SONY

AITRIOS

Cloud SDK
Sample Application
Python
Functional Specifications

Version 0.2.0
2023 - 1 - 30

TOC

1. Change history

Date	What/Why
2022/12/12	Initial draft
2023/1/30	Unified the swinging of expressions Fixed the notation Updated the PDF build environment

2. Introduction

- This book is functional specifications for a sample application that provides developers with ways to use and take advantage of the Cloud SDK for Python.
  - Python is used as the function development language.
  - The application framework uses Flask.

3. Terms/Abbreviations

Terms/Abbreviations	Meaning
Cloud SDK	SDK providing a way to access the Console
Console	A cloud service that provides various functions (Deployment, Retraining, Edge AI Device Management etc.) to efficiently implement solutions from edge to cloud
Inference result	AI-processed metadata among outputs from Vision and Sensing Applications
Image	Image data captured by edge AI devices among outputs from Vision and Sensing Applications

4. Reference materials

- Cloud SDK for Python used in sample applications
    - https://github.com/SonySemiconductorSolutions/aitrios-sdk-console-access-lib-python

5. Expected use case

- Provide ways to use and take advantage of the Cloud SDK for Python.
    - Users can see how applications using the Cloud SDK work by launching applications in the repository.
    - Users can see how to use the Cloud SDK by reviewing the source code.

6. Functional overview/Algorithm

Functional overview

- Users can see the latest image and inference results on the screen.
  - The base AI model only supports Object Detection.
- The Start/Stop button will appear by selecting the DeviceID.
- By pressing the START button, the latest image/inference results is gotten and displayed on the screen.
- By pressing the STOP button, getting the latest image/inference result is stopped.

Algorithm

1. Launch the screen.
   a. Call the getDeviceData.
   b. Display the returned data in the DeviceID selection field.
2. DeviceID is entered, the START button is pressed.
   a. Call the getCommandParameterFile to check that the settings are as follows. (Display a message if there is an error.)
      - Mode=1(Image&Inference Result)
      - UploadMethodIR="Mqtt"
   b. Call the startUpload to start upload of inference results and images.
   c. Call getImageAndInference periodically to get inference results and images.
   d. Display the gotten data on the screen.
3. Press the STOP button.
   a. Call the stopUpload.

Under what condition

- Have access to the Console.
- A Python development environment has been built.
  - A Codespaces environment is also available.
  - Python version is 3.10.

- An edge AI device is connected to the Console and ready to accept operations from the Console.

API

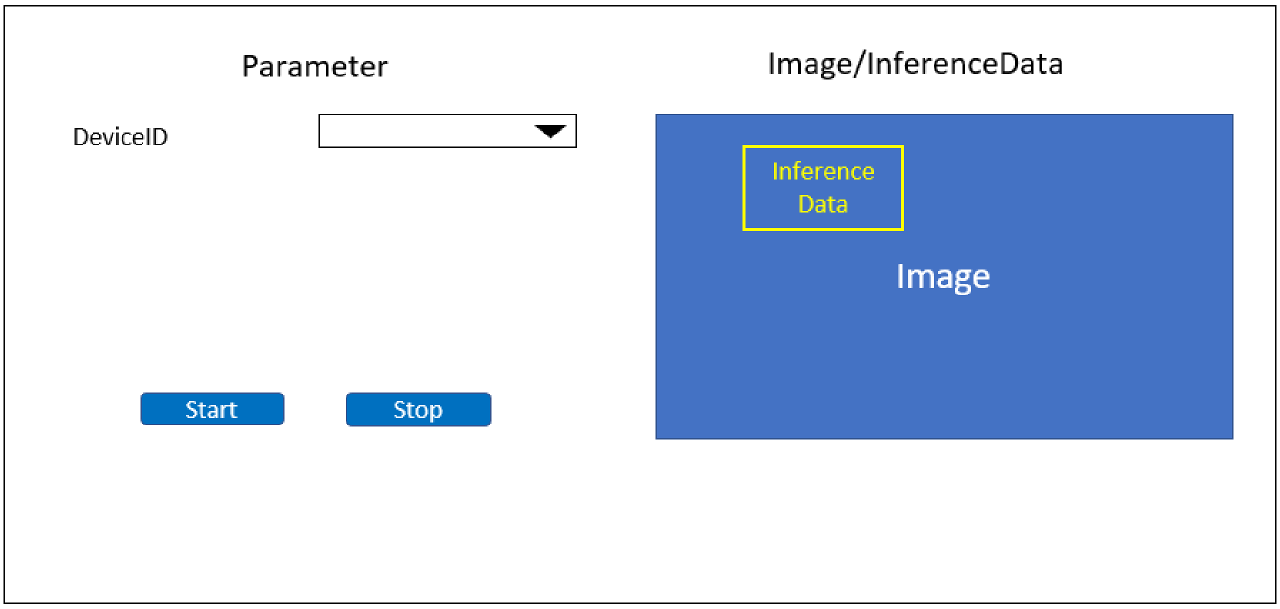
- GET
  - {base_url}/getDeviceData
  - {base_url}/getCommandParameterFile/device_id
  - {base_url}/getImageAndInference/device_id/sub_directory_name
- POST
  - {base_url}/startUpload/device_id
  - {base_url}/stopUpload/device_id

Others exclusive conditions/Specifications

- None

7. User interface specifications

Screen specifications



Operability Specifications

Operation to launch the sample application

When to use Codespaces

1. Developers open the repository of the sample application from any browser and launch Codespaces.
2. Build containers in the cloud with reference to configuration files that exist in repositories.
3. Use the built container in the browser or from VS Code.
4. Launch the sample application.

When not to use Codespaces

1. Developers open the repository of the sample application from any browser and clone the repository.
2. Install the necessary packages for the cloned sample application.
3. Launch the sample application.

After starting the sample application

1. Select the [**DeviceID**].

2. By pressing the [**START**] button, the latest image/inference results is gotten and displayed on the screen.
3. By pressing the [**STOP**] button, getting the latest image/inference result is stopped.

8. API parameters in each block

GET

- {base_url}/getDeviceData
    - Get and return the list of DeviceIDs.

Query Parameter's name	Meaning	Range of parameter
-	-	-

Return value	Meaning
device_data	Object where DeviceIDs are stored

- {base_url}/getCommandParameterFile/device_id
    - Get the list of Command Parameter Files registered in the Console and return the settings.

Query Parameter's name	Meaning	Range of parameter
device_id	DeviceID uploading images and inference results	Not specified

Return value	Meaning
mode	Mode settings registered in the Console
upload_methodIR	UploadMethodIR settings registered in the Console

- {base_url}/getImageAndInference/device_id/sub_directory_name
    - Get and return inference results and images for the specified edge AI device.

Query Parameter's name	Meaning	Range of parameter
device_id	DeviceID uploading images and inference results	Not specified
sub_directory_name	Path where images are stored	Not specified

Return value	Meaning
image_and_inference	Object where image paths and inference results are stored

POST

- {base_url}/startUpload/device_id
  - Request to start uploading inference results and images for the specified DeviceID.

Body Parameter's name	Meaning	Range of parameter
device_id	DeviceID to start uploading images and inference results	Not specified

Return value	Meaning
result	SUCCESS or ERROR string
output_sub_directory	Input image storage path

- {base_url}/stopUpload/device_id
  - Request to stop uploading inference results and images for the specified DeviceID.

Body Parameter's name	Meaning	Range of parameter
device_id	DeviceID to stop uploading images and inference results	Not specified

Return value	Meaning
result	SUCCESS or ERROR string

9. Target performances/Impact on performances

- None

10. Assumption/Restriction

- From the Console UI, set the Command Parameter File to the following setting.
  - Mode=1(Image&Inference Result)
  - UploadMethodIR="Mqtt"
- Object detection is deployed as the base AI model.
- If you select an edge AI device that does not have an AI model or application deployed at runtime, it will not work properly.

11. Remarks

- Image uploads from edge AI devices to the cloud can experience delays of up to several minutes.

12. Unconfirmed items

- None