HT_Pi & Pi Plus Policy Conversion User Manual

This manual serves as a practical guide for the strategy format conversion of HT_Pi and Pi Plus robots. Its core objective is to help users successfully convert ONNX-format models to RKNN format compatible with the robots, laying a solid foundation for subsequent strategy deployment, with the operating platform.

1. Environment Preparation (Operating Platform: Ubuntu)

Solution A: Using Miniconda/Anaconda (Recommended)

To avoid environment conflicts, it is recommended to create an independent virtual environment:

代码块

```
# Create an RKNN conversion environment (Python 3.8)
conda create -n rknn_model python=3.8

# Activate the environment
conda activate rknn_model

# Install the RKNN conversion toolkit
pip install rknn-toolkit2

# Update the Pillow library
pip install --upgrade pillow
```

Solution B: Direct Installation (No Miniconda/Anaconda)

Ensure the system Python version is 3.8 or above, then execute:

代码块

```
# Install the RKNN conversion toolkit
pip3 install rknn-toolkit2

# Update the Pillow library
pip3 install --upgrade pillow
```

2. Code Acquisition and Configuration

2.1 Obtain the Conversion Code

```
代码块
# Clone the code repository
git clone https://github.com/HighTorque-Robotics/Policy-Format-Convert.git

# Enter the project directory
cd Policy-Format-Convert
```

2.2 Modify File Paths

Open the conversion script `onnx2rknn.py` with a text editor:

- Modify the model loading path:

```
代码块
# Modify the loading path - Find the following code and replace it with your ONNX model path
print("--> Loading model")
ret = rknn.load_onnx("/path/to/your/input_model.onnx") # Replace with the actual ONNX file path
```

- Modify the output path configuration:

```
代码块
# Modify the output path - Find the following code and replace it with your output directory
OUT_DIR = "/path/to/your/output_directory" # Replace with the actual output directory
RKNN_MODEL_PATH = "{}/policy_from_onnx.rknn".format(OUT_DIR)
```

3. Execute Model Conversion

```
代码块
# Run the conversion script
python onnx2rknn.py
```

4. Verify Conversion Results

After successful conversion, the following will be generated in the specified output directory:

- `policy_from_onnx.rknn` - The converted RKNN model file
- Conversion log information (displayed in the terminal)

5. Notes

5.1 Environment Requirements

- Python version must be 3.8.x
- Ensure the system has installed all necessary dependency libraries

5.2 Path Specifications

- Avoid spaces and special characters in the path
- It is recommended to use absolute paths to ensure correctness

5.3 Model Compatibility

- Ensure the input ONNX model is compatible with the operator set supported by RKNN
- Complex models may require operator adaptation

5.4 Error Handling

- If conversion fails, check the error log
- Common issues include: unsupported operators, incorrect input/output formats, path problems, etc.

This manual provides a complete RKNN model conversion process. Following these steps can successfully convert ONNX-format models to RKNN format.