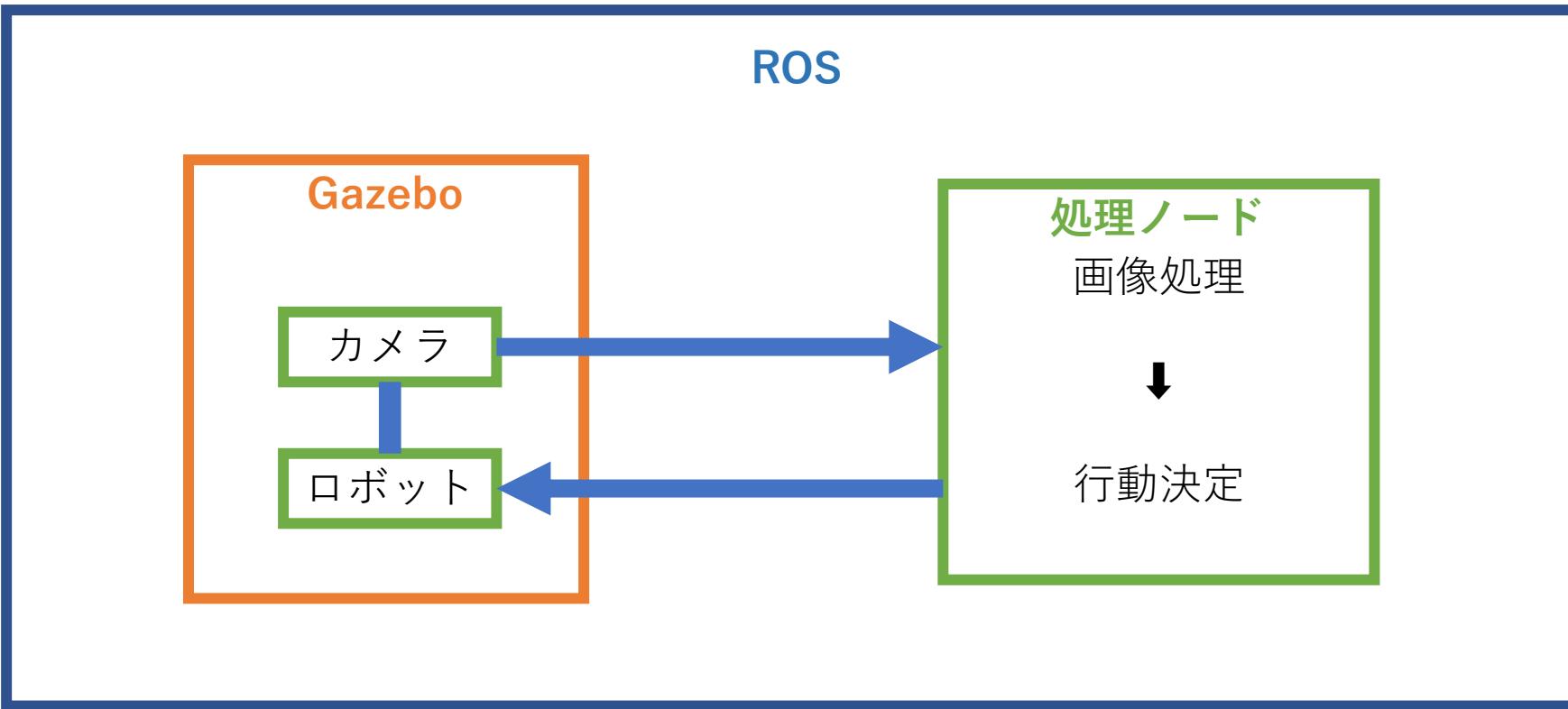


# シミュレータの使い方

京都大学情報学研究科 田村 爽

# ROSのシミュレーター



# シミュレーター概要

- DockerHub sousou1/fpt-simulatorに公開中
- 中身：仮想デスクトップ+ROS環境+シミュレーター+オリジナルのモデルデータ+サンプルコード
- ノートPCでは重いので、サーバーやデスクトップPC推奨
  - ノートPCで2fps程度、サーバーを使えば10fps

# 備考：サーバーで動かした場合のつなぎ方

- ・同ネットワークにいる場合、もしくはグローバルにIP公開されていて直接IPを叩いて繋げられる場合

Host lab\_server

Hostname xxx.xxx.xxx.xxx

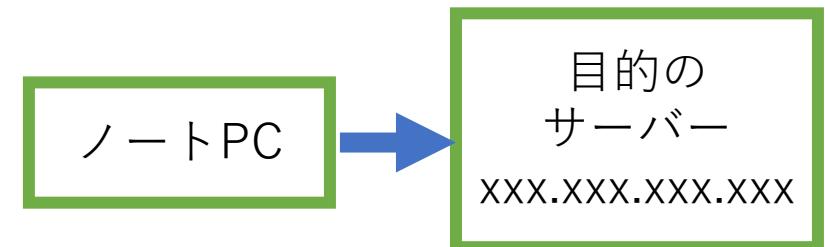
Port 22

User tamura

LocalForward 6080 xxx.xxx.xxx.xxx:6080

Identityfile ~/.ssh/id\_dsa

- ・ ssh lab\_server



# サーバーで動かした場合のつなぎ方2

外部から踏み台サーバー経由でつなげる場合

Host outlab

User tamura

```
ProxyCommand ssh vvv.vvv.vvv.vvv nc -w 10 vvv.vvv.vvv.vvv 22
```

```
LocalForward 6080 xxx.xxx.xxx.xxx:6080
```

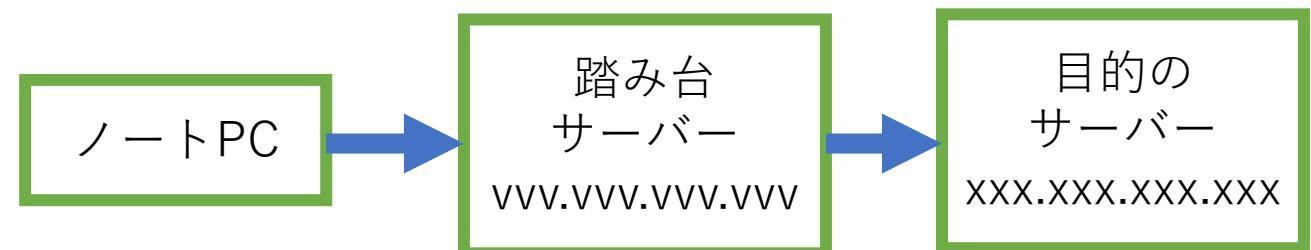
Host vvv.vvv.vvv.vvv

Hostname vvv.vvv.vvv.vvv

Port 22

User tamura

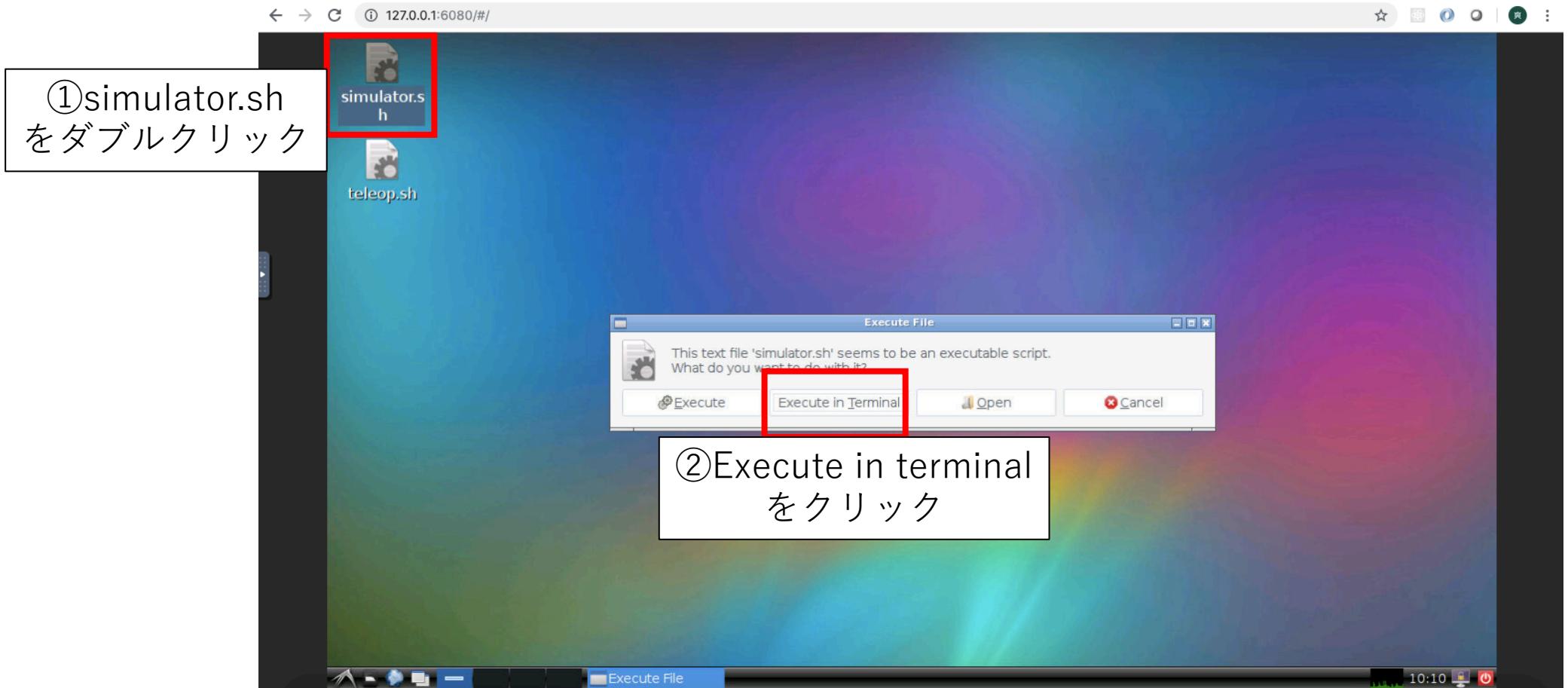
Identityfile ~/.ssh/id\_dsa



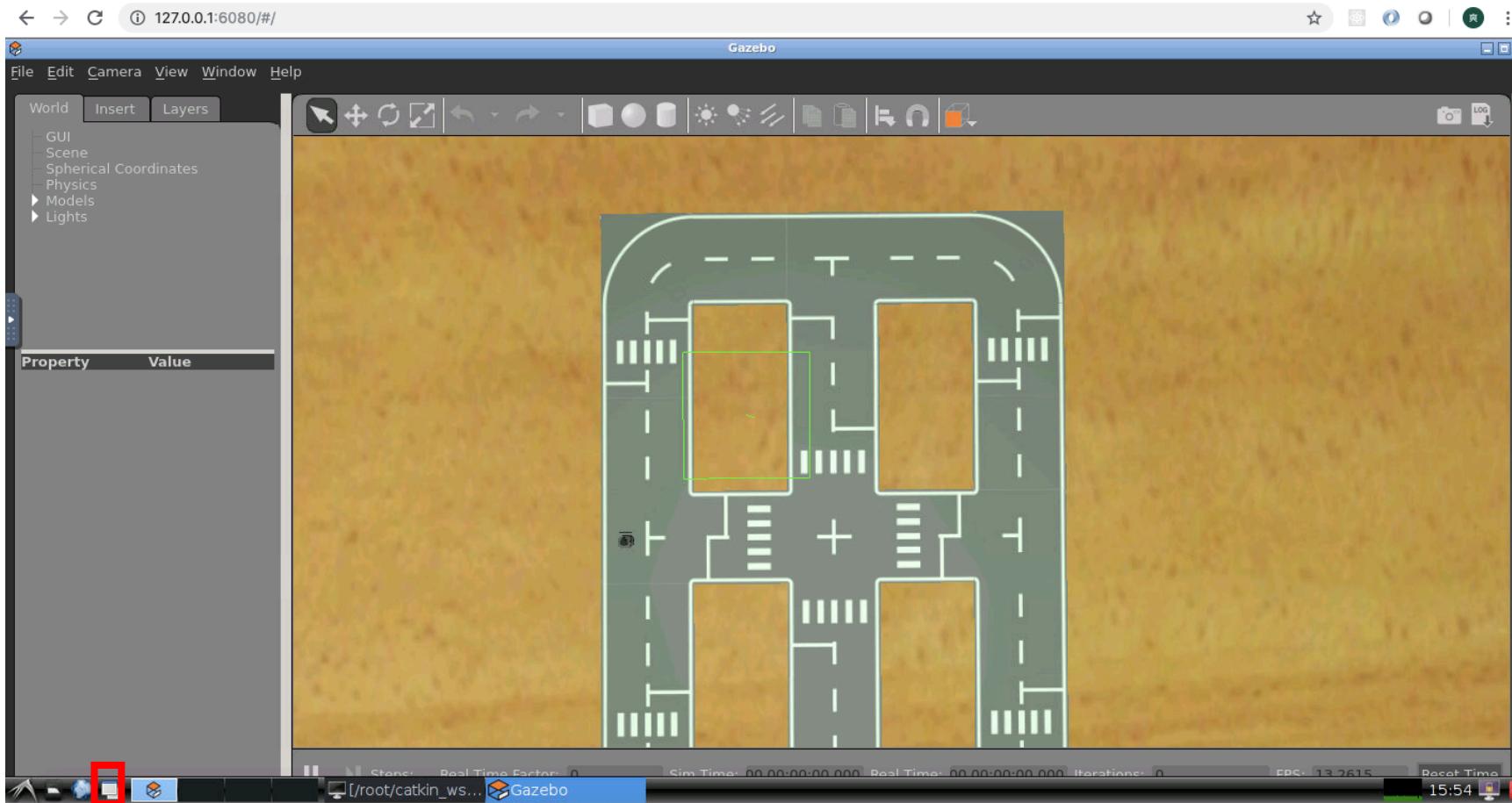
# シミュレーター起動方法

- 事前準備 dockerが入ってるPC (or サーバー)
- \$ docker run -p 6080:80 -it sousou1/fpt-simulator:latest
- <http://127.0.0.1:6080/#/>
- いきなりデスクトップでできます

# チュートリアル1

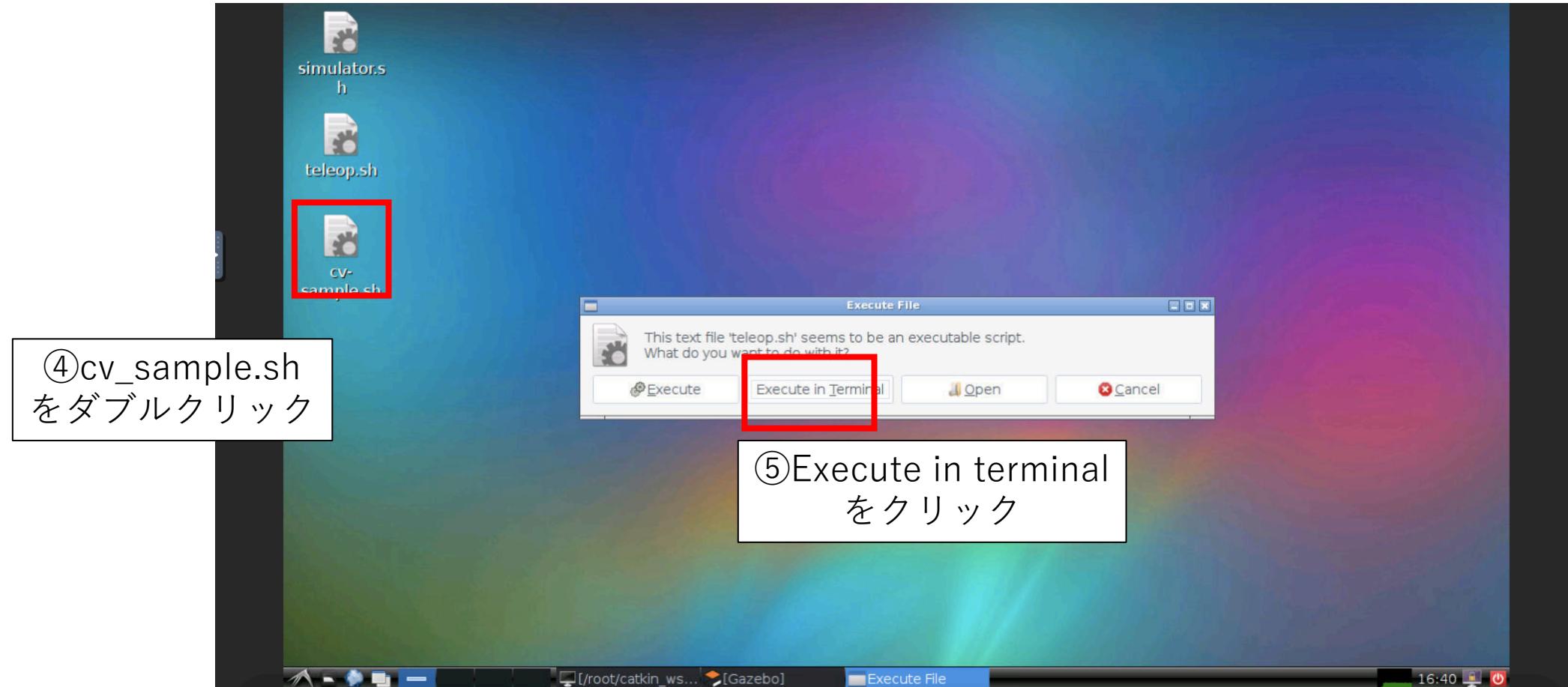


# チュートリアル2

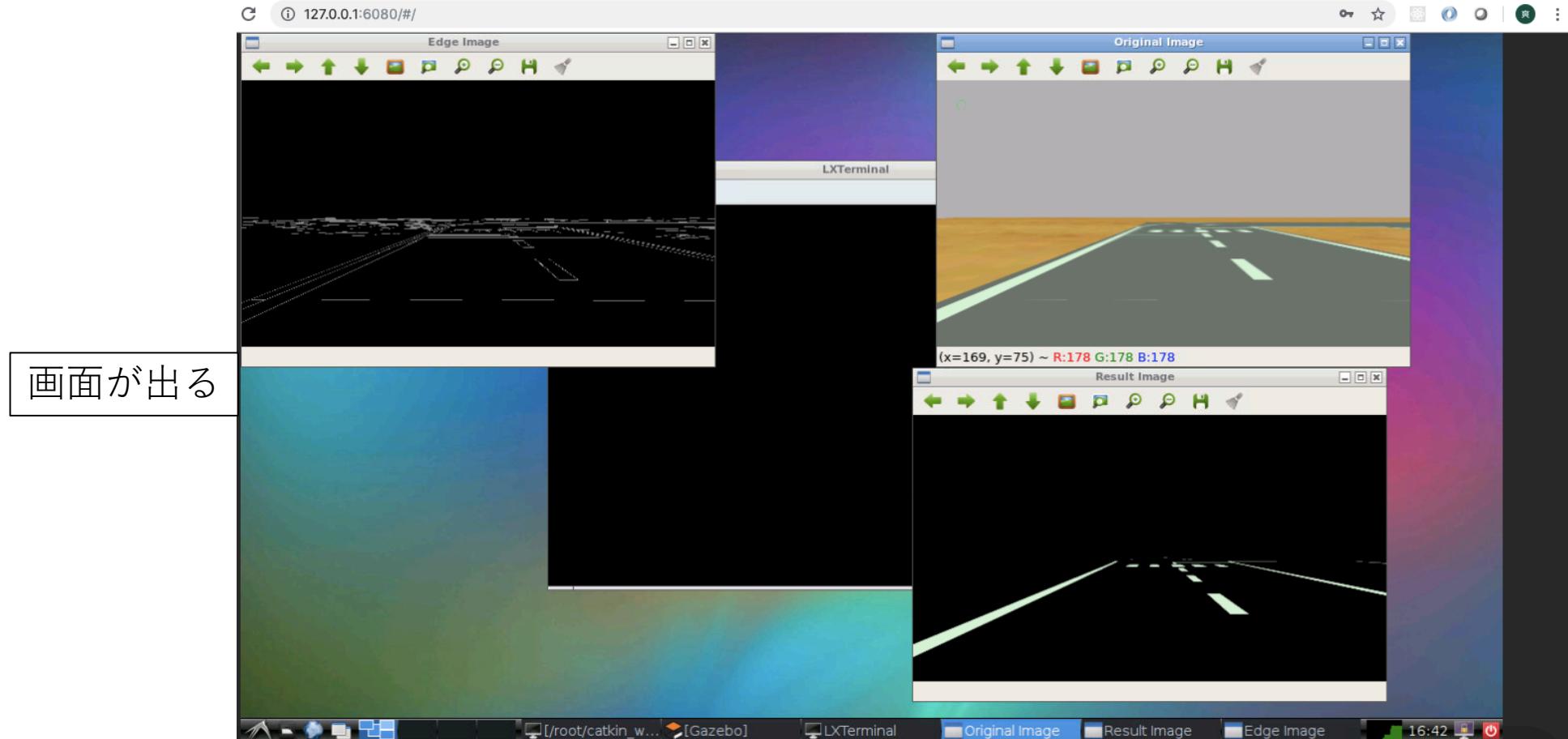


③最小化する

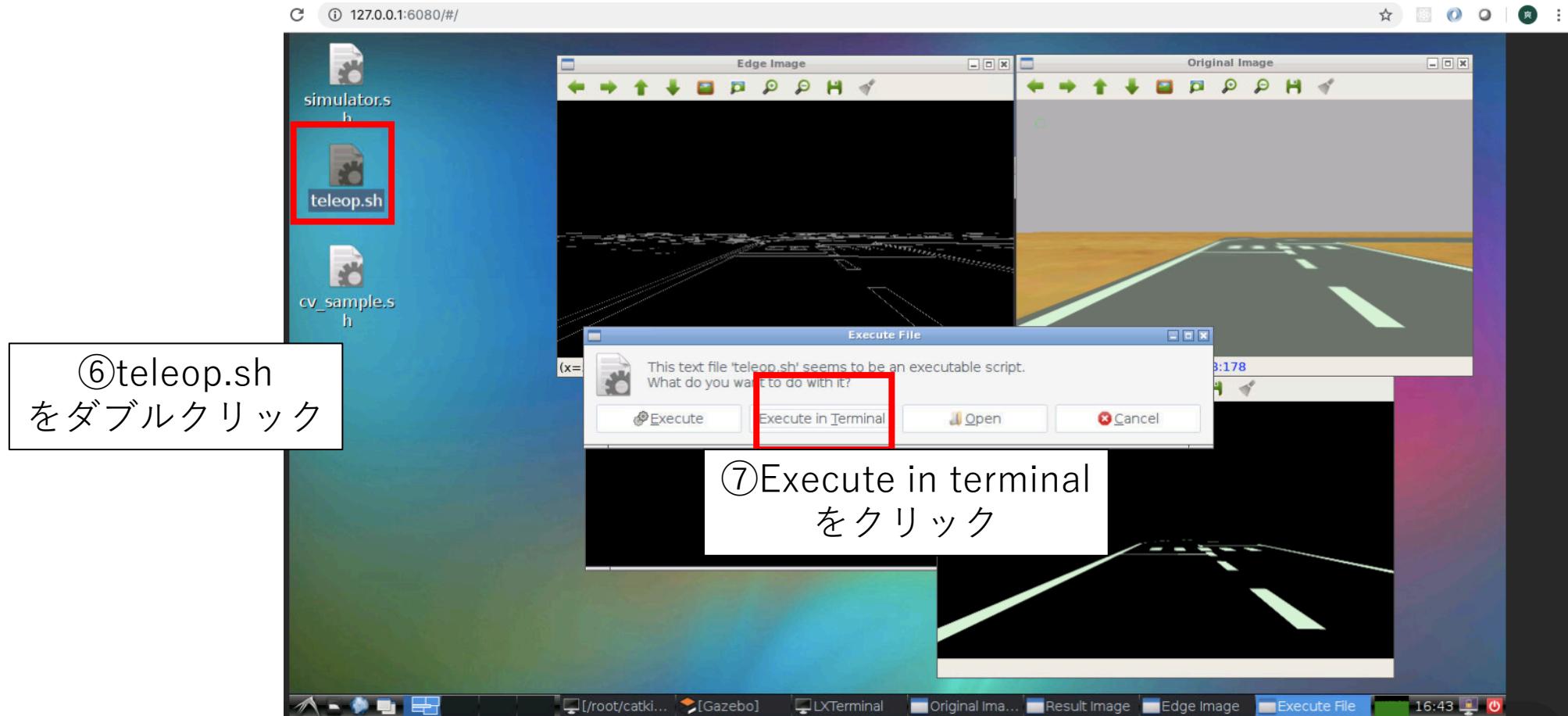
# チュートリアル3



# チュートリアル3

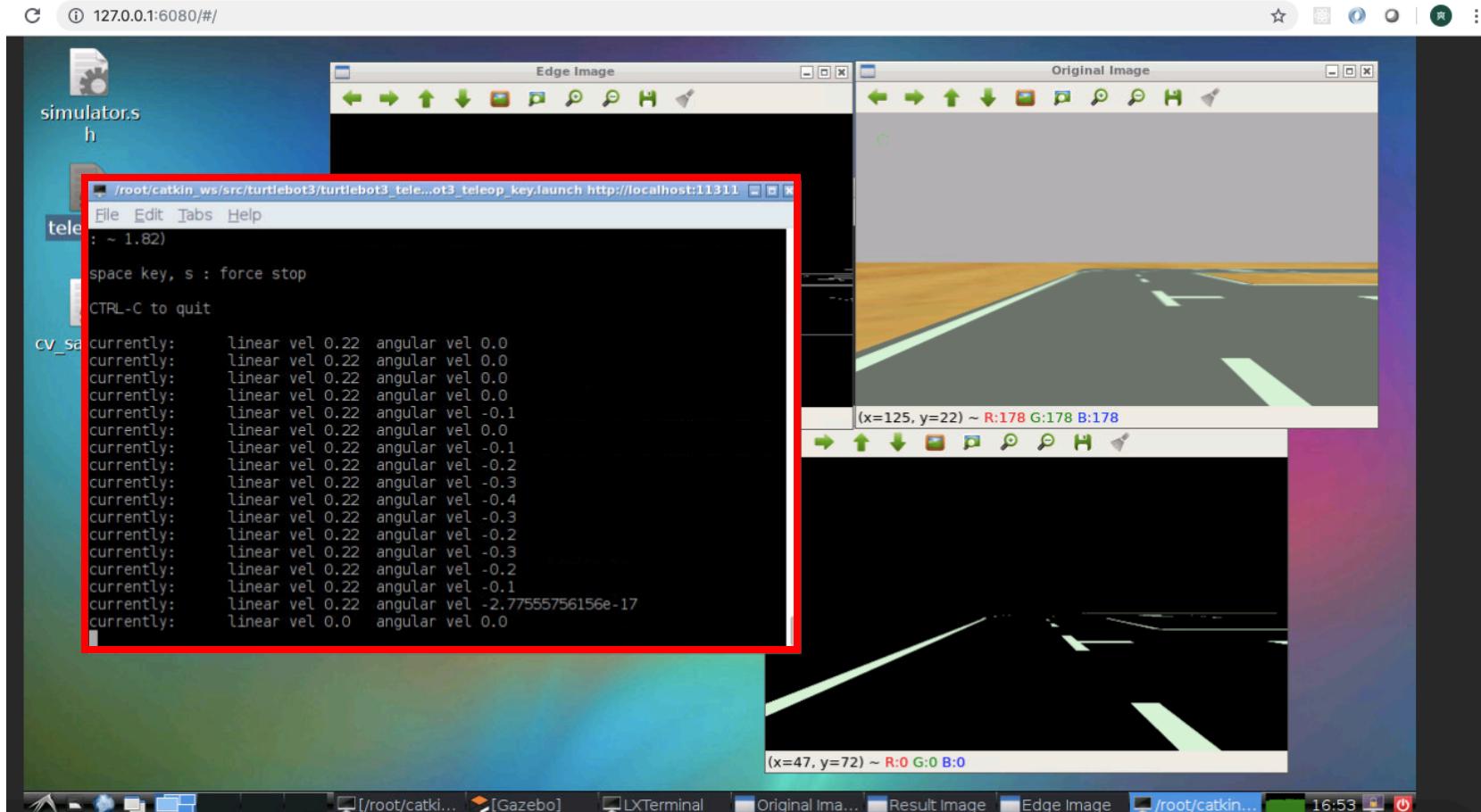


# チュートリアル4

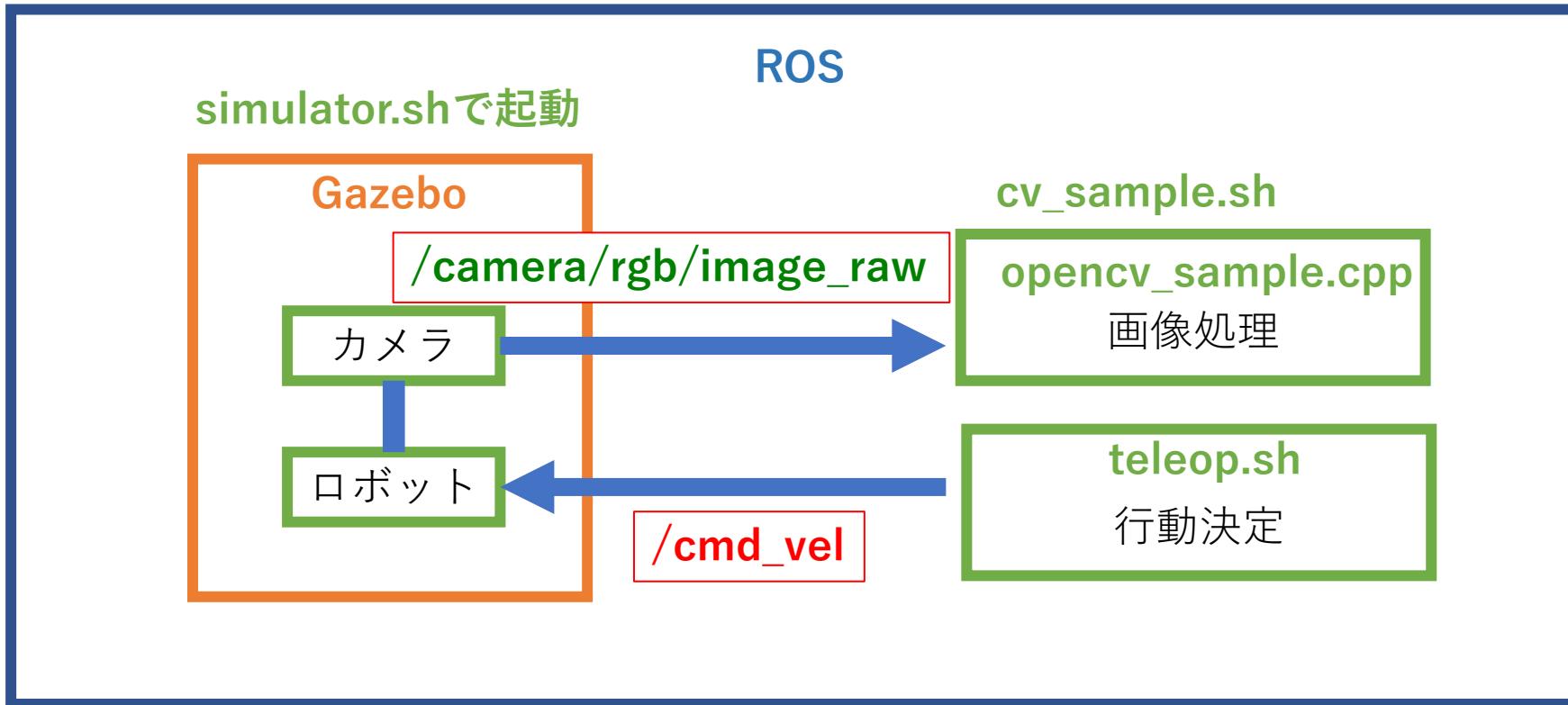


# チュートリアル5

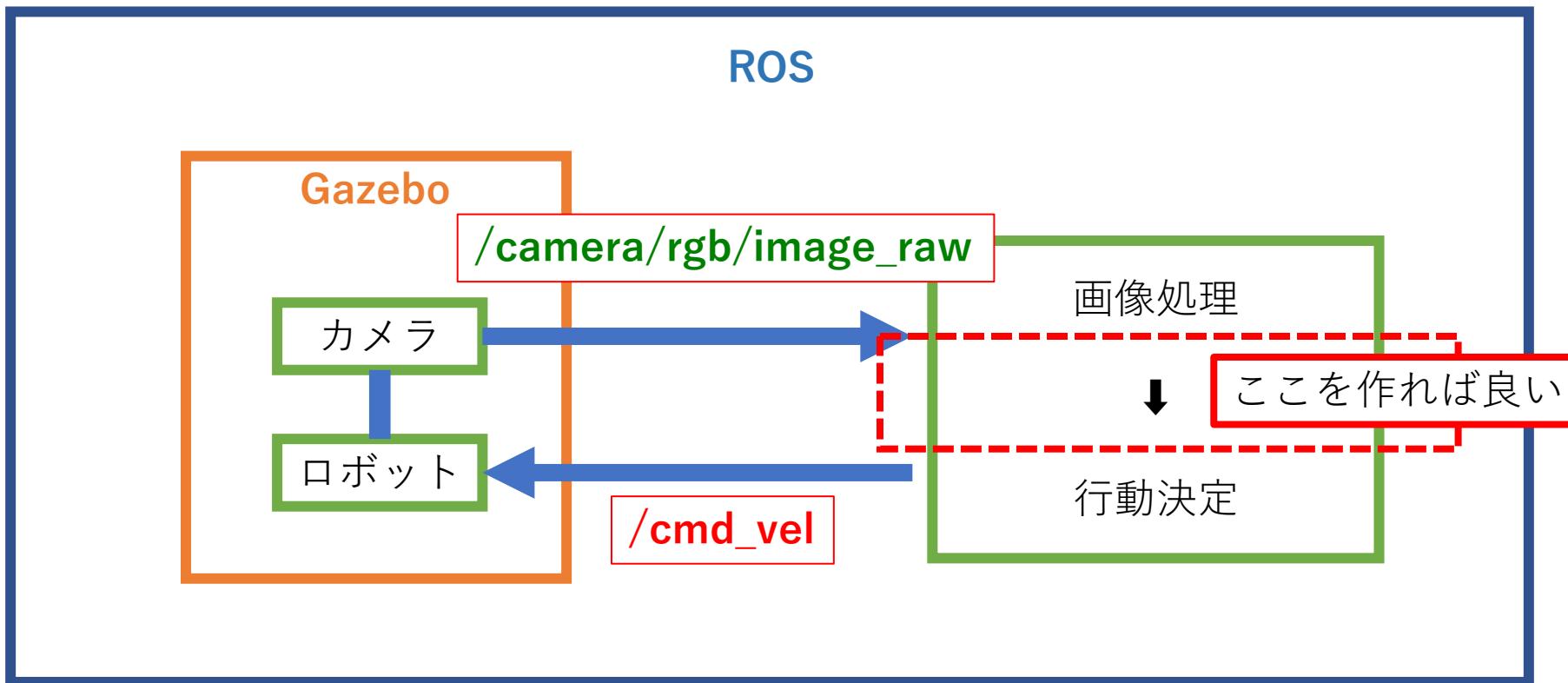
⑥  
W  
A S D  
space  
で運転！



# 仕組み



# 自動運転のために



# サンプルコード(opencv\_sample.cpp)解説

/root/catkin\_ws/src/opencv\_sample/src/opencv\_sample.cpp

(前略)

```
void imageCb(const sensor_msgs::ImageConstPtr& msg)
{
```

(中略)

```
    cv_ptr = cv_bridge::toCvCopy(msg,
        sensor_msgs::image_encodings::BGR8)
```

```
    cv::imshow("Original Image", cv_ptr->image);
```

```
}
```

画像が送られてくるたび呼び出される

# これだけは必要なROS API

- `twist_pub.publish(twist);`
    - 速度 : `twist.linear.x = 0.1;` ( $0.01 \sim 0.22$ )
    - 角度 : `twist.angular.z = 0.1;` ( $-2.84 \sim 2.84$  右回り)
- のように値をセットして、このAPIを呼び出すことで、ロボットに指令が飛ぶ

# autonomous.cppの作り方

(前略)

```
void imageCb(const sensor_msgs::ImageConstPtr& msg)
{
```

(中略)

```
    cv_ptr = cv_bridge::toCvCopy(msg,
        sensor_msgs::image_encodings::BGR8)
```

画像が送られてくるたび呼び出される

cv\_ptrを画像処理して、twist.linear.x やtwist.angular.z を決定する処理

```
    twist_pub.publish(twist);
```

```
}
```

ここに任意のアルゴリズムを入れるとで、ロボットを自動運転できる！

# 編集するファイル&ビルド

- /root/catkin\_ws/src/opencv\_sample/src/autonomous.cpp
- ビルド  
\$ cd catkin\_ws  
\$ catkin\_make
- 実行  
rosrun opencv\_sample autonomous  
もしくはデスクトップの autonomous.sh

# 実機へのインテグレーション

