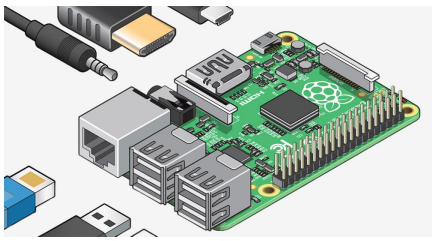
# 大模型AI机械臂使用手册



本手册旨在指导用户如何使用大模型 AI 技术与机械臂进行交互，完成如物体识别、坐标转换、动作执行等任务。系统主要依赖语音识别、图像识别与坐标转换来控制机械臂的抓取和放置操作。

目录

# 一、前期准备

## 1、大模型 AI 机械臂前期知识准备

### 1.1 人工智能与大模型

大模型（如 GPT、BERT）能够处理复杂的自然语言理解与生成任务。它在系统中主要用于：

- **语音识别与解析**：将用户的指令（如“把浅蓝色方块放到车上”）转化为可执行的任务。
- **图像理解**：通过视觉模型从图像中提取关键信息，如物体的位置、颜色和形状。

这些大模型依托庞大的数据集进行预训练，并通过微调适应特定任务场景。

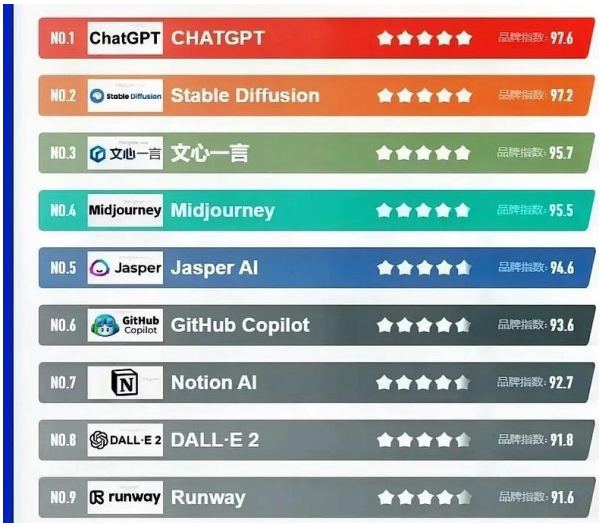



图 1：AI 模型排行

### 1.2 机械臂基础

机械臂是机器人领域的关键设备，它通过多个关节和自由度（DOF）完成各种复杂操作。在本系统中，机械臂具备 6 个自由度（图 2：myCobot 280），能够在三维空间内完成精确的抓取与放置任务。

- **自由度（DOF）**：指机械臂在空间中的独立运动方向，6 个自由度的机械臂可以进行前后、左右、上下、旋转等动作。
- **末端执行器**：机械臂的末端使用的是一个气泵，用于抓取物体并放置到指定位置。


图 2：myCobot 280

## 1.3 计算机视觉与图像识别

在本系统中，图像识别用于确定机械臂操作的对象，并通过坐标转换将像素位置映射到机械臂的工作空间。

- **图像识别**：通过拍摄的图片，提取如物体颜色、形状等关键信息，并确定物体的坐标。
- **坐标转换**：将图像中的像素坐标转换为机械臂可操作的 XYZ 坐标，用于精确执行操作。

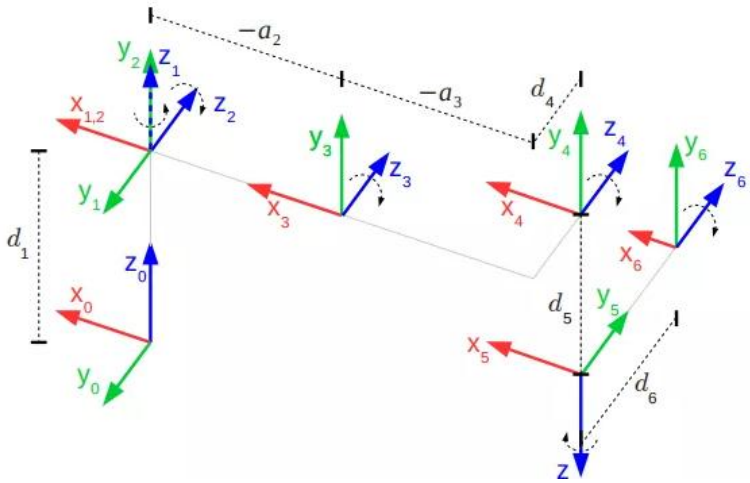



图 3：机械臂坐标逆解图解

## 1.4 坐标系统与转换

- **图像坐标系**：图像中的像素位置，如 (x, y)，是二维平面的坐标系。
- **机械臂的 XYZ 坐标系**：机械臂使用的是三维坐标系统，通过坐标转换模块，系统将图像中的二维坐标映射为机械臂的三维坐标。

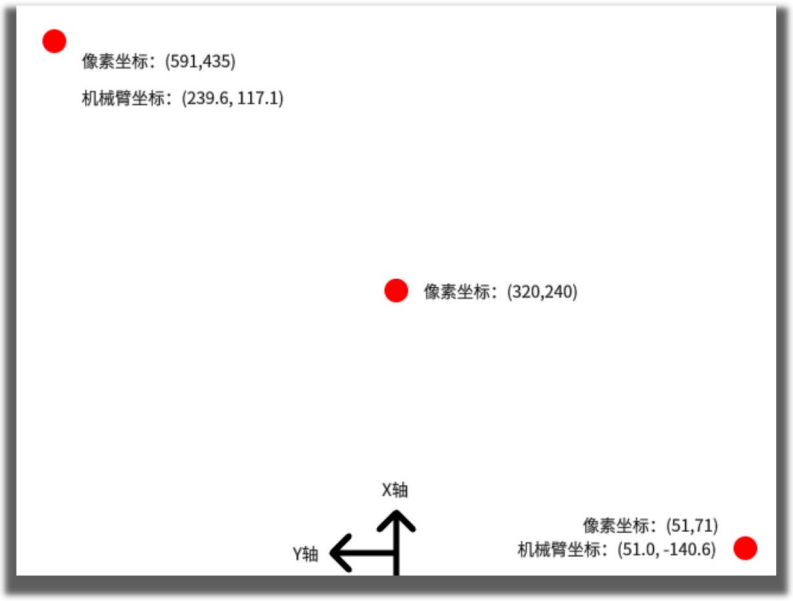



图 4：机械臂图像坐标系（像素为单位）

## 1.5. 机械臂工作流程

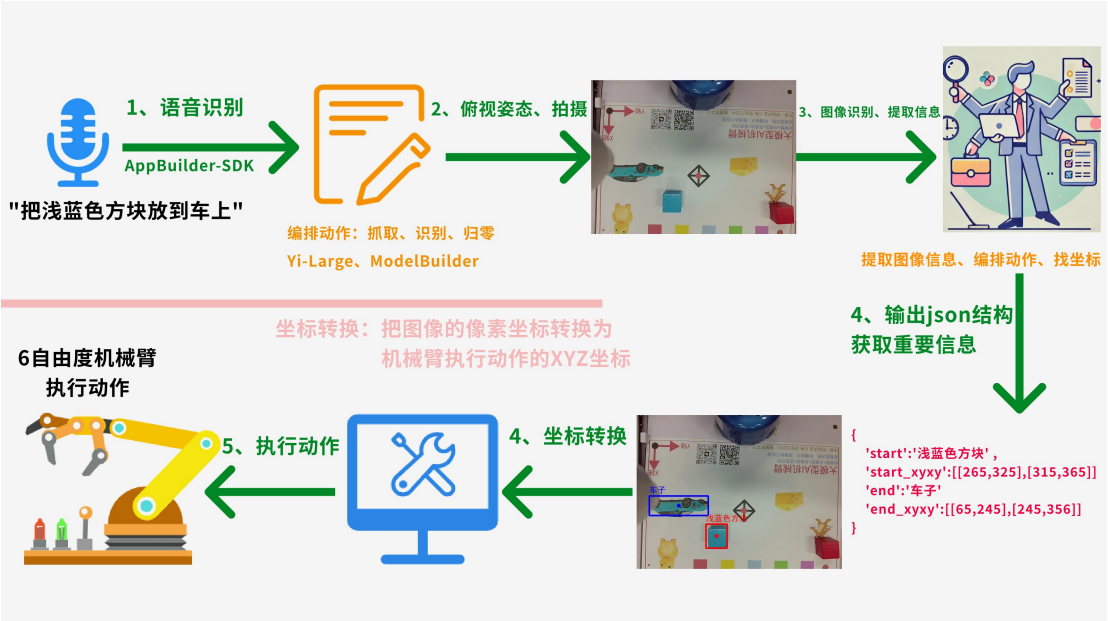



图 5：大模型 AI 机械臂工作流程

### 1.5.1 语音指令的输入与解析

用户通过语音输入指令（或者通过终端输入指令），系统使用语音识别模型进行解析，并将自然语言转化为结构化的任务。例如，用户说：“把浅蓝色方块放到车上”，系统会解析出抓取目标物体（浅蓝色方块）和放置目标位置（车上）。

### 1.5.2 图像识别与信息提取

机械臂进入到俯视状态，系统拍摄当前工作区域的图像，并使用视觉模型识别图像中的物体和位置。提取的信息包括：

- 目标物体的颜色、形状
- 目标物体和目标位置的坐标

### 1.5.3 坐标转换

提取到的物体在图像中的像素坐标需要转换为机械臂可以执行的 XYZ 坐标。通过内置的转换算法，系统将图像中的像素坐标（如 `[265, 325]`）转换为机械臂的工作空间坐标（如 `[x, y, z]`）。

### 1.5.4 机械臂的动作执行

转换完成后，系统将生成的 XYZ 坐标发送给机械臂控制模块。机械臂根据这些坐标执行抓取、移动、放置等操作。整个过程包括：

1. 移动机械臂到物体所在位置。
2. 抓取物体。
3. 移动到目标位置并放置物体。

### 1.5.5 输出结果

任务完成后，系统会生成一个 JSON 结构化数据，包含物体的起始坐标、目标位置和执行的任务状态。这些数据可以用于记录或后续的分析。

为什么是 JSON 格式呢？

JSON（JavaScript Object Notation）格式是一种轻量级的数据交换格式，易于阅读和编写，同时也便于机器解析和生成。在大模型 AI 和机械臂的应用中，选择使用 JSON 格式来输出任务结果有以下几个原因：

**为什么选择 JSON 格式？**

JSON 格式使用键值对的结构，数据清晰明确，易于人类和机器阅读。例如，在机械臂任务的输出中，JSON 可以清晰表示物体的起始位置、目标位置和任务状态。

可读性强：JSON 格式使用键值对的结构，数据清晰明确，易于人类和机器阅读。例如，在机械臂任务的输出中，JSON 可以清晰表示物体的起始位置、目标位置和任务状态。

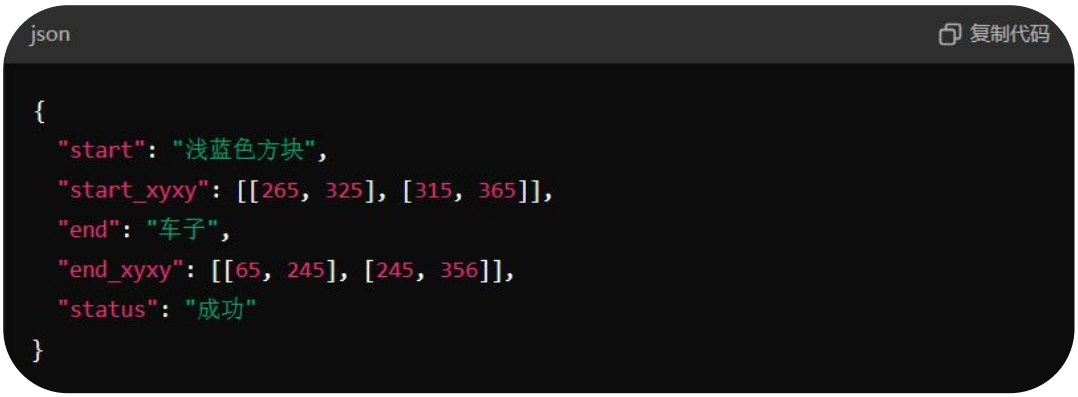



图 6： json 格式输出

标准化的数据结构：JSON 是一种标准化的数据结构，广泛应用于网络数据传输和系统交互中。它可以在不同的编程语言和系统之间轻松传递和解析。在复杂系统中（如 AI 模型与机械臂的交互），标准化的格式能确保各个模块之间无缝通信，减少格式转换的复杂性。

灵活性高：JSON 支持各种数据类型（如字符串、数字、数组和对象），可以灵活适应不同的输出需求。例如，AI 模型可以输出坐标、状态、动作指令等多种信息，并且这些信息可以根据需要添加或调整。

易于解析与扩展：在实际应用中，机械臂需要根据 AI 模型的输出进行具体操作。JSON 格式可以方便地被解析，并将数据用于控制系统。例如，解析后的 JSON 可以直接转换为具体的抓取或移动指令。

在模型的使用中，我们需要给模型一种明确的指令格式。通过 JSON 这种结构化的格式，可以精确告知模型所需的输出内容，从而避免输出中夹杂多余或不相关的数据。例如，如果我们只需要物体的坐标信息和状态，我们可以通过 JSON 格式明确这些需求，而模型会根据这些指令提供相应的结果，避免了不必要的信息输出。

如果在操作时不指定输出格式，模型可能会按照其默认的训练方式输出，而这些数据可能会包含我们不需要的额外信息，导致后续操作复杂化。因此，在每次使用之前，明确告诉模型需要输出的格式和内容是至关重要的。

**为什么在模型使用中提前指定输出格式很重要？**

减少冗余数据：如果没有提前告知模型输出格式，它可能会按照训练时的方式输出大量我们不需要的数据。通过 JSON 格式，我们可以精确指定模型只输出我们需要的数据，例如物体坐标、颜色、任务状态等。

提高模型效率：模型按照预定的格式输出，能减少数据后处理的工作量，也可以提高系统的运行效率。例如，如果只需要提取坐标和状态，通过 JSON 输出可以直接获得关键信息，而无需额外的步骤筛选数据。

便于调试和维护：结构化的 JSON 格式让调试过程更加方便。开发者可以快速定位问题，例如某个字段是否输出正确、状态是否符合预期等。同时，这也让后续的维护工作更为简单。

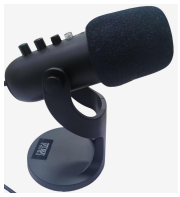
# 2、硬件：

1.1、MyCobot 280 PI 固件：https://www.elephantrobotics.com/download/

| | | | |
|---|---|---|---|
| myCobot 280 PI | ubuntu 18.04 | 点击下载 >> | 04e40af5b637ec003a8b23ef9012e353361fd336db4e17cf9a65feb75e92927e |
| | ubuntu 20.04 | 点击下载 >> | |

1.2、MyCobot 280 API 使用说明：
https://docs.elephantrobotics.com/docs/mycobot_280_pi_cn/3-FunctionsAndApplications/6.developmentGuide/python/7.2_API.html　在使用和开发过程中，可以在 MyCobot 280 PI 官网找到机械臂 API 的使用方法

1.3、录音器材（无要求，可以录音就行）
1.4、外放设备（无要求，可以外放声音就行）
1.5、12v 电源线
1.6、屏幕和键盘（初次使用需要配置网络）



# 3、软件：

## 3.1、图像识别：零一万物

https://platform.lingyiwanwu.com/apikeys

### 3.1.1 打开零一万物网址

选择“开发模式”，因为我们要实现图像的识别，所以模型选择“yi-vision”，其它参数默认。模型的选择可以参考---图 2：“零一万物”模型的种类。


图 7：“零一万物”模型使用

| | | | | |
|---|---|---|---|---|
| yi-lightning | 16K | 最新高性能模型，保证高质量输出同时，推理速度大幅提升 | 适用于实时交互，高复杂推理场景，极高的性价比能够为商业产品提供极好的产品支撑 | ¥0.99 |
| yi-large | 32K | 最新版本的yi-large模型。<br>千亿参数大尺寸模型，提供超强问答及文本生成能力，具备极强的推理能力。并且对 System Prompt 做了专属强化。 | 适合于复杂语言理解、深度内容创作设计等复杂场景。 | ¥20 |
| yi-medium | 16K | 中型尺寸模型升级微调，能力均衡，性价比高。深度优化指令遵循能力。 | 适用于日常聊天、问答、写作、翻译等通用场景，是企业级应用和AI大规模部署的理想选择。 | ¥2.5 |
| yi-vision | 16K | 复杂视觉任务模型，提供基于多张图片的高性能理解、分析能力。 | 适合需要分析和解释图像、图表的场景，如图片问答、图表理解、OCR、视觉推理、教育、研究报告理解或多语种文档阅读等。 | ¥6 |
| yi-medium-200k | 200K | 200K超长上下文窗口，提供长文本深度理解和生成能力。 | 适用于长文本的理解和生成，如文档阅读、问答、构建知识库等场景。 | ¥12 |
| yi-spark | 16K | 小而精悍，轻量极速模型。提供强化数学运算和代码编写能力。 | 适用于轻量化数学分析、代码生成、文本聊天等场景。 | ¥1 |
| yi-large-rag | 16K | 实时全网检索信息服务，模型进阶能力。基于yi-large模型，结合检索与生成技术提供精准答案。<br>支持客户私有知识库（请联系客服申请）。 | 适用于需要结合实时信息，进行复杂推理、文本生成等场景。<br>（私有化知识库，可用于商品推荐、客服问答等特定场景） | ¥25 |
| yi-large-fc | 32K | 在 yi-large 模型的基础上支持并强化了工具调用的能力，模型可以根据用户传入的工具定义，决定是否需要调用，并按照指定格式输出调用方式。 | 适用于各种需要搭建 agent 或 workflow 的业务场景。 | ¥20 |
| yi-large-turbo | 16K | 超高性价比、卓越性能。根据性能和推理速度、成本，进行平衡性高精度调优。 | 适用于全场景、高品质的推理及文本生成等场景。 | ¥12 |

图 8：“零一万物”模型的种类

## 3.1.2、模型使用

在模型的使用中，我们需要给模型一种指令，我们需要模型输出什么，我们就要告诉模型该怎么做，如果没有提前把格式告诉模型，它会默认按照工程师训练的方式输出，这样就会有很多我们不需要的数据，所以在每次使用之前，告诉模型它要做什么，是至关重要的。

例如，在我们这个项目当中，机械臂需要知道抓取物体是哪个，需要放置在哪里，所以需要模型输出抓取物体的位置、放置物体的位置，这样，机械臂才能完成我们下发的任务。

我即将说一句给机械臂的指令，你帮我从这句话中提取出起始物体和终止物体，并从这张图中分别找到这两个物体左上角和右下角的像素坐标，输出 json 数据结构。 例如，如果我的指令是：请帮我把红色方块放在车上。 你输出这样的格式： { "start":"红色方块", "start_xyxy":[[102,505],[324,860]], "end":"车子", "end_xyxy":[[300,150],[476,310]] } 只回复 json 本身即可，不要回复其它内容 我现在的指令是：把红色方块放到老鼠身上


图 9：“零一万物”图像识别截图

在获得 JSON 输出后，系统会将这些位置信息传递给机械臂的控制模块。机械臂接收到的关键信息包括：

起始物体的位置（用于抓取）
目标物体的位置（用于放置）

执行步骤：
机械臂根据起始坐标（start_xyxy）移动到红色方块的位置。
机械臂抓取红色方块。
机械臂根据终止坐标（end_xyxy）移动到老鼠的位置，并将红色方块放置在老鼠上。

获取到的信息将在后面章节“二、机械臂的运动---2、起点和目的点”用到。

## 3.2、语音识别和合成---百度智能云

### 3.2.1、登录百度智能云控制台

https://console.bce.baidu.com/ai/#/ai/speech/overview/index

出现下面的界面就是可以的了，如果不是，可以点左上角 9 个点的图标那里，选择我们的“语音技术”，就可以来到下一步。

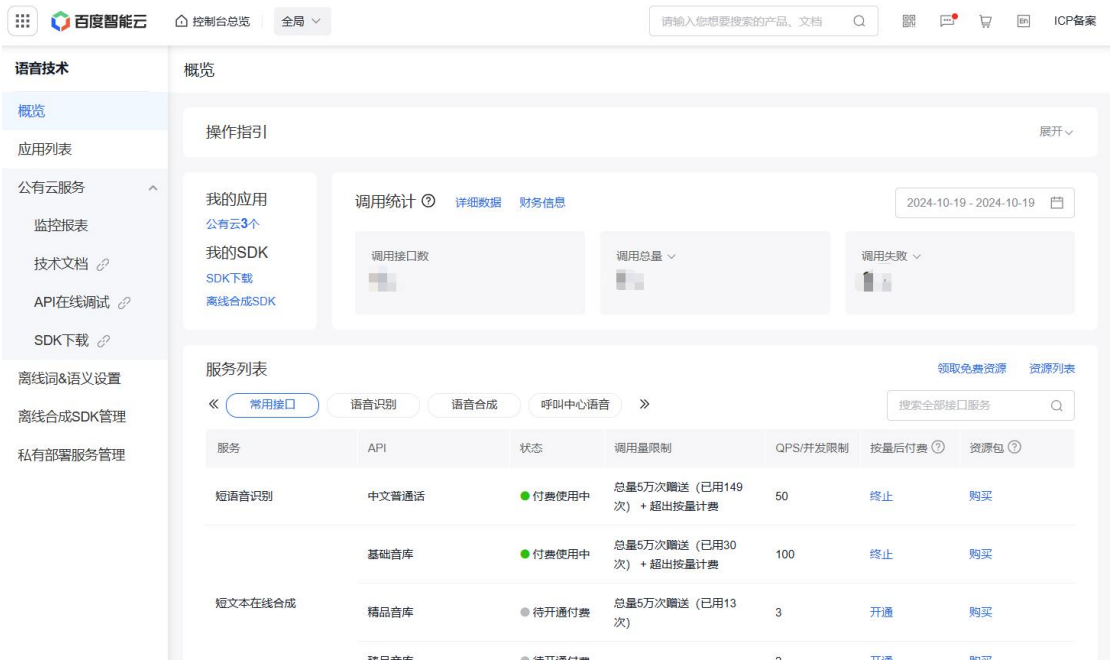



图 10.1：百度智能云控制台

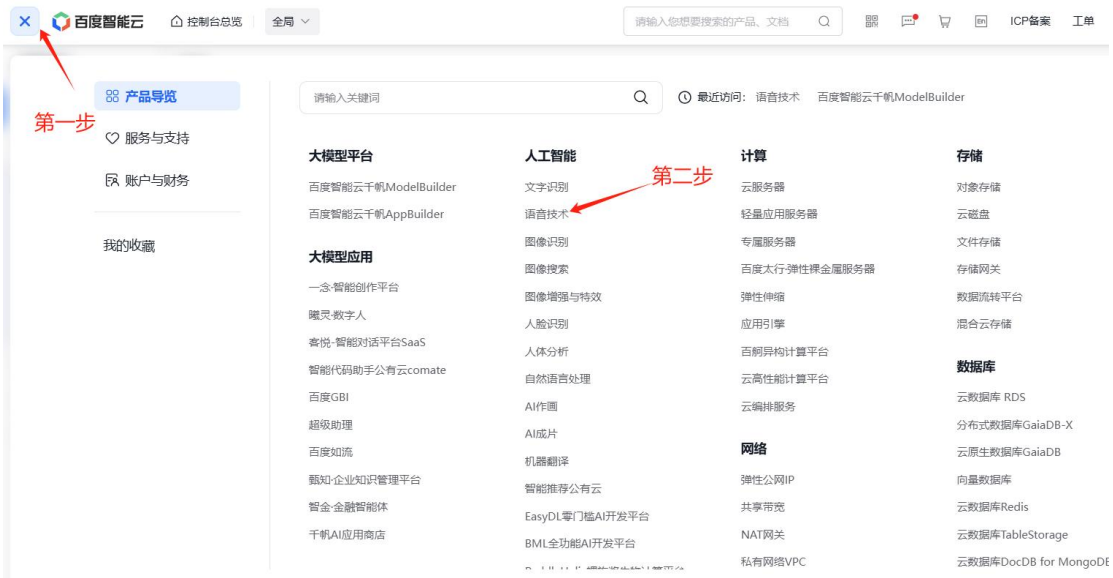



图 10.2：百度智能云控制台 2

## 3.2.2、创建应用

点击创建应用，填写好所需内容（应用名称可以中文），这里我们只选择了我们需要的语音合成和识别，后期需要别的再添加也可以。

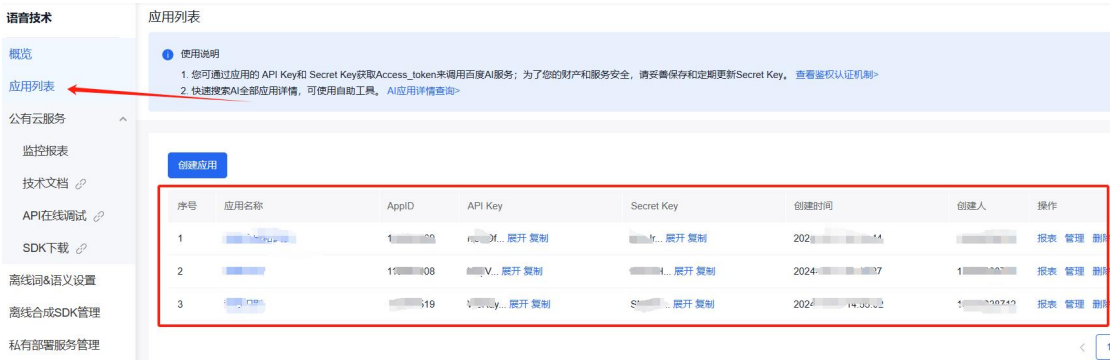



图 11：创建应用


图 12.1：填写应用信息


图 12.2：填写应用信息

回到开始的界面，我们就可以看到我们刚刚新添加的应用名称在这里了，那么我们就进行下一步的模型使用。

这里的 API Key、Secret Key 参数值是每一个应用自动生成，要注意保管，每次调用都需要耗费资源，一定要谨慎保管！


图 13：创建的应用

### 3.2.3、鉴权认证机制

在左侧选择“API 在线调试”。这里我们将进行三步来调用模型，测试模型是否可以使用。

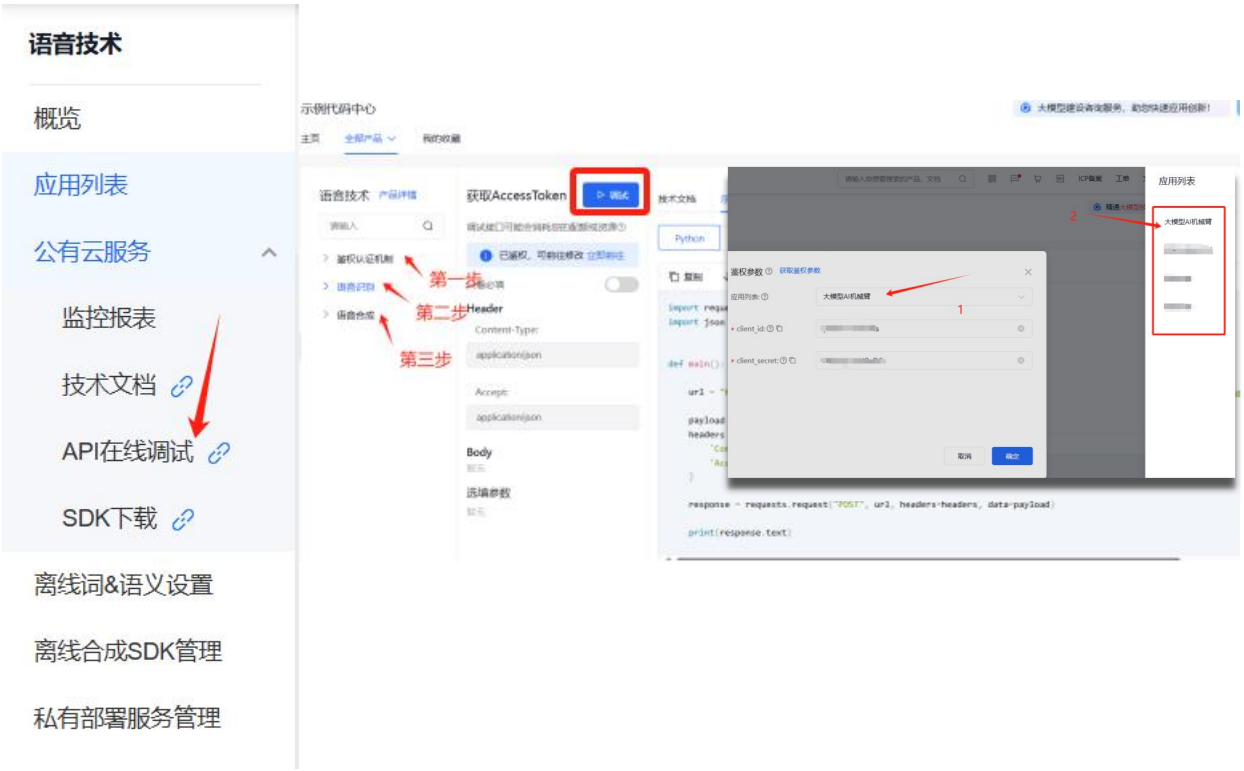



图 14.1：获取 AccessToken

第一步：在左侧点击“鉴权认证机制”---“获取 AccessToken”---“立即前往”---“应用列表”（选择我们刚刚新建的应用，这里的 ID 和 secret 会自动填写）


图 14.2：获取 AccessToken

## 3.2.4、语音合成

选择“语音合成”中间这里的参数我们选择默认的就行，具体用法可以在每个参数前的问号里找到讲解。在“tex”（或者右边输入都是一样的可以的）这里我们填写我们需要合成的文本，点击“调试”。


图 15.1：语音合成

这里的音库有很多种，可以选择我们喜欢的音色（不影响我们后期程序的运行），音频格式我们选择“wav”，部署在开发板中才容易使用。点击下面的 3 个点，就可以把合成的语音内容保存到本地，方便我们下一步“语音识别”的应用。

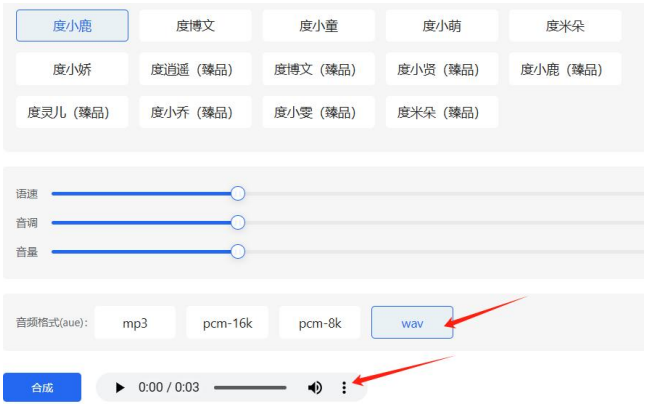



图 15.2：语音合成

在上面我们是在官网使用的，那么我们的机械臂怎么去使用这个功能呢？
在上面，找到“示例代码”---“Python”。


图 16：示例代码

官网给出的示例代码中，我们只需要修改“text”里面的内容就可以调用大模型（示例代码里面的已经是通过解码的了，所以呈现是这样，我们只需要按照正常的输入一串字符串就可以）

API_KEY、SECRET_KEY 是在我们创建应用后生成的，可以参考本目录的“2.2.2 创建应用”查看。

```
import requests

API_KEY = "y                    Ms"
SECRET_KEY = "                    "

def main():

    url = "https://tsn.baidu.com/text2audio"

    payload='tex=%E4%BD%A0%E5%A5%BD%EF%BC%8C%E6%88%91%E6%98%AF%E5%B0%8F%E6%99%BA%E5%A4%A7%E6%A8%A1%E5%9E%8BAI%E6%9C%BA%E6%A2%B0%E8%87%82&t
    headers = {
        'Content-Type': 'application/x-www-form-urlencoded',
        'Accept': '*/*'
    }

    response = requests.request("POST", url, headers=headers, data=payload)

    print(response.text)


def get_access_token():
    """
    使用 AK，SK 生成鉴权签名（Access Token）
    :return: access_token，或是None(如果错误)
    """
    url = "https://aip.baidubce.com/oauth/2.0/token"
    params = {"grant_type": "client_credentials", "client_id": API_KEY, "client_secret": SECRET_KEY}
    return str(requests.post(url, params=params).json().get("access_token"))

if __name__ == '__main__':
    main()
```

图 17：示例代码

## 3.2.5、语音识别

选择“语音识别”中间这里的参数我们选择默认的就行，具体用法可以在每个参数前的问号里找到讲解。在“speech”这里我们上传我们提前准备的录音文件，如果没有，可以使用我们在“语音合成”的文件。

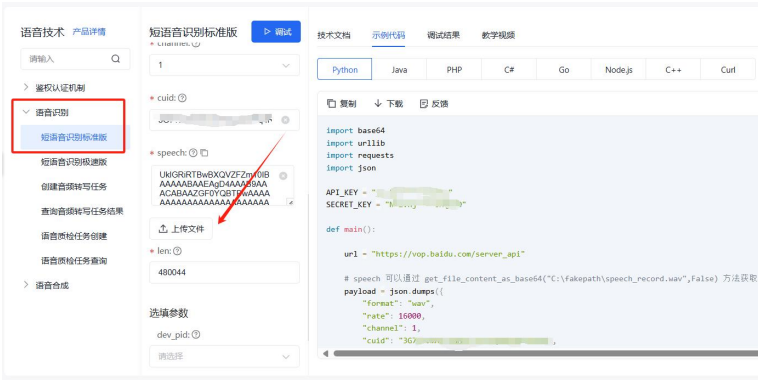



图 18：语音识别

点击“调试”，在界面右边下滑可以看到“响应数据”，result 就是识别到的内容。


图 19：响应数据

在上面我们是在官网使用的，那么我们的机械臂怎么去使用这个功能呢？
在上面，找到“示例代码”---“Python”。


图 20：示例代码

官网给出的示例代码中，我们需要修改“speech”里面的内容就可以调用大模型，我们在这里会看到，speech 的值是很长的，因为音频需要 Base64 编码才可以被大模型接受使用，关于 Base64 编码，我们将在“”章节讲到。API_KEY、SECRET_KEY 是在我们创建应用后生成的，可以参考本目录的“2.2.2 创建应用”查看。

```
API_KEY = "
SECRET_KEY = "

def main():

    url = "https://vop.baidu.com/server_api"

    # speech 可以通过 get_file_content_as_base64("C:\fakepath\text2audio.wav",False) 方法获取
    payload = json.dumps({
        "format": "pcm",
        "rate": 16000,
        "channel": 1,
        "cuid": "h
        "speech": "
        "len": 110124,
        "token": get_access_token()
    })
    headers = {
        'Content-Type': 'application/json',
        'Accept': 'application/json'
    }

    response = requests.request("POST", url, headers=headers, data=payload)

    print(response.text)
```

图 21：示例代码