Advanced Controller for Robot Manipulator

Jwu-Sheng Hu
Professor, Dept. of Electrical and Computer Engineering, NCTU, Taiwan
Technical Director, MSL, ITRI, Taiwan
Director, Intelligent Robotics Division, MSL, ITRI, Taiwan
Outline

• Advanced Robot Arm
• Real-time OS
• EtherCat Protocol
• Robotic Arm Dynamic Control Technology
• 3D Objects Pose Estimation
• Intelligent Graphic User Interface System
• Advanced Hand-Eye-Workspace Calibration
Advanced Robot Arm

- Designed and assembled by Industrial Technology Research Institute.
- Small and compact size
- Seven degree of freedom

<table>
<thead>
<tr>
<th>主控制器</th>
<th>視覺模組</th>
<th>人機介面</th>
<th>即時集線器</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trajectory following control</td>
<td>Robot Calibration</td>
<td>Task Planning/teaching</td>
<td>Real-time Traffic Control</td>
</tr>
<tr>
<td>Trajectory generation Inverse kinematics</td>
<td>Hand-eye Calibration</td>
<td>Robot/work space model</td>
<td>Gateway</td>
</tr>
<tr>
<td>State machine</td>
<td>Visual Servo</td>
<td>Virtual Reality Platform</td>
<td>DNS/DHCP</td>
</tr>
<tr>
<td>Real-time message</td>
<td>Object Recognition</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EtherCAT</td>
<td>RT Ethernet</td>
<td>Data client</td>
<td></td>
</tr>
<tr>
<td><strong>Ethernet1</strong></td>
<td><strong>Ethernet2</strong></td>
<td><strong>Ethernet 3</strong></td>
<td></td>
</tr>
</tbody>
</table>

EtherCAT servo drives of robot joint motors
Automation Interface
Advanced Robot Arm

- EtherCAT communication
- No control box, and the motor driver is embedded in each link.
- Hollow shaft design for routing
Outline

- Advanced Robot Arm
- **Real-time OS**
- EtherCat Protocol
- Robotic Arm Dynamic Control Technology
- 3D Objects Pose Estimation
- Intelligent Graphic User Interface System
- Advanced Hand-Eye-Workspace Calibration
Real-time OS
Introduction

• Growing penetration of Linux
• OSS EtherCAT has been developed
• Control applications in user space is possible
  – IgH version 1.5.0 or above

• However,
  – IgH can’t guarantee RT capability
  – Even RTAI is adopted
Real-time OS
IgH EtherCAT

• IgH EtherCAT is composed of
  – master module
    • Provide functions and data structures to allow applications accessing the master functionalities via API
  – control application
    • Interface with the master modules by means of API
    • For the cyclic exchange of process data with EC slaves
    • Can be implemented with IgH EC libraries
Real-time OS
Cereia’s Work

• “A user space EtherCAT master architecture for hard real-time control systems”
  • 2012 IEEE ETFA Conference

• Implementation
  • Linux kernel 2.6.32.11 + RTAI 3.8.1
  • IgH 1.5.0

• Improve the user space control application
  • Modify the control application to be RTAI tasks
  • Make it to reach real-time capability
Real-time OS
Drive via Character Device
Real-time OS

Jitter: Kernel vs. User Space
**Kernel space timing**

<table>
<thead>
<tr>
<th>Kernel</th>
</tr>
</thead>
</table>
| while(1) {
|   **rt_task_wait_period();**
|   write_outputs();
|   read_inputs();
|   signal(U-sem);
|   wait(U-sem);
|   wait(K-sem);
| } |

<table>
<thead>
<tr>
<th>User</th>
</tr>
</thead>
</table>
| while(1) {
|   wait(U-sem);
|   /*data processing*/
|   signal(K-sem);
| } |

**User space timing**

<table>
<thead>
<tr>
<th>Kernel</th>
</tr>
</thead>
</table>
| while(1) {
|   wait(K-sem);
|   write_outputs();
|   read_inputs();
|   signal(U-sem);
| } |

<table>
<thead>
<tr>
<th>User</th>
</tr>
</thead>
</table>
| while(1) {
|   **rt_task_wait_period();**
|   signal(K-sem);
|   wait(U-sem);
|   wait(U-sem); /*data processing*/
| } |
Real-time OS
Character Device vs. Shared Memory
Real-time OS
Jitter: User Space/Cereia’s Design
Real-time OS
EC-based Robot OS

EC-Master
ROS

In
out
Slave modules

Motor A
Motor B
Motor C
Real-time OS
Conclusion

• Control application can be implemented at user space based on RTAI
  • Real time guaranteed

• Cereia’s Design can be used to realize a Robot OS
  • For real-time control
Outline

• Advanced Robot Arm
• Real-time OS
• **EtherCat Protocol**
• Robotic Arm Dynamic Control Technology
• 3D Objects Pose Estimation
• Intelligent Graphic User Interface System
• Advanced Hand-Eye-Workspace Calibration
EtherCat Protocol Outline

- Industrial Communication Technique Comparison
- COE Communication Structure
  - Cyclic Data Transmission
  - Acyclic Data Transmission
- EtherCAT Master Processing Procedure
- Device Description File
  - Device Description File Analysis
  - Package Analysis
  - Establish Motion Control Parameters
- EtherCAT Master User Interface Demo
# EtherCAT Protocol

## Industrial Communication Technique Comparison

<table>
<thead>
<tr>
<th>Protocols</th>
<th>Real-time class</th>
<th>Throughput</th>
<th>Max devices</th>
<th>Synchronisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAN</td>
<td>2</td>
<td>10 kb/s–1 Mb/s, 31.25 kb/s, 1 Mb/s, 2.5 Mb/s (5 Mb/s optical fiber)</td>
<td>32</td>
<td>Y</td>
</tr>
<tr>
<td>PROFIBUS</td>
<td>2</td>
<td>9.6 kb/s-12 Mb/s</td>
<td>126</td>
<td>N</td>
</tr>
<tr>
<td>PROFIBUS-PA</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MODBUS</td>
<td>1</td>
<td>10,100 Mb/s, 1 Gb/s</td>
<td>247</td>
<td>N</td>
</tr>
<tr>
<td>Ethernet / IP</td>
<td>1</td>
<td>10,100 Mb/s, 1 Gb/s</td>
<td>Almost Unlimited</td>
<td>Y</td>
</tr>
<tr>
<td>PROFINET</td>
<td></td>
<td>100 Mb/s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PowerLink</td>
<td>3</td>
<td>100 Mb/s</td>
<td>240</td>
<td>Y</td>
</tr>
<tr>
<td>SERCOS</td>
<td>3</td>
<td>100 Mb/s</td>
<td>254</td>
<td>Y</td>
</tr>
<tr>
<td>EtherCAT</td>
<td>3</td>
<td>100 Mb/s</td>
<td>65535</td>
<td>Y</td>
</tr>
</tbody>
</table>

*Real-time Level*
EtherCat Protocol
COE Communication Structure

COE (CANopen Over EtherCAT) includes the following communication mode:

(1) Mailbox mode – Use SDO (Service Data Object) for acyclic transmission.

(2) Buffer mode – Use PDO (Process Data Object) for cyclic transmission.
EtherCat Protocol
COE Communication Structure

• Acyclic Data Transmission (Mailbox mode)
When the sender writes the buffer, the buffer is locked for writing until the receiver has read it out.
EtherCat Protocol
COE Communication Structure

- Cyclic Data Transmission (Buffer mode)

The sender can always update the content of the buffer. If the buffer is written faster than it is read out by the receiver, old data is dropped. Thus, the receiver always gets the latest consistent buffer content which was written by the sender.
EtherCat Protocol
EtherCAT Master Processing Procedure
EtherCAT Protocol
Device Description File

- EtherCAT slave device describe

Set up parameter to go Mapping
Set up Motion control parameters to transmit command
EtherCat Protocol
Device Description File

- Device description file analysis

<table>
<thead>
<tr>
<th>DeviceDescriptionFile Analysis</th>
<th>MappingPhysicalAddress</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>DefaultSize</th>
<th>StartAddress</th>
<th>ControlByte</th>
<th>Enable</th>
<th>MinSize</th>
<th>MaxSize</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>#x1000</td>
<td>#x26</td>
<td>1</td>
<td>34</td>
<td>192</td>
<td>MEoXOut</td>
</tr>
<tr>
<td>2</td>
<td>#x1400</td>
<td>#x22</td>
<td>1</td>
<td>34</td>
<td>192</td>
<td>MEoXIn</td>
</tr>
<tr>
<td>3</td>
<td>#x1800</td>
<td>#x64</td>
<td>1</td>
<td></td>
<td></td>
<td>Outputs</td>
</tr>
<tr>
<td>4</td>
<td>#x1C00</td>
<td>#x20</td>
<td>1</td>
<td></td>
<td></td>
<td>Inputs</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>RxPDO</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed</td>
<td>1</td>
</tr>
<tr>
<td>Sm</td>
<td>2</td>
</tr>
<tr>
<td>Index</td>
<td>DependOnSlot=true</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Outputs</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Index</th>
<th>SubIndex</th>
<th>BitLen</th>
<th>Name</th>
<th>Comment</th>
<th>DataType</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>16</td>
<td>ControlWord</td>
<td>object0x6040:0</td>
<td>UINT7</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>32</td>
<td>TargetPosition</td>
<td>object0x607A:0</td>
<td>DINT</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>TxPDO</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed</td>
<td>1</td>
</tr>
<tr>
<td>Sm</td>
<td>3</td>
</tr>
<tr>
<td>Index</td>
<td>DependOnSlot=true</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Inputs</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Index</th>
<th>SubIndex</th>
<th>BitLen</th>
<th>Name</th>
<th>Comment</th>
<th>DataType</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>16</td>
<td>StatusWord</td>
<td>object0x6041:0</td>
<td>UINT7</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>32</td>
<td>ActualPosition</td>
<td>object0x6064:0</td>
<td>DINT</td>
</tr>
</tbody>
</table>
EtherCAT Protocol
Device Description File

• Package analysis

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>21317</td>
<td>21.192077000</td>
<td>Beckhoff_10:1c:9c</td>
<td>Beckhoff_01:01:00:00</td>
<td>ECAT</td>
</tr>
<tr>
<td>21318</td>
<td>21.192086000</td>
<td>Beckhoff_01:00:00</td>
<td>MS-NLB-PhysServer-01_05:10:1c:9c</td>
<td>ECAT</td>
</tr>
<tr>
<td>21319</td>
<td>21.194077000</td>
<td>Beckhoff_10:1c:9c</td>
<td>Beckhoff_01:00:00</td>
<td>ECAT</td>
</tr>
<tr>
<td>21320</td>
<td>21.194085000</td>
<td>Beckhoff_01:00:00</td>
<td>MS-NLB-PhysServer-01_05:10:1c:9c</td>
<td>ECAT</td>
</tr>
<tr>
<td>21321</td>
<td>21.196077000</td>
<td>Beckhoff_10:1c:9c</td>
<td>Beckhoff_01:00:00</td>
<td>ECAT</td>
</tr>
<tr>
<td>21322</td>
<td>21.196086000</td>
<td>Beckhoff_01:00:00</td>
<td>MS-NLB-PhysServer-01_05:10:1c:9c</td>
<td>ECAT</td>
</tr>
<tr>
<td>21323</td>
<td>21.198075000</td>
<td>Beckhoff_10:1c:9c</td>
<td>Beckhoff_01:00:00</td>
<td>ECAT</td>
</tr>
<tr>
<td>21324</td>
<td>21.198083000</td>
<td>Beckhoff_01:00:00</td>
<td>MS-NLB-PhysServer-01_05:10:1c:9c</td>
<td>ECAT</td>
</tr>
<tr>
<td>21325</td>
<td>21.200077000</td>
<td>Beckhoff_10:1c:9c</td>
<td>Beckhoff_01:00:00</td>
<td>ECAT</td>
</tr>
<tr>
<td>21326</td>
<td>21.200085000</td>
<td>Beckhoff_01:00:00</td>
<td>MS-NLB-PhysServer-01_05:10:1c:9c</td>
<td>ECAT</td>
</tr>
<tr>
<td>21327</td>
<td>21.202077000</td>
<td>Beckhoff_10:1c:9c</td>
<td>Beckhoff_01:00:00</td>
<td>ECAT</td>
</tr>
</tbody>
</table>

Master Logical Address: Log Start: 0x10000000, Log Length: 0x00000000, Log StartBit: 0x00, Log EndBit: 0x00
Slave Physical Address: Phys Start: 0x1800, Phys StartBit: 0x00

Network Monitor
WireShark

9 μs
2.002ms
EtherCat Protocol
Device Description File

- Establish motion control parameters
  - When controller parameter is used, it musts be set up.
  - Using handshake mechanism makes sure whether

<table>
<thead>
<tr>
<th>Slave addr</th>
<th>Offset addr</th>
<th>DefaultSize</th>
<th>StartAddress</th>
<th>ControlByte</th>
<th>Enable</th>
<th>MinSize</th>
<th>MaxSize</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slave node</td>
<td>Physical address</td>
<td>128</td>
<td>#x1000</td>
<td>#x26</td>
<td>1</td>
<td>24</td>
<td>192</td>
<td>MBoxOut</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>#x1400</td>
<td>#x22</td>
<td>1</td>
<td>34</td>
<td>192</td>
<td>MBoxIn</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>2</td>
<td>#x1900</td>
<td>#x64</td>
<td>1</td>
<td></td>
<td></td>
<td>Outputs</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>6</td>
<td>#x1C00</td>
<td>#x20</td>
<td>1</td>
<td></td>
<td></td>
<td>Inputs</td>
</tr>
</tbody>
</table>

CANopen over EtherCAT
Slave addr : Slave node , offset addr : Physical address
EtherCat Protocol
EtherCAT Master User Interface Demo

• EtherCAT master interface program develops in Window XP and uses Visual Studio 2010 C++ to develop.

• The master UI includes the following feature:
  1) Reading ENI file acquires network topology and slave station information
  2) Sending the initializing packet of ENI file initializes the slave station in different state machine transformation.
  3) Acyclic Data transmission. Ex: Set up parameter, Write into fixed command...
  4) Write into the acyclic data for two different slaves at the same time
EtherCat Protocol
EtherCAT Master User Interface Demo

Read ENI file
State Machine
Logical Mapping

Status word
Servo on and off
Outline

• Advanced Robot Arm
• Real-time OS
• EtherCat Protocol
• Robotic Arm Dynamic Control Technology
• 3D Objects Pose Estimation
• Intelligent Graphic User Interface System
• Advanced Hand-Eye-Workspace Calibration
Robotic Arm Dynamic Control Technology
Introduction

• We develop high-precision, high-response-speed multi-axis control laws for 7-dof robotic arm. The control laws are robust, not affected by the change of the load arm. We regulate the control parameters with systematic rule.
Robotic Arm Dynamic Control Technology
Kinematics

- Forward kinematics:

We use D-H model to get the end-effector pose corresponding to angle of each axis. We transfer joint-space coordinate into Cartesian coordinate.

<table>
<thead>
<tr>
<th>Joint</th>
<th>$\theta$ (degree)</th>
<th>$\alpha$</th>
<th>$a$</th>
<th>$d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\theta_1$</td>
<td>-90</td>
<td>0</td>
<td>D1</td>
</tr>
<tr>
<td>2</td>
<td>$\theta_2$</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>$\theta_3$</td>
<td>-90</td>
<td>0</td>
<td>D3</td>
</tr>
<tr>
<td>4</td>
<td>$\theta_4$</td>
<td>90</td>
<td>0</td>
<td>D4</td>
</tr>
<tr>
<td>5</td>
<td>$\theta_5$</td>
<td>-90</td>
<td>0</td>
<td>D5</td>
</tr>
<tr>
<td>6</td>
<td>$\theta_6$</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>$\theta_7$</td>
<td>0</td>
<td>0</td>
<td>D7</td>
</tr>
</tbody>
</table>
Robotic Arm Dynamic Control Technology

Kinematics
Robotic Arm Dynamic Control Technology
Kinematics

• **Inverse kinematics:**
  
  We calculate angle of each axis by the position of end-effector. We transfer Cartesian coordinate into joint-space coordinate. The verification through 3D simulation are shown as follow:
Robotic Arm Dynamic Control Technology
Dynamic Equations

Robotic arm robot arm dynamic equations describe the relationship between position, velocity, acceleration and torque, which can be expressed as the following formula:

$$\tau = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + G(\theta)$$

where $\theta$ is the rotation angle of each axis, $M(\theta)$ is the mass inertia matrix, $C(\theta, \dot{\theta})$ is the centripetal force and the Coriolis force matrix, the above two matrices are $7 \times 7$ matrix, $G(\theta)$ is the $7 \times 1$ gravity vector, $\tau$ is the torque of each axis from the actuator.
Robotic Arm Dynamic Control Technology
Dynamic Equations

• Newton-Euler equation and the Euler-Lagrange equation are common methods to derive dynamic equations. We represent length, mass, inertia and centroid location of robotic arm as symbols through MATLAB symbolic toolbox, and then derive the dynamic equations by the Newton-Euler equation and Euler-Lagrange equation, respectively, to get $M(\theta)$, $C(\theta, \dot{\theta})$, $G(\theta)$. After verification, the $\tau$ derived by the Newton-Euler equation and by the Euler-Lagrange equation are matched.
Robotic Arm Dynamic Control Technology
Current Loop System Identification

- Accelnet Plus driver provides position, velocity and current mode. Due to that current is proportional to torque, the dynamic control is implemented using the current mode. In order to determine whether we need to compensate or not, we have to get the frequency response of the current loop of the servo motor.

The driver control loop block diagram
Robotic Arm Dynamic Control Technology
Current Loop System Identification

• The process of the system identification is as follows:
  1) Set the parameters of current loop controller (PI controller's Kp, Ki)
  2) Input different frequency of sinusoidal current command to motor by CME2, respectively, and record the magnitude and phase of input and output current to plot bode plot.
  3) Combine the input current command and output current in step b, and identify the system by using MATLAB identification toolbox.
  4) Input the test sinusoidal command to the model from step 3, and then compare the model output with the actual model output to verify the result of identification.
Robotic Arm Dynamic Control Technology
Experimental Result

• DC brushless motor: KBM-10H01-C00
• Digital servo drive: Accelnet Plus Module EtherCAT AEM-090-14
• Absolute encoder: DS-58[20] (18 bits Angular resolution)

• Obtained model: \( \frac{37760 s + 18490}{s^2 + 38470 s + 20080} \)
Robotic Arm Dynamic Control Technology
Experimental Result

- Comparing the bode plot of models and servo motor as follows:
Robotic Arm Dynamic Control Technology
Experimental Result

- Input the sinusoidal test command and compare the outputs of the model and the motor.

![Graph showing measured and simulated model output for 100Hz and 300Hz](image)

- **100Hz**

- **300Hz**
Robotic Arm Dynamic Control Technology
Future Work

• Solve the singularity problem of 7-axis Jacobian Matrix.
• Build the process of robot arm system identification to identify the parameters of dynamic model.
• Develop high efficiency motion control law of robot arm.
Outline

• Advanced Robot Arm
• Real-time OS
• EtherCat Protocol
• Robotic Arm Dynamic Control Technology
• 3D Objects Pose Estimation
• Intelligent Graphic User Interface System
• Advanced Hand-Eye-Workspace Calibration
3D Objects Pose Estimation
Outline

• Introduction
• Problem Statement
• CAD Database system
• Hand-Eye Calibration
• Experimental Result
• Conclusion and Feature Work
3D Objects Pose Estimation

Introduction

• With the advent of new-generation depth sensors, using 3D data is becoming increasingly popular. As these sensors are commodity hardware and sold at low cost, a rapidly growing group of people can acquire 3-D data cheaply and in real time.

• For a texture-less object like a mechanical part, conventional visual feature matching usually fails due to the absence of rich texture features.

• The vision system is easily set up to recognize different objects by using CAD models database.
3D Objects Pose Estimation

Problem Statement

• How to carefully chosen feature descriptor encode more information compactly and thereby provide higher accuracy 6-DOF pose estimation and enable faster computation.

• How to use CAD data to establish the CAD-model database, so that the system not only work even in highly cluttered environment, but also to accurately estimate 6-DOF pose of the workpieces.
System Architecture
3D Objects Pose Estimation
6-DOF Pose Estimation System

- The system can convert scene depth image to point cloud format, and estimate the 6-DOF pose of object by using CAD Database which has been established in offline phase.
3D Objects Pose Estimation
6-DOF Pose Estimation System

• The Voxel Grid Filter created 3D voxel grids from the input point cloud data, then points in each voxel (i.e. 3D box) will be describe as their centroid.

The centroid point of all the points in the voxel.
(1) Reference point Sr is paired with every other point Si, and their point pair feature F is calculated.

(2) Feature F is matched to the hash table, which returns a set of point pairs on the model that have similar distance and orientation.

(3) For each point pair on the model matched to the point pair in the scene, the local coordinate is calculated by solving $s_i = T_{z\rightarrow g} R_x(\alpha) T_{z\rightarrow g} m_i$.

(4) After is calculated, a vote is cast for the local coordinate $(m_i, \alpha)$.
3D Objects Pose Estimation
Pose Clustering

• The retrieved poses are clustered such that all poses in one cluster do not differ in translation and rotation for more than a predefined threshold.

\[
if \ ( trans_{pose\_n} < threshold_{trans\_n} ) \ and \ ( rotation_{pose\_n} < threshold_{rotation\_n} )
\]

\[=> \ pose\_n \ is \ put \ in \ cluster\_n \]

• The score of a cluster is the sum of the scores of the contained poses, after finding the cluster which vote is bigger than threshold, the resulting pose is calculated by averaging the poses contained in the cluster.

\[
if \ vote_{cluster\_n} > threshold
\]

\[=> \ cluster\_n \ is \ final \ pose \ estimation \]
3D Objects Pose Estimation
Eye-in-Hand Calibration

• We set up an eye-in-hand configuration using the Kuka 6-DOF robot arm installed with a Kinect camera.
• We first calibrated the eye-in-hand configuration by using the DLR Camera Calibration Toolbox.
3D Objects Pose Estimation
Calibration Result

- We took 15 images of the calibration pattern with varying orientations and translation. The error means are 2.54 mm, 7.99 mm and 2.74 mm in x-coordinate, y-coordinate and z-coordinate, respectively.
3D Objects Pose Estimation
Experimental Result
3D Objects Pose Estimation
Conclusion and Feature Work

• The 6-DOF object pose estimation based on CAD-model database has been proposed.

• We have implemented the 6-DOF pose estimation system in kuka robot. The experimental result shown that the proposed method can pick 3-4 bins in one minute.

• In the feature, we will speed up the cycle time of the method.
Outline

• Advanced Robot Arm
• Real-time OS
• EtherCat Protocol
• Robotic Arm Dynamic Control Technology
• 3D Objects Pose Estimation
• **Intelligent Graphic User Interface System**
• Advanced Hand-Eye-Workspace Calibration
Intelligent Graphic User Interface System

Goal

• The graphic user interface system is developed for operating an *industrial robotic arm*, with the implementation on the tablet of an *Android based application*.

• The graphic user interface system:
  • Providing a more ease-to-use virtual interface for the operator to execute the task.
  • Providing a platform to realize the remote control.
Intelligent Graphic User Interface System
Communication

• The WiFi technology makes it possible for us to control robot remotely, while tablet makes it possible for us to control the robot in the visual interface.

• The programming based on TCP Windows Sockets is used to realize the communication.
  – TCP Windows Sockets:
    • A reliable transport protocol
    • Ordered
    • Streaming
Intelligent Graphic User Interface System Communication

• It’s a simple way to realize the *Client / Server* model.
  
  • Server:
    - PC: Microsoft Visual Studio
    - Program: C language
  
  • Client:
    - Android tablet: eclipse
    - Program: Java language
Intelligent Graphic User Interface System Communication

- PC: The program uses multi-threads.
- Multi-thread is the ability of a program to manage multiple requests at one time in the computer.
Intelligent Graphic User Interface System Specifications

• Tablet
  – ASUS Memo Pad FHD ME302KL-1B024A
  – Qualcomm 8064 Quad-core 1.5GHz
  – OS: Android4.2
  – Size: 264 x 182 x 9.5mm
  – LCD size: 10.1-inch IPS
  – WiFi、Bluetooth
Intelligent Graphic User Interface System
Operation Interface

• Tablet
  – Joint Mode
Intelligent Graphic User Interface System

Operation Interface

- Tablet
  - xyz Mode
Intelligent Graphic User Interface System
Operation Interface

• Tablet
  – Teach Mode

Menu of Points
Run buttons
Intelligent Graphic User Interface System
Operation Interface

- Tablet
  - Knob Mode

Joint buttons

Knob

Seekbar to adjust the scale of the knob
Intelligent Graphic User Interface System Demo
Outline

• Advanced Robot Arm
• Real-time OS
• EtherCat Protocol
• Robotic Arm Dynamic Control Technology
• 3D Objects Pose Estimation
• Intelligent Graphic User Interface System
• Advanced Hand-Eye-Workspace Calibration
Advanced Hand-Eye-Workspace Calibration

Introduction

• The proposed Hand-Eye-Workspace calibration system including a laser pointer installed on the robot arm, a stationary camera, and a planar surface. The laser pointer beams to the surface, and the camera observes the projected laser spot. It is not necessary for robot arm to enter the view scope of the camera for calibration. The configuration of camera is more flexible.
Advanced Hand-Eye-Workspace Calibration Definition

- $^C_H_B$: The transformation matrix between the fixed camera and the robot base. It represents the camera extrinsic matrix.

- $^B_H_E$: The transformation matrix between the robot base and the end-effector. It represents the robot kinematics.

- The relationship between the laser pointer and the end-effector.

- Plane equation

- Camera intrinsic parameters
Advanced Hand-Eye-Workspace Calibration

Definition

The laser line is defined in the end-effector coordinate with two position parameters \((y, z)\) and two Euler angles \((\theta, \phi)\). The values of four parameters are unique. The direction of line is computed with \((\theta, \phi)\) as

\[
\begin{bmatrix}
\cos^E_L \theta \\
\sin^E_L \phi \\
\sin^E_L \theta \sin^E_L \phi \\
\cos^E_L \phi
\end{bmatrix}^T
\]
Advanced Hand-Eye-Workspace Calibration
Line-Plane Intersection

• Laser line

The laser line in the camera coordinate can be obtained using $c_H B$, $B_H E$ and $\left(\begin{array}{c}E_L y, E_L z, E_L \theta, E_L \phi\end{array}\right)$. The direction is denoted as $c_z_{laser}$. The position is denoted as $c_p_{laser}$.

• Plane definition

A plane is generally described by the normal vector $n$ and the position vector $p$ of any point on the plane, but it can be simply defined by the unique position vector whose direction is the normal vector as

$$a = \begin{cases} n, & \text{if } n \cdot p = 0 \\ (n \cdot p) n & \text{otherwise} \end{cases}$$
Line-Plane intersection

The laser spot on the plane can be defined with the distance $d_{laser}$ as $^c p_{spot} = d_{laser}^c z_{laser} + ^c p_{laser}$. It matches the geometrical constraint as

$$(d_{laser}^c z_{laser} + ^c p_{laser} - a) \cdot a = 0.$$  

The distance $d_{laser}$ can be obtained as

$$d_{laser} = \frac{\|a\| - ^c p_{laser} \cdot a}{^c z_{laser} \cdot a}.$$
Advanced Hand-Eye-Workspace Calibration
Camera Model

- The camera model is a pin-hole type including lens distortion. The 2D position \((u, v)\) in an image of a 3D point \(Cp = [Cx, Cy, Cz]^T\) can be obtained with the intrinsic matrix \(K\) and the lens distortion parameters \((\kappa_1, \kappa_2, \rho_1, \rho_2)\) as

\[
\begin{bmatrix}
u \\
v \\
1
\end{bmatrix} = K \cdot \begin{bmatrix} x_d \\
y_d \\
1
\end{bmatrix} \quad \text{with} \quad K = \begin{bmatrix} f_u & \alpha_c \cdot f_u & u_0 \\
0 & f_v & v_0 \\
0 & 0 & 1
\end{bmatrix}
\]
Advanced Hand-Eye-Workspace Calibration
Camera Model

- The camera model is a pin-hole type including lens distortion. The 2D position \((u, v)\) in an image of a 3D point \(\mathbf{c}_p = [c_x, c_y, c_z]^T\) can be obtained with the intrinsic matrix \(\mathbf{K}\) and the lens distortion parameters \((\kappa_1, \kappa_2, \rho_1, \rho_2)\) as

\[
\begin{bmatrix}
x_d \\
y_d
\end{bmatrix} = \left(1 + \kappa_1 \|\mathbf{x}_r\|^2 + \kappa_2 \|\mathbf{x}_r\|^4\right) \mathbf{x}_r + \begin{bmatrix}
2\rho_1 x_r y_r + \rho_2 (\|\mathbf{x}_r\|^2 + 2x_r^2) \\
\rho_1 (\|\mathbf{x}_r\|^2 + 2y_r^2) + 2\rho_2 x_r y_r
\end{bmatrix}
\]

where \(\mathbf{x}_r = [x_r \quad y_r]^T = \begin{bmatrix}
c x/c z \\
c y/c z
\end{bmatrix}^T\) is the ray direction from the camera.
Advanced Hand-Eye-Workspace Calibration
All Parameters

- $^{C}H_{B}$: The transformation matrix between the fixed camera and the robot base. It is defined with the Euler angles $(^{B}_{C}\theta, ^{B}_{C}\phi, ^{B}_{C}\phi)$ and the position $^{B}_{C}t$.
- $^{B}H_{E}$: The transformation matrix between the robot base and the end-effector. The robot kinematic parameters.
- The relationship between the laser pointer and the end-effector is defined with Euler angles $(^{E}_{L}\theta, ^{E}_{L}\phi)$ and the position $^{E}_{L}t$.
- Plane equation: $a$
- Camera intrinsic matrix $K$ and the lens distortion parameters $(\kappa_1, \kappa_2, \rho_1, \rho_2)$
Advanced Hand-Eye-Workspace Calibration Experiment

- For Hand-Eye-Workspace calibration, the goal is to estimate the relationship between the robot arm, the camera and the workspace plane. Assuming the robot arm and the camera are pre-calibrated. The estimated parameters include:
  - \( (^B_C \theta, ^B_C \phi, ^B_C \phi) \) and \(^B_C t\) : the transformation matrix between the fixed camera and the robot base.
  - \( (^L_E \theta, ^L_E \phi) \) and \(^E_L t\) : the relationship between the laser pointer and the end-effector.
  - \( a \) : the plane parameters.
Advanced Hand-Eye-Workspace Calibration Experiment

Experiment Configuration
Advanced Hand-Eye-Workspace Calibration Experiment

• The robot arm, Stäubli TX60, is operated to generate multiple laser spots on the plane in the view of camera. The measurement is the \((u, v)\) position of laser spot. The optimal solution is obtained by minimizing the total position errors using nonlinear optimization method (the Levenberg-Marquardt method).

• The laser spot position in the image is obtained by two steps. First, the grey image converts to binary image using the simple threshold method. It can reject the image noise. Second, the average position of light pixels is the laser spot position.
## Advanced Hand-Eye-Workspace Calibration Experiment

### Calibration Result

<table>
<thead>
<tr>
<th>Parameter</th>
<th>30 samples</th>
<th>50 samples</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$({}<em>{C}^{B}\theta, {}</em>{C}^{B}\phi, {}_{C}^{B}\varphi)$</td>
<td>(-100.14, -3.29, -172.69)</td>
<td>(-102.55, -6.15, -174.22)</td>
<td>(-102.60, -5.88, -173.53)</td>
</tr>
<tr>
<td>$B_{c}t^T$</td>
<td>[1343.73, -4.34, 319.28]</td>
<td>[965.56, -47.41, 440.90]</td>
<td>[976.59, -38.35, 436.62]</td>
</tr>
<tr>
<td>$({}<em>{L}^{E}\theta, {}</em>{L}^{E}\phi)$</td>
<td>(-89.93, 80.21)</td>
<td>(-91.15, 84.45)</td>
<td>(-91.13, 84.36)</td>
</tr>
<tr>
<td>$E_{l}t^T$</td>
<td>[0, -30.14, 423.31]</td>
<td>[0, 8.55, 135.06]</td>
<td>[0, 7.49, 140.77]</td>
</tr>
<tr>
<td>$a^T$</td>
<td>[-70.39, 55.95, 690.76]</td>
<td>[-50.57, 35.52, 705.07]</td>
<td>[-61.34, 33.18, 702.78]</td>
</tr>
<tr>
<td>RMS</td>
<td>30.32</td>
<td>1.0626</td>
<td>1.1494</td>
</tr>
</tbody>
</table>
Advanced Hand-Eye-Workspace Calibration

Conclusion

• The proposed calibration system is cost-efficient and flexible for any manipulator. It can be extended to calibrate the camera intrinsic parameters and the kinematic parameters of robot arm.