

# Zadanie do zrealizowania

---

Celem projektu było stworzenie węzła ROS umożliwiającego badanie błędów dla różnych źródeł odometrii.

Etapy projektu:

- Uruchomienie symulacji z `tune_controller`,
- Uruchomienie symulacji z `diff_drive_controller`,
- Poprawa lokalizacji względnej wykorzystując dodatkowe czujniki (pakiet `Laser_scan_matcher`)

Badanie błędu dla każdego z powyższych pomiarów odometrii przeprowadzone zostało poprzez:

- Wykonanie ruchu w przód i w tył o długości 1m,
- Wykonanie pełnego obrotu,
- Przejazd po kwadracie o boku 1m.

Błędy bezwzględne położenia dla testu linii oraz kwadratu wyznaczone zostały poprzez obliczenie odległości pomiędzy położeniem publikowanym na tematach `/gazebo_odom` i

`/elektron/mobile_base_controller/odom` (`/pose2D` w przypadku korzystania z czujnika laserowego).

Błąd ten liczony jest jako pierwiastek sumy kwadratów różnicy poszczególnych współrzędnych robota.

Natomiast błąd bezwzględny orientacji dla testu obrotu wyznaczany został jako wartość bezwzględna najmniejszego kąta pomiędzy orientacjami robota publikowanymi na tematach `/gazebo_odom` i

`/elektron/mobile_base_controller/odom` (`/pose2D` w przypadku korzystania z czujnika laserowego).

Uwaga! Zaszumienie wykresów (widoczne zwłaszcza dla małych błędów) w znacznym stopniu wynika z faktu odbierania przez węzeł położeń z różnych tematów publikowanych w różnych odstępach czasu (nie jest możliwe odczytanie położień z obydwu tematów w tym samym momencie). Jest to szczególnie widoczne w sytuacji gdy wiadomości na nasłuchiwanym tematach pojawiają się z różną częstotliwością.

## Uruchamianie

---

Przed uruchamianiem komend w nowych oknach terminala należy wpisać:

```
source ~/ws_mobile-devel/setup.bash
```

gdzie `ws_mobile` to nazwa utworzonego katalogu roboczego.

W celu przeprowadzenia poszczególnych testów należy wpisać następujące komendy.

W pierwszym oknie terminala:

```
roslaunch stereo_mobile_init elektron_world.launch - inicjalizacja środowiska z  
diff_drive_controller,
```

lub:

```
roslaunch stereo_mobile_init tune_controller.launch - inicjalizacja środowiska z tune_controller,
```

W kolejnym oknie w celu przeprowadzenia odpowiedniego testu wpisujemy:

- test linii:

```
rosrun stero_mobile_init test_lini.py
```

- test obrotu:

```
rosrun stero_mobile_init test_lini.py
```

- test kwadratu:

```
rosrun stero_mobile_init test_kwadratu.py
```

W celu poprawy lokalizacji względnej z wykorzystaniem pakietu `laser_scan_matcher` w kolejnym oknie terminala uruchamiamy dodatkowo:

`roslaunch stero_mobile_init laser_no_odom.launch` - wyznaczanie lokalizacji robota tylko i wyłącznie za pomocą czujnika laserowego

lub:

`roslaunch stero_mobile_init laser_odom.launch` - wyznaczanie lokalizacji robota z wykorzystaniem czujnika laserowego oraz `tune_controller` lub `diff_drive_controller` (w zależności od tego, za pomocą której komendy zainicjalizowane zostało środowisko).

Przy wykorzystaniu pakietu `laser_scan_matcher` odpowiednie testy uruchamiamy za pomocą następujących komend:

- test linii:

```
rosrun stero_mobile_init LaserTL.py
```

- test obrotu:

```
rosrun stero_mobile_init LaserTO.py
```

- test kwadratu:

```
rosrun stero_mobile_init LaserTK.py
```

Jeśli chcemy nagrywać trasę robota, w kolejnym terminalu uruchamiamy:

```
rosbag record -o nazwa_pliku.bag /topic1 /topic2
```

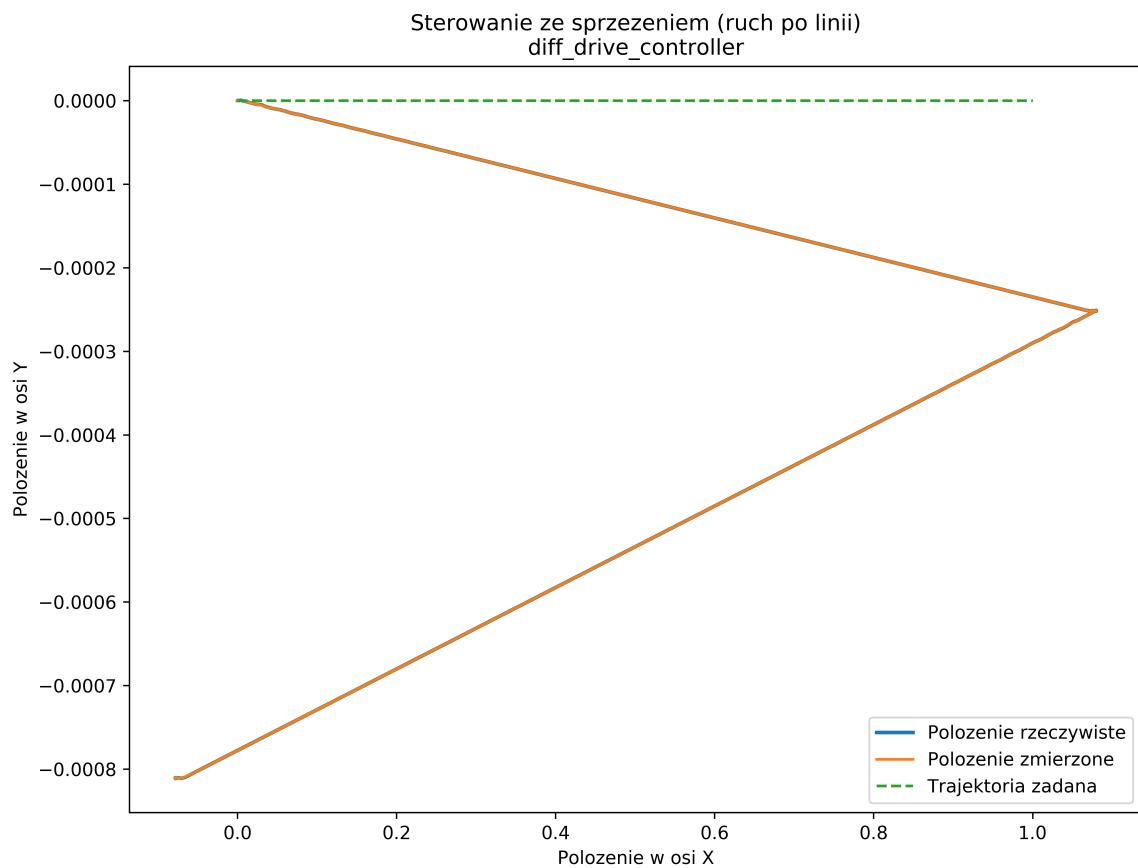
Domyślnie wszystkie węzły automatycznie zapisują położenia oraz błędy do plików bag.

## Przedstawienie uzyskanych trajektorii oraz porównanie wyznaczonych błędów

### Test linii

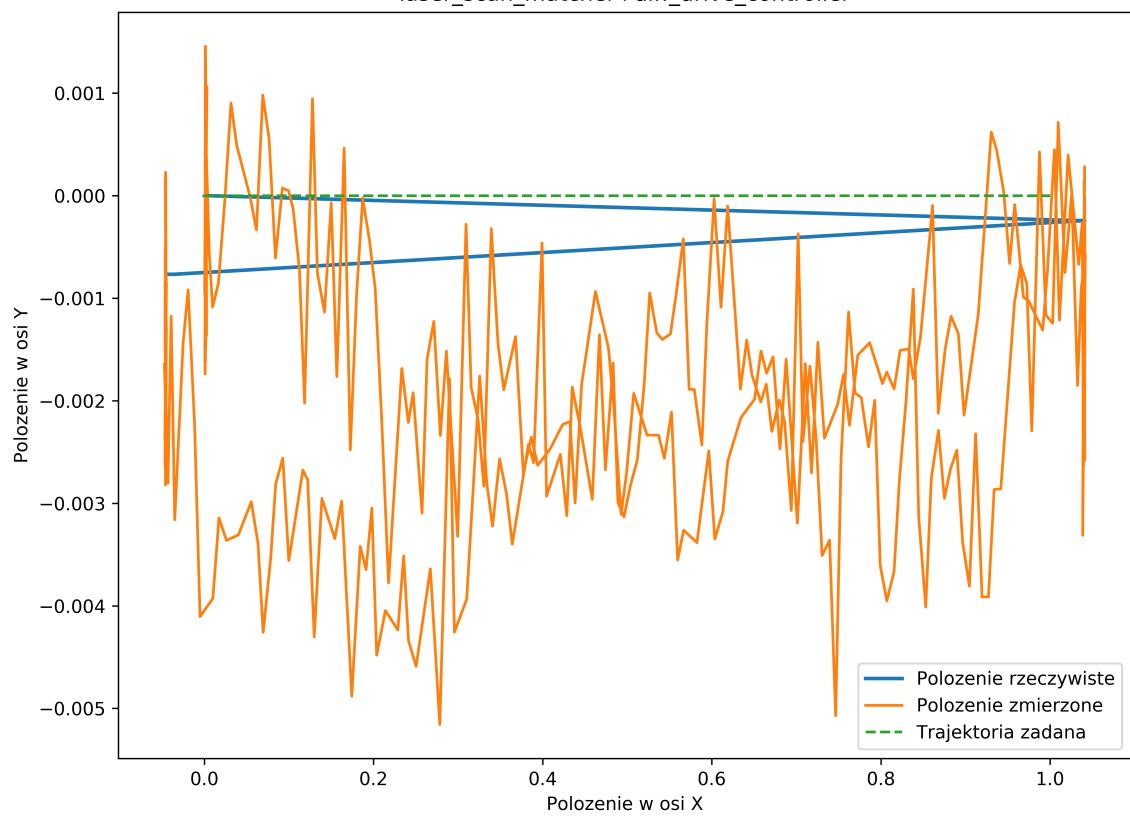
Test linii polegał na wykonaniu przez robota ruchu do przodu z położenia (0, 0) do punktu (1, 0) i z powrotem. Szybkość ruchu była stała i wynosiła 0.3m/s. Poniżej przedstawione zostały uzyskane rezultaty.

#### Trajektoria uzyskana przy wykorzystaniu `diff_drive_controller`:



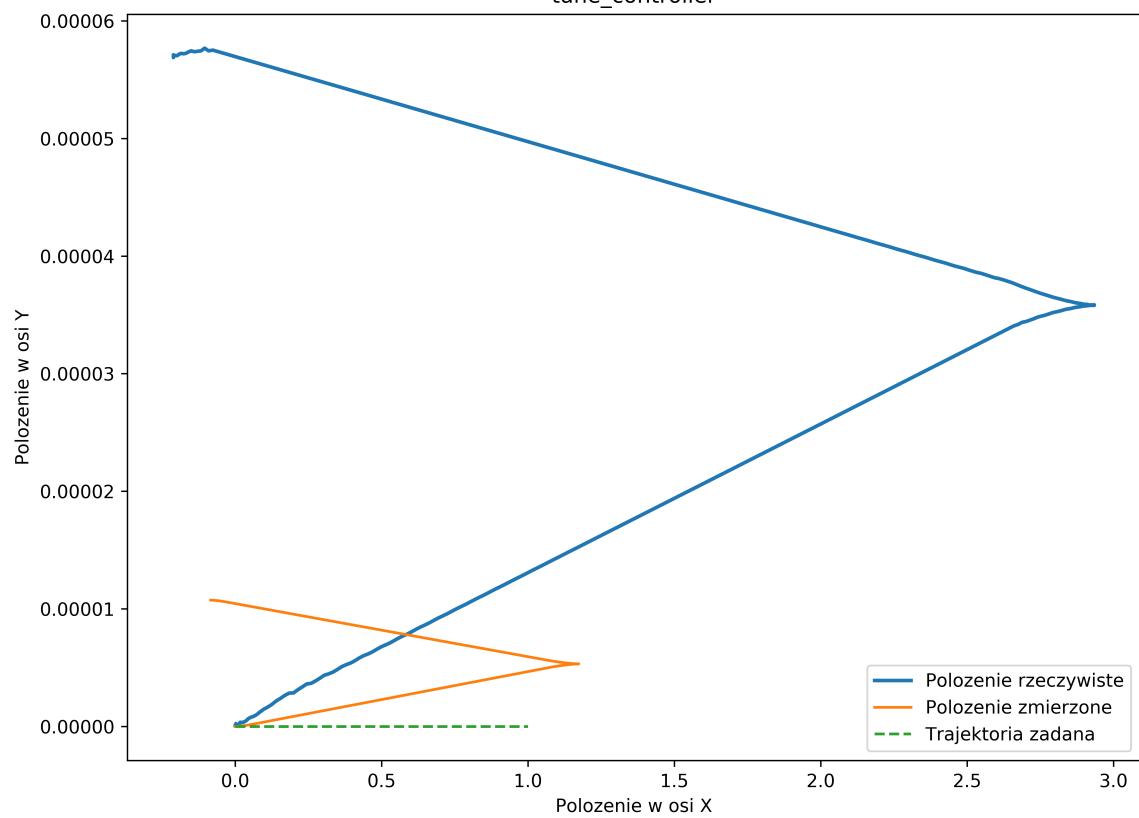
**Trajektoria uzyskana przy wykorzystaniu  
`diff_drive_controller` w połączaniu z pakietem  
`laser_scan_matcher`**

Sterowanie ze sprzezaniem (ruch po linii)  
laser\_scan\_matcher+diff\_drive\_controller

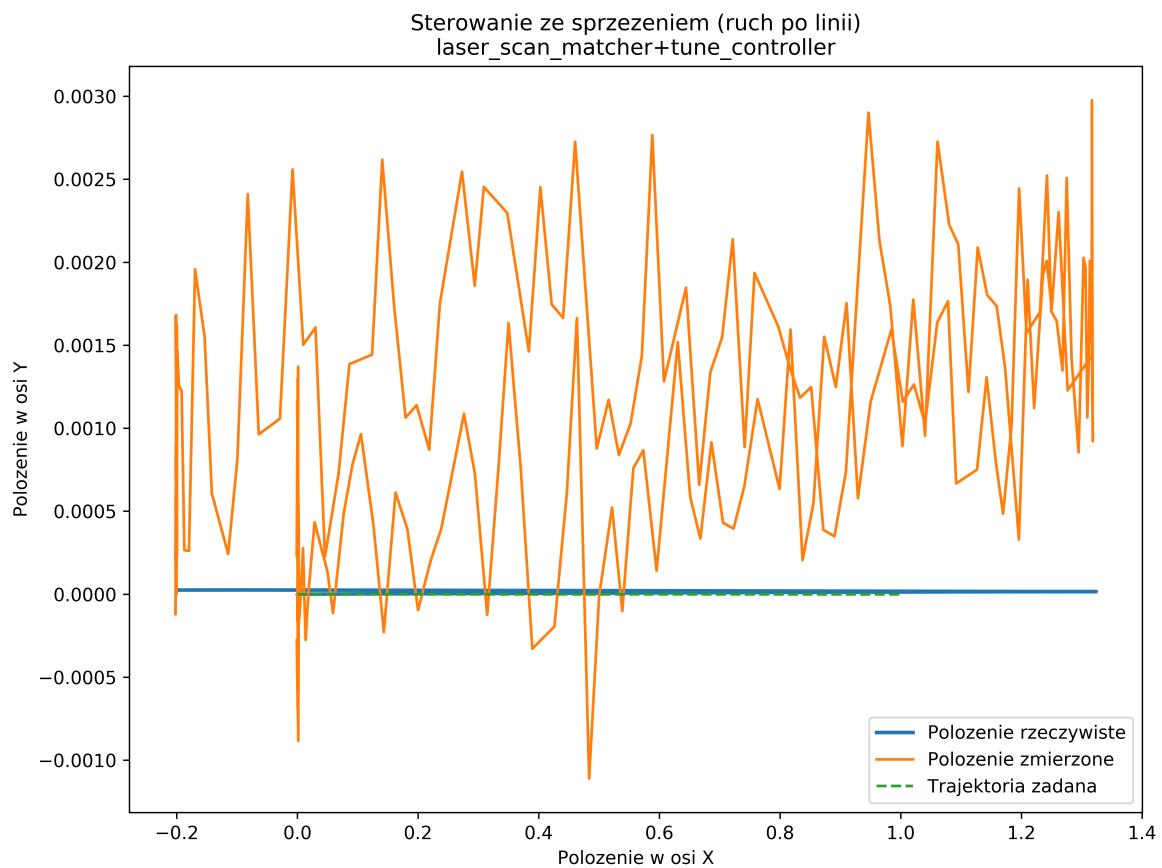


Trajektoria uzyskana przy wykorzystaniu `tune_controller`:

Sterowanie ze sprzezaniem (ruch po linii)  
tune\_controller

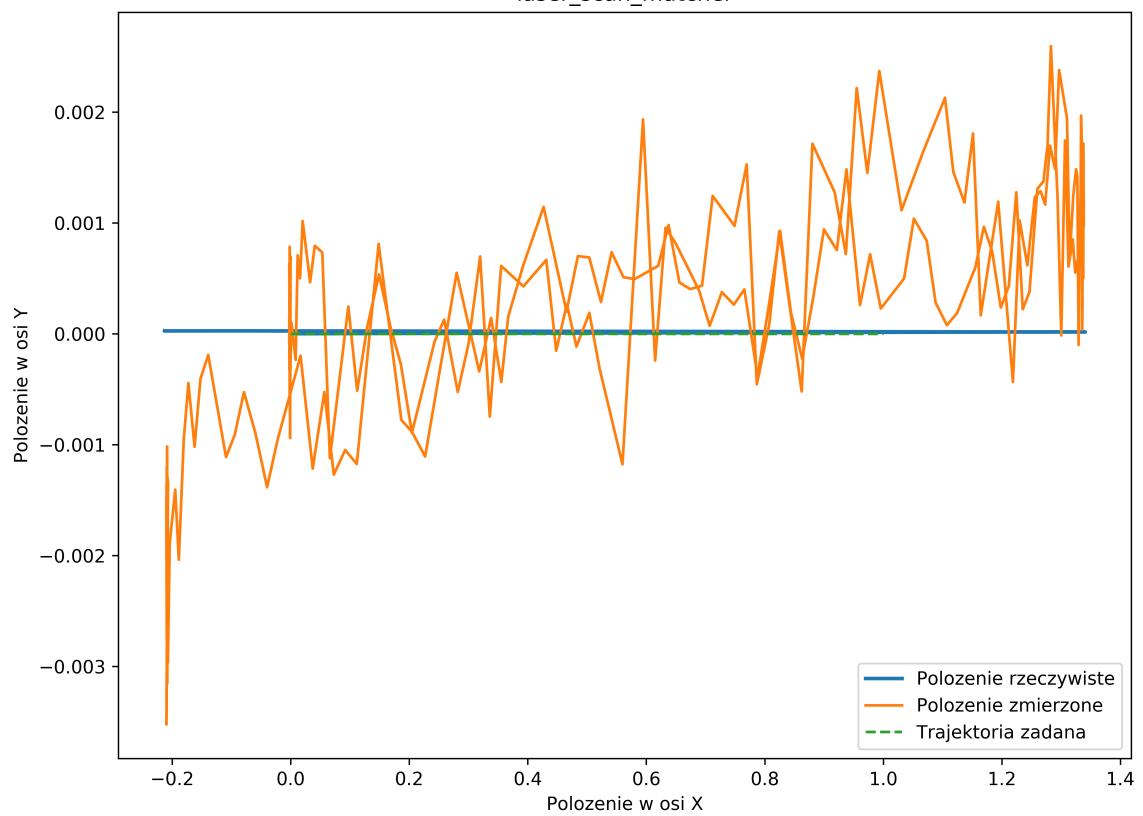


Trajektoria uzyskana przy wykorzystaniu `tune_controller` w połączeniu z pakietem `laser_scan_matcher`

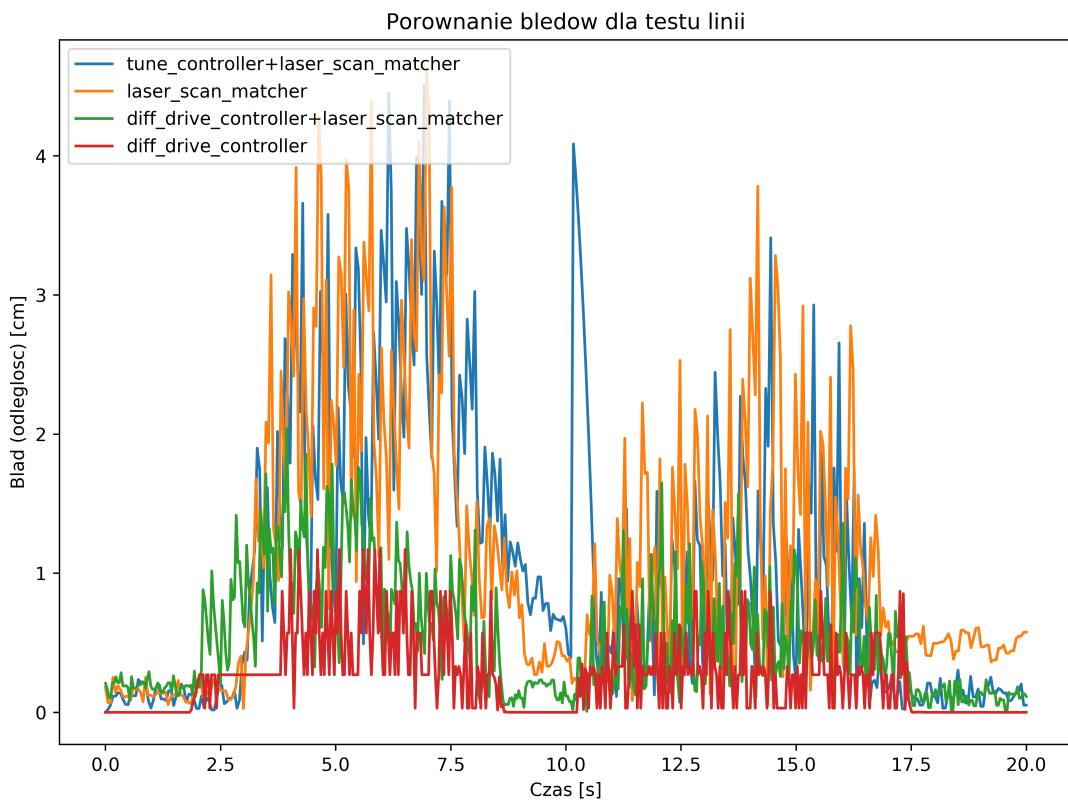


Dla porównania poniżej przedstawiona została uzyskana trajektoria przy wykorzystaniu samego pakietu `laser_scan_matcher`:

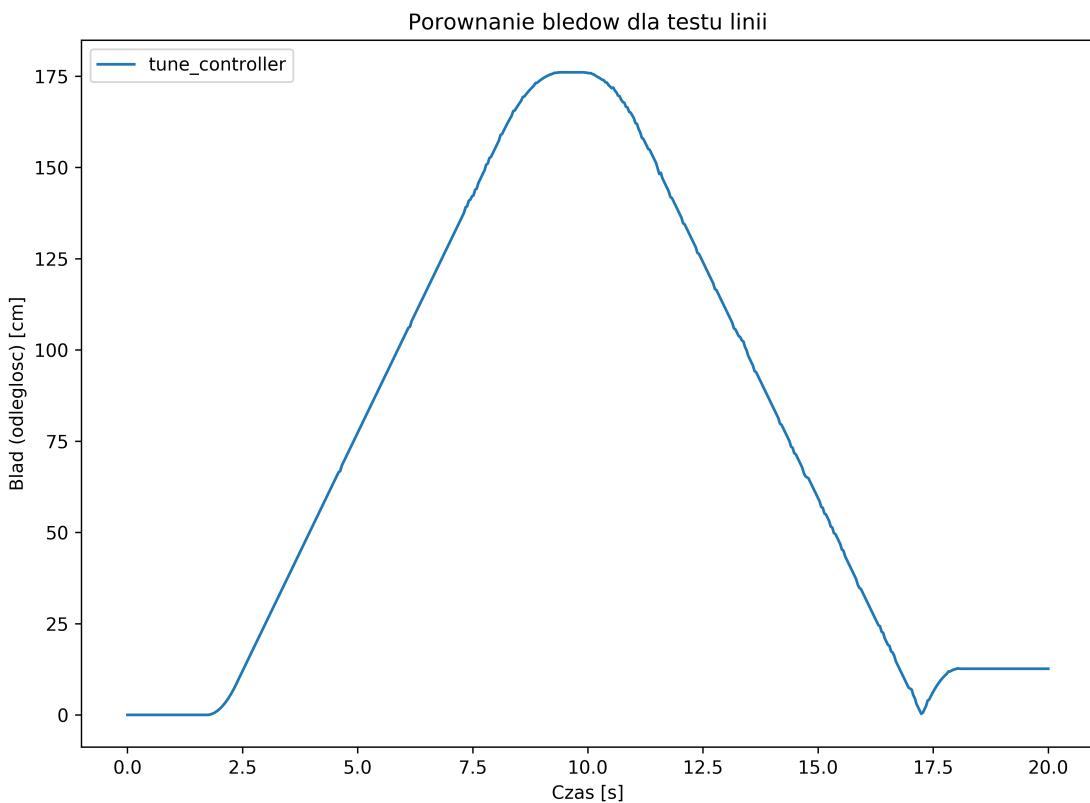
Sterowanie ze sprzezaniem (ruch po linii)  
laser\_scan\_matcher



## Porównanie błędów



Oddzielnie przedstawiony został wykres błędu dla `tune_controller` ze względu na to, że błąd ten jest znacznie większy od błędów pochodzących z innych źródeł odometrii:



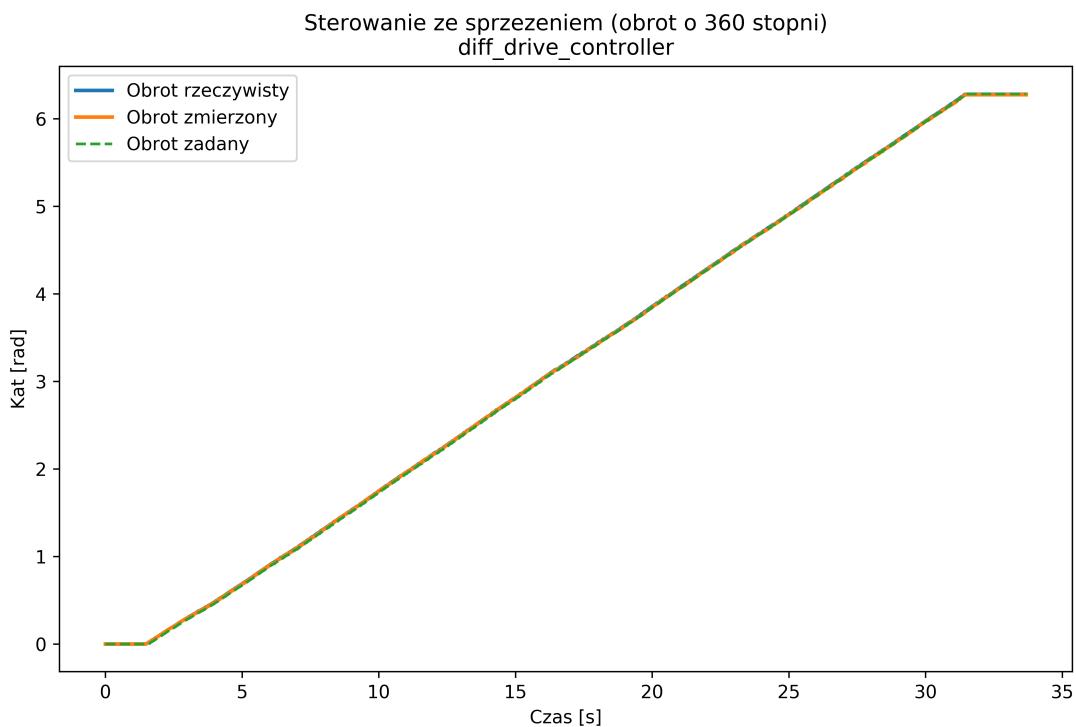
## Błędy końcowe

Źródło odometrii	Błąd bezwzględny [cm]
diff_drive_controller	6.91457e-07
tune_controller	12.64528
laser_scan_matcher	0.57553
diff_drive_controller + laser_scan_matcher	0.11456
tune_controller + laser_scan_matcher	0.05187

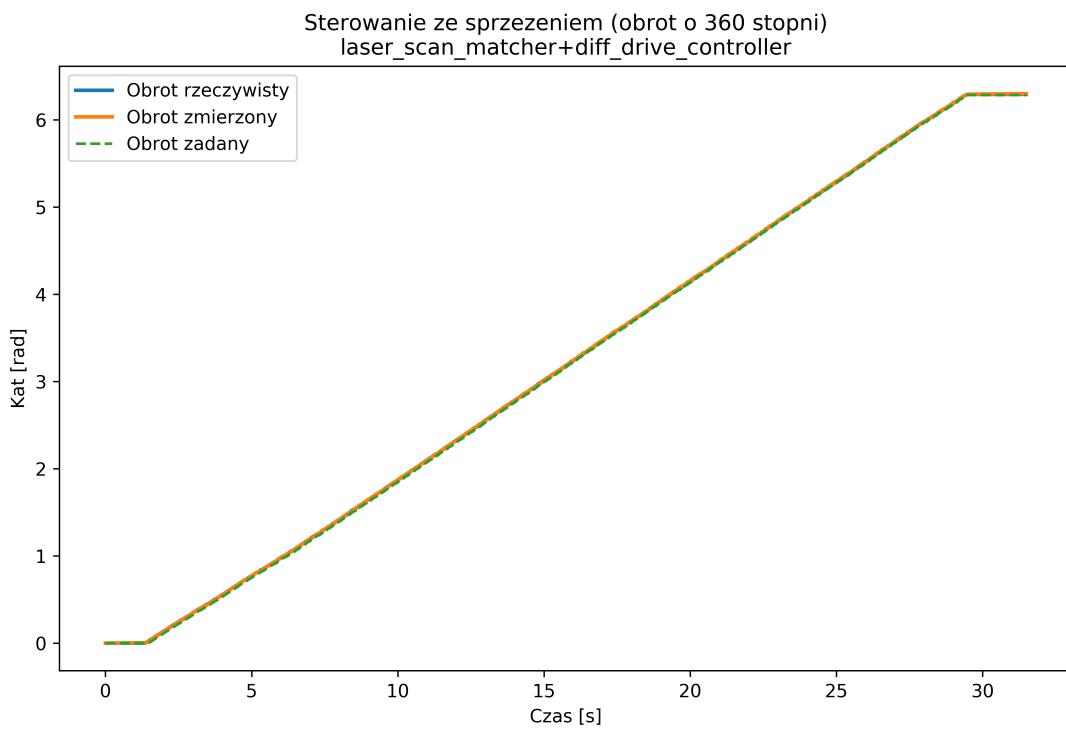
## Test obrotu

Test obrotu polegał na wykonaniu przez robota pełnego obrotu przeciwnie do ruchu wskazówek zegara z prędkością kątową równą 0.3rad/s. Poniżej przedstawione zostały uzyskane rezultaty.

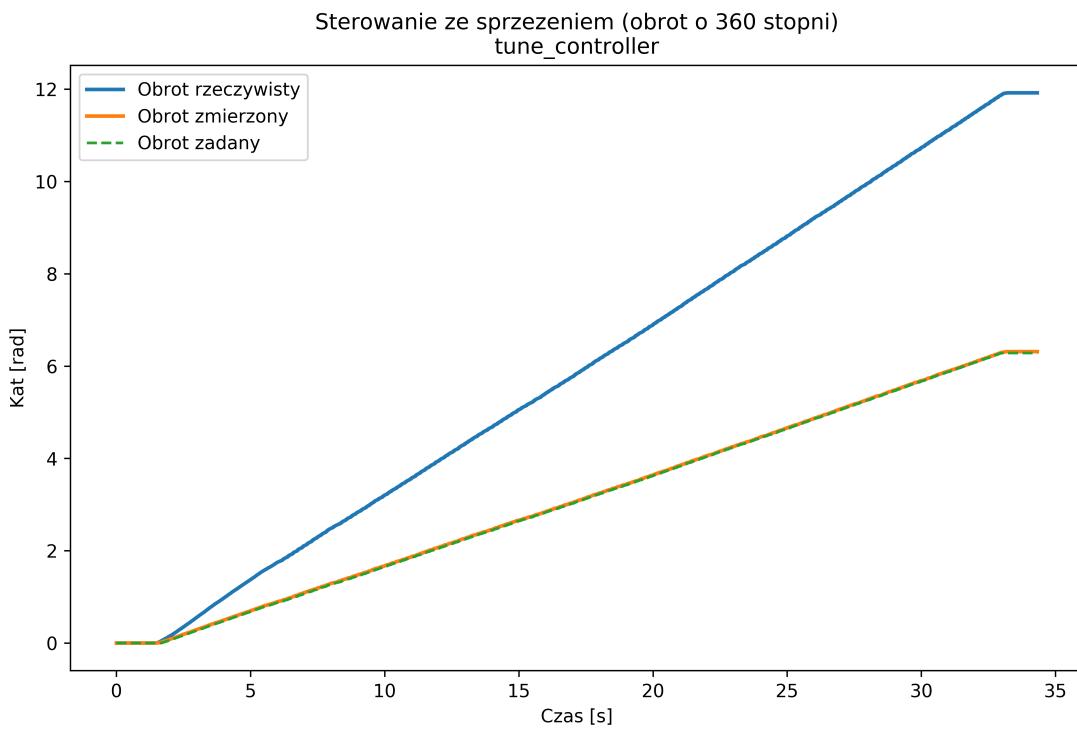
### Trajektoria uzyskana przy wykorzystaniu diff\_drive\_controller:



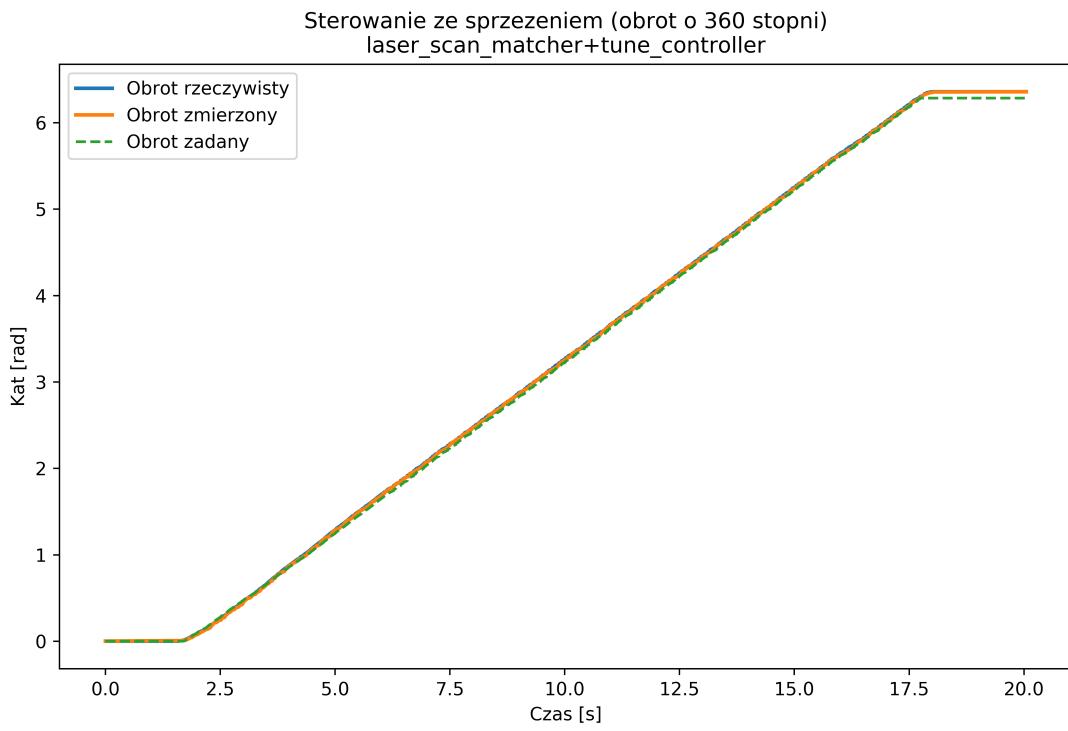
### Trajektoria uzyskana przy wykorzystaniu diff\_drive\_controller w połączeniu z pakietem laser\_scan\_matcher



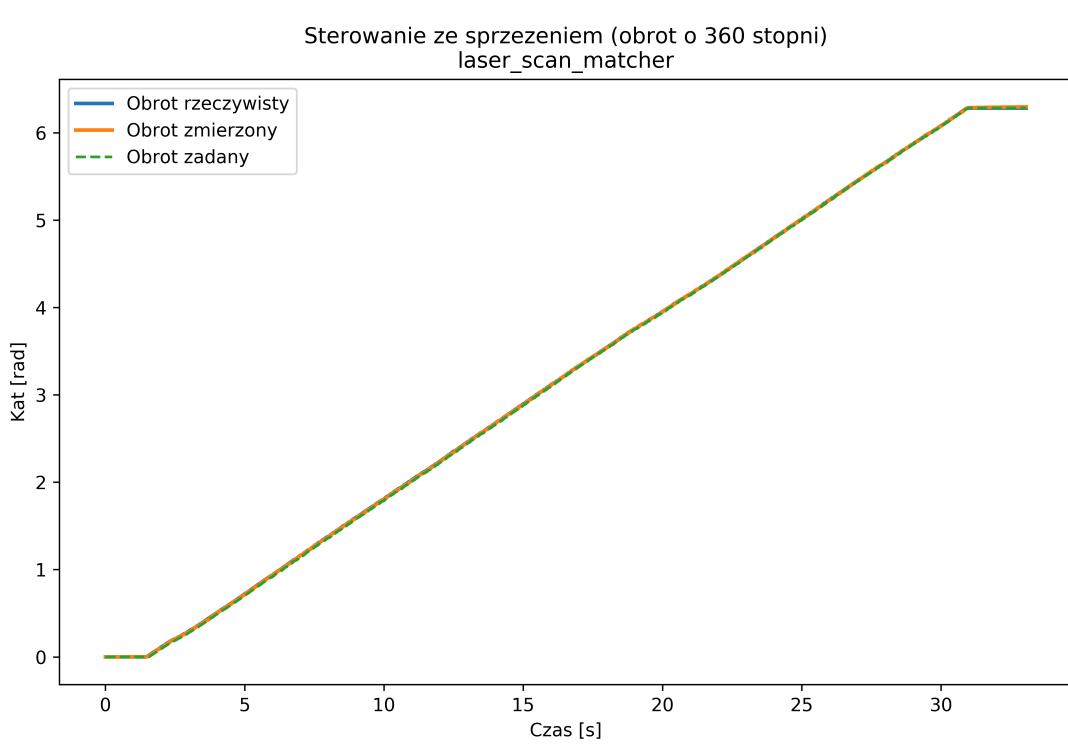
Trajektoria uzyskana przy wykorzystaniu `tune_controller`:



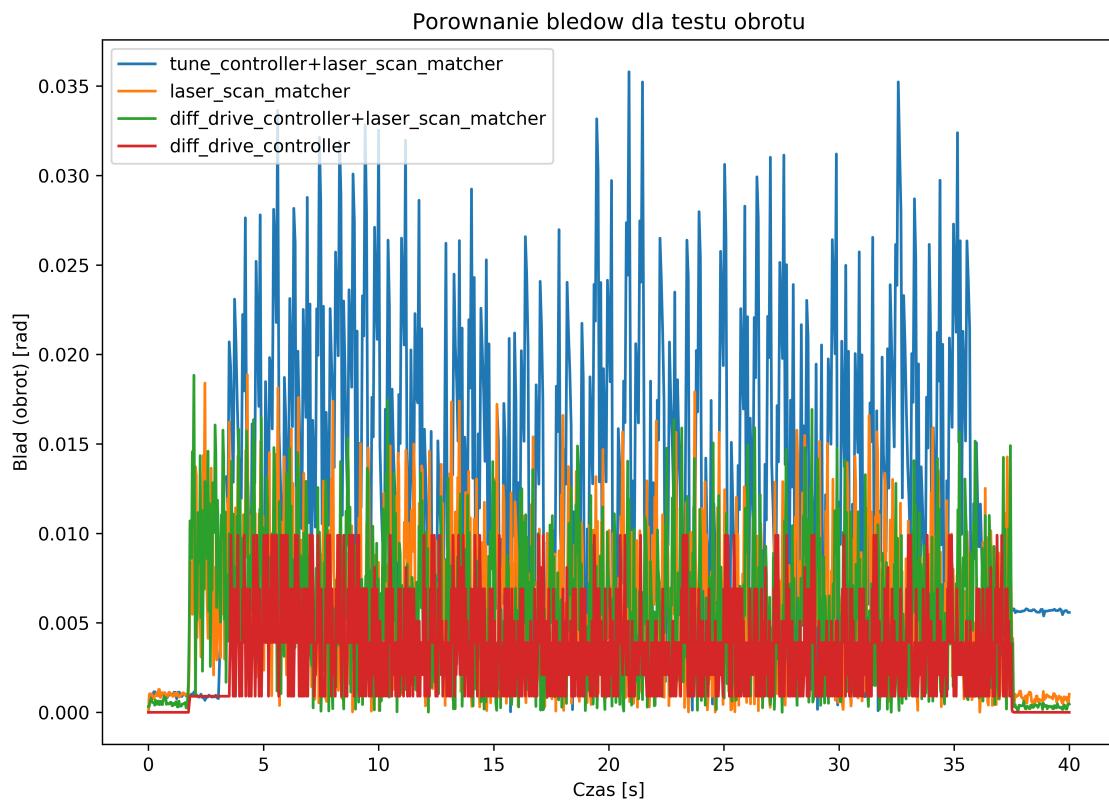
Trajektoria uzyskana przy wykorzystaniu `tune_controller` w połączeniu z pakietem `laser_scan_matcher`



Dla porównania poniżej przedstawiona została uzyskana trajektoria przy wykorzystaniu samego pakietu `laser_scan_matcher`:

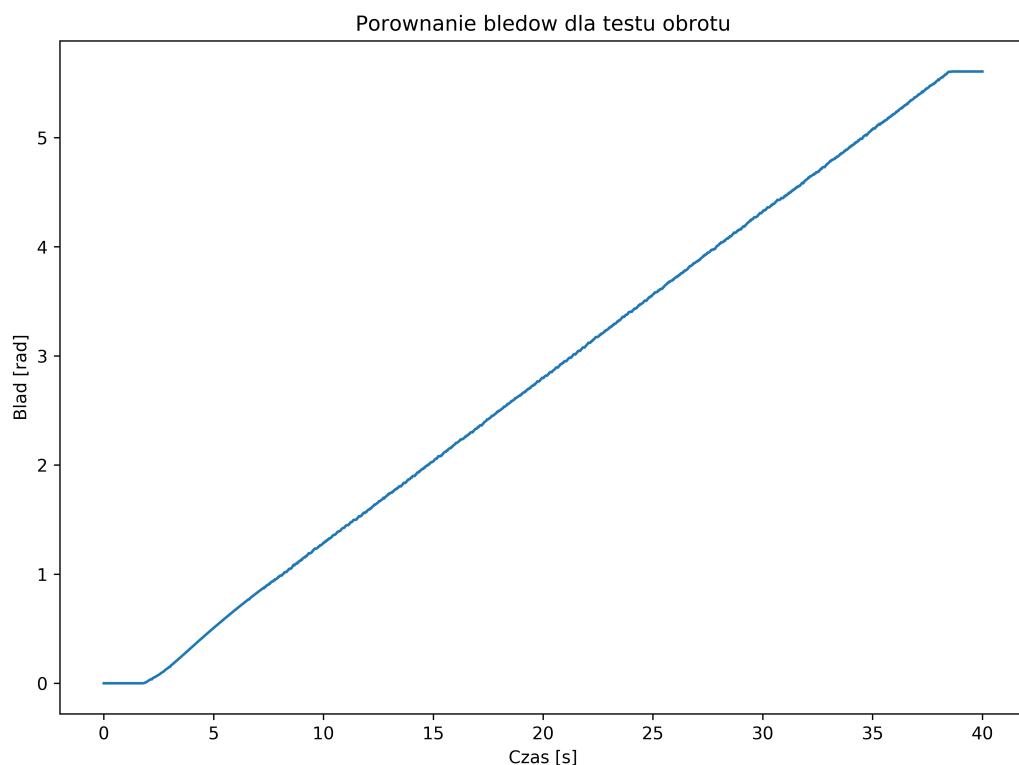


## Porównanie błędów

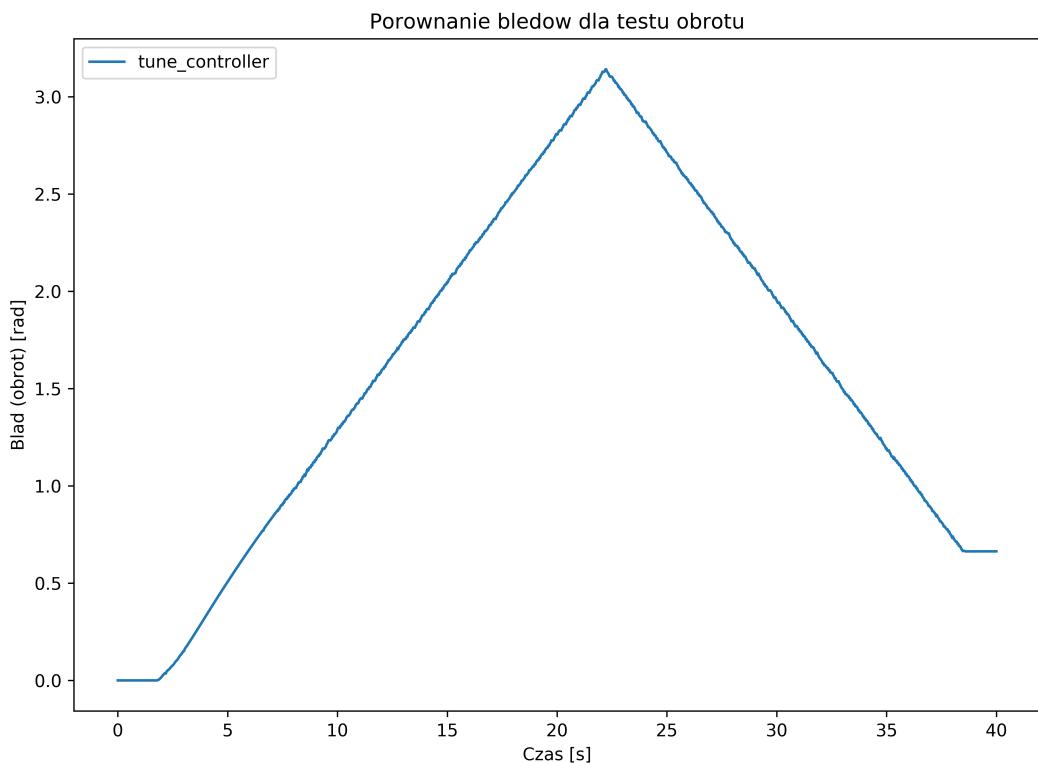


Oddzielnie przedstawiony został wykres błędu dla `tune_controller` ze względu na to, że błąd ten jest znacznie większy od błędów pochodzących z innych źródeł odometrii:

- Całkowity skumulowany błąd:



- Błąd orientacji robota:



## Błędy końcowe

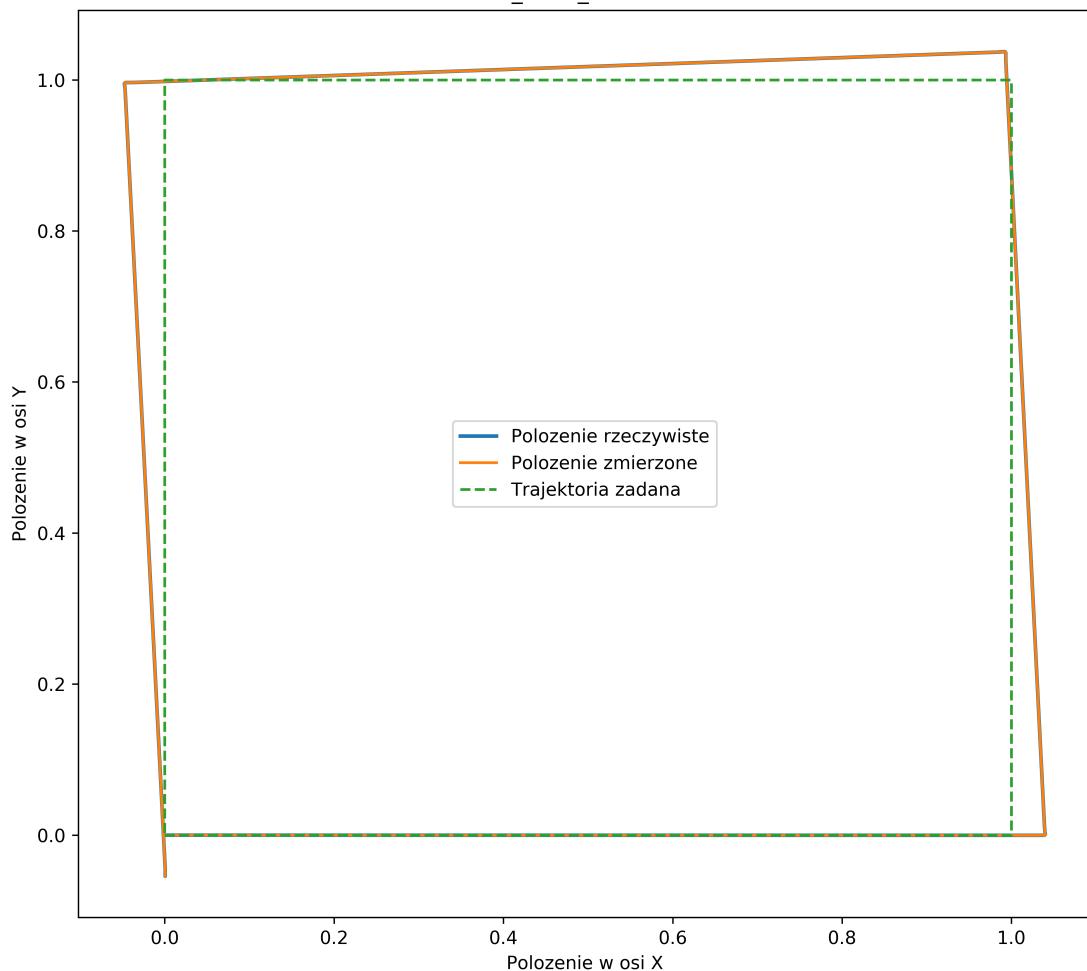
Źródło odometrii	Błąd bezwzględny [rad]
diff_drive_controller	3.28817e-08
tune_controller	0.66342
laser_scan_matcher	0.00546
diff_drive_controller + laser_scan_matcher	0.00046
tune_controller + laser_scan_matcher	0.00473

## Test kwadratu

Test kwadratu polegał na wykonaniu przez robota ruchu z położenia  $(0, 0)$  kolejno do punktów  $(1, 0)$ ,  $(1, 1)$ ,  $(0, 1)$ ,  $(0, 0)$  a więc wykonaniu ruchu po kwadracie o boku 1m. Prędkość ruchu dla każdego testu wynosiła 0.3m/s natomiast prędkość kątowa (przy ustawianiu się do ruchu do następnego punktu) wynosiła 0.3rad/s. Poniżej przedstawione zostały uzyskane rezultaty.

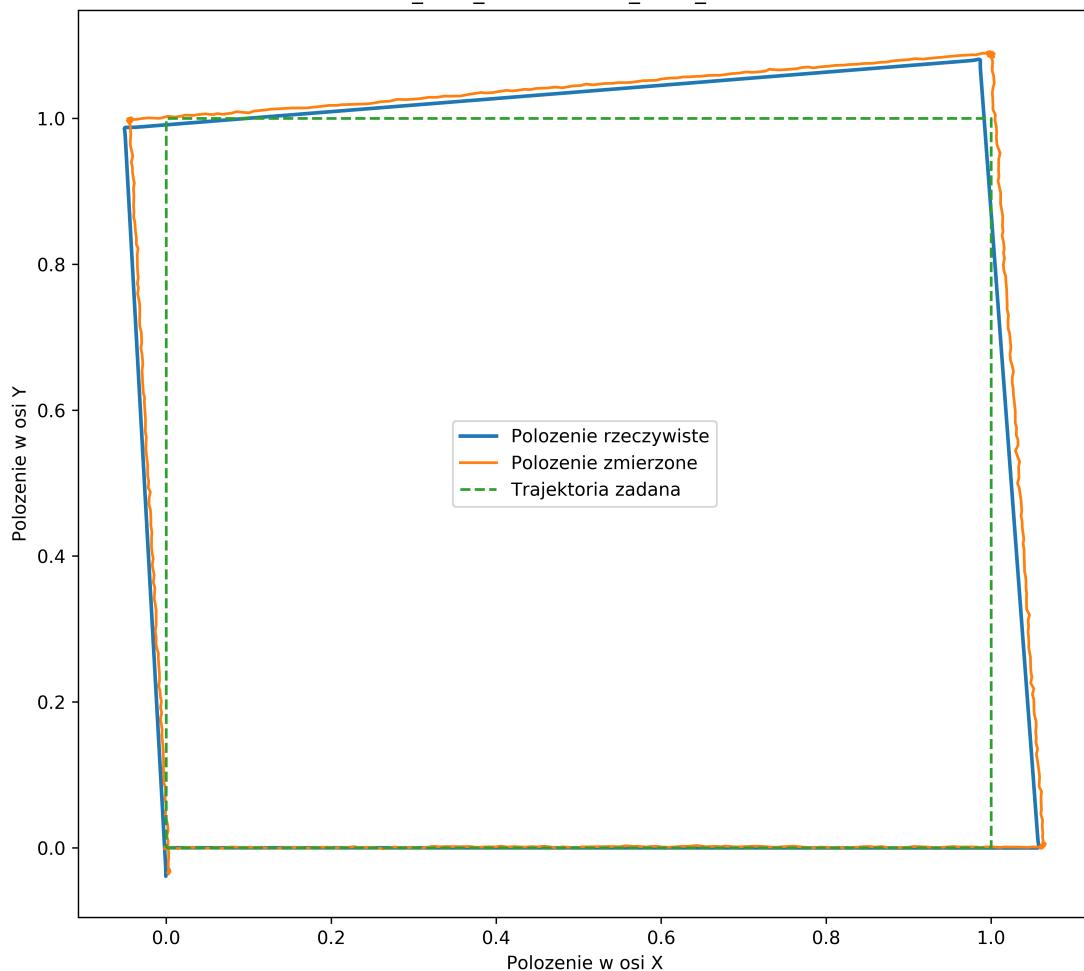
**Trajektoria uzyskana przy wykorzystaniu  
diff\_drive\_controller:**

Sterowanie ze sprzezaniem (ruch po kwadracie)  
diff\_drive\_controller



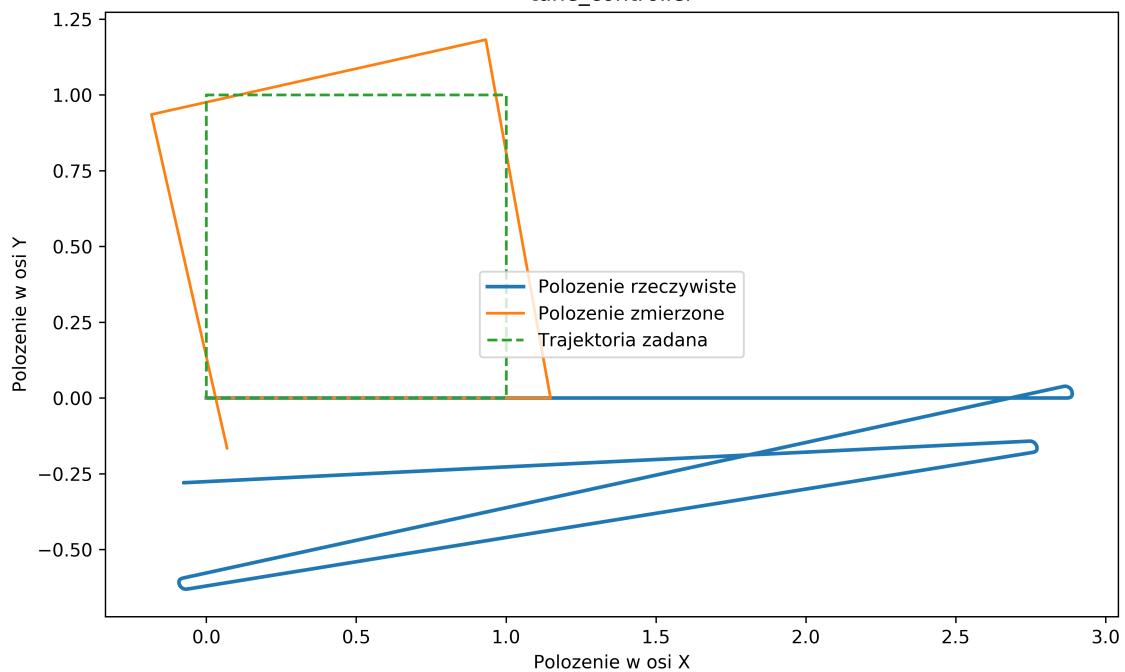
**Trajektoria uzyskana przy wykorzystaniu  
diff\_drive\_controller w połączaniu z pakietem  
laser\_scan\_matcher**

Sterowanie ze sprzezaniem (ruch po kwadracie)  
laser\_scan\_matcher+diff\_drive\_controller

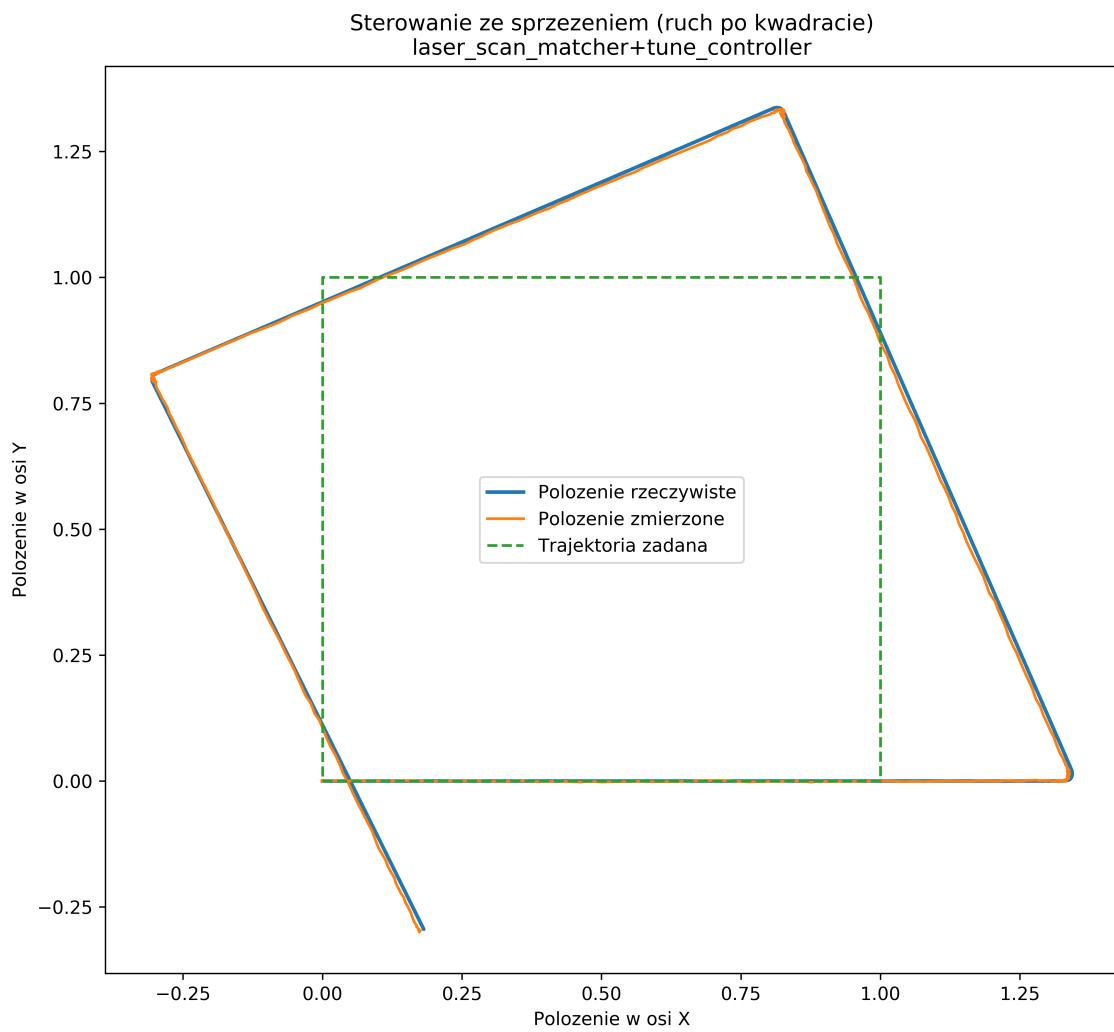


Trajektoria uzyskana przy wykorzystaniu `tune_controller`:

Sterowanie ze sprzezaniem (ruch po kwadracie)  
tune\_controller

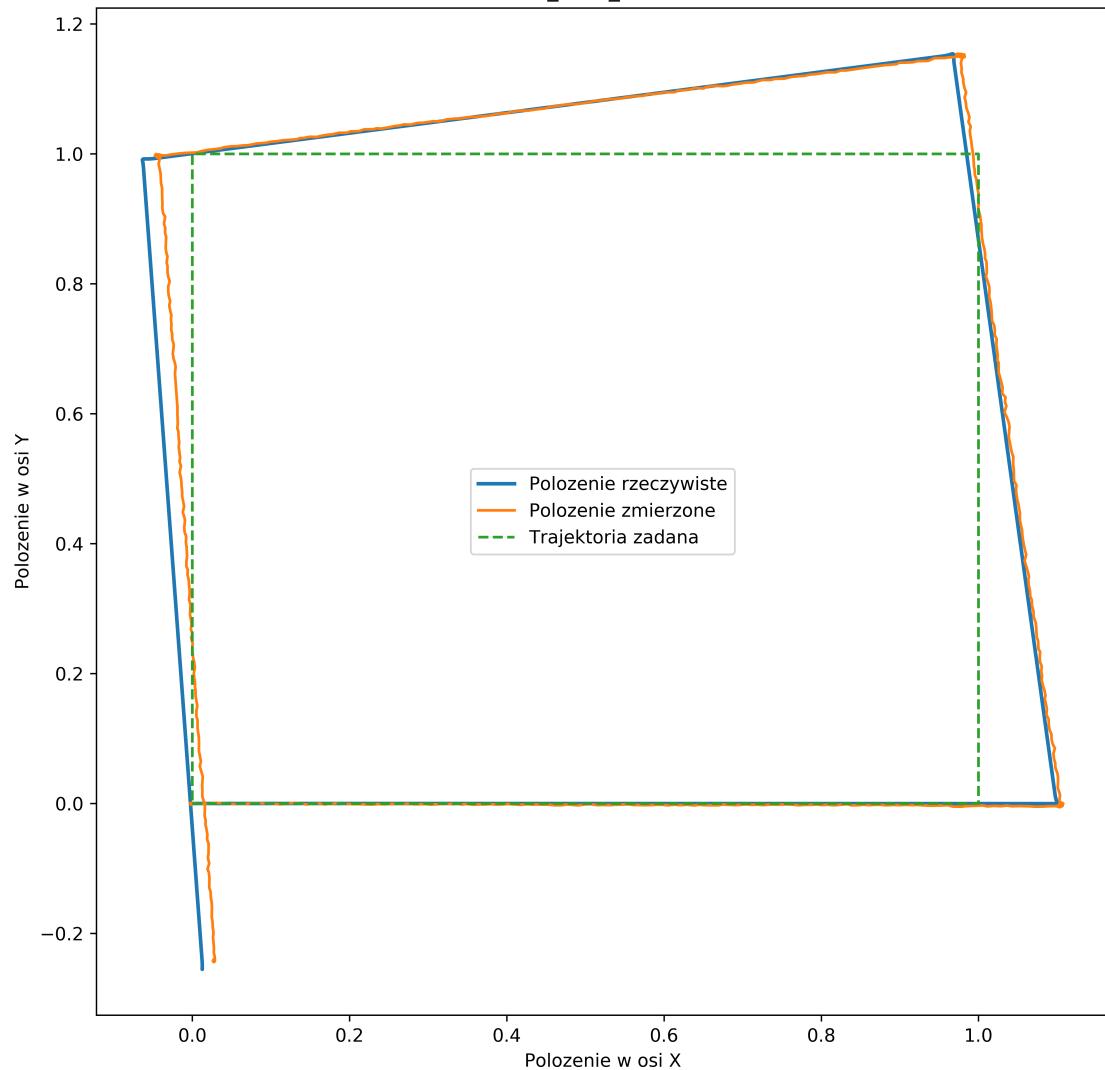


**Trajektoria uzyskana przy wykorzystaniu `tune_controller` w połączaniu z pakietem `laser_scan_matcher`**

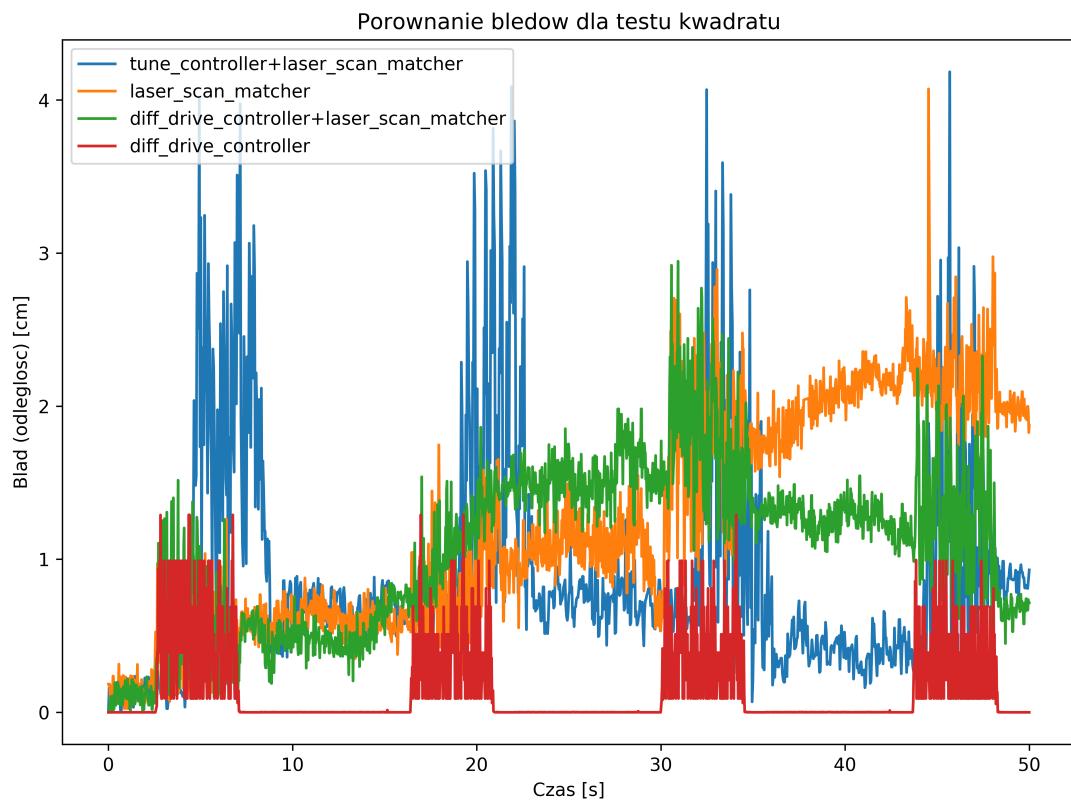


**Dla porównania poniżej przedstawiona została uzyskana trajektoria przy wykorzystaniu samego pakietu `laser_scan_matcher`:**

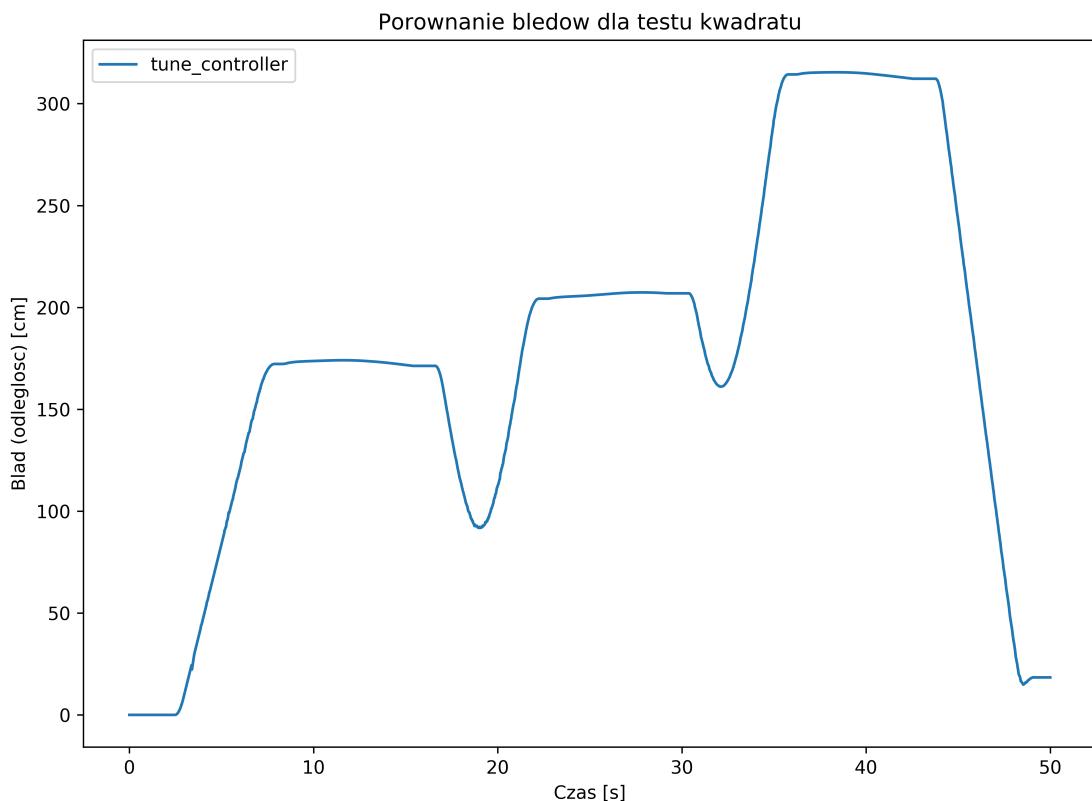
Sterowanie ze sprzezeniem (ruch po kwadracie)  
laser\_scan\_matcher



## Porównanie błędów



Oddzielnie przedstawiony został wykres błędu dla `tune_controller` ze względu na to, że błąd ten jest znacznie większy od błędów pochodzących z innych źródeł odometrii:



## Błędy końcowe

Źródło odometrii	Błąd bezwzględny [cm]
diff_drive_controller	4.91142e-06
tune_controller	18.39553
laser_scan_matcher	1.21292
diff_drive_controller + laser_scan_matcher	0.71498
tune_controller + laser_scan_matcher	0.93035

## Podsumowanie

Najlepsze rezultaty, dla każdego z testów uzyskane zostały w przypadku korzystania z samego `diff_drive_controllera`, położenia przez niego publikowane są niemalże identyczne z rzeczywistymi. W tym przypadku poprawa wzgórnej lokalizacji za pomocą czujnika laserowego nie przyniosła żądanego rezultatów gdyż czujnik laserowy wprowadza większy błąd oraz zaszumienie pomiaru, jednakże połączenie obu tych źródeł odometrii przynosi lepsze rezultaty niż przy wykorzystaniu samego czujnika laserowego.

W kwestii uzyskanego błędu lokalizacji wzgórnej najgorzej wypadł nieskalibrowany `tune_controller`. Uzyskane błędy przy jego wykorzystaniu są kilka rzędów wielkości większe od błędów uzyskanych z użyciem innych źródeł odometrii. Jednakże w tym przypadku poprawa lokalizacji wzgórnej przy użyciu pakietu `laser_scan_matcher` umożliwiła uzyskanie błędów zbliżonych do tych uzyskanych z użyciem innych źródeł odometrii. Co więcej połączenie pakietu `laser_scan_matcher` oraz `tune_controllera` umożliwia dokładniejsze określenie położenia robota niż w przypadku wykorzystania samego czujnika laserowego. Wadą korzystania z `tune_controllera` nawet w przypadku połączenia go z pakietem `laser_scan_matcher` jest pogorszona jakość sterowania. Widoczne jest to na wykresach przedstawiających przebytą przez robota trajektorię. Ciekawym zjawiskiem jest to, że w przypadku testu linii oraz kwadratu błędy generowane przez `tune_controller` pomimo pośrednich dużych wartości, po zakończeniu ruchu nie są aż tak duże (z teoretycznego punktu widzenia powinny one stale narastać). Da się to解释 tym, że robot przy korzystaniu z `tune_controllera` porusza się znacznie szybciej niż wynikałoby to z zadawanych prędkości, lecz sam `tune_controller` nie dostrzega tej różnicy. Jest to szczególnie widoczne w przypadku testu linii - rzeczywista przebyta przez robota trasa jest kilka razy dłuższa niż ta zmierzona przez `tune_controller` (wykres przebytej przez robota trajektorii). W przypadku testu kwadratu końcowy błąd jest tak mały tylko dzięki temu, że robot zamiast obracać się o 90 stopni, wykonywał obrót około dwa razy większy, dzięki czemu końcowe położenie znalazło się blisko rzeczywistego. Była to więc tylko i wyłącznie kwestia przypadku.