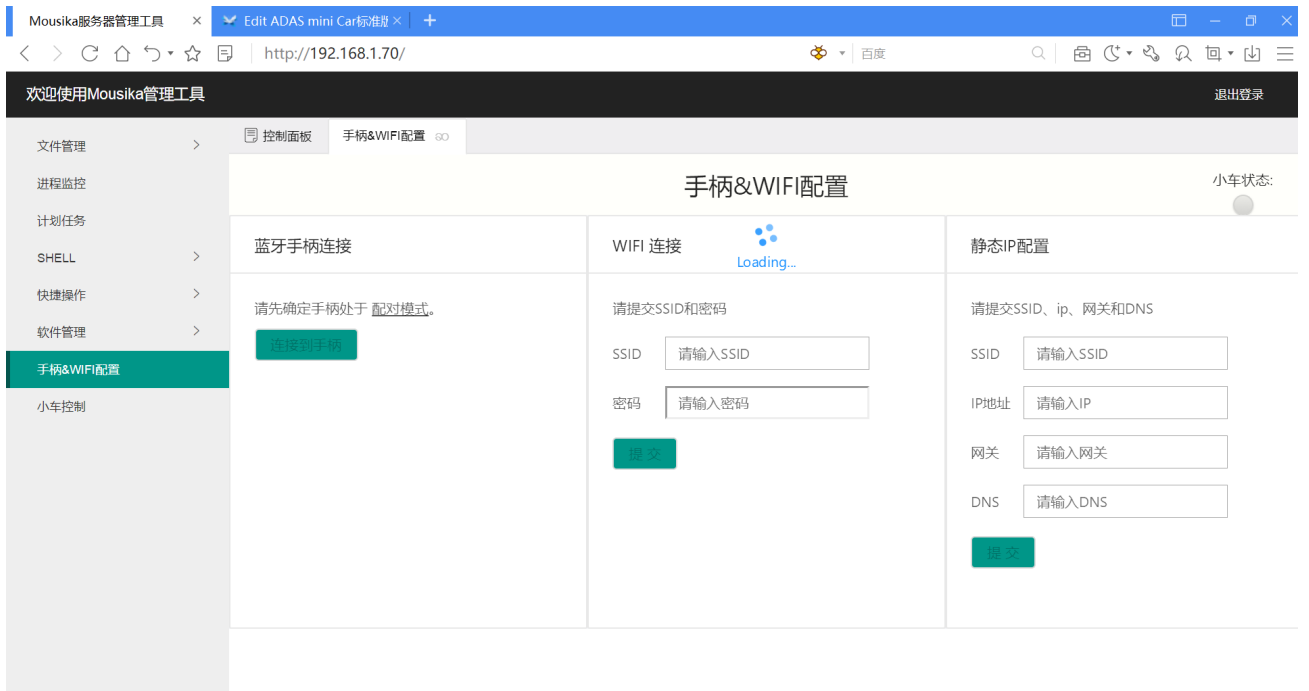
灯闪烁时松开Share键和图标键。



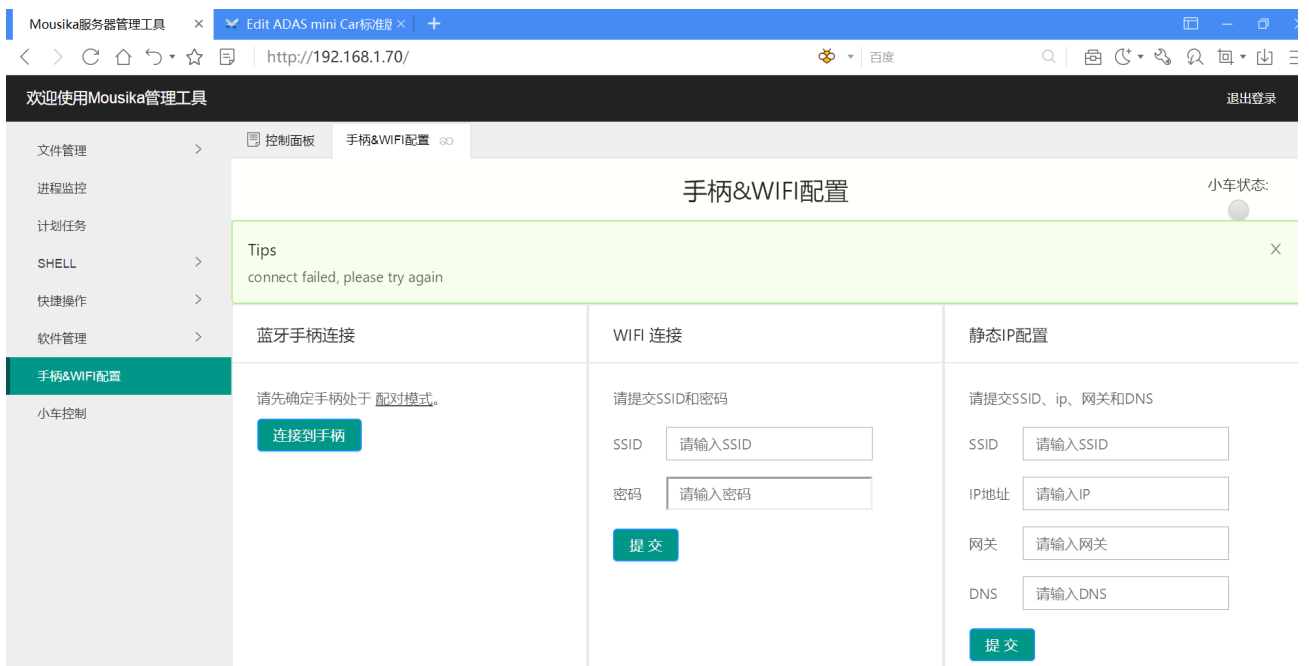
2. 点击页面中的“连接到手柄”即可自动连接手柄。



1) 等待手柄连接。

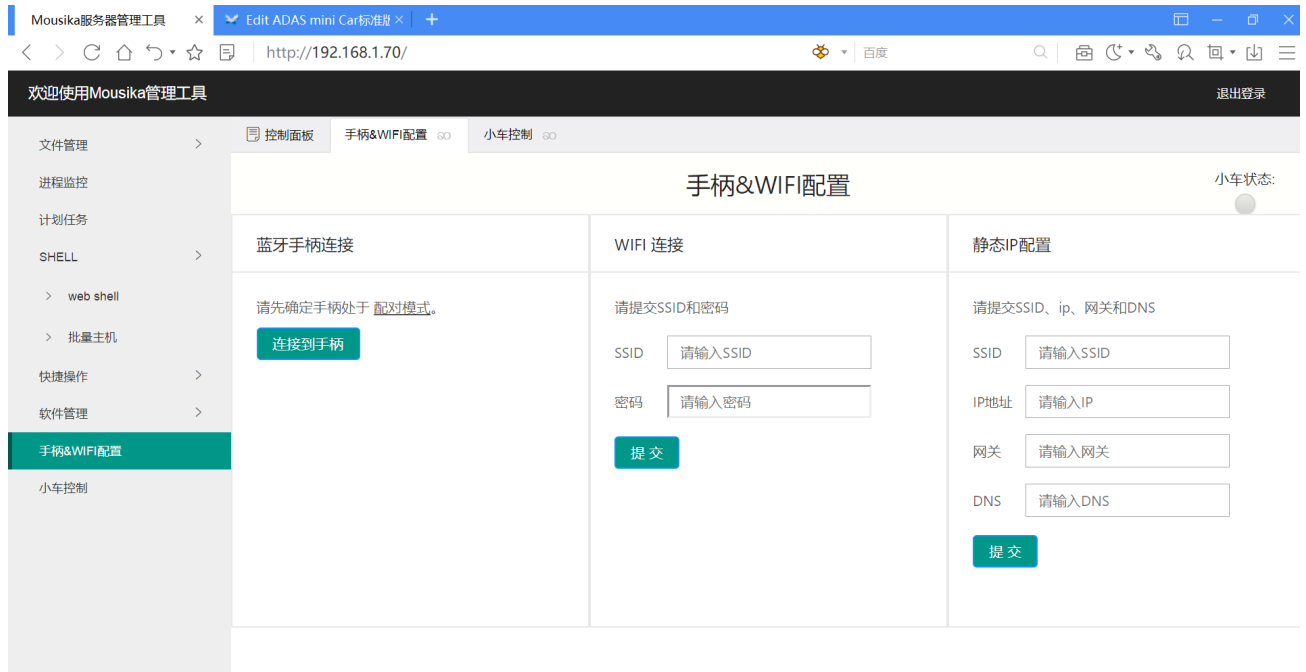


2) 当Web页面返回“Connect failed, please try again”时，重复以上步骤。（注意观察手柄指示灯，如果指示灯不亮需要立即重新按住share键和图标键，直至指示灯闪烁。）当手柄指示灯常亮时代表连接成功。

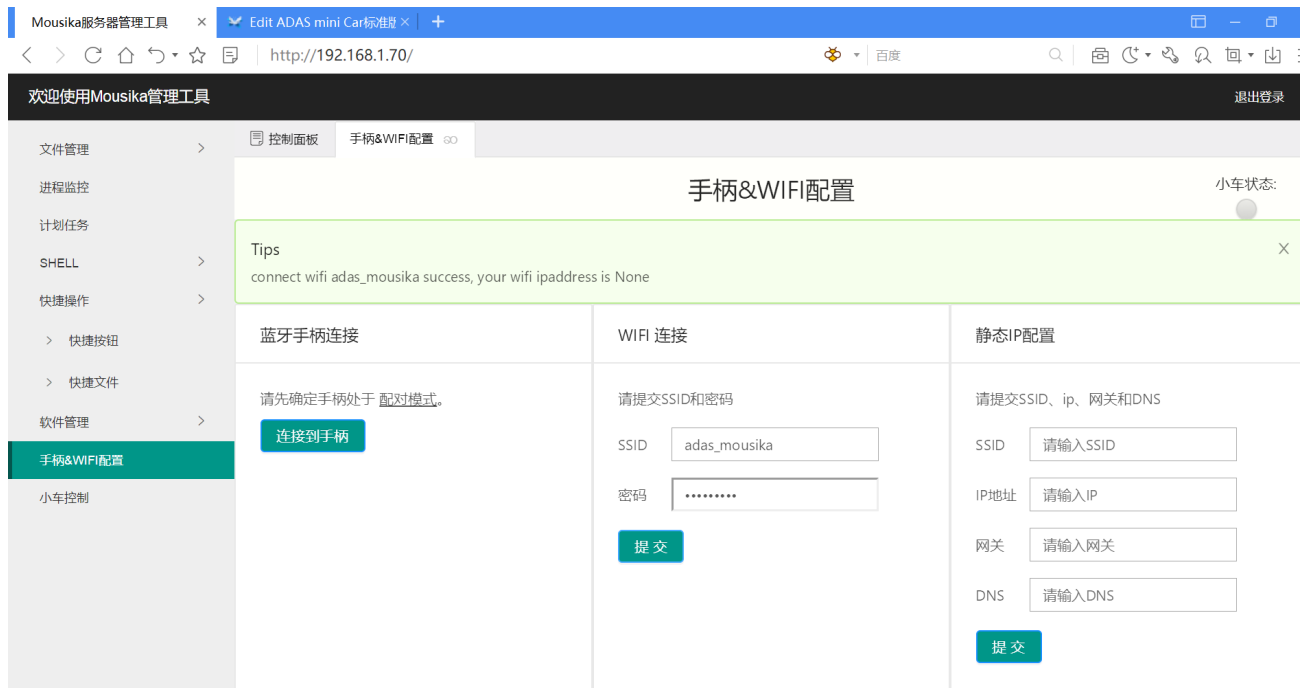


WIFI连接

1. WIFI连接时，在WIFI连接目录下输入SSID和密码并提交。



2. 如果WIFI连接成功，会在页面上显示“success”。



静态IP配置

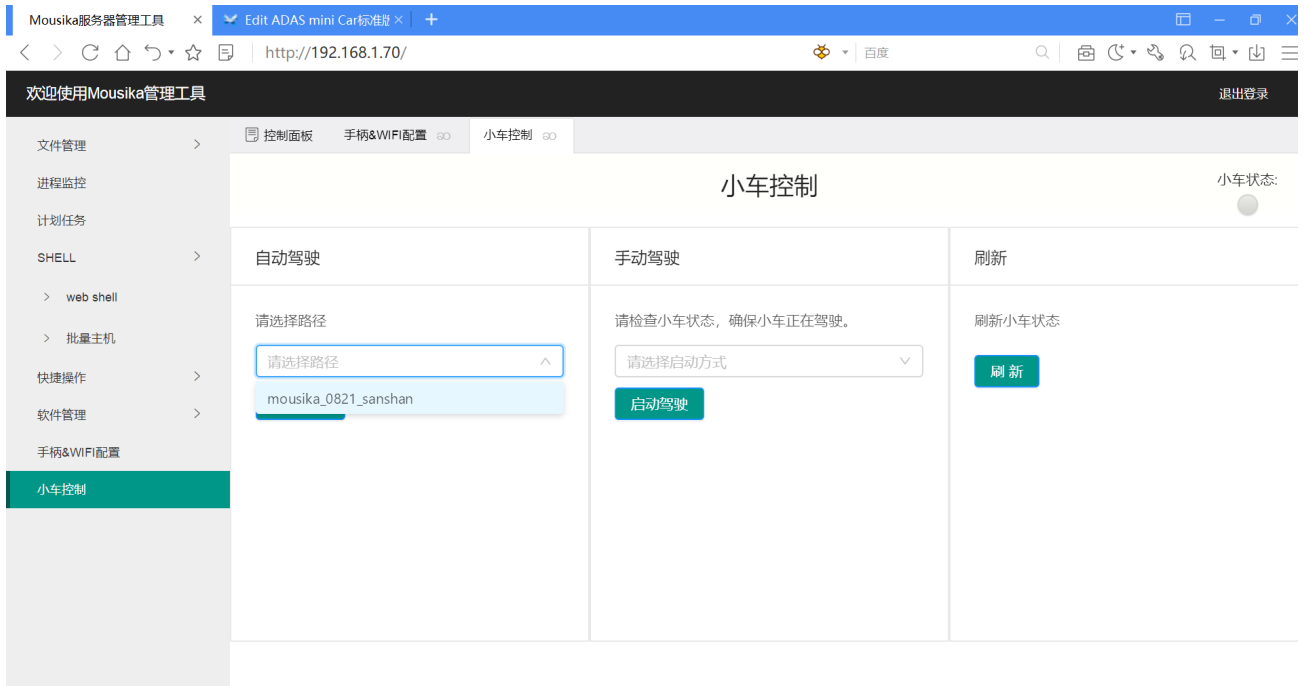
静态IP配置时，在静态IP配置目录下输入SSID、ip、网关和DNS并提交。



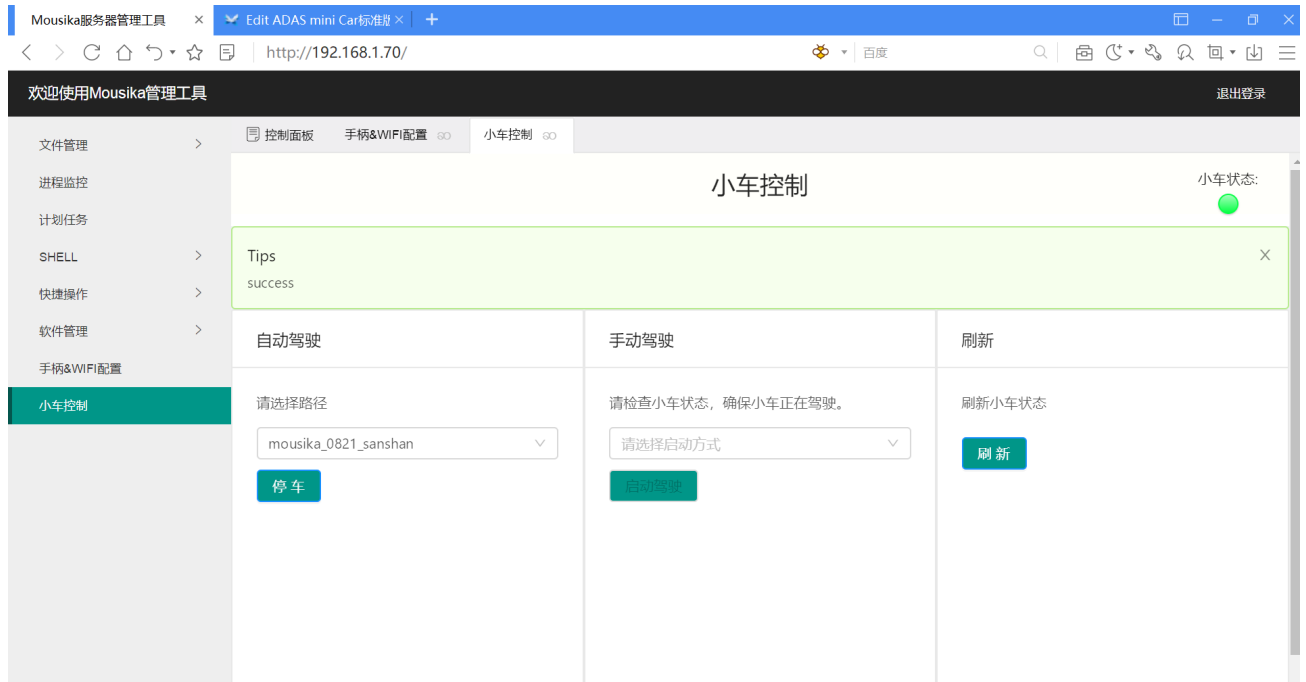
小车控制

自动驾驶

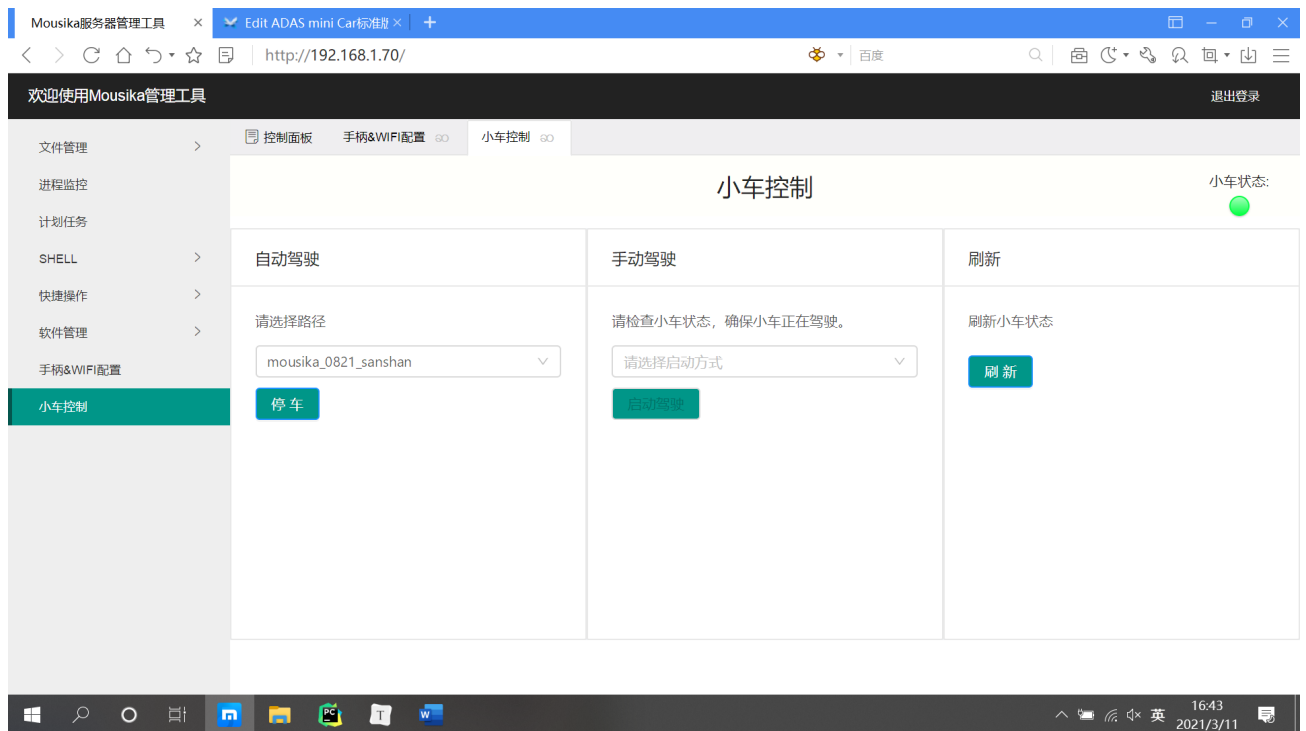
1. 点击“小车控制”进入驾驶控制页面。
2. 在自动驾驶目录下，点击“请选择路径”下拉框选择想要运行的模型。
3. 点击“启动驾驶”启动自动驾驶。



4. 程序成功运行提示Success。



5. 当需要停止时点击“停车”按钮。

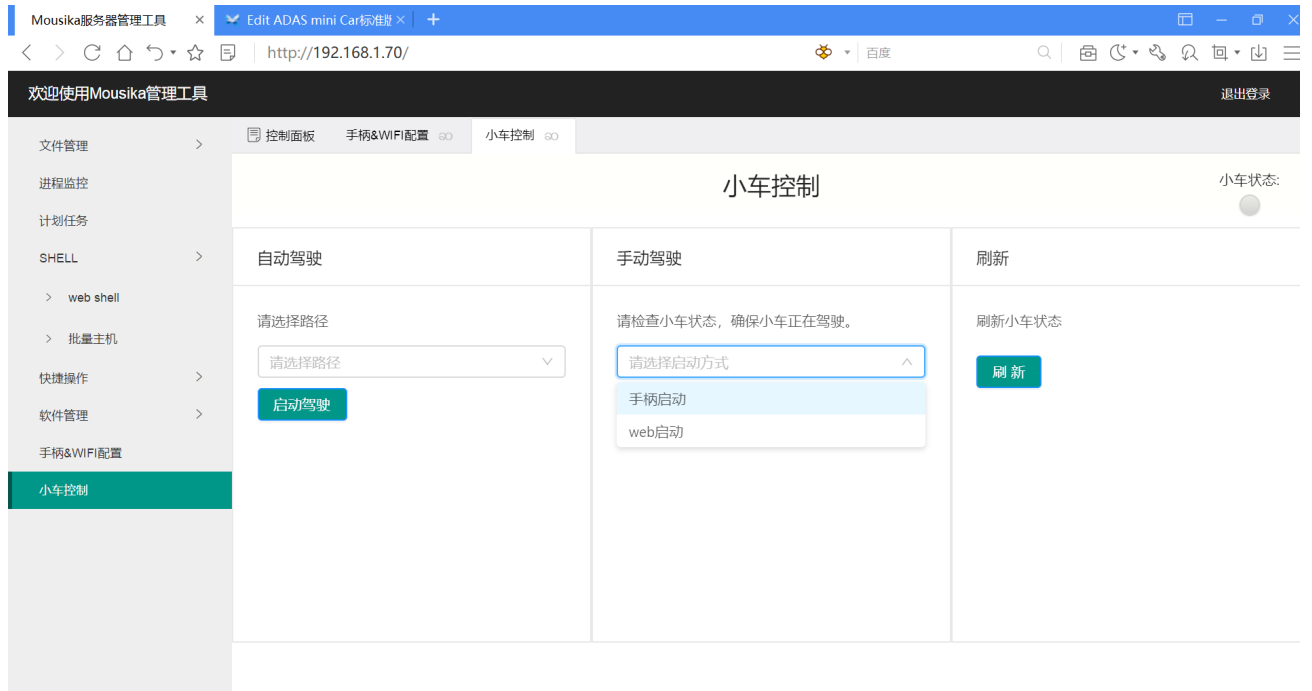


6. 受性能约束，NANO模组使用web界面停车时，需要等待5s左右的时间才能继续使用web界面

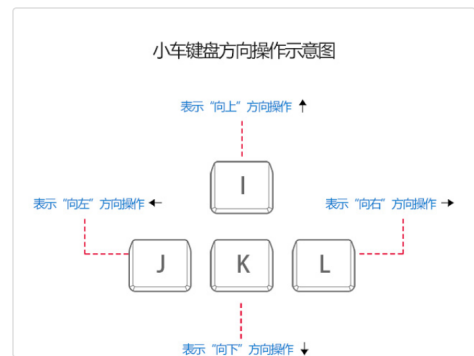
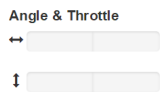
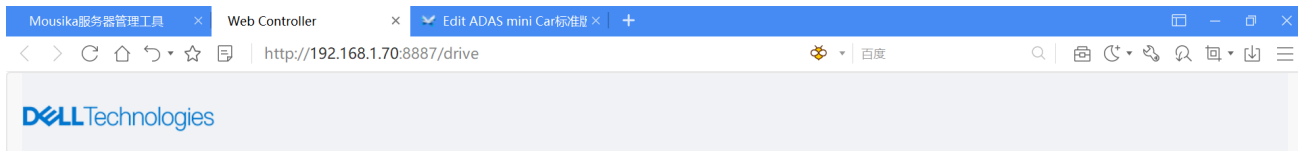
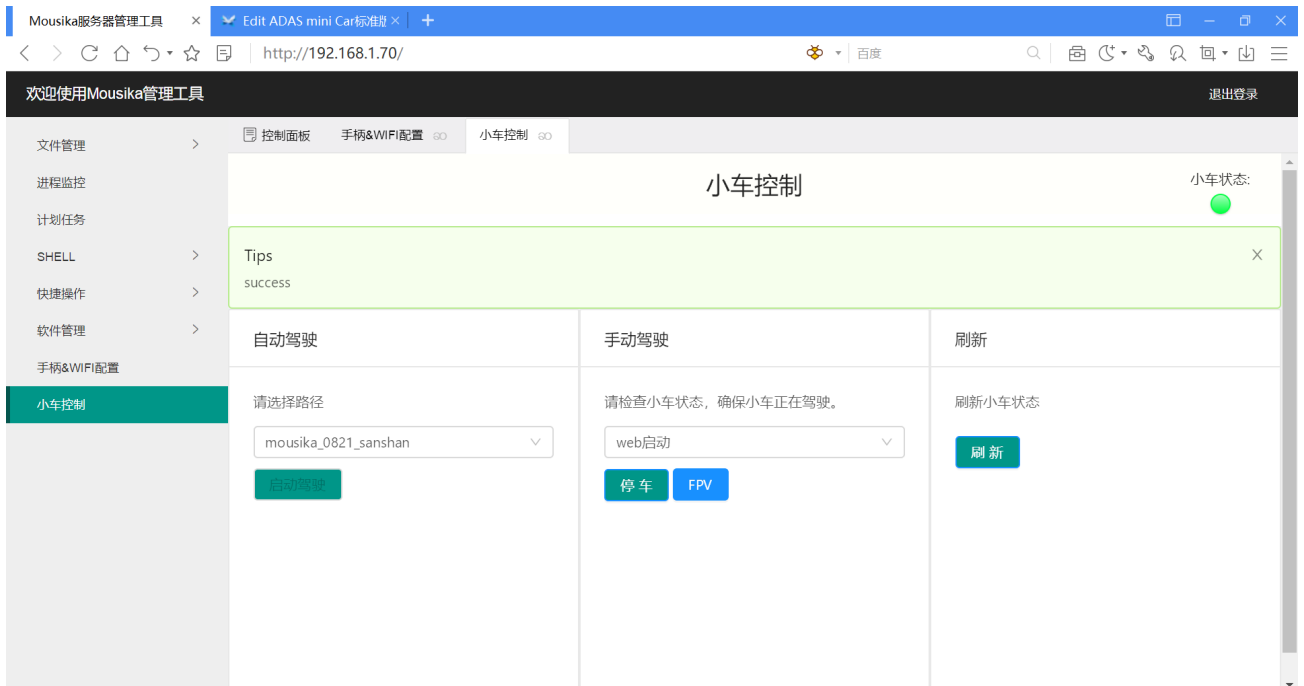
手动驾驶

1. 点击“小车控制”进入驾驶控制页面。选择手动驾驶时，首先要检查小车状态，确保小车处于空闲（web指示灯灰色）的状态。
2. 点击“请选择启动方式”下拉框选择“手柄启动”或者“web启动”。“手柄启动”通过手柄控制小车的运行，“web启动”通过web界面上的按钮控制小车的运行。

3. 点击“启动驾驶”启动手动驾驶。

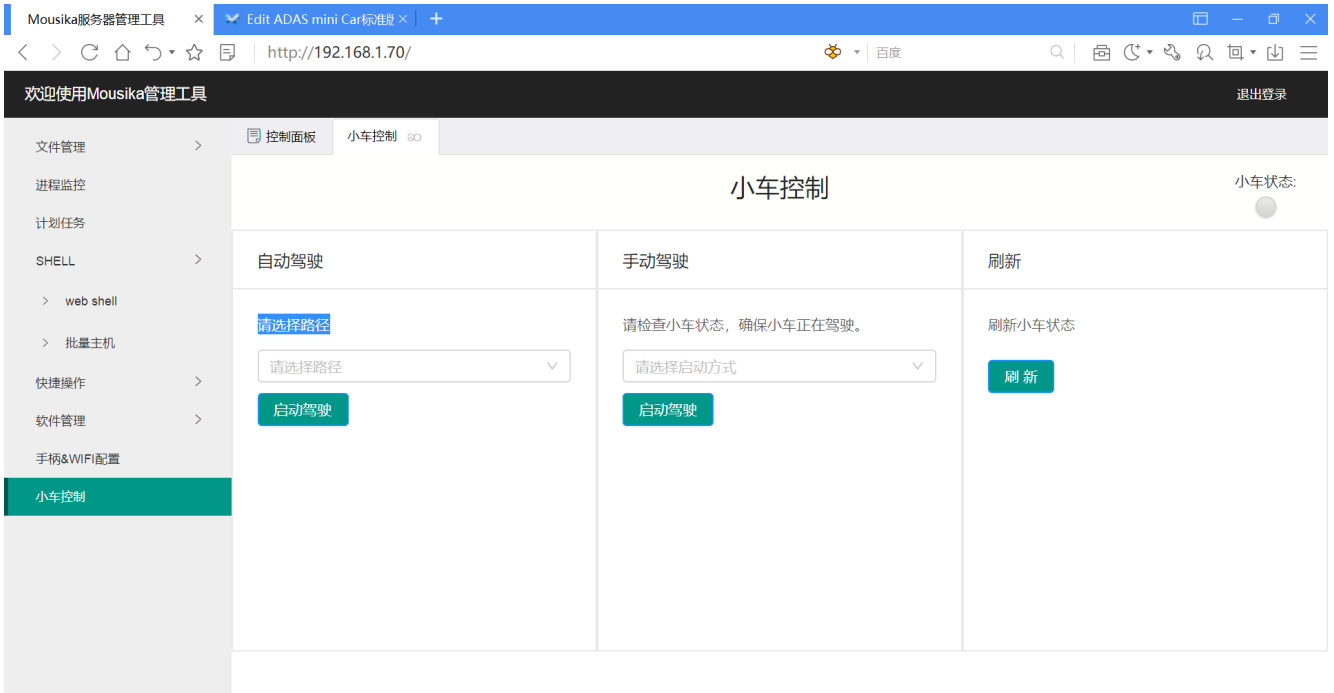


4. 当选中“web启动”时，点击“FPV”按钮进入小车的web控制页面。当需要停止时点击“停车”按钮。



刷新

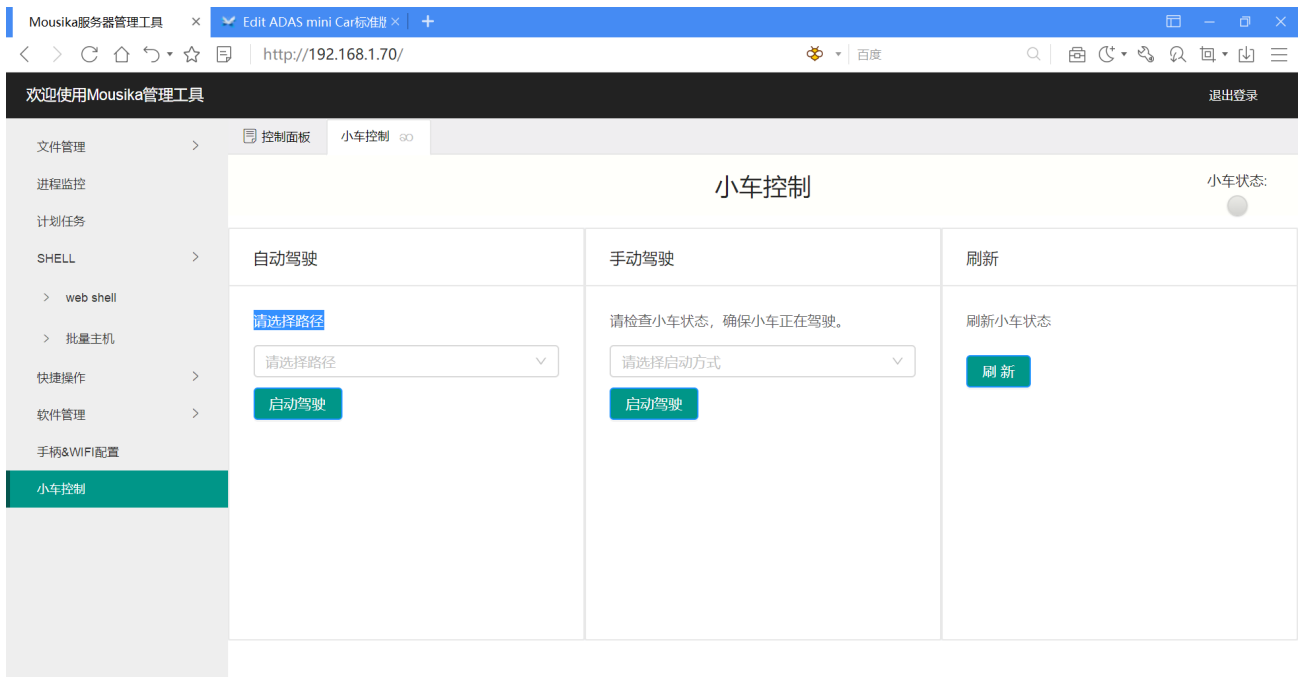
- ▶ 点击“刷新”按钮刷新小车的状态。无论小车状态是什么，点击刷新会立即将小车恢复到空闲状态。



小车状态

点击“小车控制”进入驾驶控制页面。在页面的右上角有一个检测小车当前状态的图标，一共有三种颜色，灰色，绿色和红色。

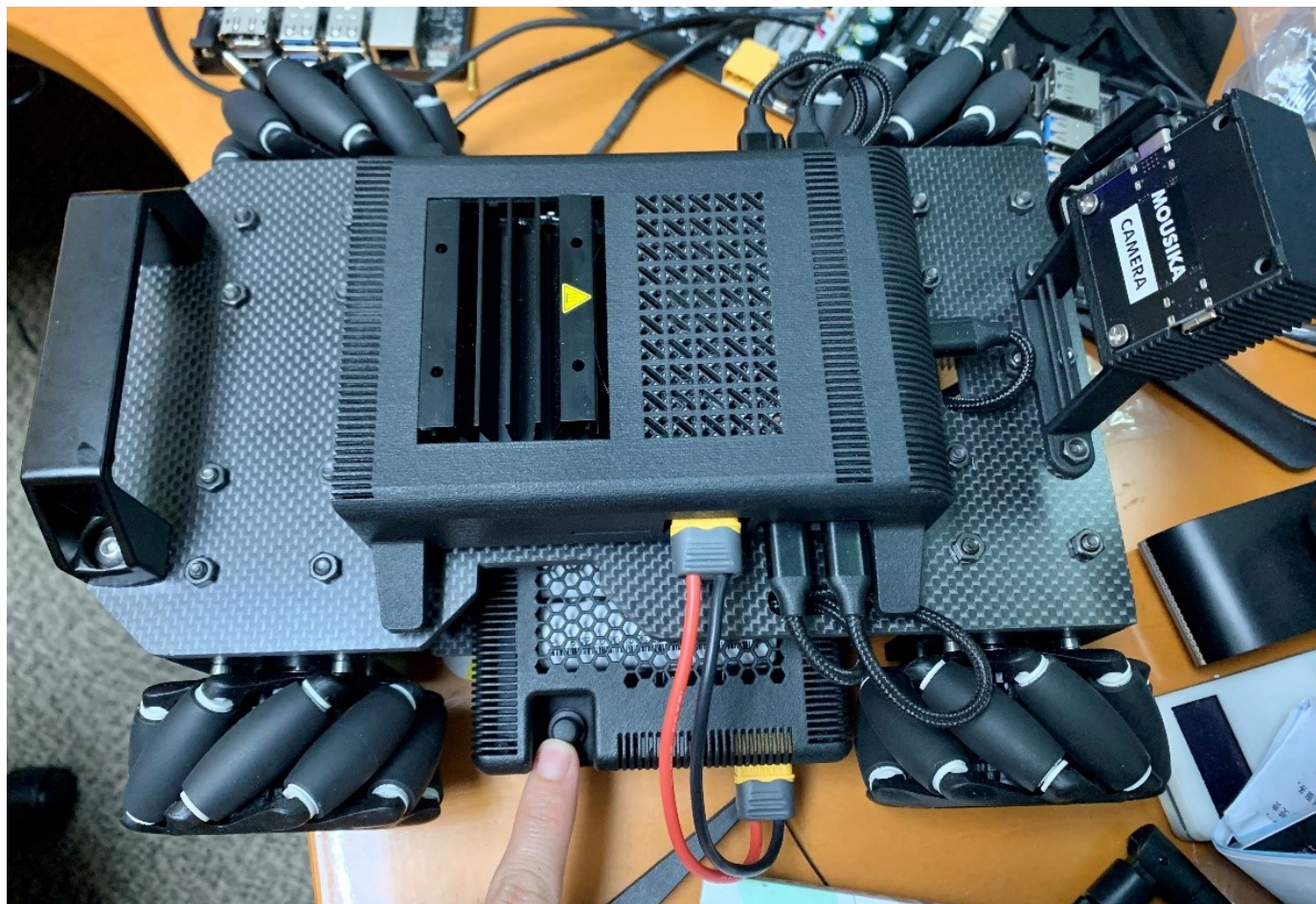
- ▶ 1. 小车处于空闲状态时，“小车状态”显示灰色。
- ▶ 2. 小车正在启动驾驶模块时，“小车状态”显示黄色。
- ▶ 3. 小车处于正常启动状态时，“小车状态”显示绿色。
- ▶ 4. 小车程序遇到问题不能正常工作时，“小车状态”显示红色。



小车充电

小车开机

按下电池上的按钮开机

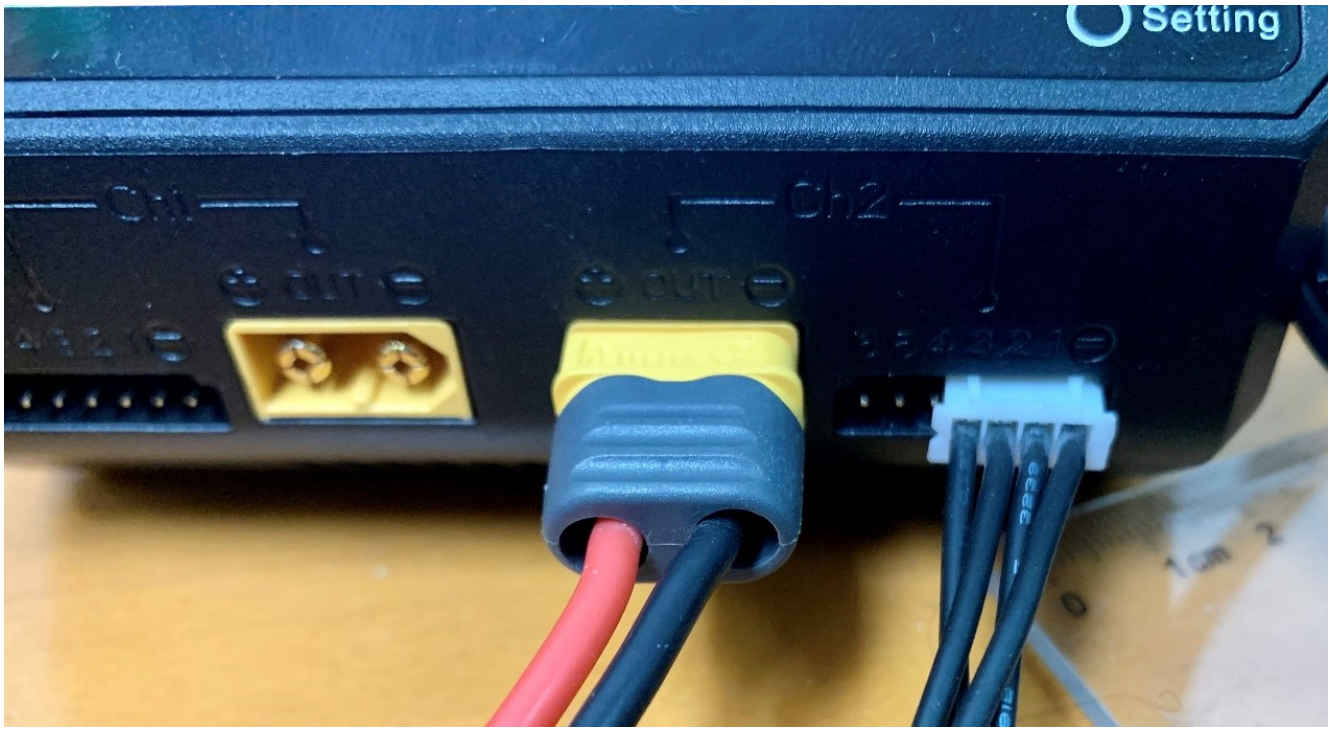


什么时候需要充电

当小车主控两侧的灯变成红色时小车电池需要充电

开始充电

▶ 1.电源线如图所示，注意将电源线右侧对准充电器的“负号”





▶ 2. 点击充电器左侧按钮选择频道，此时插在CH2，所以选择CH2



- ▶ 3. 点击充电器右侧按钮，进行电流设置，一般设置为1.0A



- ▶ 4. 设置完成后点击“开始任务”



▶ 5.开始充电



在GPU上进行模型训练

训练程序的目录结构

```
|— config.py ☑ ----->>配置文件
|— dellcar
| |— config.py ☑ ----->>读取配置文件
| |— \_\_init\_\_.py
| |— log.py ☑ ----->>记录log
| |— parts
| | |— datastore.py ☑ ----->>处理数据
| | |— keras.py ☑ ----->>训练参数&网络结构
| |— util
|— data.py ☑ ----->>数据格式化
|— files.py ☑ ----->>解析文件路径
```

修改配置文件

```
1 | import os
2 |
3 | # PATHS
4 | CAR_PATH = PACKAGE_PATH = os.path.dirname(os.path.realpath(__file__))
5 | DATA_PATH = os.path.join(CAR_PATH, 'data')
6 |
7 | # CAMERA
8 | CAMERA_RESOLUTION = (480, 640) # 如果使用多摄像头拼接，这里是拼接后的分辨率
9 |
10 | # TRAINING
11 | BATCH_SIZE = 128
12 | TRAIN_TEST_SPLIT = 0.8
13 | NEURAL_FUNCTION_LIST = ["default_categorical", "optimal_categorical"]
14 | NEURAL_FUNCTION = NEURAL_FUNCTION_LIST[0]
15 |
16 | # Behavior
17 | SMOOTH = False # Change it to True if you want to train the smooth behavior
```

配置参数说明

DATA_PATH: 不手动添加tub参数时默认读取数据的目录

CAMERA_RESOLUTION: 输入图像的分辨率

TRAIN_TEST_SPLIT: 训练集的比例

NEURAL_FUNCTION_LIST: 已有的网络结构

NEURAL_FUNCTION: 选择用哪个网络结构

SMOOTH: 是否训练横移功能

必须要修改的参数:

- ▶ CAMERA_RESOLUTION, 根据要训练的图片的分辨率进行修改
- ▶ NEURAL_FUNCTION, 有两种网络结构可以选择, default_categorical训练的时间短一些, 参数较多, 在车上推演的时间长。optimal_categorical训练的时间长一些, 参数较少, 在车上推演的时间短。

训练前对数据进行错误检查

这个步骤会对一些异常数据进行检查和删除, 小车突然断电, 数据清洗失误等原因可能会导致数据异常

```
1 | python3 train.py --tub <tub_path> --check_only True
```

训练前对多组数据同时进行错误检查

```
1 | python3 train.py --tub <tub_path_1>,<tub_path_2>,<tub_path_3> --check_only True
```

从零开始训练

```
1 | python3 train.py --tub <tub_path> --model <new_model_path>
```

用base model进行叠加训练

注意：default_categorical网络训练出来的base_model在optimal_categorical网络上不能进行叠加训练，反之亦然。

```
1 | python3 train.py --tub <tub_path> --model <new_model_path> --base_model <base_
```

用多组数据进行训练

方法1：在命令行中追加数据路径

```
1 | python3 train.py --tub <tub_path_1>,<tub_path_2>,<tub_path_3> --model <new_mod
```

方法2：在mycar目录下创建一个data目录，将所有的数据全部放在data目录中，这样在训练命令不需要传入--tub参数，默认去data目录获取所有目录

```
1 | python3 train.py --model <new_model_path>
```

查看API文档

查看小车的API文档

将小车上的~/mycar/api_document下载到本地，使用Chrome浏览器打开
api_document/index.html

Search docs

CONTENTS:

- mycar package
 - Subpackages
 - mycar.dellcar package
 - Submodules
 - mycar.code_template module
 - mycar.config module
 - mycar.convert-to-uff module
 - mycar.drive module
 - mycar.sample module
 - Module contents
- mycar
 - mycar package
 - Subpackages
 - mycar.dellcar package
 - Submodules
 - mycar.code_template module
 - mycar.config module
 - mycar.convert-to-uff module
 - mycar.drive module
 - mycar.sample module
 - Module contents

Subpackages

- mycar.dellcar package
 - Subpackages
 - mycar.dellcar.parts package
 - Subpackages
 - mycar.dellcar.parts.sign_detect package
 - Submodules
 - mycar.dellcar.parts.sign_detect.camera module
 - mycar.dellcar.parts.sign_detect.display module
 - mycar.dellcar.parts.sign_detect.sign_detect module
 - mycar.dellcar.parts.sign_detect.visualization module
 - mycar.dellcar.parts.sign_detect.yolo_classes module
 - mycar.dellcar.parts.sign_detect.yolo_with_plugins module

查看服务器的API文档

将服务器上的~/mycar_server/api_document下载到本地，使用Chrome浏览器打开api_document/index.html

mousika_competition_server

Search docs

CONTENTS:

- mycar_server package
 - Subpackages
 - mycar_server.dellcar package
 - Submodules
 - mycar_server.train module
 - mycar_server.config module
- mycar_server
 - mycar_server package
 - Subpackages
 - mycar_server.dellcar package
 - Submodules
 - mycar_server.train module
 - mycar_server.config module

Subpackages

- mycar_server.dellcar package
 - Subpackages
 - mycar_server.dellcar.util package
 - Submodules
 - mycar_server.dellcar.util.data module
 - mycar_server.dellcar.util.files module
 - Module contents
 - mycar_server.dellcar.parts package
 - Submodules
 - mycar_server.dellcar.parts.keras module
 - mycar_server.dellcar.parts.datastore module
 - Module contents

Submodules

mycar_server.train module

mycar_server.train.check(path) [源代码]

将异常的数据清除

参数: path (str) 要检测的数据的路径, 多个数据的详细清理

常见的问题

- ▶ 1. 我怎么添加自己的模块?
可以参考~/mycar/code_template.py,上面有详细的注释
- ▶ 2. 我怎么将添加的模块应用到小车上?
可以参考~/mycar/sample.py, 上面有详细的注释

- ▶ 3. 我训练时总是报错怎么办?
查看小车上配置文件，确保小车上相机的分辨率和服务服务器上配置文件的分辨率一致
查看小车上配置文件，确保小车上SMOOTH和服务服务器上SMOOTH一致
在服务器输出nvidia-smi或者是nvidia-smi查看gpu显存是否还有剩余
- ▶ 4. 我想在小车上编辑代码应该怎么操作?
使用vim编辑
在本地安装Xming，配置好之后在小车的终端输入charm,小车上预装了pycharm
使用web的文件编辑功能进行文件编辑
- ▶ 5. 我在小车上装python库总是失败怎么办?
输入date查看小车的的时间是否自动更新成正确的时间，如果不是，使用更新成当前时间,下面是示例

```
1 | sudo date -s "2021/5/1 11:11"  
2 | date
```

更新完成之后再重新安装python库

- ▶ 6. 我训练的模型在车上运行总是报错怎么办?
确保训练数据的分辨率和小车当前的分辨率一致
确保训练时SMOOTH的值和当前一致
确保训练时的tensorflow版本和当前tensorflow版本一致（默认tensorflow版本是1.15.2）
- ▶ 7. 我用命令程序卡在加载配置文件一直不往下执行怎么办?
执行命令：

```
1 | ps -ef | grep python3  
2 | sudo kill -9 <pid>
```

在web上查看小车状态，如果小车状态不是灰色的话点击刷新按钮进行刷新。

欢迎使用Mousika管理工具 退出登录

控制面板 小车控制

小车控制 小车状态: ●

自动驾驶	手动驾驶	刷新
<p>请选择路径</p> <p>请选择路径</p> <p>启动驾驶</p>	<p>请检查小车状态，确保小车正在驾驶。</p> <p>请选择启动方式</p> <p>启动驾驶</p>	<p>刷新小车状态</p> <p>刷新</p>