

---

# ZytleBot

自律移動ロボットへのFPGA活用のすゝめと  
ROS 2移行への道

田村 爽 新田 泰大 高瀬 英希  
(京都大学)

# Agenda

---

- ROSにおけるFPGAの活用
  - FPGA Design competition
  - LinuxによるFPGAの扱い方
  - 活用事例紹介
- ROS 2への移行
  - ROS 2対応のモチベーション
  - ROS 2ツール使ってみた



---

# ROSにおけるFPGAの活用

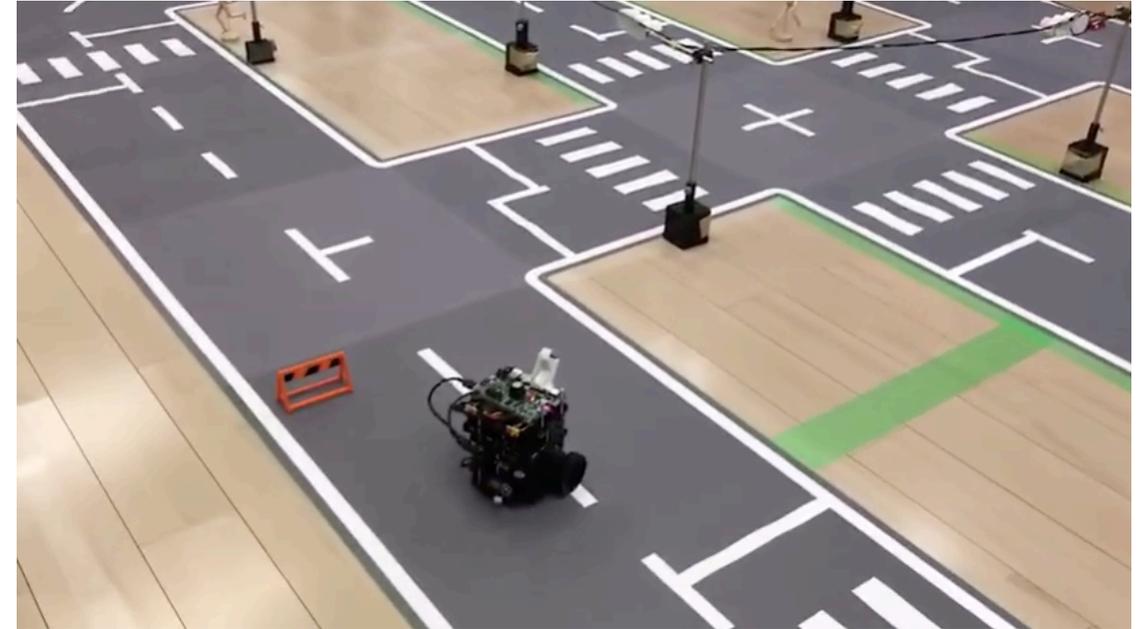
- FPGA Design competition
- LinuxによるFPGAの扱い方
- 活用事例紹介

# FPGA Design competition

- FPT2018およびHEART2019において併催された  
**自律移動ロボット開発コンテスト**

- **概要**

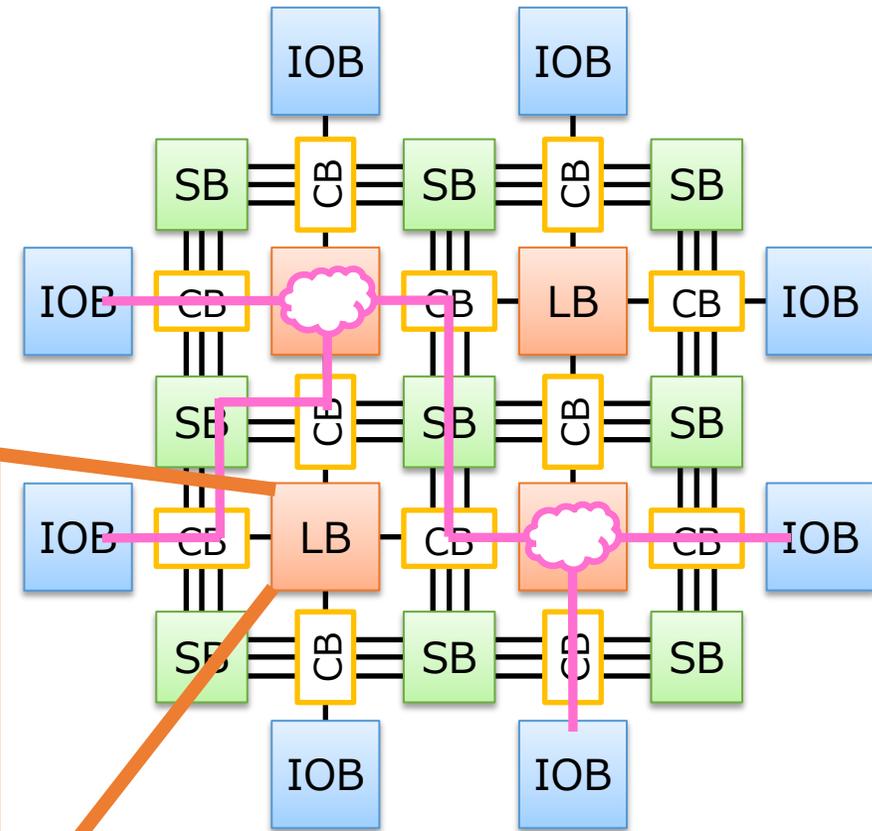
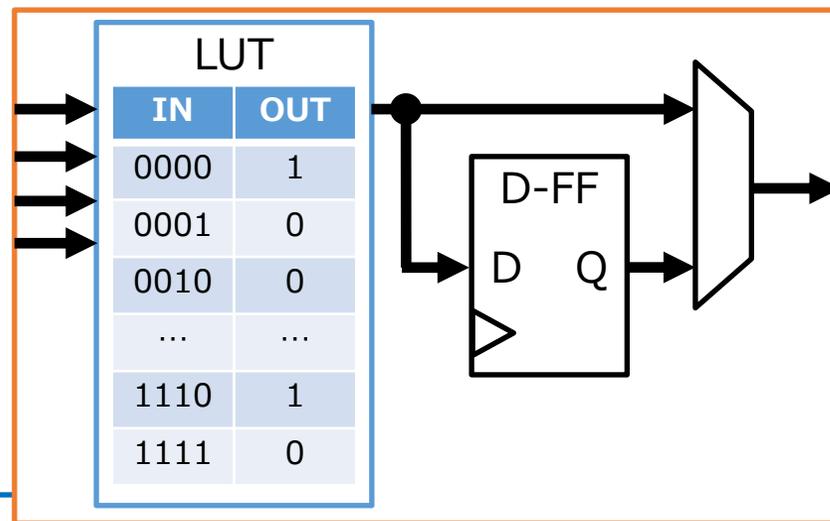
- 道路を模したコースを走行
- センサ類はCMOS**カメラのみ**
- プロセッサは組み込みCPUのみ、さらに外部との通信は禁止されているため、**FPGAをうまく活用することが求められる**
- 車体に関しては**自由!**
  - ✓ **ロボットは自作でも既製品でもOK!**



# FPGAとは？

- Field Programmable Gate Array

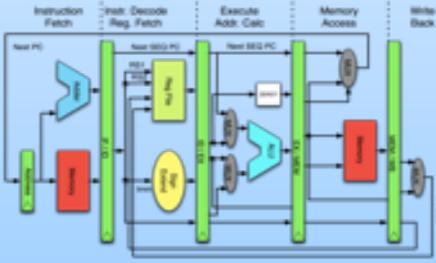
- 中身を改変可能なLSI
- ハードウェアそのものの振る舞いを変えられる
- 独自のデジタル回路を自由に何回でも形成できる



LB 論理ブロック      CB コネクションブロック  
SB スイッチブロック      IOB 入出力ブロック

# FPGAの強み

プロセッサ  
&  
ソフトウェア



性能・並列性



設計容易性・柔軟性

省電力性



開発製造コスト

FPGA

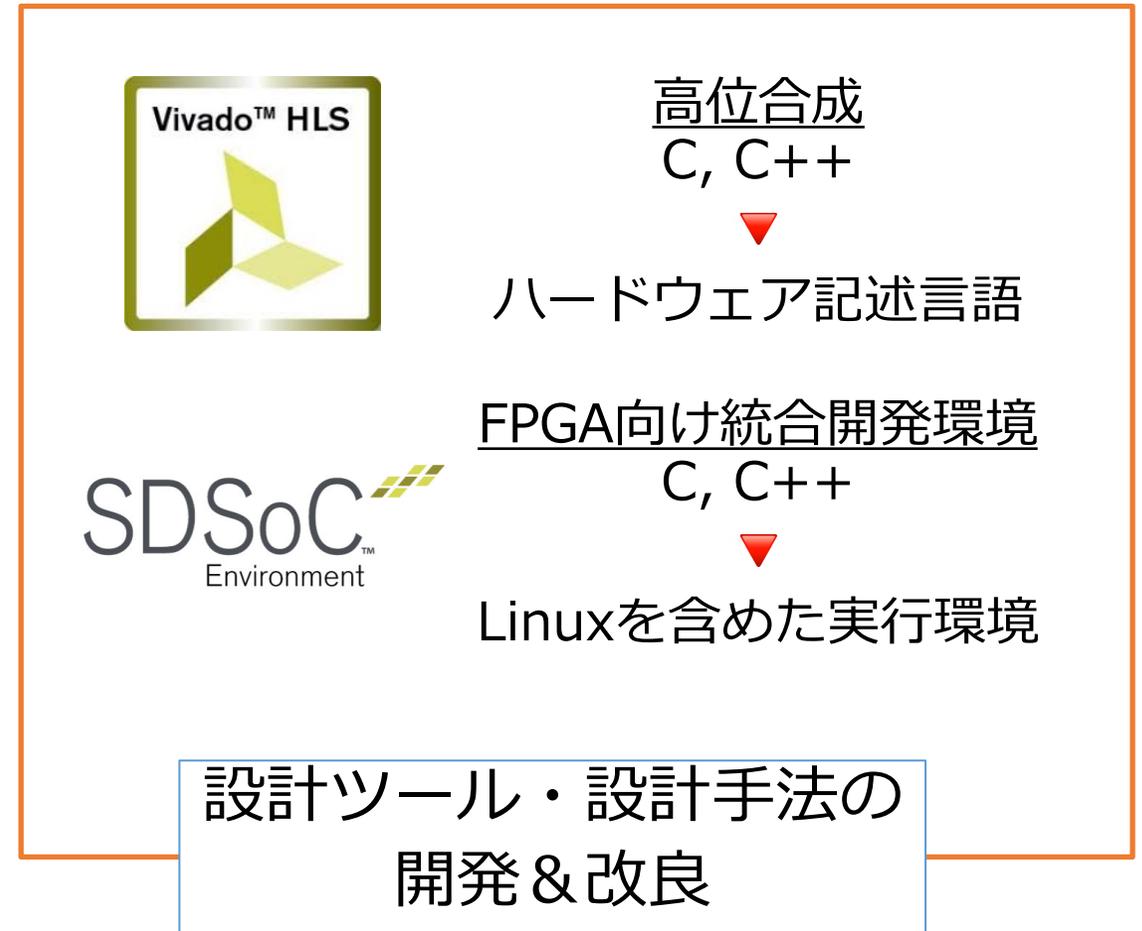


ASIC  
(専用回路)  
ハードウェア



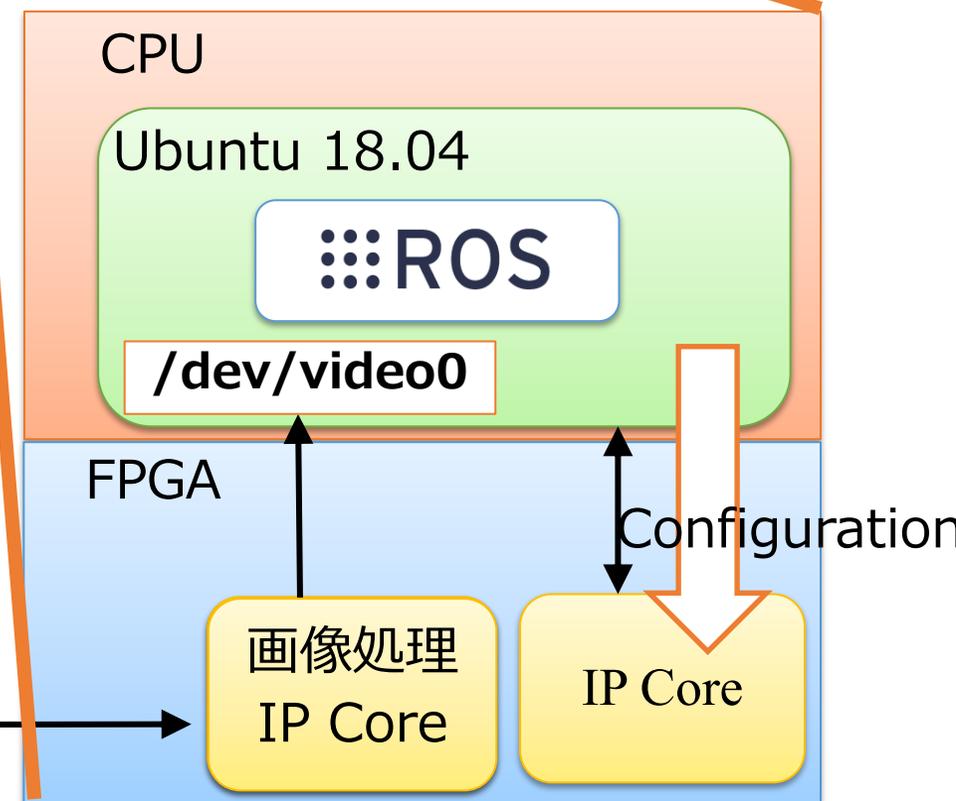
# FPGA vs GPU

- Pros
  - 消費電力
  - リアルタイム性
- Cons
  - 開発コスト ←
- 備考
  - 性能は設計とやりたいこと次第
  - 値段も安いとは言い切れない



# プログラマブルSoC

- FPGAのみでは柔軟性に乏しい
- CPUとFPGAを1チップに収めた  
**プログラマブルSoC**
  - Xilinx : Zynq
  - Intel : SoC FPGA
- CPUにはRTOSやLinuxを搭載可能
  - もちろんROSも
- デバイスドライバを介して  
**FPGAをデバイス**として扱うことができる
- FPGAの回路の再書き換えをLinuxから行うことも可能に！  
(Linux kernel 4.10以降)



# LinuxからFPGAを操作

- FPGAの設計ツールで、CPUとFPGA回路間を配線
  - ツールによりLinuxの物理アドレスへの割り当てが行われる

- 物理アドレスを操作

- a. メモリマップドIO

- /dev/memで制御（危険！）

- b. UIO(User space IO)

- ユーザプロセスから指定範囲のメモリを操作する機能

- c. デバイスドライバの作成

- 安全・手間がかかる
- 再利用性◎

```
int uio1_fd = open("/dev/uio1", O_RDWR);
hls_regs = mmap(NULL, 0x10000, PROT_READ|PROT_WRITE, MAP_SHARED, uio1_fd, 0);

int uio2_fd = open("/dev/uio2", O_RDWR);
dma_regs = mmap(NULL, 0x1000, PROT_READ|PROT_WRITE, MAP_SHARED, uio2_fd, 0);
```

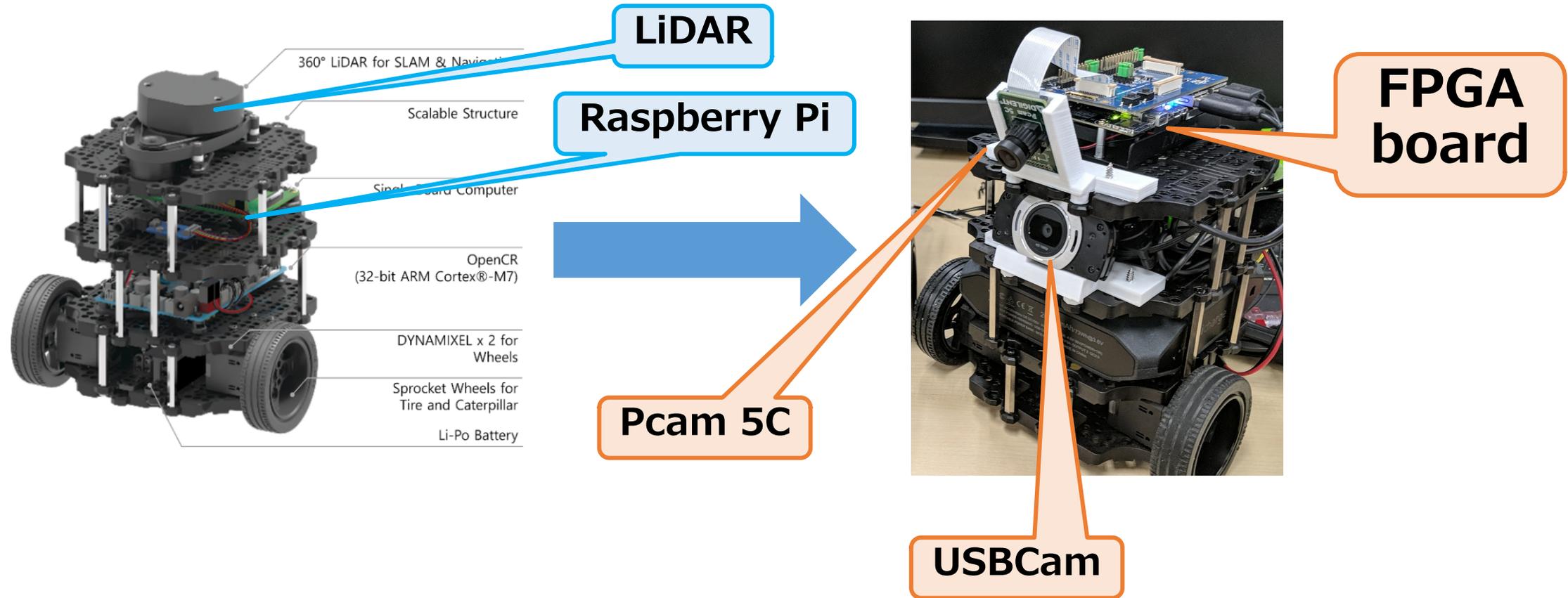
```
v4l2_fd = open("/dev/pcamvideo0", O_RDWR);
```

# 活用事例 : ZytleBot

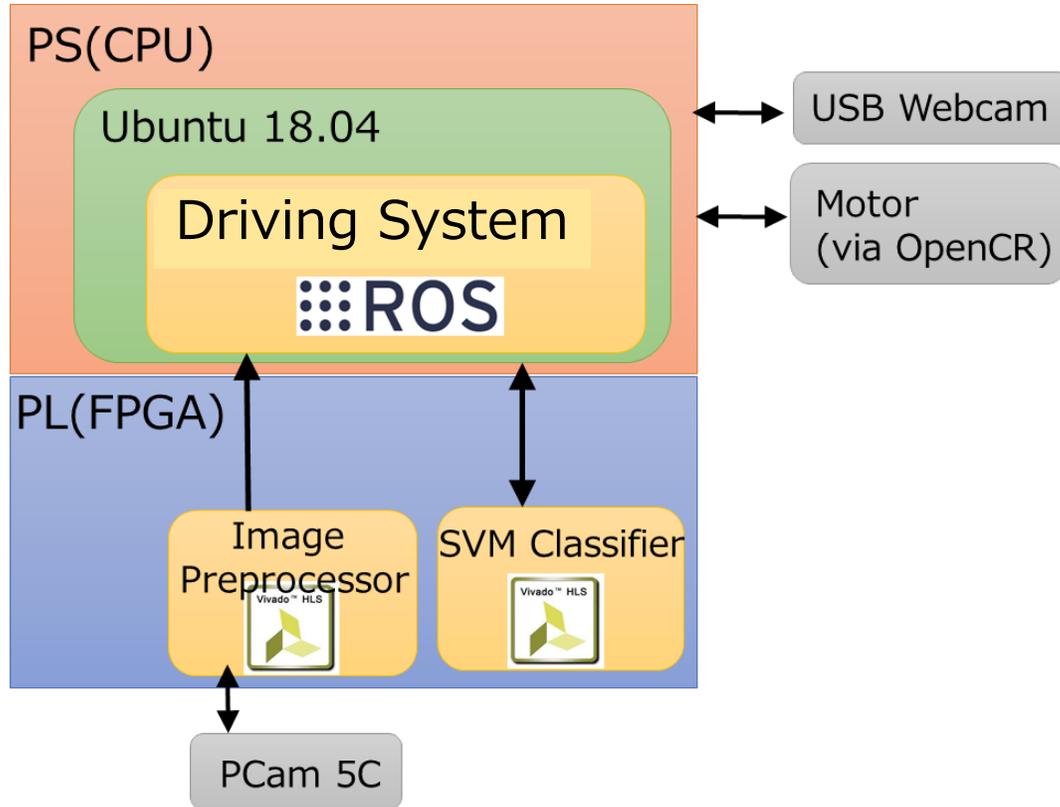
- **Zynq** + **TurtleBot**
  - Zynq UltraScale+ MPSoC
  - TurtleBot3 : **ROS** standard platform robot kit



# TurtleBot3 & ZytelBot



# システム構成&開発メンバー



## • 学生2人で開発

– SW担当：田村（本日発表）

- ✓ **Gazebo**シミュレータ用の環境作成
- ✓ 自律移動アルゴリズム全般
- ✓ 3Dモデリングツールによる  
パーツ作成・印刷

– HW担当：新田

- ✓ FPGAですべきアルゴリズムの  
検討とその実装
- ✓ FPGAの回路設計

# FPGAによる赤信号検出

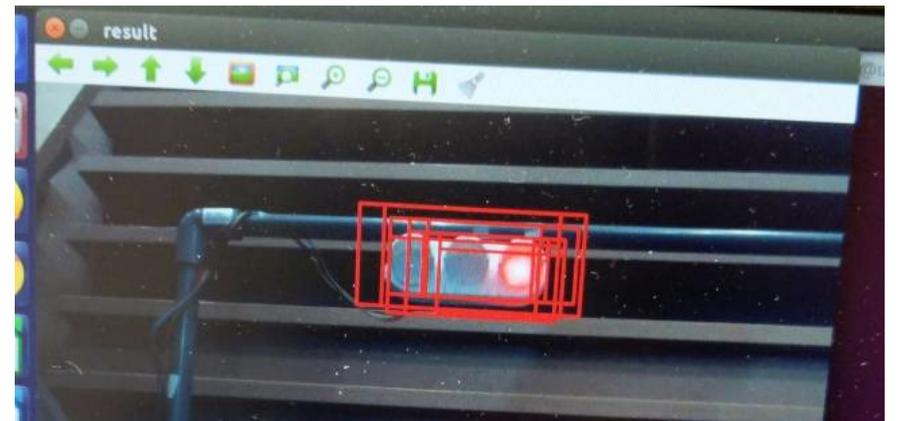
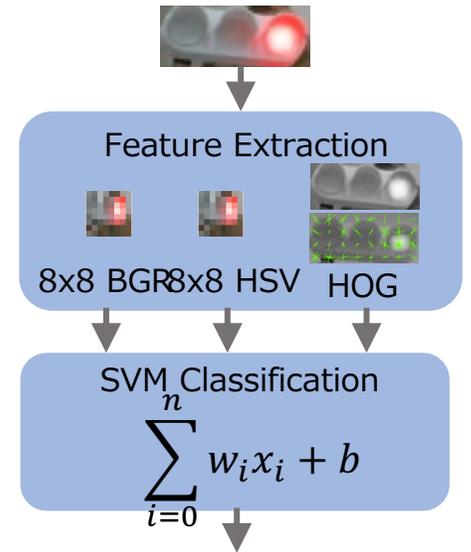
- スライディングウィンドウ法 + SVM
- 入力: 320pix\*240pixフレーム画像
- 出力: 891個のウィンドウの推定結果
  - レイテンシ: 2milisec
  - HW使用 : 6.22milisec = 160fps
  - SWのみ : 1700milisec = 0.58fps

✓ **275倍高速化**

<https://www.slideshare.net/takasehideki/zytlebot-rosfpga/takasehideki/zytlebot-rosfpga>



<https://github.com/lp6m/ImageDetectionHW2>



# コンテスト結果

- 2018年12月に沖縄で開催されたFPT2018にて**優勝**！
- 14チーム中 3チームがROS
- 半数以上のチームがロボットが動かなかった中、**ROS**を使っていたチームは全て動作、内2つが**決勝進出**

|   |                                   |     |     |     |     |                   |
|---|-----------------------------------|-----|-----|-----|-----|-------------------|
| 8 | Kyoto University                  | 420 | 160 | 200 | 120 | 670 + 900<br>1570 |
| 5 | Hiroshima City University         | 150 | 0   | 0   | 120 | 510 + 270<br>780  |
| 3 | <del>Ritsumeikan University</del> |     |     |     |     |                   |
| 1 | The University of Aizu            | 110 | 0   | 0   | 0   | 350 + 110<br>460  |
| 6 | Utsunomiya University             | 0   | 0   | 0   | 0   | 300 + 0<br>300    |

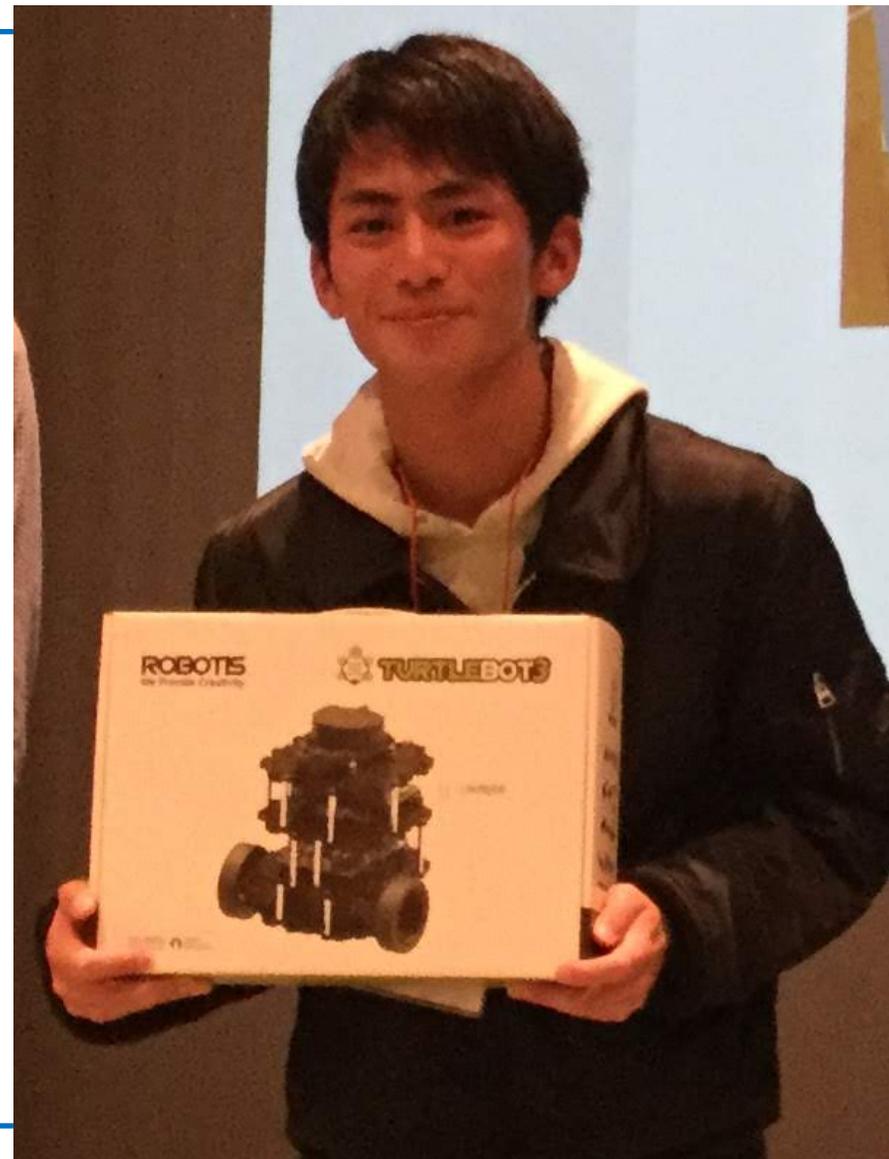


• **Welcome to ROS × FPGA**

- 優勝商品はなんと…

## *TurtleBot3*

⋮ 2号機につづく

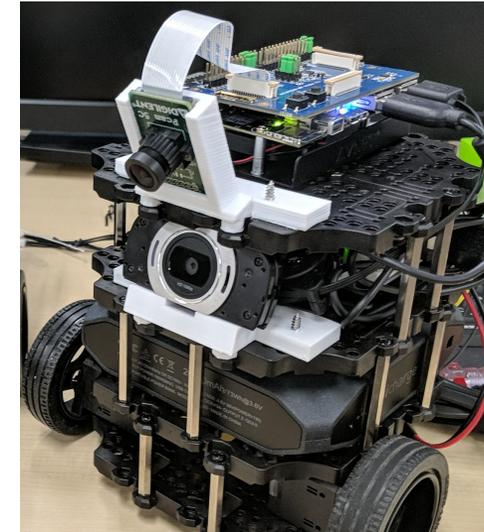
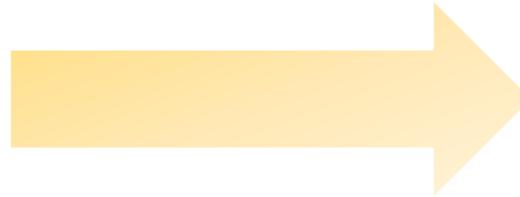
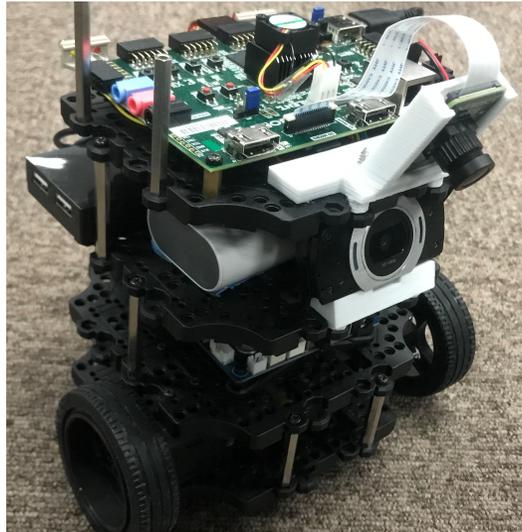


---

# ROS 2への移行

- ROS 2移行へのモチベーション
- 移行の流れ

# ZytleBot to ROS 2!



|            |                     |
|------------|---------------------|
| 車体         | TurtleBot3          |
| メインボード     | <i>Zybo Z7-20</i>   |
| OS         | Ubuntu <i>16.04</i> |
| ROS Distro | <b>Kinetic Kame</b> |

|            |                        |
|------------|------------------------|
| 車体         | TurtleBot3             |
| メインボード     | <i>Ultra96</i>         |
| OS         | Ubuntu <i>18.04</i>    |
| ROS Distro | <b>Crystal Clemmys</b> |

現在は**Dashing**に対応

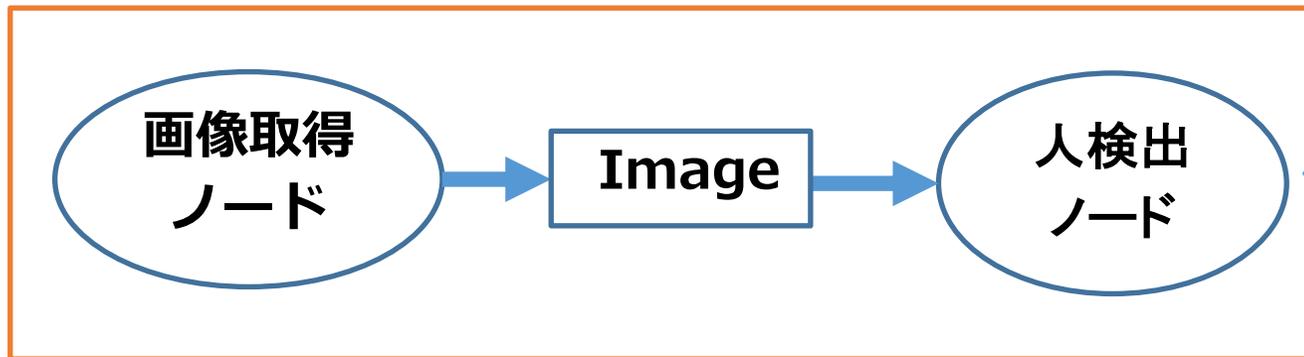
# Why ROS 2?

---

- とりあえずNodeletをやめたい
- ノートPCをMacBookにしたい
- launchファイルがPythonに…?

# Why ROS 2?

- とりあえずNodeletをやめたい
- ノートPCをMacBookにしたい
- launchファイルがPythonに…?



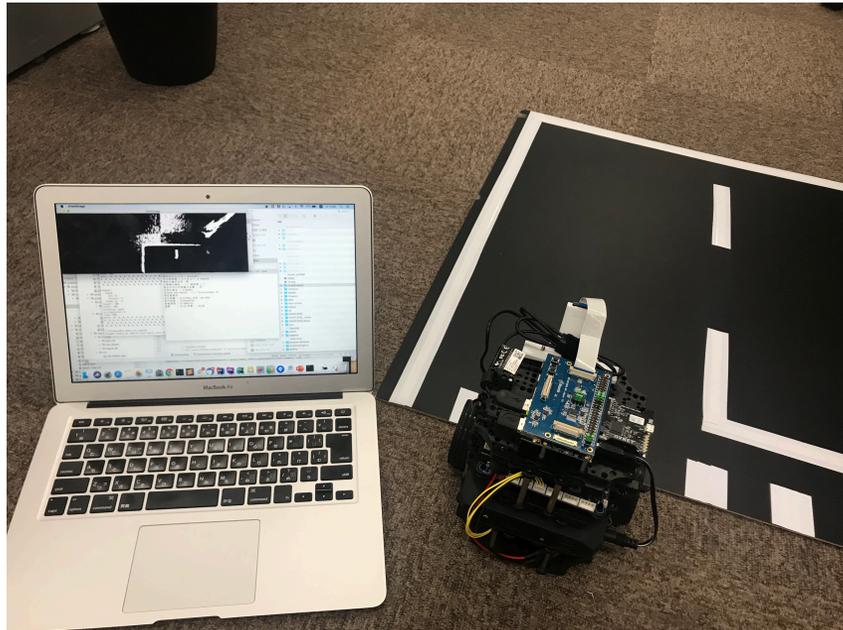
ノードごとに1プロセス  
▼  
通信コストがかかる

**Nodelet**  
複数ノードをまとめて1プロセスで起動  
◎スマートポインタによるデータコピー  
×保守性 再利用性

**ROS 2では標準機能**  
コンポーネントとしての  
記法を定めて  
**保守性 再利用性**◎

# Why ROS 2?

- とりあえずNodeletをやめたい
- ノートPCをMacBookにしたい
- launchファイルがPythonに…?



## Tier 1 platforms:

- Ubuntu 18.04 (Bionic): amd64 and arm64
- Mac OS X 10.12 (Sierra)
- Windows 10 (Visual Studio 2019)

## Tier 2 platforms:

- Ubuntu 18.04 (Bionic): arm32

## Tier 3 platforms:

- Debian Stretch (9): amd64, arm64 and arm32
- OpenEmbedded Thud (2.6) / webOS OSE: arm32 and x86

<https://index.ros.org/doc/ros2/Releases/Release-Dashing-Diademata/>

# Why ROS 2?

- とりあえずNodeletをやめたい
- ノートPCをMacBookにしたい
- **launchファイルがPythonに…?**

```
def generate_launch_description():
    server = launch_ros.actions.Node(
        package='demo_nodes_cpp', node_executable='add_two_ints_server', output='screen')
    client = launch_ros.actions.Node(
        package='demo_nodes_cpp', node_executable='add_two_ints_client', output='screen')
    return launch.LaunchDescription([
        server,
        client,
```

xml ➡ **Python**

※現時点では開発中

# Why ROS 2?

- とりあえずNodeletをやめたい
- ノートPCをMacBookにしたい
- launchファイルがPythonに…?
- **etc..**

DDS?

FPGAとの  
親和性?

リアルタイム性?

組込み向け?



とりあえず  
やってみよう!

# ROS 2移行のための取り組み

---

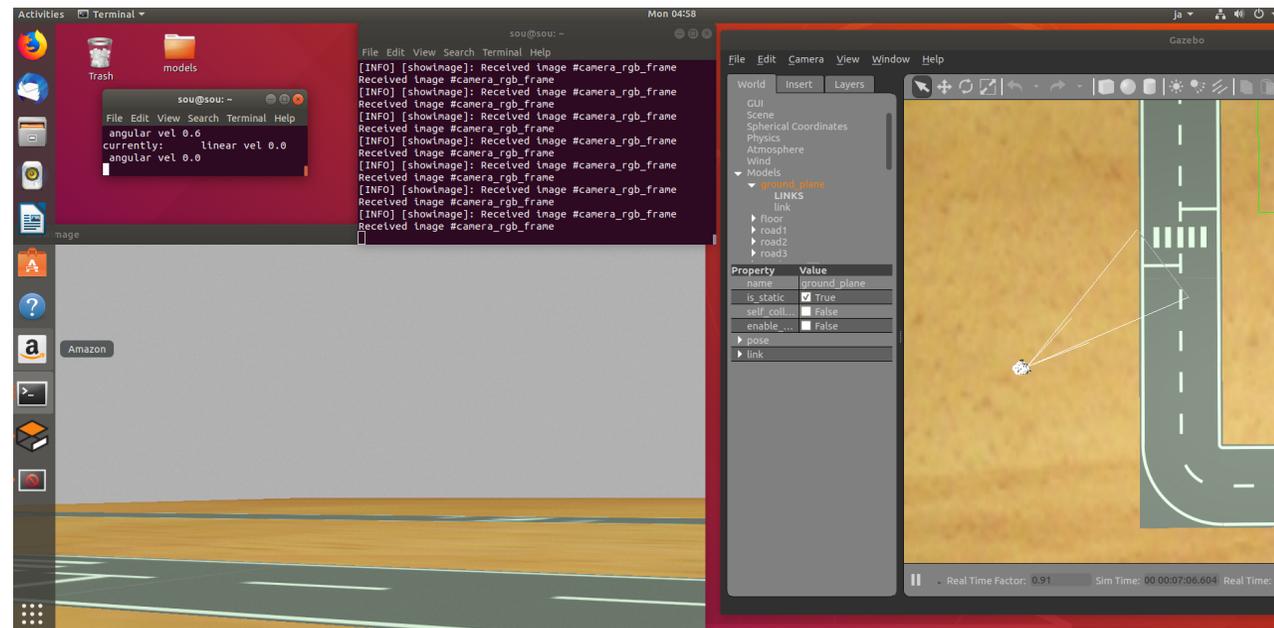
- ROS 2のコンポーネント形式にそったパッケージの作成
- Gazeboシミュレータの移行
- 実機へのインテグレーション

# ROS 2 プログラミング

- 2通りの選択肢
  - ROS 1と同様の書き方
  - **共有ライブラリ**としてノードを実装する (**ベストプラクティス**)
- 共有ライブラリとしての書き方を理解する必要があるため、プログラミング難易度は上がる
- 同一プロセス内で複数ノード実行ができる
  - データの**ゼロコピー**や、**ノードの実行順序を守らせる**ことが可能になる
- Nodeletで不満だった保守性、安全性も ◎

# Gazeboシミュレータの移行

- GazeboもROS 2対応！！
- モデルの記法は変わらず
- ROS 1プラグインを対応したROS 2プラグインに移行
  - マイグレーションガイドあり！  
gazebo\_ros\_pkgsのgitのWikiを参考



[github.com/ros-simulation/gazebo\\_ros\\_pkgs/wiki](https://github.com/ros-simulation/gazebo_ros_pkgs/wiki)

## Migration

| ROS 1 plugin                | ROS 2 plugin      | Migration guide           |
|-----------------------------|-------------------|---------------------------|
| gazebo_ros_camera           | gazebo_ros_camera | <a href="#">migration</a> |
| gazebo_ros_triggered_camera | gazebo_ros_camera | <a href="#">migration</a> |

# 実機へのインテグレーション

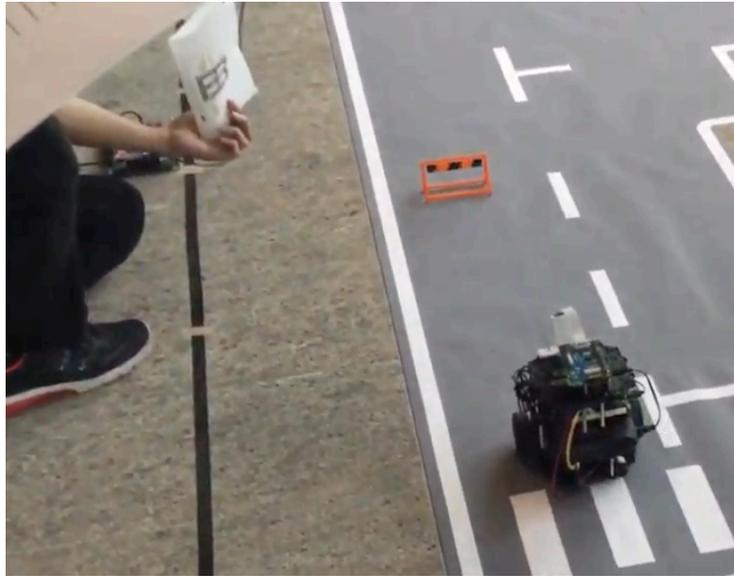
- ロボット動作に必要なパッケージがROS 2に対応しているか否かで明暗が別れる
- 今回は**TurtleBot3がROS 2対応**していたため、スムーズに移行
  - ロボットベンダー社様、ROS 2パッケージをお願いします…
  - (私たちももちろん頑張ります^^;)
- rosbag2も利用可能！
  - ただし、1で可能だった再生速度の変更などは未対応

# ROS 1 or ROS 2 ?

- ROS 2で大変だった点
  - ...特になし！（先人の方々のおかげ）
  - コア部分の開発はかなり進んでいる実感
- ROS 2に移行するならこれからが**チャンス**
  - 初の長期サポートdistroのDashingがリリース
  - 日本語の講習会、書籍、記事なども！
  - ROS 1のソフトウェア資産も魅力的だが、みんなでしがみ付いていると…



# ROS 2 コンテスト結果



Evaluation Score

| Team                      | Score for Run       | Traffic signal    | Obstacles         | Doll              | Total               |
|---------------------------|---------------------|-------------------|-------------------|-------------------|---------------------|
| Ritsumeikan Univ.         | 80<br><del>70</del> | 0<br><del>0</del> | 0<br><del>0</del> | 0<br><del>0</del> | 80<br><del>70</del> |
| Kyoto University          | 330                 | 80                | 160               | 120               | 690                 |
| Meisei University         | 10                  | 0                 | 0                 | 0                 | 10                  |
| Hiroshima City University | 140                 | 0                 | 0                 | 0                 | 140                 |

- HEART2019 @長崎
- ROS 2でも優勝！

 [https://github.com/sousou1/zylebot\\_ros2](https://github.com/sousou1/zylebot_ros2)

**Welcome to ROS 2 and FPGA !**

*To Be Continued FPT2019*