Build Advanced Industrial Robot Usages with Intel OpenVINO and Movelt

Yu Yan
Intel SSP Robotics Software Engineering
Agenda

- Who We Are
- Grasp Detection
- OpenVINO™ Toolkit Grasp Detection
- OpenVINO™ Toolkit Grasp Library as MoveIt Plug-in
- MoveIt Hand-Eye Calibration
- MoveIt Example Apps
- Build MoveIt into Advanced Industrial Robot Controllers with ACRN
Who We Are

- Who we are:
  - SSP Robotics Software Engineering Team from Intel Open Source Technology Center (OTC)

- Intel ROS2 projects:
  - ROS2 Realsense Camera: https://github.com/intel/ros2_intel_realsense
  - ROS2 OpenVINO: https://github.com/intel/ros2_openvino_toolkit
  - ROS2 Movidius NCS: https://github.com/intel/ros2_intel_movidius_ncs
  - ROS2 Object Analytics: https://github.com/intel/ros2_object_analytics
  - ROS2 Object Map: https://github.com/intel/ros2_object_map
  - ROS2 Grasp Library: https://github.com/intel/ros2_grasp_library
  - ROS2 Navigation: https://github.com/ros-planning/navigation2
Grasp Detection


- Convolutional Neural Networks (CNN)-based grasp detection
  - Dex-Net*

*Other names and brands may be claimed as the property of others.

OpenVINO™ Toolkit Grasp Detection

- 3~4X(2018), 6~8X(2019) performance gain on inference time
- 25% CPU offload
- [https://github.com/atenpas/gpd](https://github.com/atenpas/gpd)

Other names and brands may be claimed as the property of others.
OpenVINO™ Toolkit Grasp Library as MoveIt Plug-in

Visualization of grasp detection results

https://github.com/intel/ros2_grasp_library
Added to website “moveit.ros.org”

*Other names and brands may be claimed as the property of others.
How to use ros2_grasp_library

- Launch grasp planner

ros2 run grasp_ros2 grasp_ros2 __params:=src/ros2_grasp_library/grasp_apps/random_pick/cfg/random_pick.yaml

```cpp
using GraspPlanning = moveit_msgs::srv::GraspPlanning;
static std::shared_ptr<GraspPlanning::Response> result_ = nullptr;
......
// create client for grasp planning
auto client = node_->create_client<GraspPlanning>("plan_grasps");

// request grasp poses
auto request = std::make_shared<GraspPlanning::Request>();
auto result_future = client->async_send_request(request);
......
if (moveit_msgs::msg::MoveItErrorCodes::SUCCESS == result_future.get()->error_code.val) {
    result_ = result_future.get();
    RCLCPP_INFO(node_->get_logger(), "Response received %d", result_->error_code.val);
} else continue;
geometry_msgs::msg::PoseStamped p = result_->grasps[0].grasp_pose;
......
```

GraspDetectorGPD:
ros__parameters:
- cloud_topic: /camera/pointcloud
device: 1 # 0:CPU, 1:GPU, 2:VPU
workspace: [-0.35, 0.35, -0.6, 0.1, 0.0, 1.0]
finger_width: 0.005
hand_outer_diameter: 0.100
hand_depth: 0.038
hand_height: 0.020
......

GraspPlanner:
ros__parameters:
grasp_frame_id: "base"
grasp_approach: [0.0, 0.0, -1.0]
grasp_approach_angle: 0.523
grasp_offset: [-0.000, -0.002, 0.000]
eef_offset: 0.174
eef_yaw_offset: -0.7854 # M_PI/4
......

Create Grasp Service Client
Request Grasp Pose
Get Grasp Pose Result
Movelt Hand-eye Calibration

- Consists of four Movelt plugins:
  - Rviz GUI Plugin
  - Marker Detection Plugin
  - Calibration Algorithm Plugin
  - Covariance Analysis Plugin (WIP)
- Flexible architecture, each plugin can be replaceable
- Easy to use interface
- Calibration process understandable and visualized in 2/3D
- State-of-the-art algorithms integrated
- Can be used to:
  - Eye-to-hand calibration
  - Eye-in-hand calibration
- PRs under review in Movelt:
  - Movelt#1558
  - Movelt#1559
  - Movelt#1560
MoveIt Hand-eye Calibration

- Rviz GUI Plugin consists of:
  - Target tab widget (Used for setting and visualizing calibration board detection)
  - Context tab widget (Used for setting calibration context and initial guess)
  - Calibrate tab widget (Used for calibration computing)
MoveIt Hand-eye Calibration

- Future improvements:
  - Thank @felixvd for reviewing the tool and providing good suggestions
  - Split the calibration tab widget into four tab widgets: (Motion, Collect, Calculate and Test)
  - Add Covariance Analysis Plugin (WIP)
MoveIt Example Apps

Screen snapshot of “MoveIt Example Apps Tutorial”

Video: Intelligent Visual Grasp
(OpenVINO™ Grasp Library + MoveIt)

https://www.youtube.com/watch?v=b4EPvHdiOA&list=PLxCmGJeII.g0xq3uqcCVSYnSJiQK1L9vP

https://github.com/ros-planning/moveit_example_apps

- moveit_example_apps #PR2
- moveit_example_apps #PR3
- moveit_example_apps #PR5
Build MoveIt into Advanced Industrial Robot Controllers with ACRN

- Usual way to implement MoveIt on industrial robots:
  - Machine 1 (Hard real-time OS, motor level control)
  - Machine 2 (Non real-time OS, MoveIt motion planning)
  - Machine 3 (Non real-time OS, Perception)

- It’s useful to support real-time and non real-time development safely and effectively at a single machine

- Using multiple machines is not good at:
  - System safety and reliability
  - Space possession
  - Adaptation challenges
  - Power consumption

*Other names and brands may be claimed as the property of others.*
Build MoveIt into Advanced Industrial Robot Controllers with ACRN

Real-Time / Safety Critical tasks

- Real-Time Tasks (robot control, etc)
- DDS
- Comms
- UOS (RT Linux*, Xenomai, VxWorks, QNX, Zephyr)

Non Real-Time Tasks

- DDS
- Comms
- ROS2 Core
- ROS2 Stacks

Hypervisor ACRN

VirtIO

Dev Passthrough

IA Hardware
Legal Notices and Disclaimers

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request. No product or component can be absolutely secure.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting www.intel.com/design/literature.htm.

Intel, the Intel logo, Intel RealSense, OpenVINO, and the OpenVINO logo are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others.

© Intel Corporation