



家庭用自律移動ロボット 「カチャカ」の開発者API公開と ROS 2インターフェイス実装

近藤 豊 <youtalk@probotics.jp> 野崎 耕平 外丸 敏宏 村瀬 和都 日台 健一 寺田 耕志

株式会社Preferred Robotics <https://www.probotics.jp>

近藤 豊

@youtalk

- 2021/11~ Preferred Robotics PjM / エンジニア
- 2019/8 #ROS2ではじめよう 出版
- 2018/5~ Preferred Networks エンジニア
- 2018~ ROSCon JP 実行委員
- 2013/4~ カワダロボティクス 主任
- 2013/3 奈良先端大 Ph.D

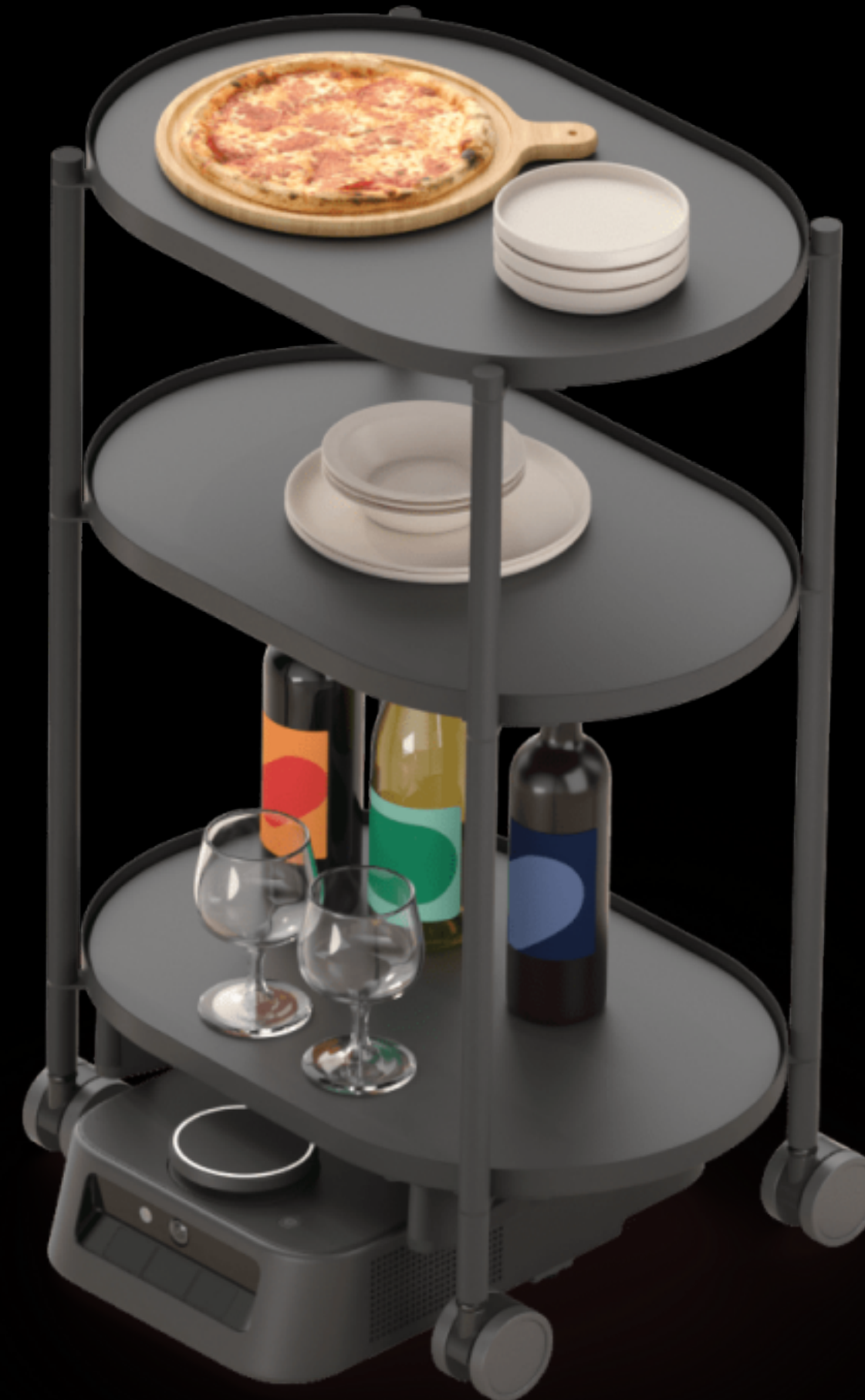




Preferred Robotics

@pfrobotics

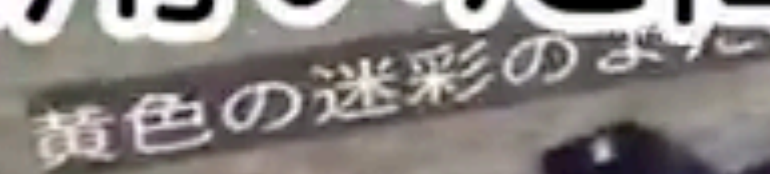
- 2021/11 Preferred Networks
より分社化
- 自律移動ロボットの製品化・
事業化に注力
- B2C / B2B両事業の同時進行



A white robotic arm, resembling a mobile service robot, is shown in a child's playroom. The arm is holding a light-colored wooden block. The background features a white shelving unit with various toys and books, and a green plant. The scene is dimly lit, with the robot's arm being the primary focus.

すべての人にロボットを

CEATEC Japan 2018



深層学習技術を用いた高速・高精度な物体認識で



スマートファニチャープラットフォーム カチャカ

<https://kachaka.life>



kachaka

@kachaka_jp

- 家庭用自律移動ロボット
- 家具の自動運搬を通じて物の移動、片付け、習慣化などをサポート
- 本体：228,000円
- カチャカシェルフ：23,800円～
- サブスクリプション：980円/月





毎週の生協宅配🚚もカチャカ🤖がいれば圧倒的に楽チン💡

配送員さん👤に
カチャカシエルフ🛒に
山盛り食材を積んでもらい...



👤 #ねえカチャカ
お風呂場に来て

カチャカ🤖は
子育て世代や共働き家庭の
良きパートナー👍



👤 #ねえカチャカ
リビングに行って

カチャカ列車🚂
家事がないときは子供の相手👶
カチャカの使い方は無限大∞



カチャカチャレンジ🤖
障害物競走🏃





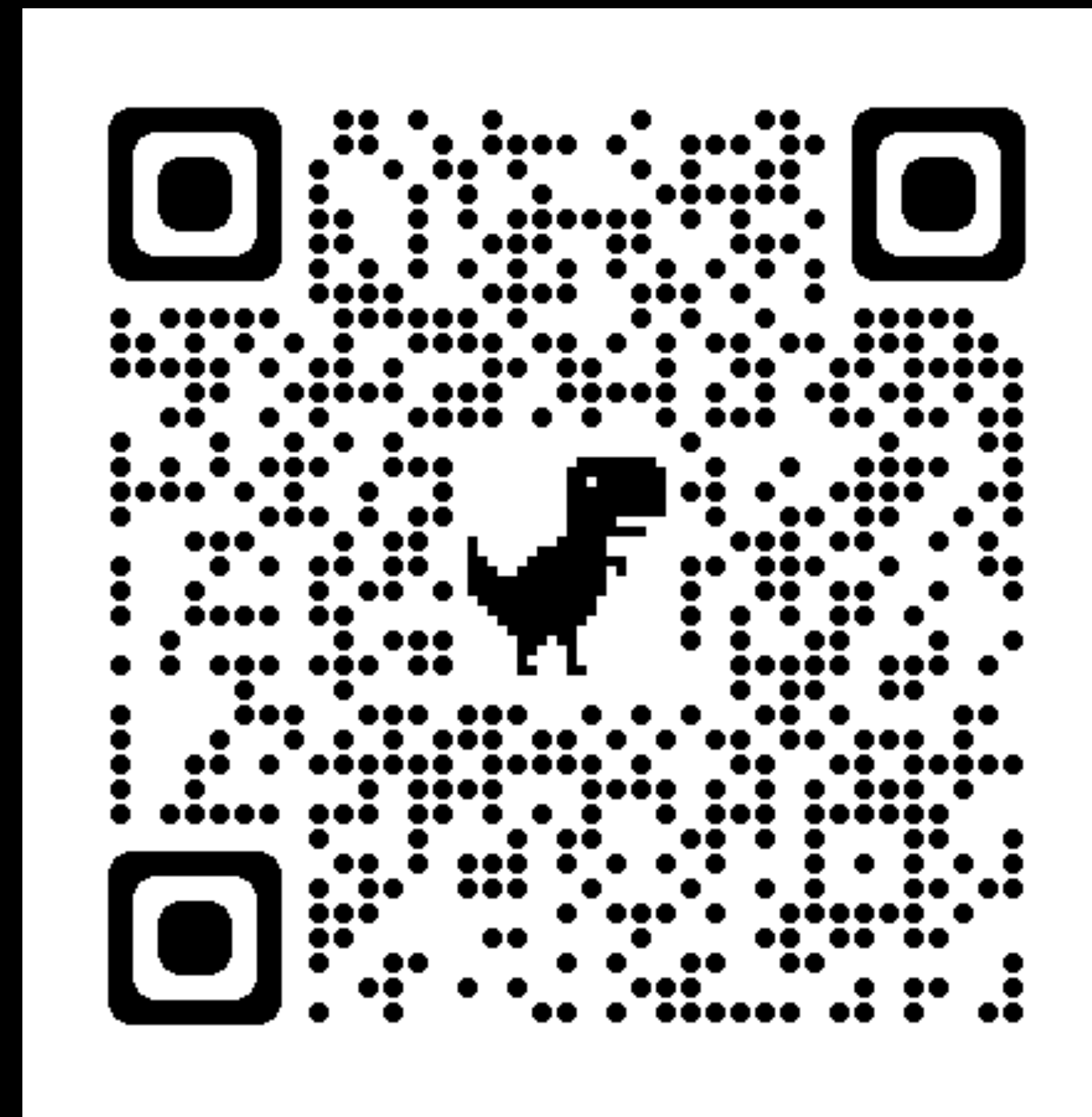
API

すべての移動ロボット開発者に
カチャカAPIを

ROSCon JP 2023

カチャカAPI

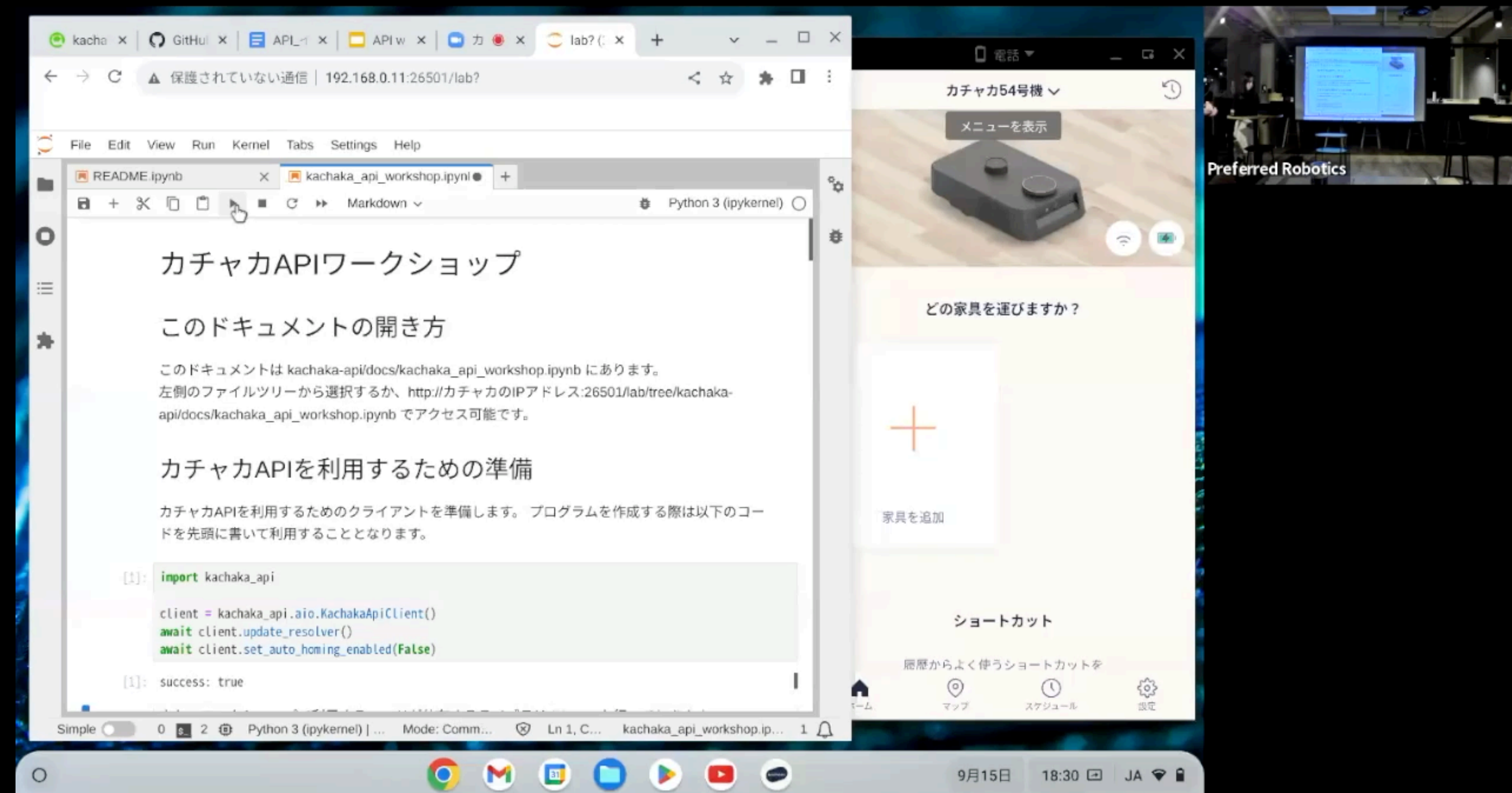
- スマフォアプリ相当のAPIを外部開発者に公開
- gRPCの.protoファイル、Pythonクライアントライブラリ、ROS 2ブリッジの提供
- ROS 2ブリッジは外部PC実行
- Apache License 2.0



<https://github.com/pf-robotics/kachaka-api>

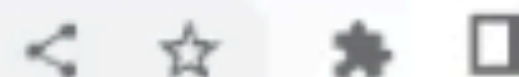
カチャカAPI

- Pythonプログラミングはカチャカ体内のJupyterLabで実行可能
- カチャカAPIの開発に必要なものはブラウザだけ



60倍速

保護されていない通信 | 192.168.0.11:26501/lab?



カチャカ54号機



充電ドックに戻す



どの家具を運びますか？

充電ドック



ワゴン



家具を追加

ショートカット

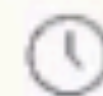
履歴からよく使うショートカットを



ホーム



マップ



スケジュール



設定

速度制御

位置ではなく速度を指定して動かすことも可能です。指定する値の単位は(m/s, rad/s)です。

下記は 0.628rad/s $\approx$ 36deg/s を10秒間与えています。

```
[13]: for i in range(10):  
        await client.set_robot_velocity(0.0, 0.628)  
        await asyncio.sleep(1)  
        await client.set_robot_stop()
```

カメラ画像

下記の実行でカメラ画像を取得、表示します。

```
[ ]: image = await client.get_front_camera_ros_compressed_image() # カメラ画像を取得  
display(Image(data=image.data, format="jpeg")) # 取得した画像を表示
```

カメラ画像の連続取得

下記のように記述すると、新しいカメラ画像が来るたびに表示を更新することができます。

実行の停止はJupyterLab上部のバーの ■ ボタンで行ってください。

カチャカAPI提供機能

コマンド実行

- 家具運搬、目的地移動
- 任意の位置姿勢移動
- 速度指令
- 発話
- 家具一覧取得、目的地一覧取得
- コマンド実行履歴一覧取得
- 自動充電、手動操作の設定

センサ出力

- オドメトリ（自己位置と現在速度）取得
- IMUデータ取得
- LiDAR点群データ取得
- 前カメラ画像取得
- 物体認識結果（人、家具、充電器、扉）取得
- カチャカマップ取得

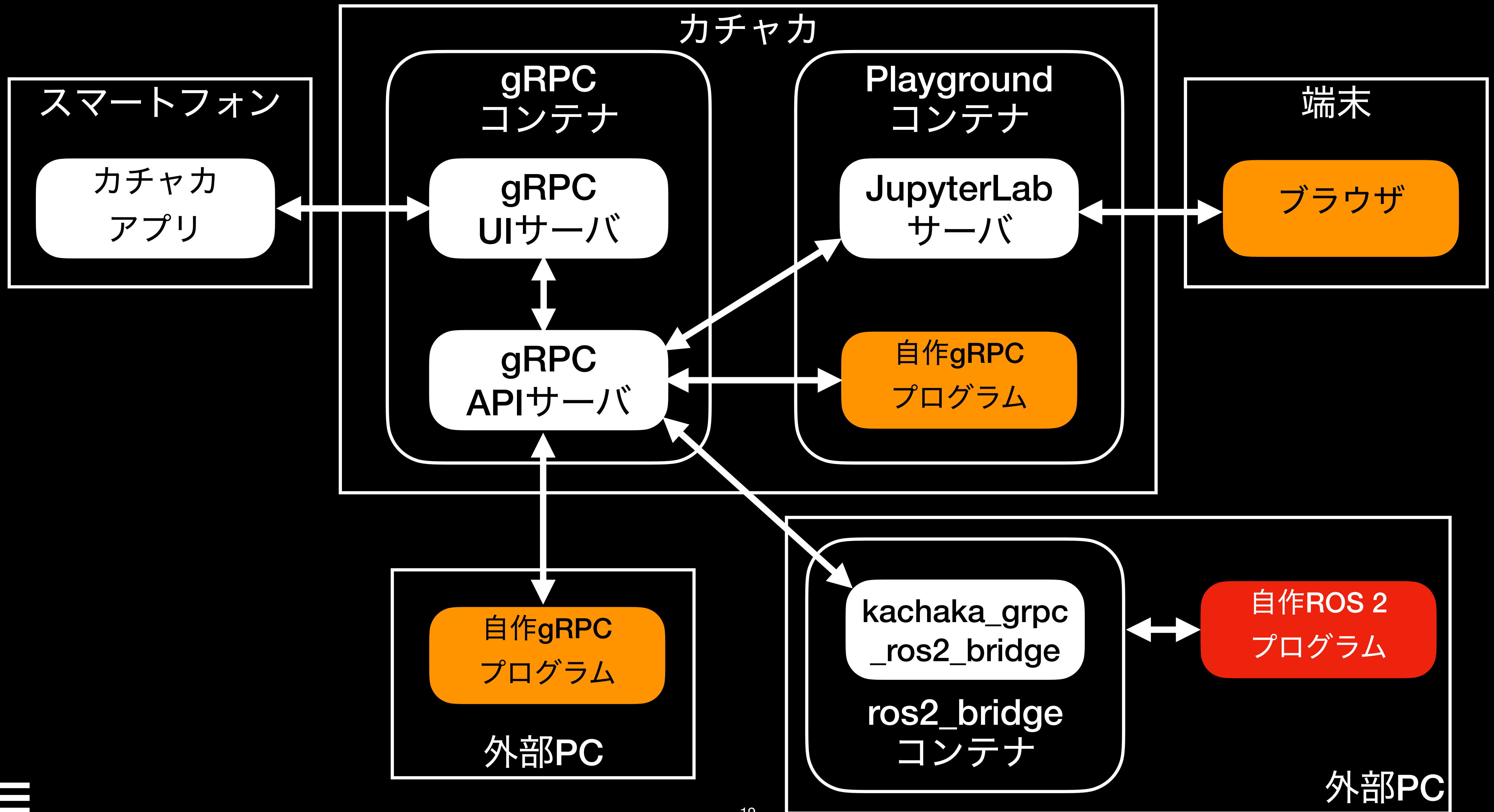
ROS 2ブリッジ提供機能

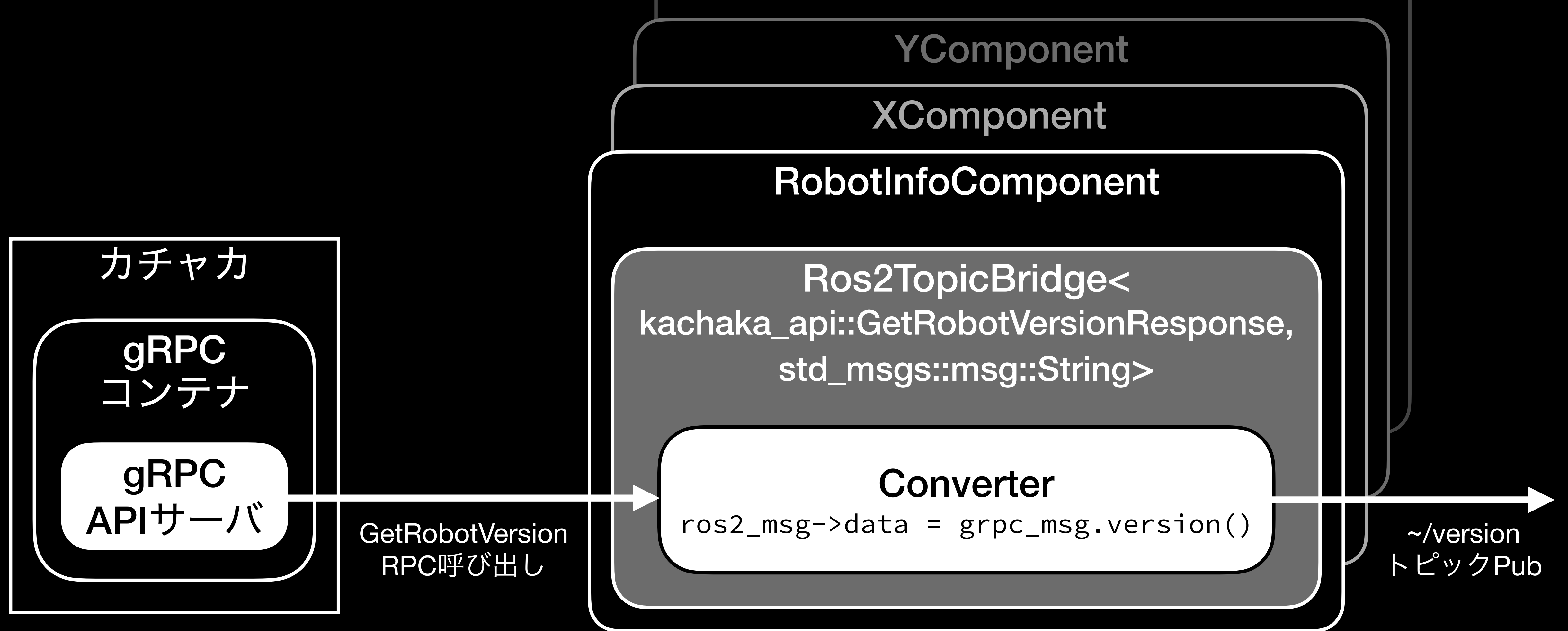
```
$ ros2 topic list | grep kachaka
/kachaka/front_camera/camera_info
/kachaka/front_camera/image_raw
/kachaka/front_camera/image_raw/compressed
/kachaka/imu/imu
/kachaka/layout/locations/list
/kachaka/layout/shelves/list
/kachaka/lidar/scan
/kachaka/manual_control/cmd_vel
/kachaka/mapping/map
/kachaka/object_detection/result
/kachaka/odometry/odometry
/kachaka/robot_info/version
/kachaka_description/joint_states
/kachaka_description/robot_description
```

```
$ ros2 service list | grep -v parameter
/kachaka/auto_homing/set_enabled
/kachaka/manual_control/set_enabled
```

```
$ ros2 action list
/kachaka/kachaka_command/execute
```

```
$ ros2 interface list | grep kachaka
.../msg/KachakaCommand
.../msg/Location
.../msg/LocationList
.../msg/ObjectDetection
.../msg/ObjectDetectionListStamped
.../msg/Shelf
.../msg/ShelfList
.../msg/ShelfSize
.../action/ExecKachakaCommand
```

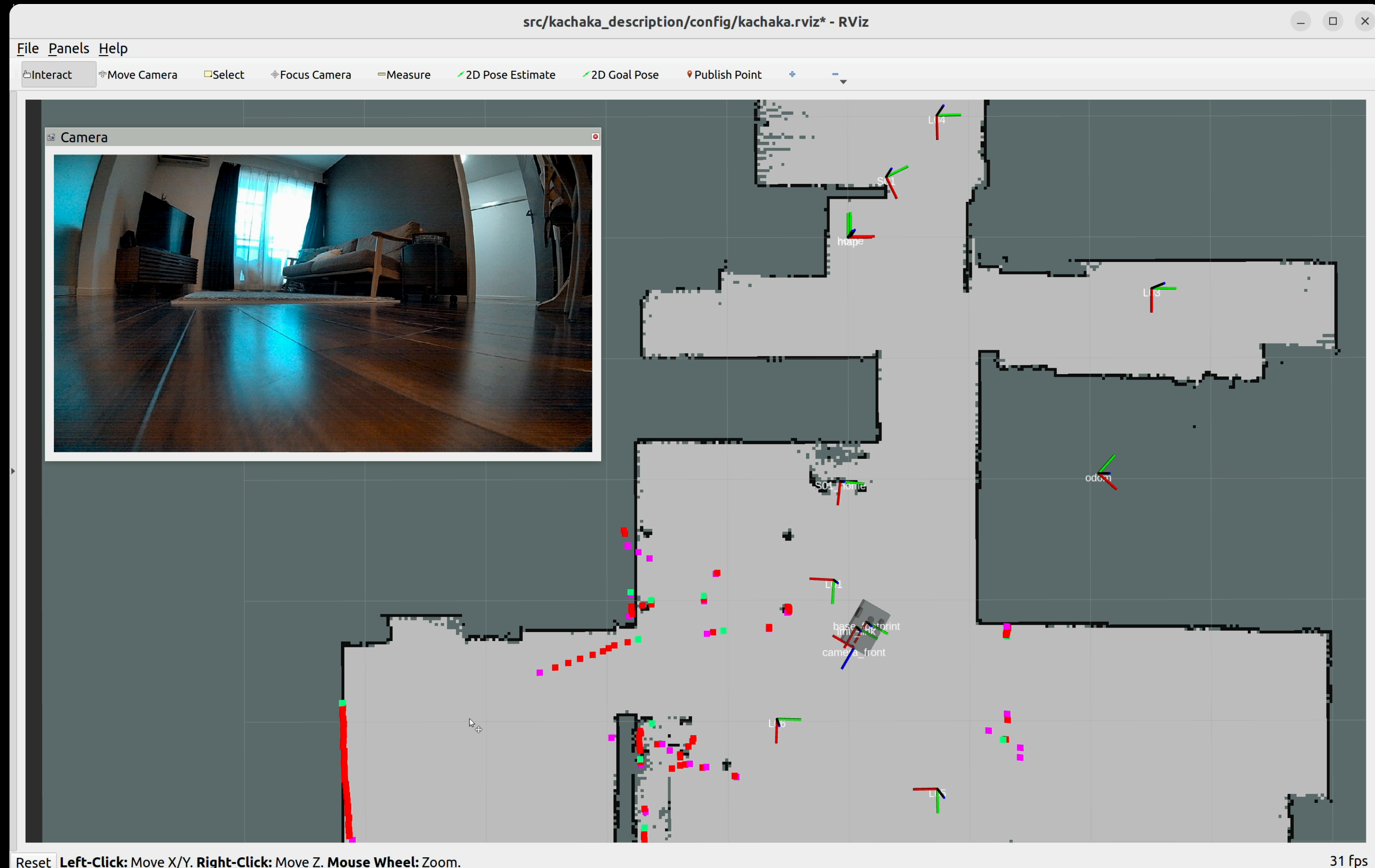





カチャカソフトウェアのバージョンを
PubするROS 2コンポーネント

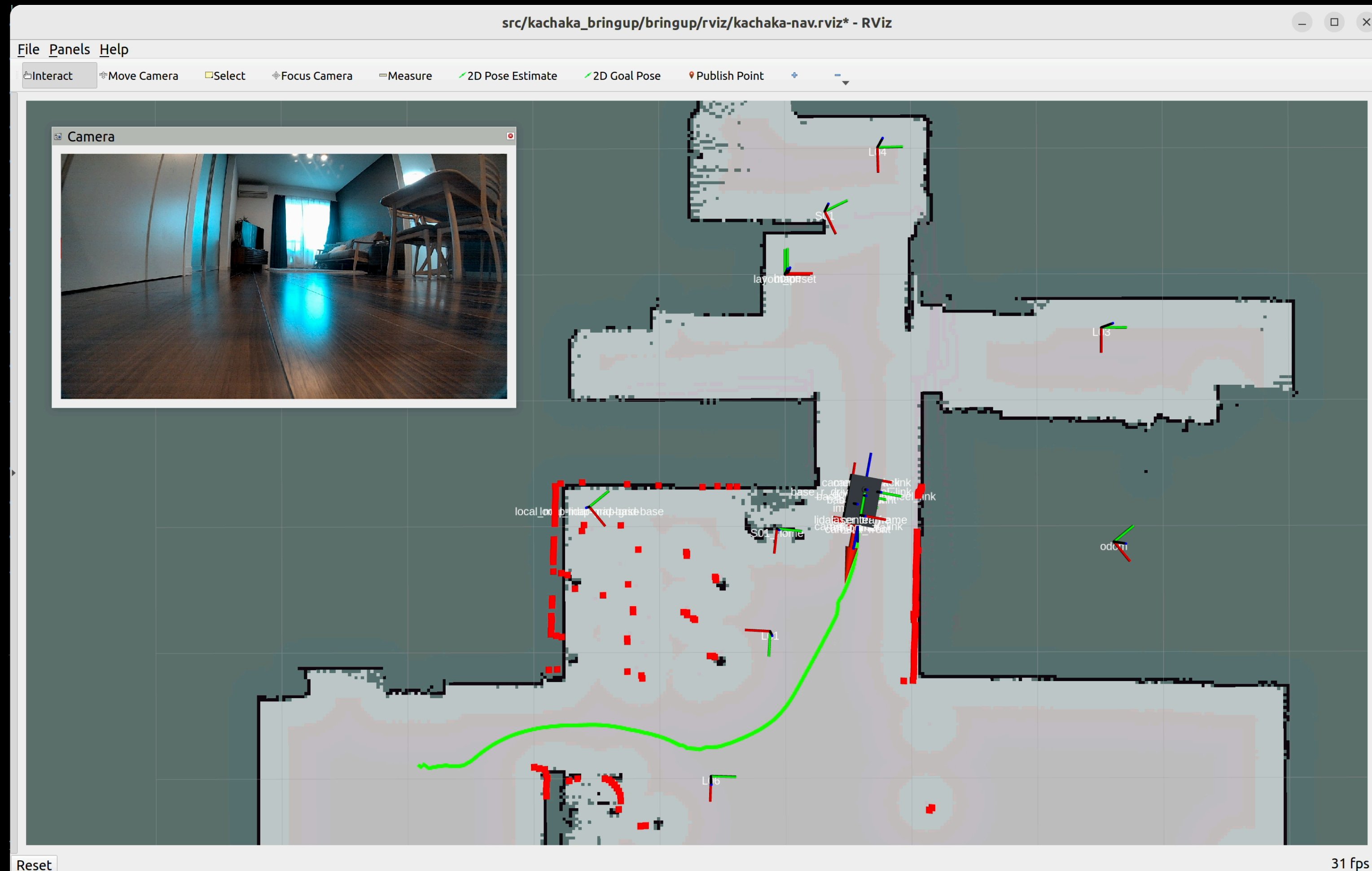
カチャカAPI経由でのナビゲーション

- ExecKachakaCommand
アクションをRViz2の2D
Goal Pose機能に適用
- 前カメラによるSemantic
Segmentation、LiDAR、
ToFすべてをセンサー統合
- 家具の積載有無でInflation
Mapも変化



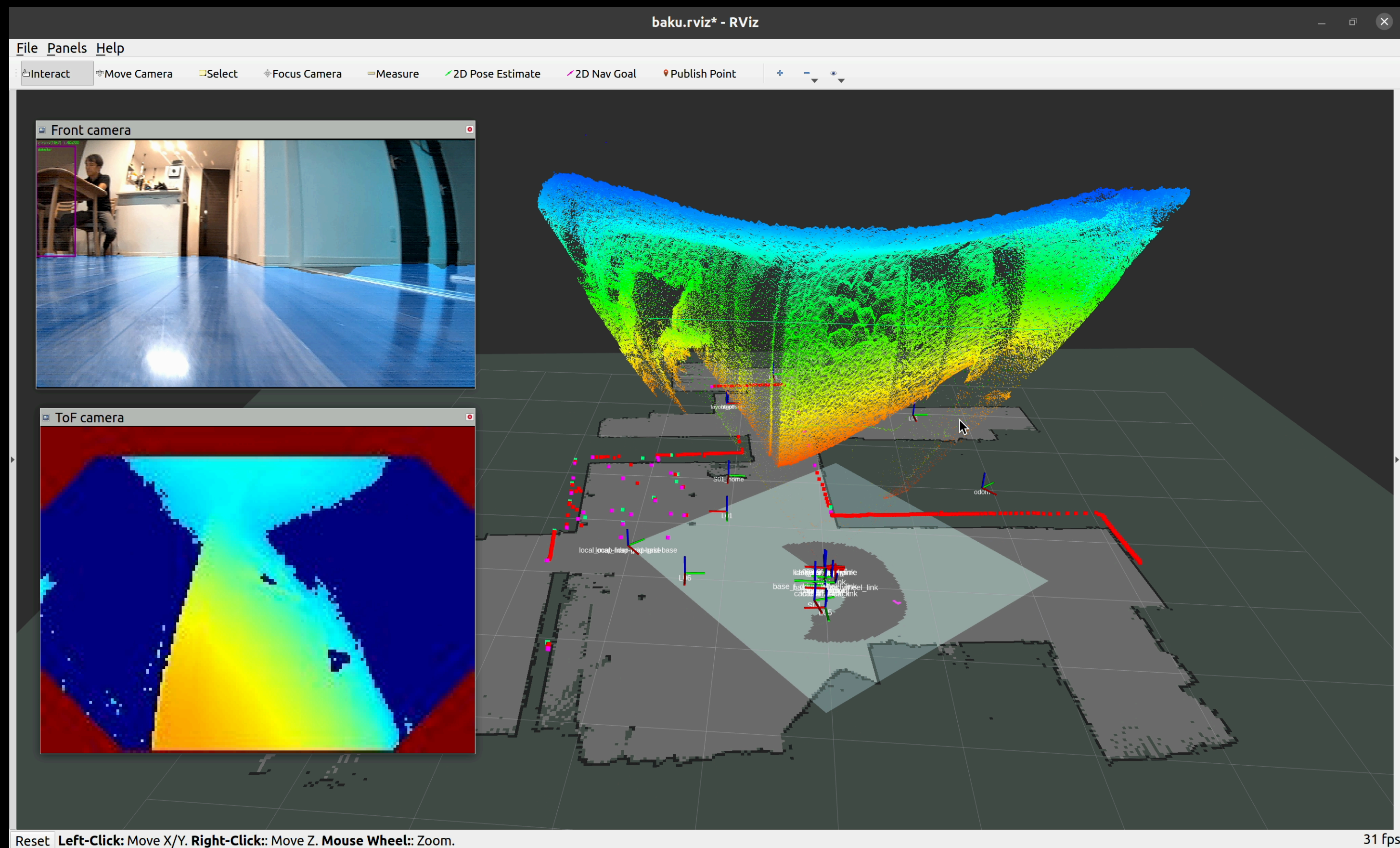
Navigation 2によるナビゲーション

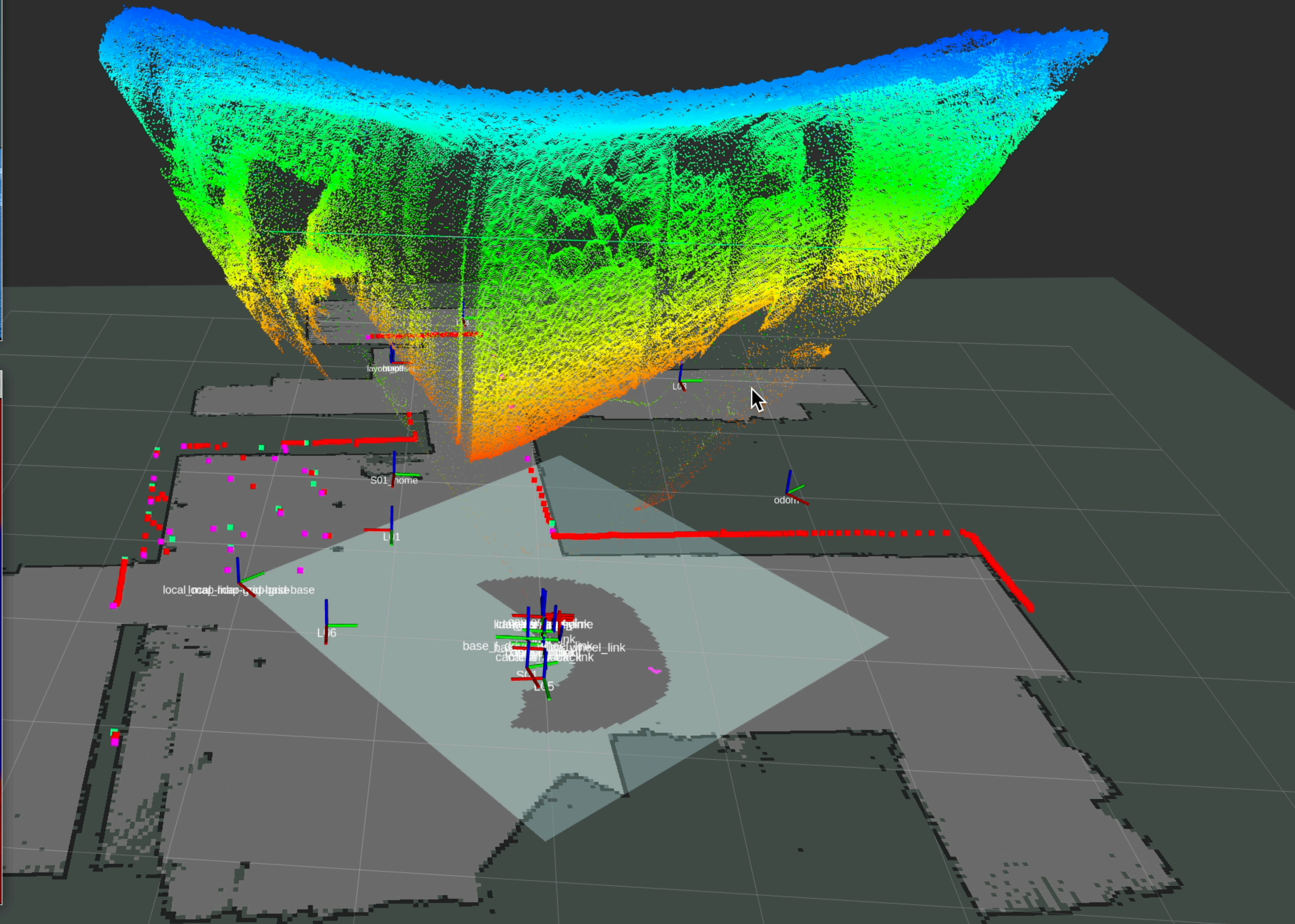
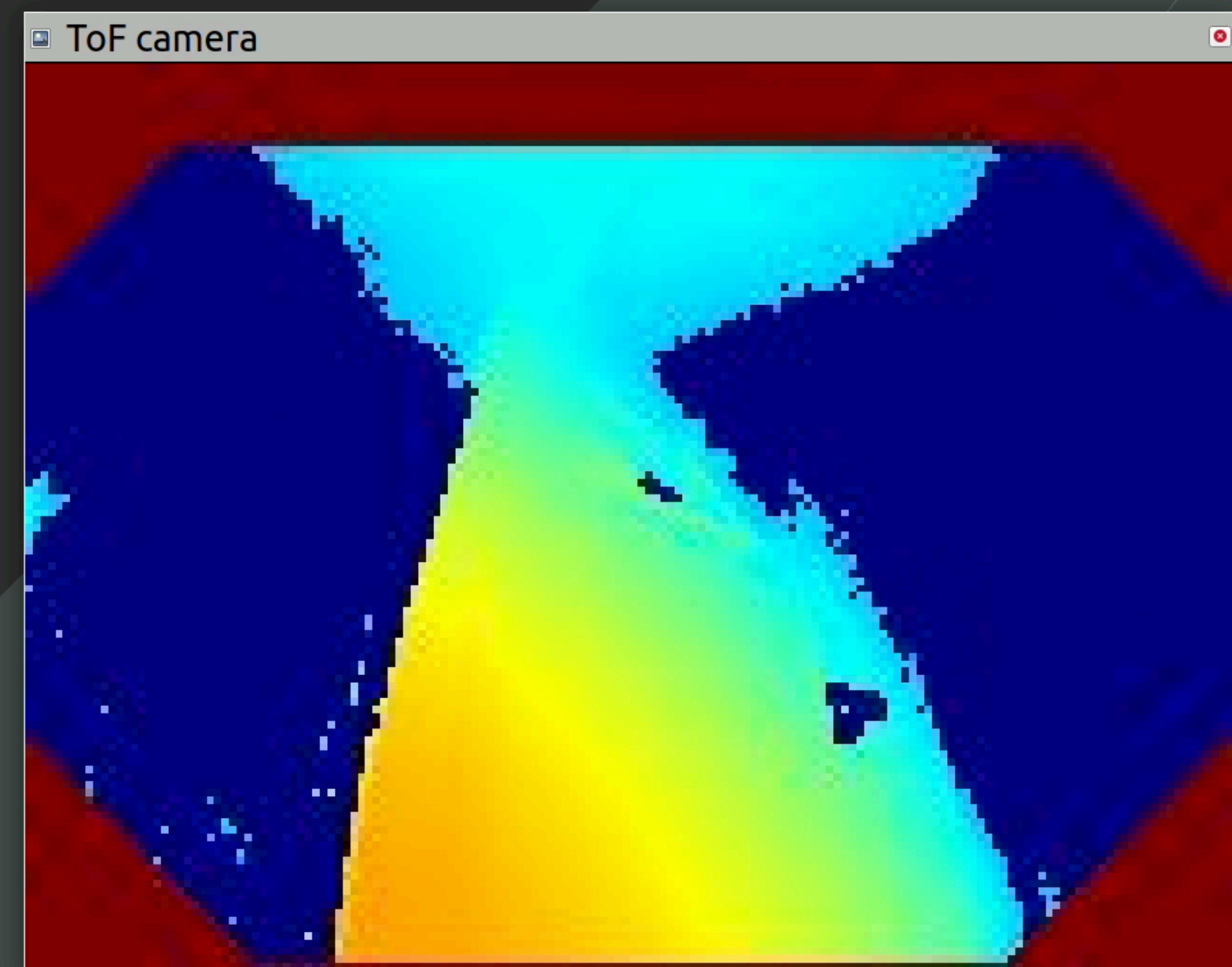
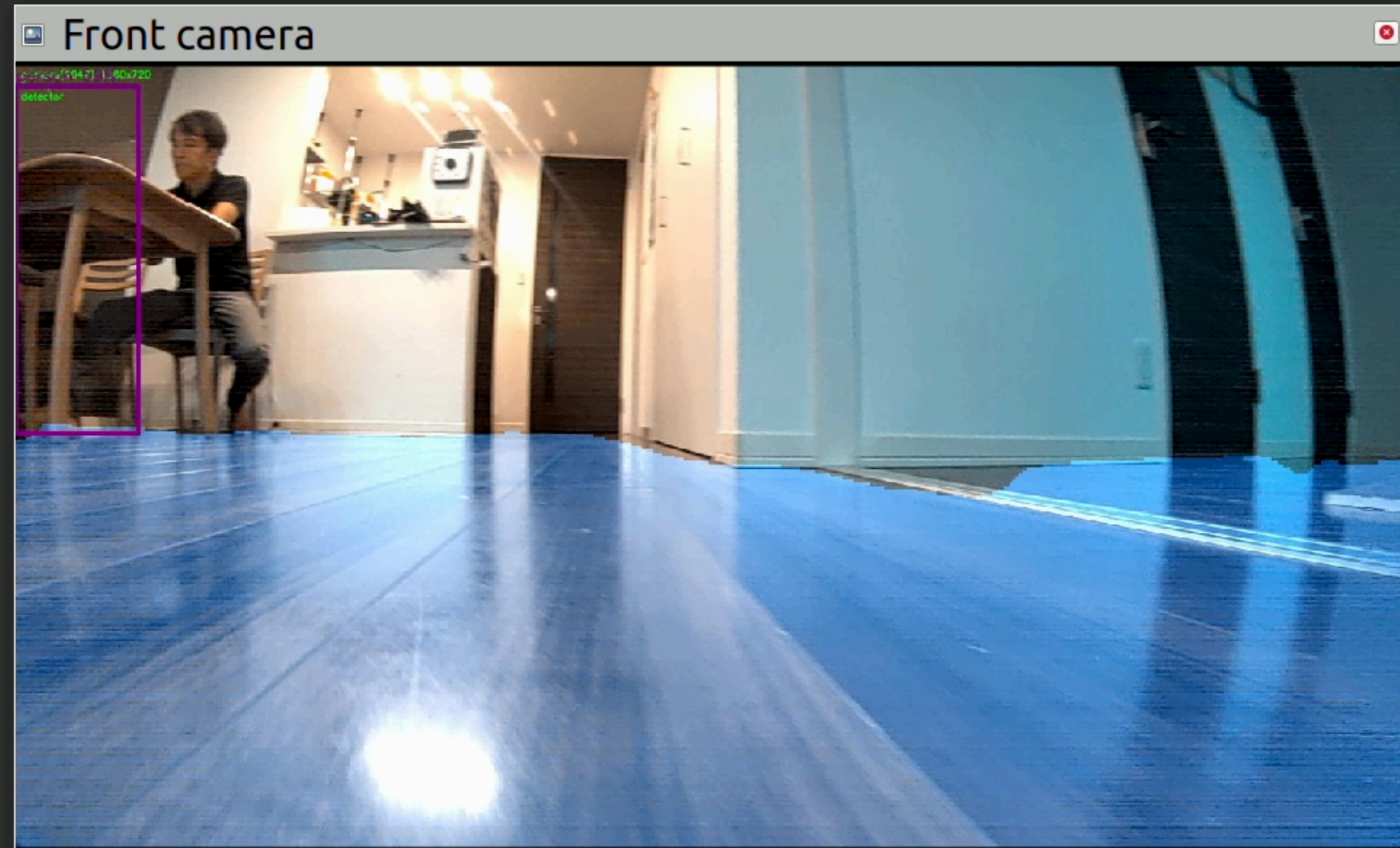
- <https://github.com/ros-planning/navigation2> をリモートブレインで実行可能
- センサー情報はLiDARのみ
- カチャカAPIでは未提供のPathやInflation Map表示に対応



カチャカソフトウェア本気のナビゲーション

- 性能自体はカチャカAPI経由とほぼ同様 (ブリッジに伴うオーバーヘッドなし)
- カチャカAPIでは未提供の Semantic Segmentation、ToFが使用可能
- ToF APIはプライバシー問題を解決後、提供予定







カチャカAPI

ROS 2ブリッジTips

CompressedImageの利用

Tips

- 非圧縮画像トピックはHzが低いが、JPEG画像トピックはHzが高い

```
$ ros2 topic hz /kachaka/front_camera/  
image_raw
```

```
average rate: 1.961
```

```
$ ros2 topic hz /kachaka/front_camera/  
image_raw/compressed
```

```
average rate: 8.672
```

- remapすればRViz2などでも描画可能

```
$ rviz2 --ros-args --remap  
/kachaka/front_camera/image_raw/camera_info:=  
/kachaka/front_camera/camera_info
```

Image



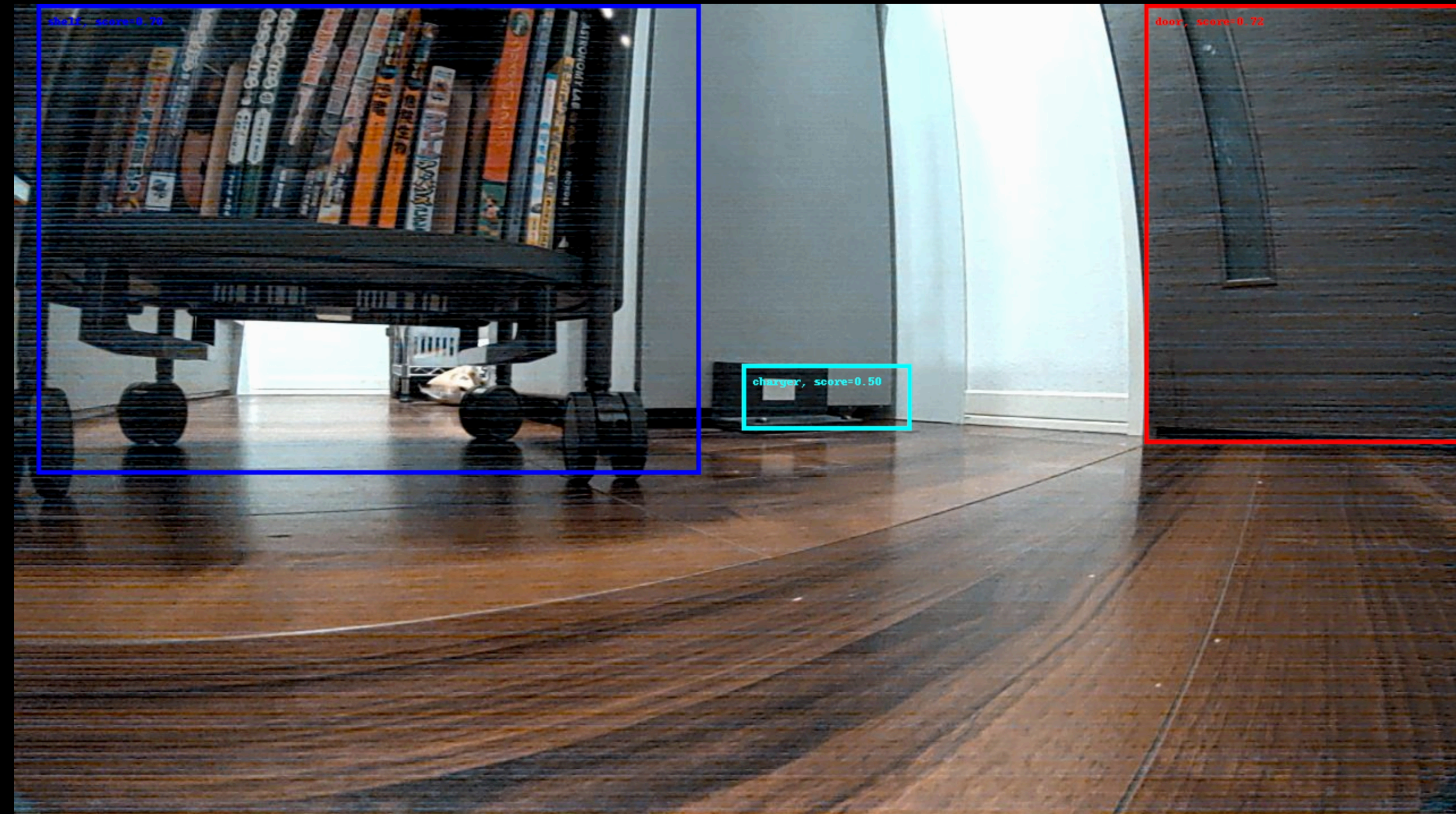
CompressedImage



物体認識アプリケーション

Tips

- 独自の学習データセットなので人は足だけでも認識可能
- カチャカシエルフはカバー有無、積載有無に限らず認識可能
- Network Architecture Searchによる限界までの軽量化
- 人追従、家具自動発見などに応用可能

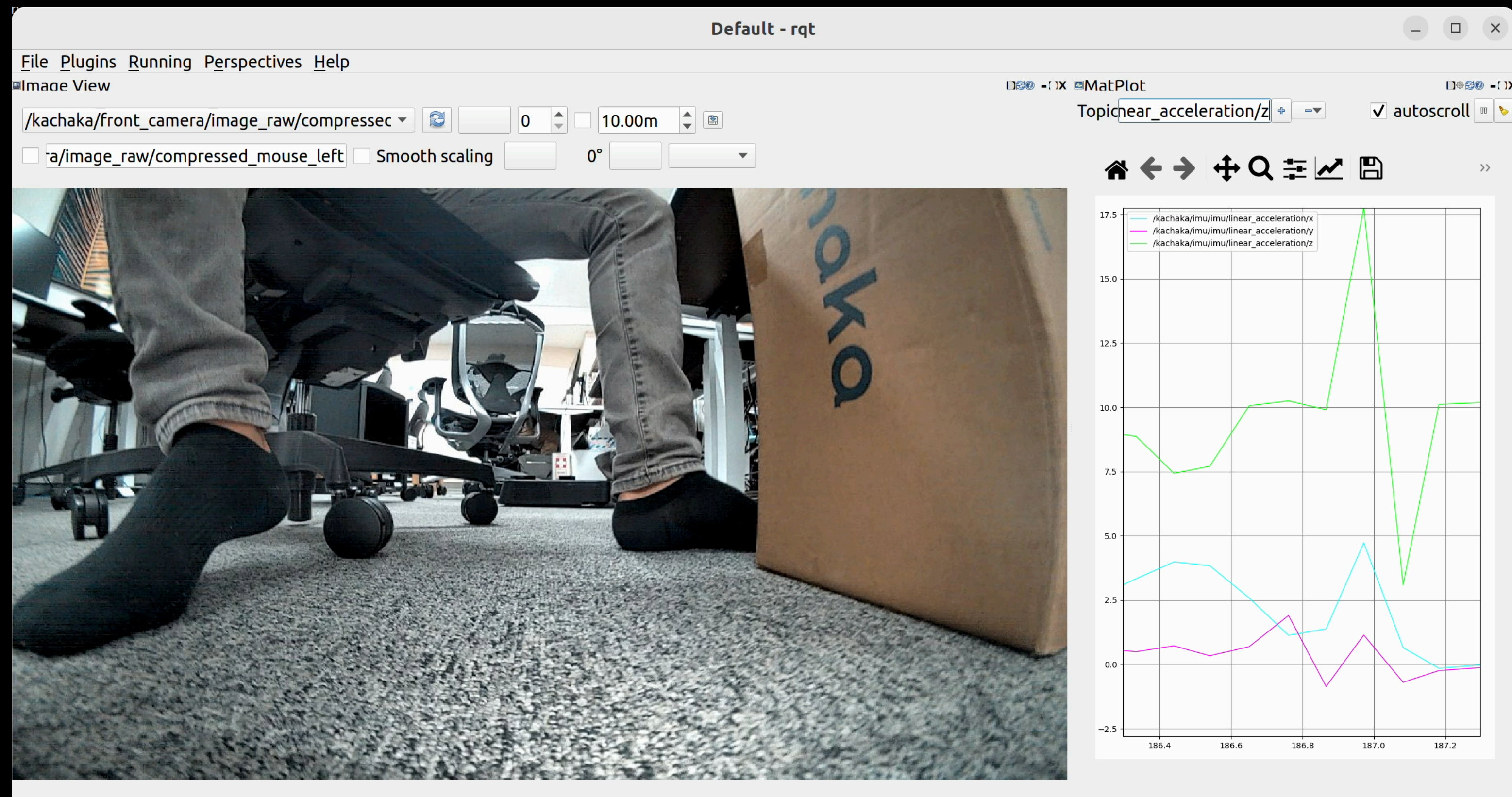


人 家具 充電器 扉

IMUアプリケーション

Tips

- XYZ軸の並進加速度とZ軸の回転速度を出力
- キッドナップ検出、転倒検出などに応用可能



センサートピックのQoS設定

Tips

- カチャカのセンサートピックはすべてrclcpp::SensorDataQoSを使用
- ReliabilityをBest Effortに設定しないと、トピックをSubできないので注意

▼  Camera	☒
▶ ✓ Status: Ok	
▶ 👁 Visibility	☒
▼ Topic	/kachaka/front_camera/ima...
Depth	5
History Policy	Keep Last
Reliability Policy	Best Effort
Durability Policy	Volatile
Image Rendering	background and overlay
Overlay Alpha	1
Zoom Factor	1
Far Plane Distance	100

cmd_velの安全機能

Tips

- 前カメラのSemantic Segmentation、LiDAR、ToFによる非常停止機能が有効
- 後退は安全機能がない（ぶつかる）
- 自動充電機能の無効化は可能

```
$ ros2 service call /kachaka/auto_homing/set_enabled std_srvs/srv/SetBool "data: true"
```

- RemapすればROS 2標準パッケージとも相性良し

```
$ ros2 run teleop_twist_keyboard teleop_twist_keyboard --ros-args --remap  
  /cmd_vel:=/kachaka/manual_control/cmd_vel
```


kachaka_grpc_ros2_bridgeのローカル実行

Tips

- ros2_bridgeコンテナを使わず、ローカル実行することも可能

```
$ cd ~/kachaka-api/ros2/kachaka_grpc_ros2_bridge
$ protoc -I ~/kachaka-api/protos --grpc_out gen-src
    --plugin=protoc-gen-grpc=/usr/bin/grpc_cpp_plugin
    --cpp_out=gen-src ~/src/kachaka-api/protos/kachaka-api.proto
$ mkdir -p ~/kachaka-api-ros/src
$ ln -s ~/kachaka-api/ros2/kachaka_* ~/kachaka-api-ros/src/
$ cd ~/kachaka-api-ros
$ rosdep install --from-paths src --ignore-src -r -y
$ colcon build
$ source ~/kachaka-api-ros/install/setup.zsh
$ ros2 launch kachaka_grpc_ros2_bridge grpc_ros2_bridge.launch.xml
    server_uri:=$KACHAKA_IP:26400
```

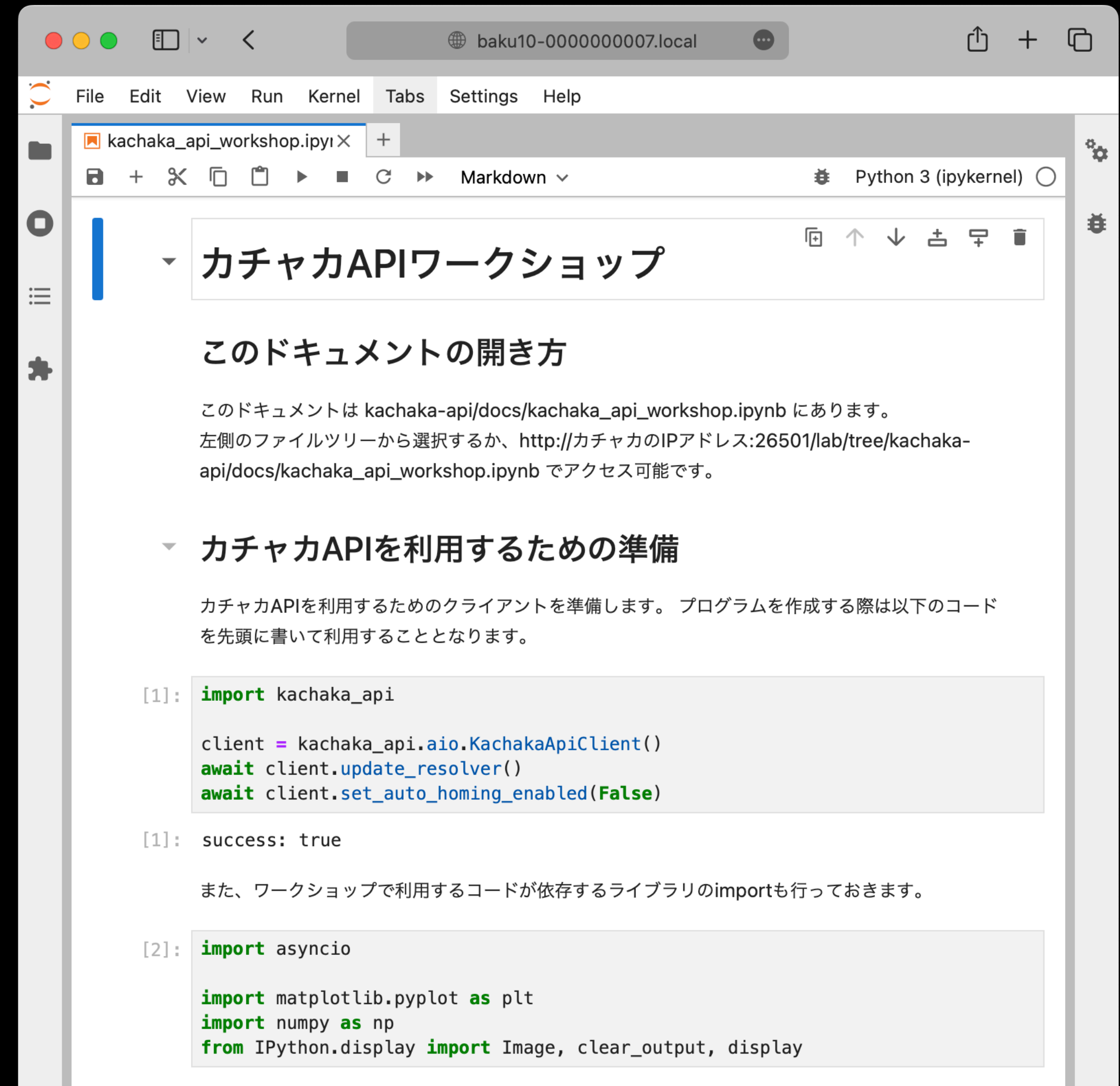
- <https://github.com/pf-robotics/kachaka-api/blob/main/Dockerfile.ros2> を参考に

カチャカAPIワークショップ資料

Tips

https://github.com/pf-robotics/kachaka-api/blob/main/docs/kachaka_api_workshop.ipynb

- 9/15に開催されたカチャカAPIワークショップの講義資料
- 15分あれば、ご自宅のカチャカでAPIワークショップの個人学習が可能



弊社エンジニアによるQiita記事

Tips

<https://qiita.com/organizations/probotics>

外部APIの利用例: 天気に応じて動かす、スクリプトの常時実行

次に、`move_shelf` APIを使って、便利な機能を実装してみます。ある時点の天気を取得して、それに応じて動作を変えます。晴れの場合は観葉植物を載せている家具を窓際に移動させ、雨の場合は観葉植物を載せている家具をダイニングに移動させます。

天気を取得するには、[OpenWeatherMap](#)のAPIを使います。[sign_up](#)からアカウントを作成し、[api_keys](#)でAPIキーを取得する必要があります。以下のコードの `API_KEY=""`の部分には取得したAPIキーを入力してください。`ZIP_CODE=""`には、天気を取得したい地域の日本の郵便番号をXXX-XXXXの形式で入力してください。

```
import datetime
import json
import requests
import time
from typing import Tuple

import kachaka_api

client = kachaka_api.KachakaApiClient()
client.update_resolver()

API_KEY = ""
ZIP_CODE = ""
TRIGGER_HOUR = 12

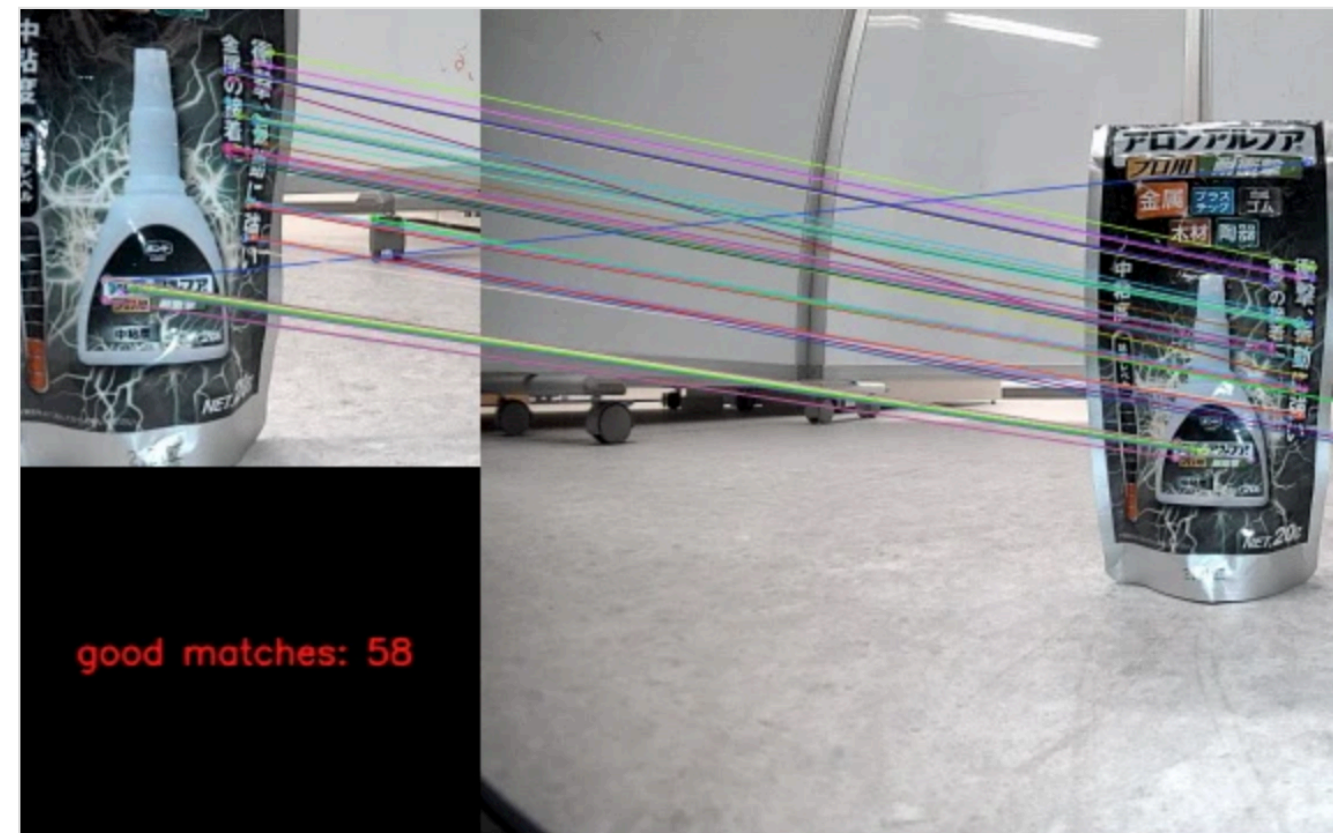
client.speak("今日もいい天気だったらいいなー")

def get_weather_and_description() -> Tuple[str, str]:
    response = requests.get(f"https://api.openweathermap.org/data/2.5/weather")
    ret = json.loads(response.text)
    weather = ret["weather"][0]
    return weather["main"], weather["description"]
```

撮像して....



ぐねぐね曲げたりしなければ認識できていますね。



起動

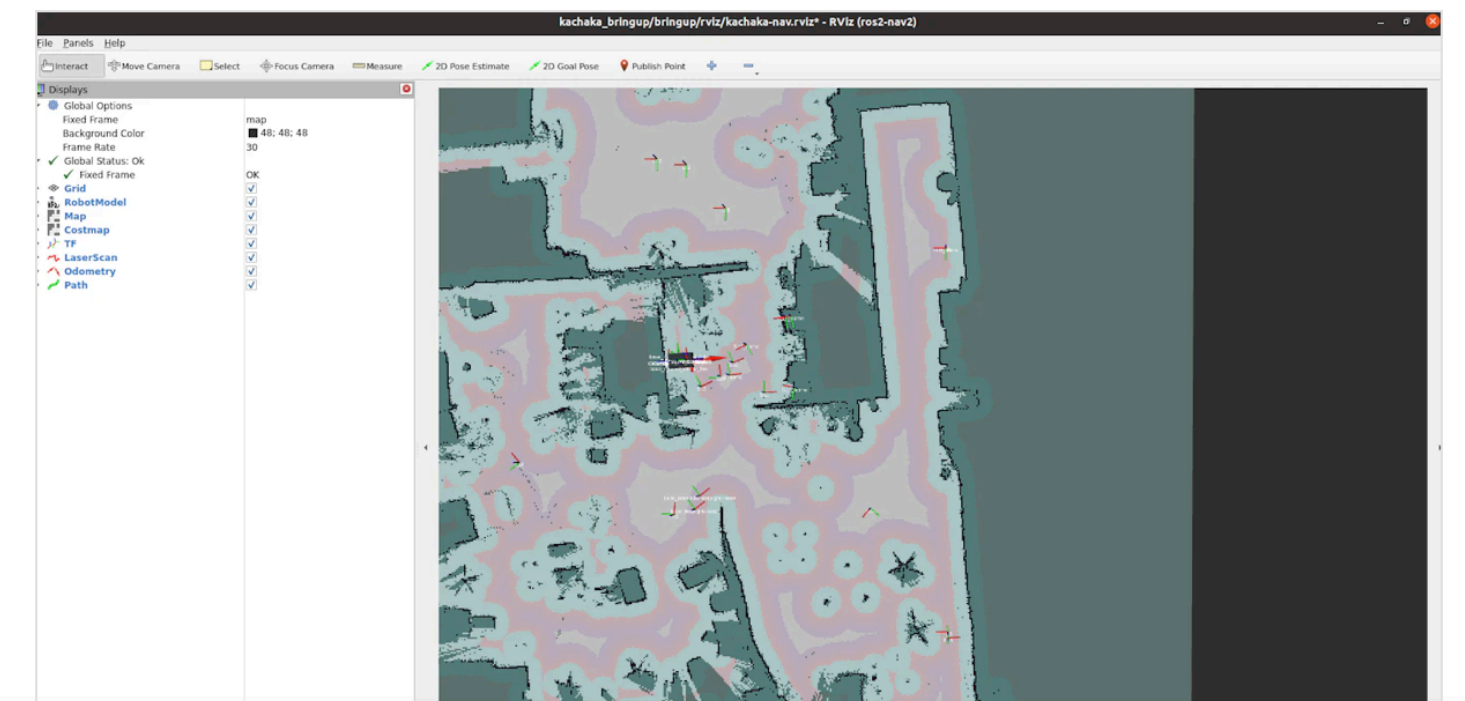
以下のコマンドでnavigationを起動します。上述のros2grpcブリッジが起動している必要があります。

```
source install/setup.bash
ros2 launch kachaka_bringup navigation_launch.py
```

可視化

rvizで可視化します。表示されているのはPreferred Roboticsのオフィスになります。

```
source install/setup.bash
rviz2 -d install/kachaka_bringup/share/kachaka_bringup/rviz/kachaka-nav.rviz
```

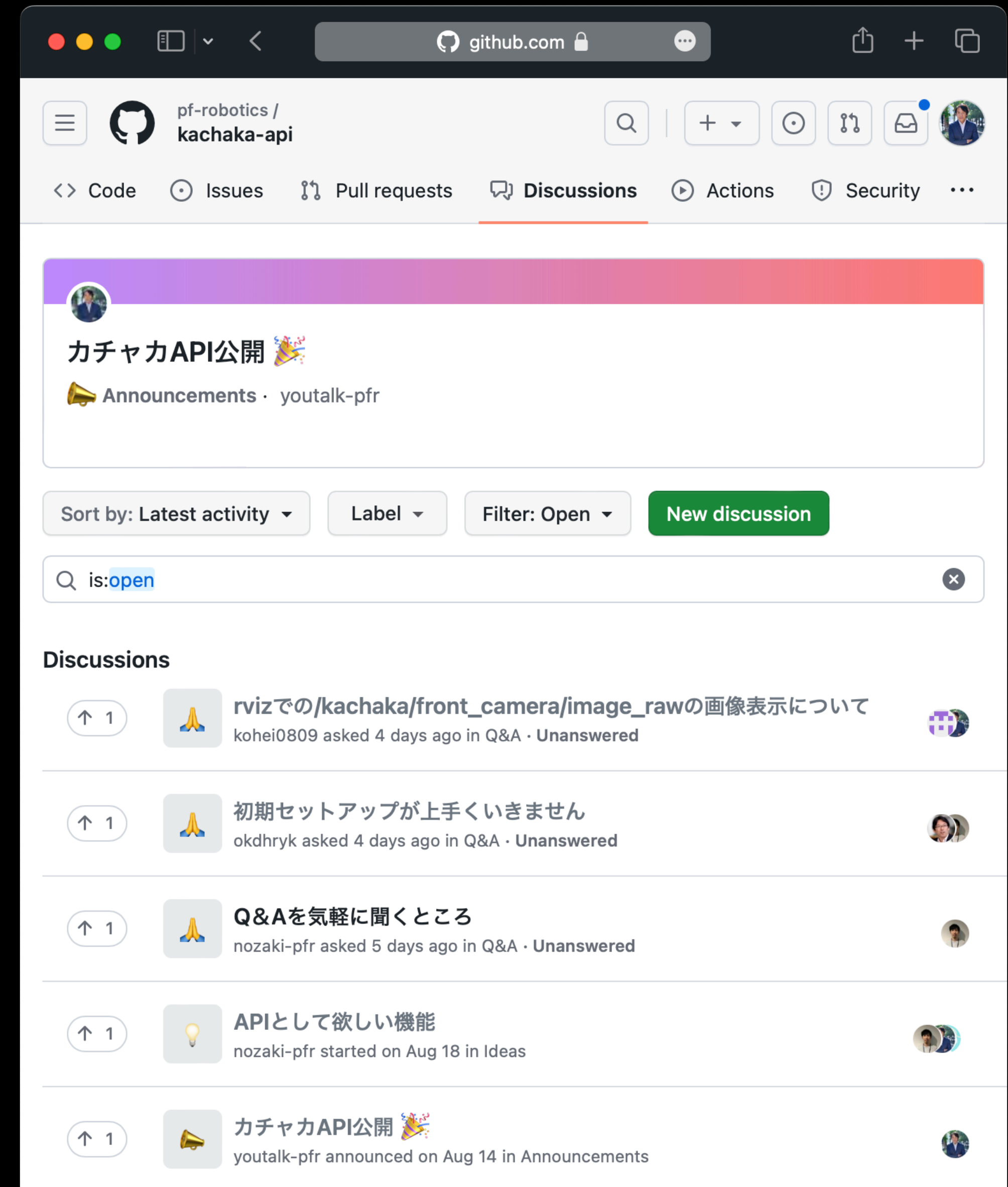


GitHub Discussionsの活用

Tips

- カチャカAPIはあくまでコミュニティーサポート
- GitHub Discussionsでの討論、GitHub Issuesでの問題提起、GitHub PRによるOSS貢献、お待ちしております

<https://github.com/pf-robotics/kachaka-api/discussions>





いちにさん | ロボット系SE

@ko1_baseba11

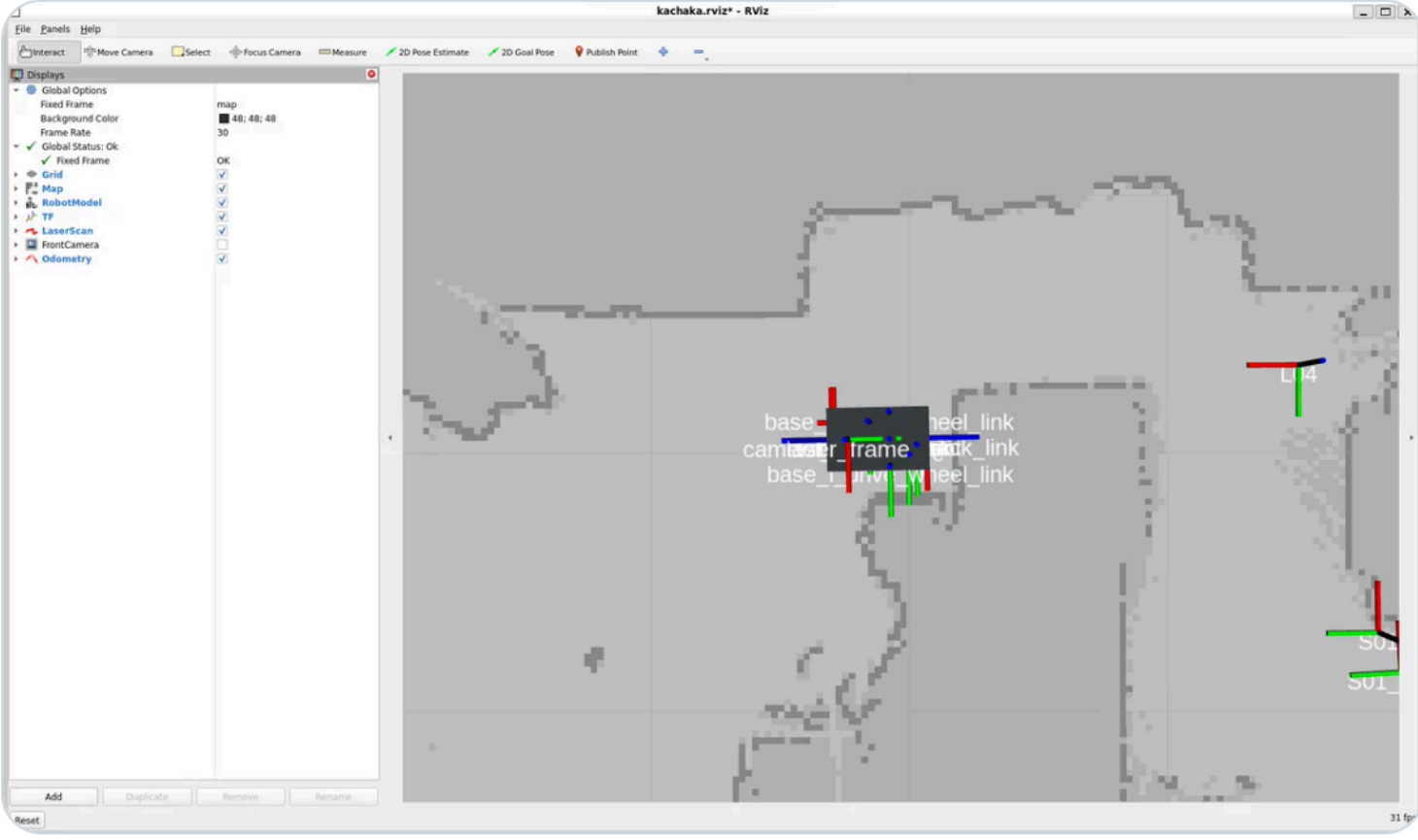
https://x.com/ko1_baseba11/status/1692509808845549620?s=20

より転載

...

Windows上のWSL2でも動くこと確認できました!!! #ねえカチャカ

Translate post





Daijiro Mori

@daijiro

https://x.com/daijiro/status/1699307339839582379?s=20

より転載

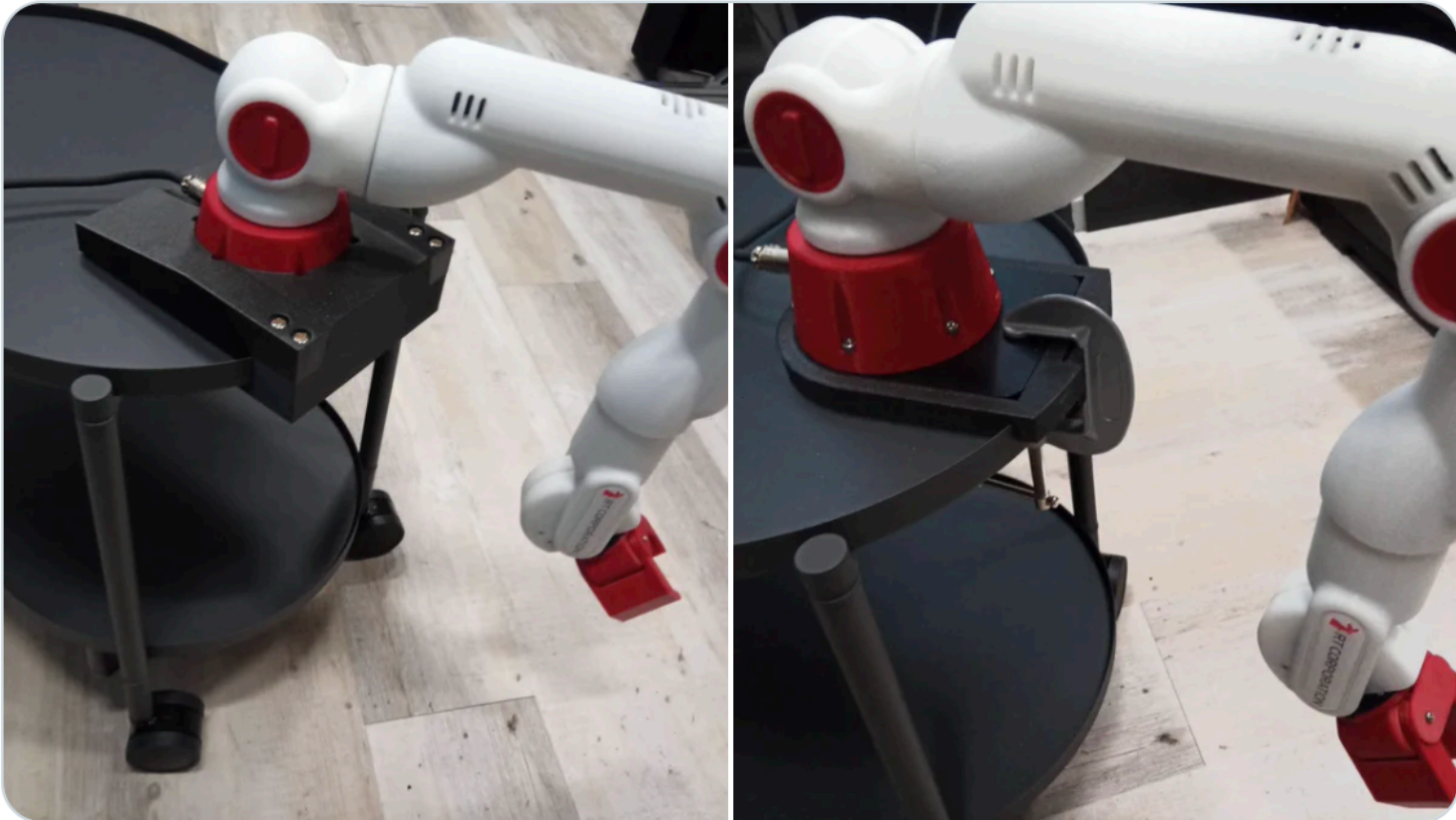
...

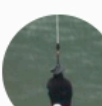
CRANE-X7標準のクランプはシェルフの縁に干渉してちょっとマウントしづらいですね。
そこでぴったりフィットする専用の治具を作りました! 良かったらご活用下さい

[github.com/reason-research...](https://github.com/reason-research/ouxt_robot)

#カチャカ

Translate post





龍

@_ame_a_

https://x.com/_ame_a_/status/1703417718018625814?s=20

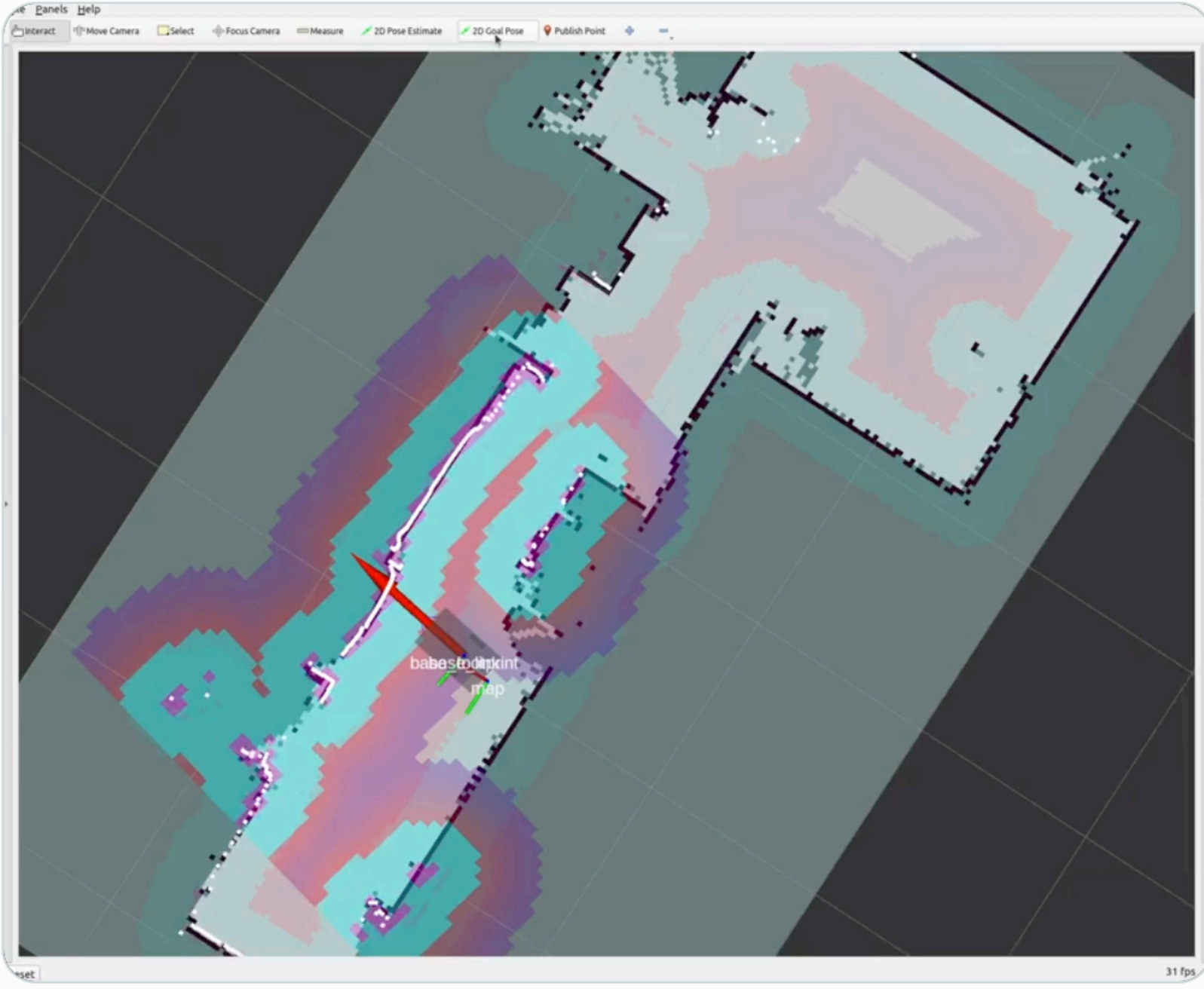
より転載

...

navigation 2 + emcl2

リモートPC側でも快適に動くので割と自作navigation普通にざりざり動かせそうな感覚

Translate post





片岡大哉

@hakuturu583

https://x.com/hakuturu583/status/1681886045212909568?s=20

より転載

...

自宅でインテグしておきます 🙏

Translate post



Shinpei KATO (加藤真平)

@ShinpeiKato

Jul 20

カチャカとAutoware、つなげちゃお! [twitter.com/hakuturu583/st...](https://twitter.com/hakuturu583/status/1681886045212909568)



龍

@_ame_a_

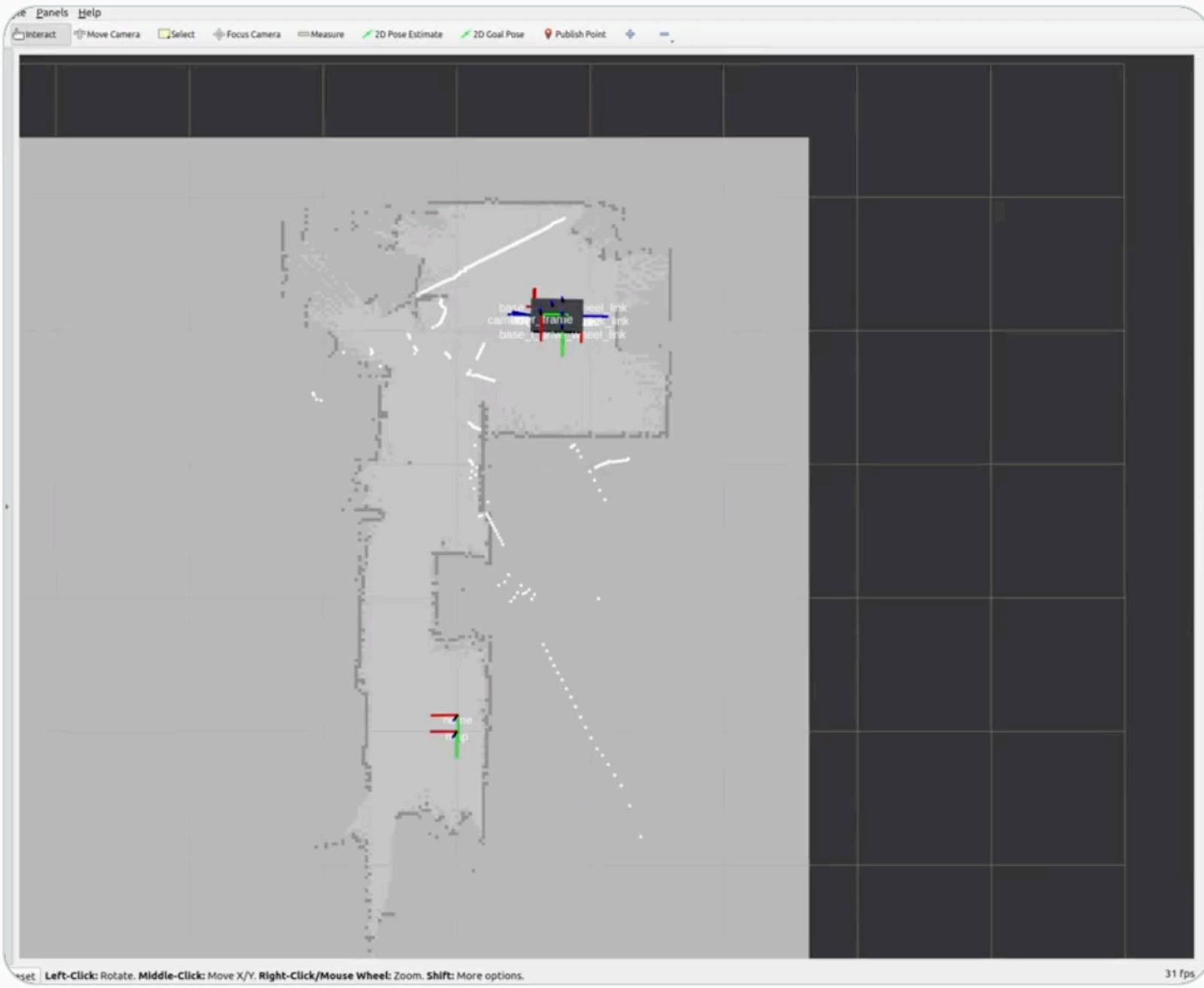
https://x.com/_ame_a_/status/1692888644951163357?s=20

より転載

...

位置がずれた状態で「落ち着いて」って言うと自己位置修正してくれました...すごい

Translate post



taki4416

@taki_4416

Aug 19

Replying to @_ame_a_

「落ち着いて」試してなかったらぜひ kachaka.zendesk.com/hc/ja/articles...



片岡大哉

@hakuturu583

https://x.com/hakuturu583/status/1702158560803828034?s=20

より転載

...

うちのカチャカでOUXTのスタック、ハードウェアCIしましょうw

Translate post

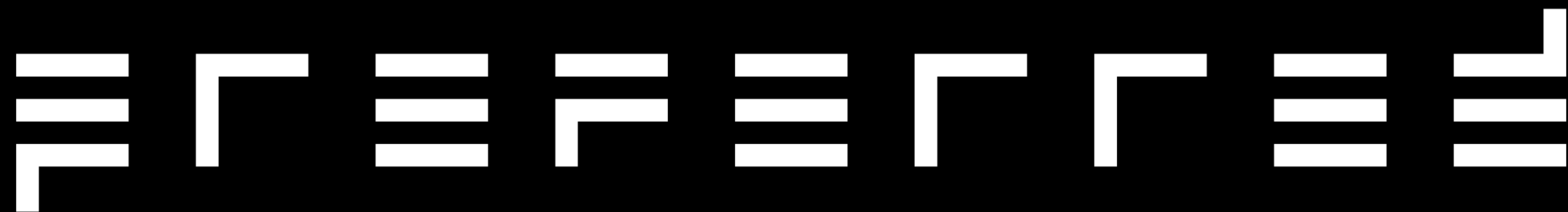


るっと

@MeRTcooking

Sep 14

読みます 🐦 [twitter.com/tera_koji/stat...](https://twitter.com/tera_koji/status/1702158560803828034)



Preferred Robotics

各種ECサイトで絶賛発売中

kachaka
Online Store

<https://store.kachaka.life/>

公式サイト



RT CORPORATION

<https://rt-net.jp/service/kachaka/>

公費・校費払い
有料サポート

amazon

<https://www.amazon.co.jp/stores/Kachaka/page/194F8B15-D2F4-46CE-B591-F722A5D0B2F7>

翌日配送

Amazonポイント

Rakuten

<https://www.rakuten.co.jp/kachaka/>

楽天経済圏
楽天ポイント

法人向け本格導入開始

- カチャカが医療・福祉、飲食業界などを中心に、事業の現場においても活用が進んでいることを受け、法人向けサイトをオープン
- <https://kachaka.life/business/>



kachaka 法人向け

カチャカラウンジ

- 丸の内テラス内にある会員制シェアスペースOCA TOKYOにカチャカのショールームがオープン
- <https://prtimes.jp/main/html/rd/p/000000012.000115855.html>
- <https://note.kachaka.life/n/nde611aa906fa>



HAPiIBOT



- アマノ様と共同開発した小型床洗浄ロボット
- Preferred Roboticsがナビゲーションシステム、AIビジョン、エッジCPUボードなどを開発

スーパーマーケット



ファクトリー

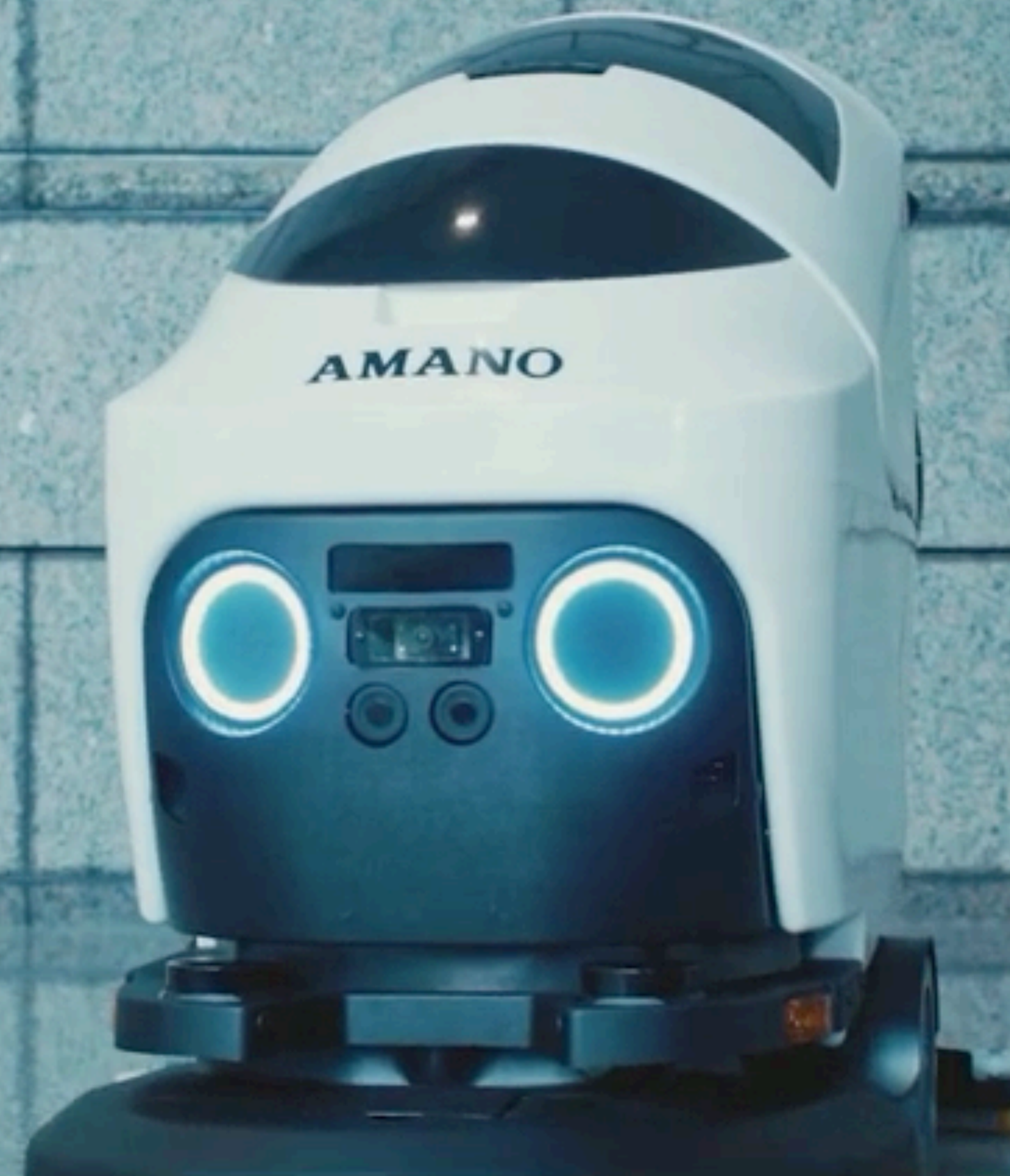


AMANO

×



Preferred Robotics



プリファードロボティクスと共同開発

Cursor

- gRPCにはない独自の仕組み
- cursor比較によるデータ取得の既読管理と
ロングポーリングを導入
- cursor = 0のとき、その時点の最新値を受信
- cursor = Xのとき、サーバ側がcursor != Xに
なるまで待機し、データを受信
- 重複受信、再送、最新データのみ受信など
の課題に対応

```
message Metadata {
    sfixed64 cursor = 1;
}

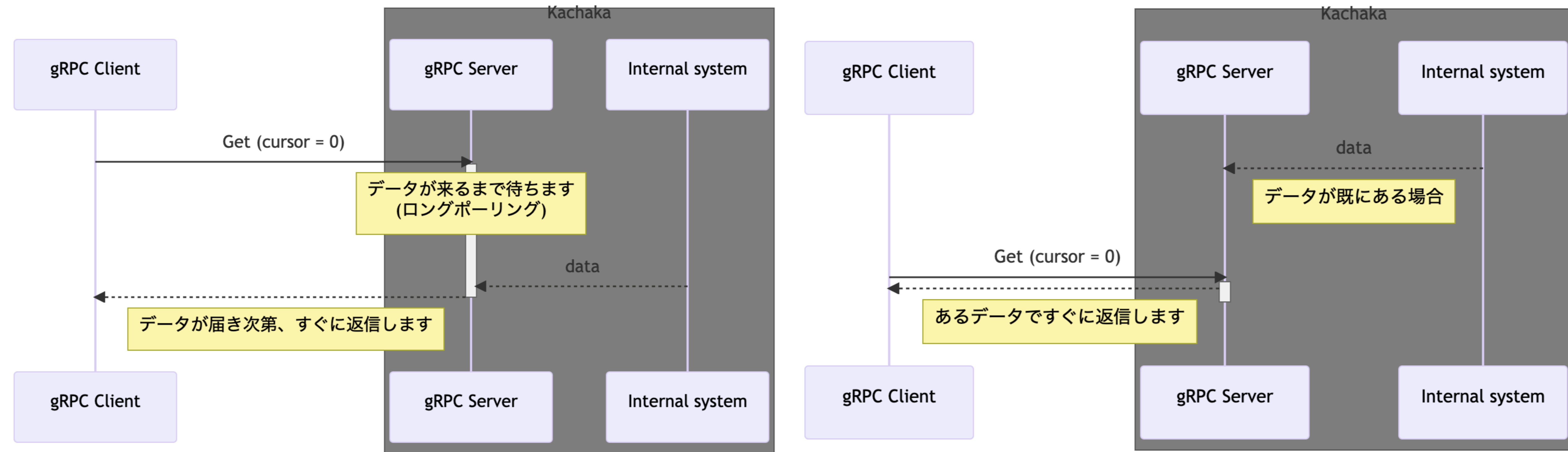
message GetRequest {
    Metadata metadata = 1;
}

message GetRobotSerialNumberResponse {
    Metadata metadata = 1;
    string serial_number = 2;
}

service KachakaApi {
    rpc GetRobotSerialNumber (GetRequest)
        returns (GetRobotSerialNumberResponse);
}
```

<https://github.com/pf-robotics/kachaka-api/blob/main/protos/kachaka-api.proto> 抜粋

cursor = 0のケース



cursor != 0のケース

