A “Product-Oriented Design” IDE
Make Creating Robots Easier
Content

1. The requirements of robot research and development
2. RoboWare and its values
3. Plan in the future
Development of Robots

Requirements
Development of Robots

- Design tool
- Software IDE
- GUI development tool
- Hardware
Development of Robots

- Design tool
- Software IDE
- GUI development tool
- Hardware

Requirements
Development of Robots

- Design tool
- Software IDE
- GUI development tool
- Hardware
Development of Robots

- Design tool
- Software IDE
- GUI development tool
- Hardware
Robot development IDE

RoboWare
Make creating robots easier
RoboWare

New Concept: POD Product Oriented Design

Designer → Studio → Viewer → Robots

Develop Robot → Develop Graphic UI
The Values of Designer
Quickly build the hardware architecture

Drag and drop, quickly finish the hardware design
Features of Designer

- Hardware selection
- Port checking
- Code workspace generating
- ROS package auto downloading
New Designer Interface
The output of Designer

Design Document (.rwd)  BOM List
RWD File

.rwd

Upload

Store
We are doing…

We are adding some common hardware components, and your participation will make a big difference.
The Values of Studio
Development process

Create ROS workspace

Write the ROS node

Local debugging

Remote deployment

Remote debugging
ROS C++/Python Code debugging function

—— Easier/ Shorten the development cycle ——

debug control

variable display

break point

call stack

Set breakpoint | Check the variable | Display call stack | Single step run | …
Built-in ROS package management

Convenient installation of ROS packages/ uninstall & check Wiki

ROS package one-click-(un)install | meta-package and package searching | integrated ROS Wiki browse
VIM VS Studio

Normal development mode

Studio development mode

VICTORY
```python
# Author: Wim Meeussen

from __future__ import with_statement

import roslib; roslib.load_manifest('dashgo_calibration')
import yaml
import rospy
import lasereye

from sensor_msgs.msg import LaserScan
from dashgo_msgs.msg import ScanAngle
from math import *
from std_msgs.msg import Int16

class ScanToAngle:
    def __init__(self):
        self.min_angle = laser_eye.get_param('min_angle', -0.4)
        self.max_angle = laser_eye.get_param('max_angle', 0.4)
        self.pub = laser_eye.Publisher('scan_angle', ScanAngle)
        self.sub = laser_eye.Subscriber('scan', LaserScan, self.scan_cb)
        self.angle_pub = laser_eye.Publisher('angle', Int16)

    def scan_cb(self, msg):
        angle = msg.angle_min
```

RoboWare Studio
Design for ROS

Robot chassis

Robot machine
Studio Project File

Studio ➔ Project File (.rws) ➔ RoboStore
Buy directly without leaving the IDE
The Values of Viewer
Efficient development of CMS and UI

- Visual interface editing
- High-efficient development
- Quick response
RWV File

RoboViewer ➔ Project File (.rwv) ➔ RoboStore
Buy directly without leaving the Viewer
We need you

You are welcome to provide components.
We will make it

OPEN SOURCE
The Values of RoboStore

- Precision of product description
- Label ROS compatibility
- Product and service provider
- Good user experience
- Provide application cases
  Solutions in different scenarios
  or Product portfolio case
RoboStore

RoboStore makes creating robots easier.

Search for products like the ASTRA 3D motion-sensing camera, giving you brand-new operation experience with a powerful 3D computer chip and accurate AI recognition.

Best sellers:
The Future of RoboStore

- Hardware
- Package (rws)
- Designer (rwd)
- Usr Interface (rwv)
- Overall Solution (res/rwd/rwv)
Join RoboStore

Your components are welcome to be online.
Further Work

Thinking about the future
Conclude
Dr. Tony Wang
+86 18605313834
tony.wang@tonyrobotics.com

There is no end to perfection, so just keep going!

Thank you!