УДК 004.43

Кокорин Александ Евгеньевич
Холухина Диана Евгеньевна
Шишкина Мария Николаевна
Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»
Россия, Санкт-Петербург

Применение языков программирования в робототехнике

Аннотация. Данная статья посвящается анализу применимости языков программирования в робототехнике. Рассмотрены такие языки, как C++, Python, MATLAB и ROS с последующими сравнительными характеристиками их гибкости, производительности, и, непосредственно применимости - результаты исследования показывают, что выбор языка зависит от специфики задачи.

Ключевые слова: языки программирования, робототехника, алгоритмы управления, ROS, C++, Python, MATLAB.

Введение

Создание робота — это всегда компромисс между "железом"и софтом. Мы можем собрать идеальную механическую конструкцию, но без грамотного ПО она останется просто дорогой грудой металла. Есть разные языки программирования и системы обеспечения роботов. C++ для прямого управления аппаратурой. Python в контексте машинного обучения и экспериментальных разработок. Функционал MATLAB/Simulink при проектировании виртуальных моделей. Совмещение языков в рамках экосистемы ROS.

Почему софт решает все?

1. Управление – даже простейшему роботу-манипулятору нужны алгоритмы, предсказывающие траекторию движения с точностью до миллиметра

2. Обратная связь – без корректной обработки данных с датчиков робот просто не понимает, что происходит вокруг

3. Интеграция – современные системы объединяют десятки компонентов, которые должны работать как единый организм C++ для прямого управления аппаратурой. Python в контексте машинного обучения и экспериментальных разработок. Функционал MATLAB/Simulink при проектировании виртуальных моделей. Совмещение языков в рамках экосистемы ROS.

Основные положения

1. C++: скорость и контроль «железа» Несмотря на рост альтернатив, C++ сохраняет статус базового инструмента для систем, где критичны скорость отклика и доступ к «голому» железу. Кейсы вроде промышленных манипуляторов Fanuc или KUKA демонстрируют: программы на C++ обеспечивают точное управление сервоприводами и координацию движений [1].

Сильные стороны:

1) Минимизация задержек в режиме реального времени.

2) Параллельная обработка данных через многопоточность.

3) Совместимость с низкоуровневыми библиотеками (Eigen, OpenCV).

4) Обратная сторона — высокая пороговая сложность для новичков и трудоемкая отладка, что замедляет исследовательские итерации.

2. Python: гибкость вместо скорости Минималистичный синтаксис Python сделал его фаворитом в задачах, где приоритет — быстрая проверка гипотез. Например, в TurtleBot3 скрипты на Python обрабатывают лидарные данные и строят маршруты [2].

Преимущества:

1) Мгновенное развертывание прототипов.

2) Готовые решения для ИИ (TensorFlow, PyTorch).

3) Интерфейс rclpy для интеграции с ROS2.

4) Цена простоты — просадки производительности при работе с потоками в реальном времени, что часто требует переписывания ядра на C++.

```python
import pypcd
# also can read from file handles.
pc = pypcd.PointCloud.from_path('foo.pcd')
# pc.pc_data has the data as a structured array
# pc.fields, pc.count, etc have the metadata

# center the x field
pc.pc_data['x'] -= pc.pc_data['x'].mean()

# save as binary compressed
pc.save_pcd('bar.pcd', compression='binary_compressed')
```

Рис. 1: Пример чтения данных с лидара .pcd с помощью pypcd на языке Python

3. MATLAB/Simulink: виртуальные полигоны MATLAB доминирует в нише цифровых двойников — от симуляции динамики манипулятора UR5 [3] до тонкой настройки ПИД-регуляторов.

Козыри платформы:

1) Шаблонные блоки для проектирования систем управления.

2) Конвертация моделей в код для микроконтроллеров.

3) Инструменты визуализации процессов.

4) Главный барьер — привязка к дорогостоящим лицензиям и закрытым алгоритмам.

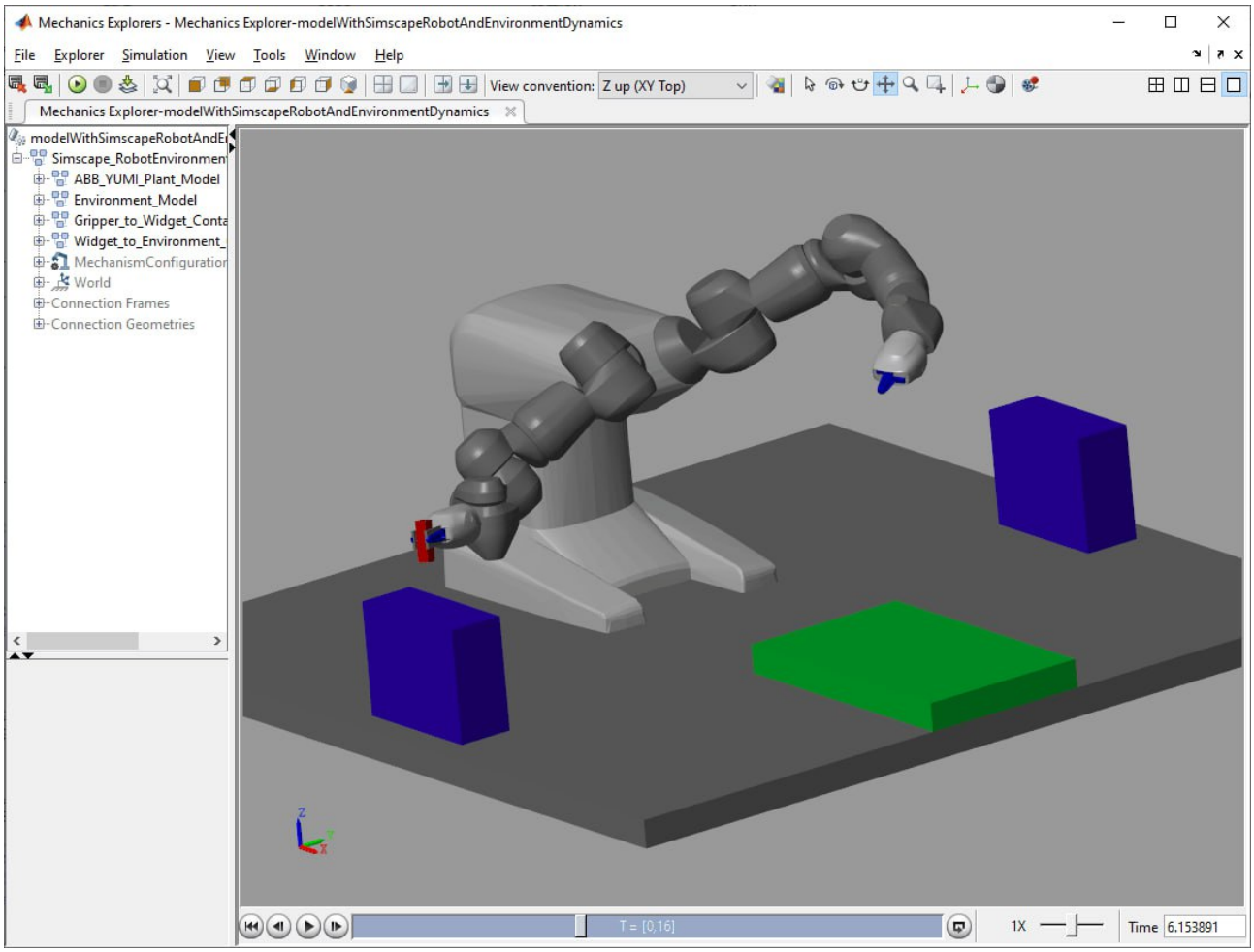

Рис. 2: Визуализация роботизированной руки в matlab

4. ROS как универсальный связующий каркас Экосистема ROS (Robot Operating System) стала мостом между разнородными компонентами робототехнических систем. Её архитектура позволяет, например, объединить C++-модуль управления мотором с Python-скриптом для распознавания объектов через:

Ноды — изолированные процессы (сбор данных с датчика, расчет траектории).

Топики — асинхронные каналы обмена сообщениями.[4]

Сервисы — синхронные запросы (калибровка, экстренная остановка).

Пакеты — контейнеры с кодом и зависимостями.

Примеры интеграции:

1) В системе координации дронов C++ отвечает за стабилизацию, а Python — за построение маршрутов.

2) Российская платформа «Сервосила» использует ROS для синхронизации промышленных роботов [5].

Сильные стороны ROS:

1) Масштабируемость под распределенные системы.

2) Нативные инструменты симуляции (Gazebo) и визуализации (RViz).

3) Совместимость с OpenCV, PCL и другими opensource-библиотеками.

Слабые места:

1) Сложность первичной настройки среды.

2) Ресурсоемкость при работе с большими массивами данных.

Сравнительные характеристики

В таблице 1 отражена эффективность языков по ключевым критериям:

Параметр	**C++**	**Python**	**MATLAB**
Производительность	Высокая	Низкая	Средняя
Простота разработки	Низкая	Высокая	Средняя
Поддержка ROS	Да	Да	Нет
Стоимость	Бесплатно	Бесплатно	Платная

Таблица 1: Сравнение языков программирования

На рис. 1 изображена гибридная архитектура ROS-системы, где C++ и Python разделены по уровням управления.

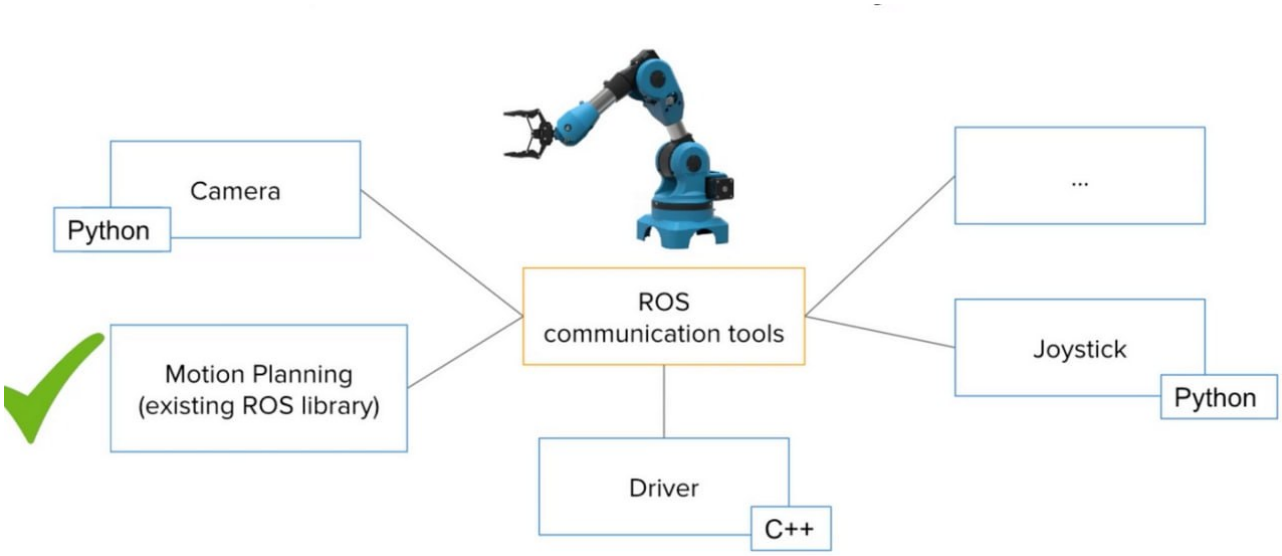

Рис. 3: Гибридная архитектура ROS-системы

Тренды и прогнозы

- DSL: Разработка узкоспециализированных языков (Urbi) для конкретных классов роботов.

- ROS 2: Внедрение механизмов реального времени и enhanced-безопасности [6].

- Симбиоз ИИ и ROS: Встраивание Python-моделей машинного обучения прямо в ROS-ноды.

Итоги

Выбор языка в робототехнике диктуется этапом проекта: C++ — для «железного» ядра, Python — для алгоритмических экспериментов, MATLAB — для виртуальных тестов. ROS устраняет барьеры между языками, выступая универсальным интегратором. Эволюция ROS 2 и DSL усилит тренд на создание адаптивных автономных систем, сочетающих скорость, гибкость и точность.

Список литературы

[1] Кокоурова М. Как написать программу для робота? [Электронный ресурс] // Блог SkillFactory. — 2024. — URL: https://blog.skillfactory.ru/kak-napisat-programmu-dlya-robota/ (дата обращения: 8.02.2025).

[2] Габов А.Е. Инструменты для обработки данных лидарной съёмки. [Электронный ресурс] — 2024. — URL: https://cyberleninka.ru/article/n/instrumenty-dlya-obrabotki-dannyh-lidarnoy-syomki/viewer (дата обращения: 9.02.2025).

[3] Кубриков М.В. Цифровой двойник в системе внешнего адаптивного управления роботами-манипуляторами. [Электронный ресурс] — 2023. — URL: https://cyberleninka.ru/article/n/tsifrovoy-dvoynik-v-sisteme-vneshnego-adaptivnogo-upravleniya-robotami-manipulyatorami/viewer (дата обращения: 10.02.2025).

[4] AlexeyMerzlyakov ROS: стань контрибьютором самого большого Open Source проекта в робототехнике [Электронный ресурс] — 2021. — URL: https://habr.com/ru/companies/samsung/articles/571302/ (дата обращения: 11.02.2025).

[5] Лаврентев Р.О. Маврин И.А. Сафин Р.Н. Магид Е.А. Робот "Сервосила Инженер": разработка сервера передачи видеопотока и интерфейса управления под фреймворк ros [Электронный ресурс] — 2018. — URL: https://cyberleninka.ru/article/n/robot-servosila-inzhener-razrabotka-servera-peredachi-videopotoka-i-interfeysa-upravleniya-pod-freymvork-ros/viewer (дата обращения: 12.02.2025).

[6] Yanlei Ye, Zhenguo Nie, Xinjun Liu, Fugui Xie, Zihao Li Peng Li ROS2 Real-time Performance Optimization and Evaluation [Электронный ресурс] — 2023. — URL: https://cjme.springeropen.com/articles/10.1186/s10033-023-00976-5 (дата обращения: 13.02.2025).