1. 运行grasp.py执行任务流。
2. 对于抓取时的末端执行器长度补充方式做了进一步优化。之前的解决方案是在graspnet预测出的平移向量中叠加夹爪长度，或者在获取当前机械臂位姿的时候沿着z轴平移夹爪长度。第二种解决方法存在一个问题，那就是手眼标定的结果是在未进行平移的坐标系上计算出的，所以这种方式并不一定是最优解。

目前该版代码采取的方式是不进行夹爪长度补偿，直接把夹爪当成不存在计算到最后。在最终抓取位姿上直接沿着末端执行器负方向平移夹爪长度距离，这种方式应该是在原理上最准确的。

1. 我选用的是Qwen2.5-VL-7b作为核心驱动，利用vllm部署，需要注意的是，假如您也想部署该系列模型进行复现，需要特殊的方法安装vllm：  
   pip install git+https://github.com/huggingface/transformers@f3f6c86582611976e72be054675e2bf0abb5f775

pip install accelerate

pip install qwen-vl-utils

pip install 'vllm>0.7.2'

具体细节请参考：<https://github.com/QwenLM/Qwen2.5-VL>

正确安装vllm后，启动脚本可参考下面：

CUDA\_VISIBLE\_DEVICES=0 vllm serve /data5/dehaozhou/LLaMA-Factory/model/Qwen2.5-VL-7B-Instruct/ --port 1034 --host 0.0.0.0 --served-model-name "qwen7b" --dtype bfloat16 --limit-mm-per-prompt image=1,video=1 （可选：--gpu-memory-utilization 0.3，这个决定了调用多少gpu资源）

您可以选择公网的大模型端口，通过api直连，效果更好！

1. ASR和TTS模型我选择的是 GPT-SoVITS和SenseVoice，您也可以自身的硬件条件更换为其他的类似模型，如果以服务形式挂载模型，在其官方github都有说明，不同的声学模型使用方式不同，再次不过多赘述。
2. 代码中的prompts是一个简单示例，您可以根据需求自行更换调整来适配不同的任务。
3. 睿尔曼的机械臂在完成部署上述模型的部署后可以一键运行该套代码，进行交互式智能抓取。