

# Lokalizacja Visual Inertial Odometry przykład użycia OpenVINS

Eryk Możdżeń

Koło Naukowe Robotyków KoNaR

8.10.2025

# Plan prezentacji

- 1 Wstęp
- 2 Visual Inertial Odometry
- 3 Dobór sprzętu
- 4 Oprogramowanie OpenVINS

# Motywacja

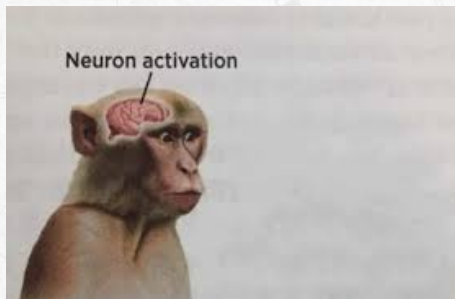
Sposoby lokalizacji w robotyce:

- enkodery na kołach
- GPS
- GPS RTK
- LIDAR + SLAM
- optical-flow
- kombinacja powyższych przez EKF
- ...

Wiele z nich są zliczeniowe i podatne na błędy, działa w specyficznych warunkach lub są duże i drogie.

# Jak robią to ludzie?

Ludzie mają oczy.  
Ludzie mają błędnik.



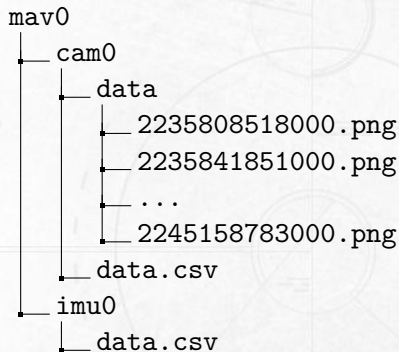
# Okulary VR



# Czym jest VIO

- znajdowanie ficzerów na obrazie (+ deskryptory): FAST, Harris, ORB
- śledzenie ficzerów: Lucas-Kanade
- EKF predykcja: IMU
- EKF korekcja: "magiczne" przekształcenie ficzerów na geometryczne ograniczenia

# Datasety



Przetestowane na własnej skórze:

- Visual-Inertial Dataset (TUM)
- The EuRoC MAV Dataset (ETH Zurich)

# Datsety

mav0/cam0/data.csv

timestamp [ns]	filename
2235808518000	2235808518000.png
2235841851000	2235841851000.png
2235875180000	2235875180000.png
2235908520000	2235908520000.png
2235958521000	2235958521000.png
2236008523000	2236008523000.png
2236058524000	2236058524000.png
2236108525000	2236108525000.png
2236158527000	2236158527000.png
2236208529000	2236208529000.png
2236258524000	2236258524000.png
2236308531000	2236308531000.png
2236358532000	2236358532000.png
2236408524000	2236408524000.png



# Datasety

mav0/imu0/data.csv

timestamp [ns]	$w_R S_{S_x} [rads^{-1}]$	$w_R S_{S_y} [rads^{-1}]$	$w_R S_{S_z} [rads^{-1}]$	$a_R$
2235150904471	-0.155081	0.0650168	0.134031	-9.3
2235155901735	-0.13856	0.0652833	0.147087	-9.3
2235160895646	-0.122573	0.0746095	0.145489	-9.3
2235165894112	-0.106052	0.0897979	0.130034	-9.3
2235170888079	-0.0889985	0.10845	0.10472	-9.3
2235175881305	-0.0730107	0.128168	0.0724778	-9.3
2235180879679	-0.0626187	0.145755	0.0421011	-9.3
2235185878757	-0.0543583	0.164141	0.0135896	-9.3
2235190870538	-0.0500949	0.181994	-0.0130567	-9.3
2235195867894	-0.0498285	0.196649	-0.0389035	-9.4
2235200863972	-0.0516937	0.205176	-0.0618193	-9.4
2235205876495	-0.0599541	0.210239	-0.0719449	-9.6
2235210852627	-0.0660827	0.2177	-0.0644820	-9.6

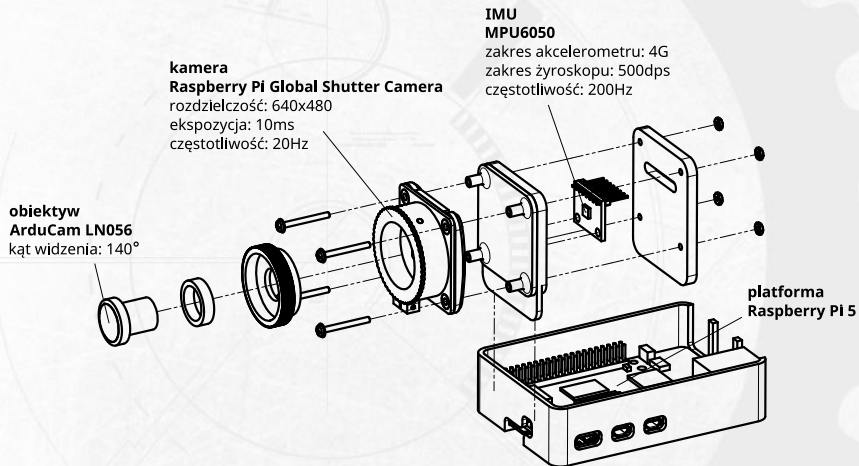
# Jakie IMU?

- akcelerometr (XYZ) i żyroskop (XYZ)
- zwykły tani MEMS wystarczy
- pomiar ze **stałą** częstotliwością: 100Hz, 200Hz, 500Hz

# Jaka kamera?

- niska rozdzielczość: 512x512, 640x480, 720x480, 1024x512
- niska **stała** częstotliwość migawki: 20Hz, 30Hz
- obraz w odcieniach szarości
- krótki czas ekspozycji
- migawka typu **global shutter** (\$\$\$)
- obiektyw szerokokątny +120° (**rybie oko**)
- można mieć jedną
- można mieć dwie (**stereowizja**)

## Mój setup



# Nietrafione zakupy i problemy

- "zasilacz z czarnej listy"
- cross-kompilacja (problem z zależnościami)
- podrobiony MPU6050 (zła wartość WHO\_I\_AM)
- obraz na żywo (laguje fest)
- kamera OV5647 (brak global shutter)
- tasiemki do kamer (są w zestawie)
- regulacja ostrości obiektywu



# Co to OpenVINS

- OpenCV + MSCKF + SLAM
- rozbudowana konfiguracja
- łatwy w użyciu
- wspiera setup z jedną i wieloma kamerami
- kompensacja bias-ów IMU
- ignorowanie wskazanych pól na obrazie
- słaba dokumentacja
- stare wersje zależności ze źródeł

# Jak zrobić żeby działało?

- 1 Otrzymywać dane z IMU (częstotliwość, timestamp)
- 2 Otrzymywać klatki z kamery (częstotliwość, timestamp)
- 3 Skompilować OpenVINS
- 4 Ustalić transformacje kamera-IMU
- 5 Skalibrować kamerę (distortion + intrinsic)

# LIVE DEMO

