STERO Laboratorium 2 - Sprawozdanie

Skład zespołu: Dominika Wyszyńska, Bartosz Kowalski
Numer stanowiska: 5

WSTĘP

Celem laboratorium 2 było stworzenie narzędzi umożliwiających badanie odometrii robota z bazą różnicową. W ramach zadania skupiono się na implementacji dwóch głównych elementów: węzła zadającego trajektorię po kwadracie oraz węzła liczącego błąd na podstawie danych z odometrii, prędkości zadanego ruchu oraz lokalizacji z symulatora *Gazebo*.

Kod projektu został zamieszczony na *gitlabie* pod poniższym adresem.
https://gitlab-stud.elka.pw.edu.pl/bkowals3/stero_lab_2

Link do filmiku z prezentacją przykładowego działania:
https://youtu.be/g803Jl6LigI?feature=shared

1. Węzeł zadający trajektorię

Węzeł ten został zaprojektowany do generowania trajektorii w kształcie kwadratu. Wartość długości boku kwadratu, kierunek wykonania (lewo/prawo) oraz profil prędkości (skokowy, trapezowy) są ustawiane jako parametry (w pliku *parameters.yaml*). Prędkość zadawana jest na topic */key_vel*. Profil trapezowy jest realizowany poprzez przyspieszanie przez 1/3 długości boku, utrzymanie stałej prędkości przez kolejną 1/3, a następnie zwalnianie, aby na końcu boku osiągnąć prędkość zerową.

2. Węzeł liczący błąd

Węzeł *Recorder* subskrybuje trzy główne tematy:

- Lokalizację z odometrii (*/mobile_base_controller/odom*).
- Prędkość zadaną (*/key_vel*).
- Lokalizację z symulatora Gazebo (*/gazebo/model_states*).

Na podstawie tych danych węzeł ten przeprowadza obliczenia w celu określenia błędu chwilowego oraz skumulowanego. Błąd chwilowy jest obliczany w oparciu o różnicę między lokalizacją rzeczywistą a zadaną. Błąd skumulowany jest następnie obliczany dla każdego rogu kwadratu, a także na koniec całej trajektorii kwadratu.

3. Uruchomienie programu

Aby wszystko działało poprawnie, trzeba jeszcze dokonać kilku modyfikacji w ***CMakeLists.txt***. Aby wywołać jednocześnie oba węzły, a także Gazebo i RViZ należy wykonać polecenie: ***roslaunch <nazwa_paczki> <nazwa_pliku_launch>.***

4. Analiza wyników

Po wykonaniu implementacji należało przejść do testowania. Najważniejszymi wynikami z testów powinny być błędy. Przy każdej zmianie parametru zapisywaliśmy błędy do plików *e_moment.txt*, *e_sum.txt* oraz *e_sides.txt*. Dzięki temu łatwiej było później rysować wykresy przedstawiające zależności błędu chwilowego w czasie eksperymentu. Kod, którym posługiwaliśmy się przy rysowaniu wykresów w *Pythonie*:

```
for trio in [[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12]]:
    for typ in ['moment', 'sides', 'sum']:
        for proba in trio:
            with open(f'stero_testy/{proba}/e_{typ}.txt', 'r') as file:
                data = [float(value) for line in file.readlines() for value in line.strip().split()]
                iterations = list(range(1,len(data)+1))
                errors = data
                plt.plot(iterations, errors, marker='o', label=f'{proba_opis_short[proba]}', linewidth=1, markersize=1)
        plt.title(f'Zależność wartości błędu od iteracji/czasu - {typ_opis[typ]} - {trio_opis[tuple(trio)]}', fontsize=8)
        plt.xlabel('Iteracja/czas')
        plt.ylabel('Wartość błędu')
        plt.grid(True)
        plt.legend()
        plt.savefig(f'img/{trio[0]}_{trio[1]}_{trio[2]}/{typ}.png', dpi=400)
        plt.cla()
```

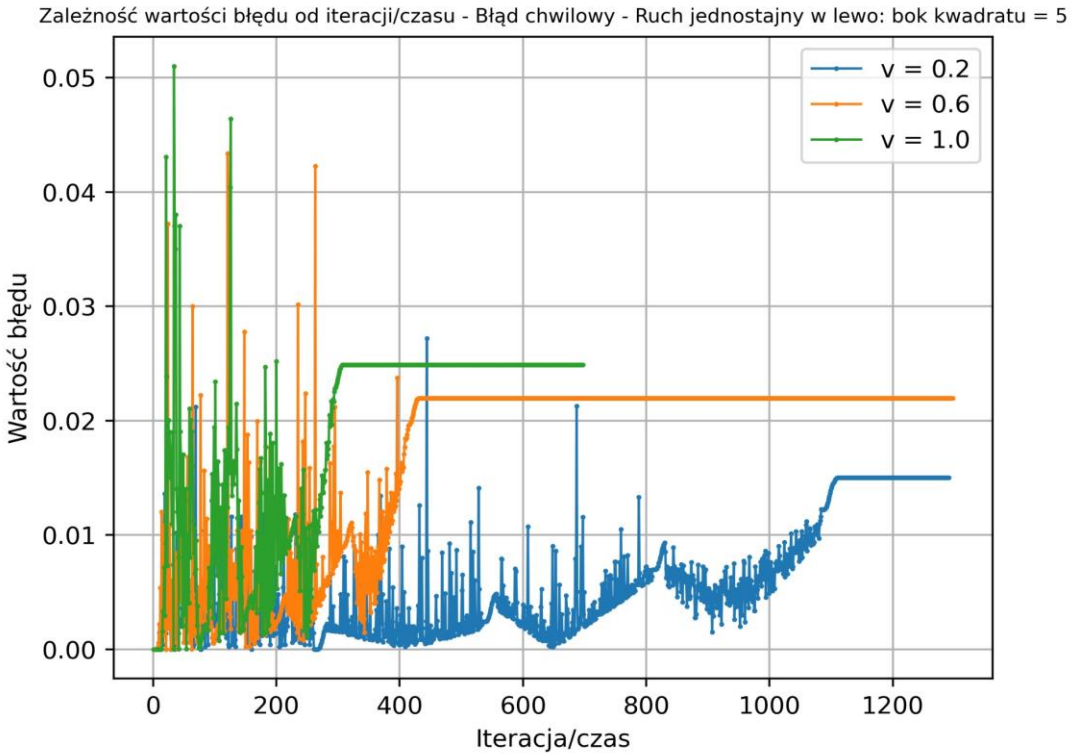
Dla wszystkich testów zastosowano długość boku kwadratu równą 5.

Odpowiedzi na pytania z projektu:

1. **Jak wpływa zmiana prędkości przy sterowaniu skokowym profilem prędkości (zakładając te same parametry kwadratu)?**

Testy wykonane zostały dla trzech różnych wartości prędkości z przedziału [0.1, 1.0] m/s - dokładnie dla 0.2, 0.6 oraz 1.0 dla wartości *lewo* - kierunku wykonywania kwadratu. Jeżeli zadamy mniejszą prędkość, to robot dokładniej wykonuje kwadrat oraz obraca się o 90 stopni, ale zajmuje mu to sporo czasu. Czasami robot nie zdąża wykonać pełnego obrotu o 90 stopni, bądź wykona trochę niedokładny bok kwadratu. Dość ważną informacją jest to, że symulator Gazebo nie do końca odzwierciedla, to co się faktycznie dzieje. W związku z tym liczone są błędy - chwilowe, skumulowane dla rogów kwadratu i całego kwadratu. Kluczowymi chwilami dla profilu prędkości skokowej są rozpoczęcie ruchu liniowego i rozpoczęcie ruchu obrotowego.

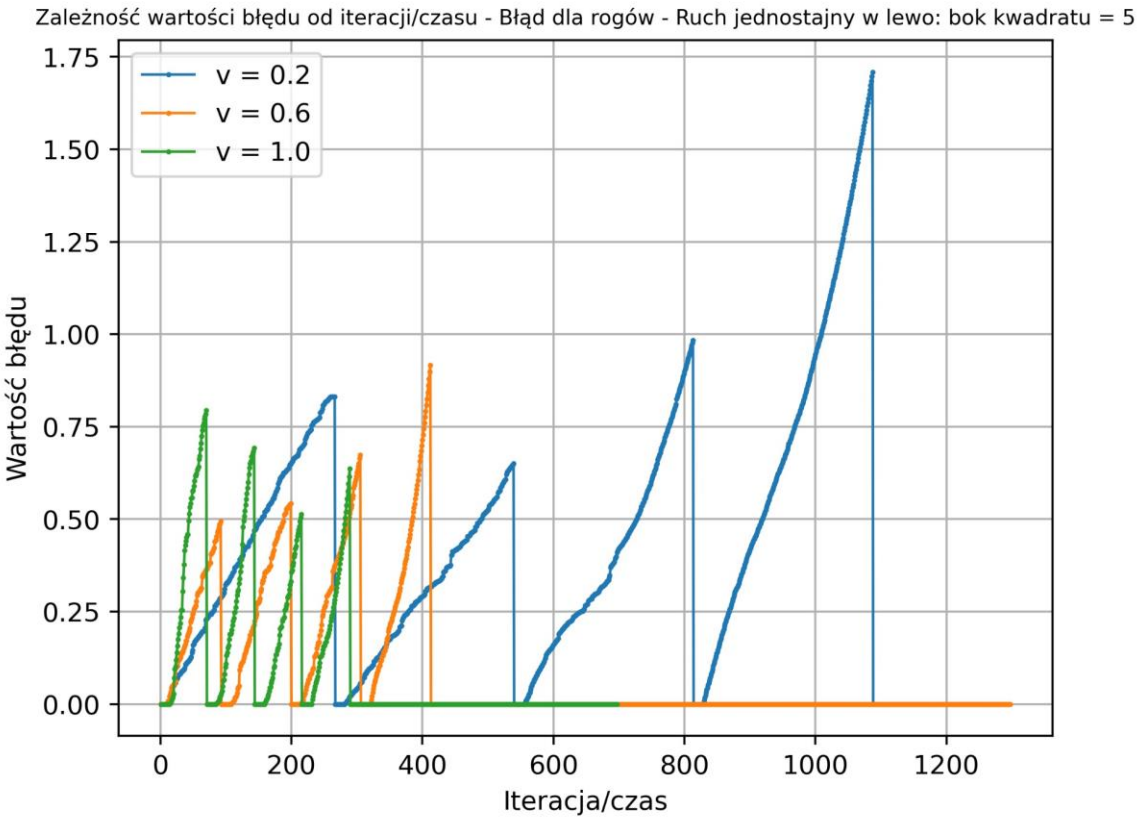
Wykresy przedstawiające zależność błędu chwilowego w czasie eksperymentu dla różnych wartości badanego parametru - prędkości:

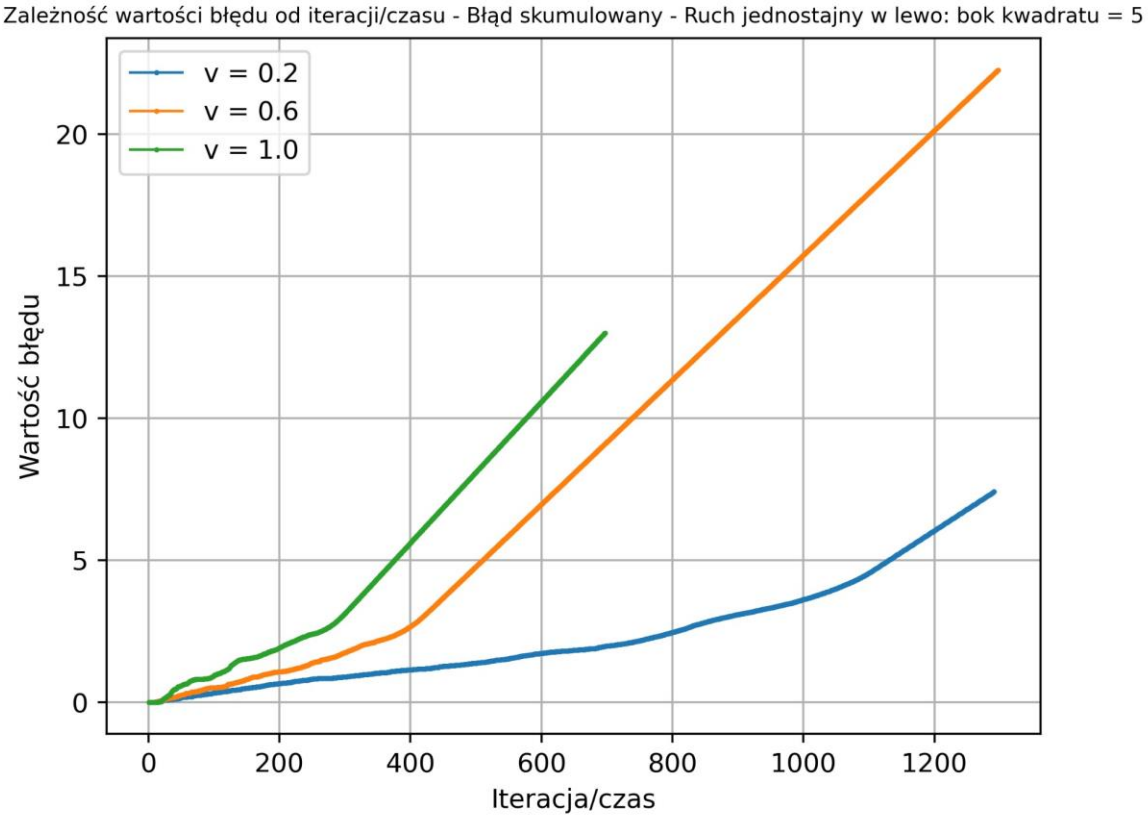


Jak widać na wykresie powyżej, im wyższa prędkość, tym większy błąd chwilowy. Dla każdego wykresu, od pewnego momentu można zauważyć funkcję stałą. Widoczna jest ona, ponieważ przedstawione są tam dane już po zakończeniu obliczeń i przepisywane są ostatnie wartości danego wektora błędów. “Górki” widoczne na wykresach powyżej występują dla sytuacji, gdy robot znajduje się na rogu kwadratu i zaczyna się obracać. Widać także, że początkowe błędy (np. niedokładne obrócenie się robota o 90 stopni) propaguje się przez kolejne iteracje.

Prędkość	0,2	0,6	1,0
Błąd skumulowany	4,389	3,281	2,897
Skumulowany błąd dla rogów kwadratu	4,175	2,626	2,637

Błąd skumulowany liczy się od początku i każda mała niedokładność będzie się propagować do końca procesu. Możemy wnioskować, iż dla małej prędkości liniowej błędy skumulowane są większe niż dla większych prędkości. Jest to spowodowane tym, że robot jedzie wolniej, więc ma więcej iteracji, w których dolicza błąd. Widać, że skumulowany błąd dla rogów kwadratu stanowi sporą część całkowitego błędu skumulowanego. Różnicę między błędem skumulowanym a sumą błędów dla rogów kwadratu ilustrują poniższe wykresy.


Zależność wartości błędu od iteracji/czasu - Błąd dla rogów - Ruch jednostajny w lewo: bok kwadratu = 5
v = 0.2
v = 0.6
v = 1.0
Wartość błędu
Iteracja/czas
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
0
200
400
600
800
1000
1200


Zależność wartości błędu od iteracji/czasu - Błąd skumulowany - Ruch jednostajny w lewo: bok kwadratu = 5
v = 0.2
v = 0.6
v = 1.0
Wartość błędu
Iteracja/czas
0
5
10
15
20
0
200
400
600
800
1000
1200

Jak widać na powyższych wykresach, błąd dla rogów kwadratu z każdym kolejnym rogiem jest coraz większy. Jest to spowodowane tym, że robot przejeżdża więcej drogi i położenia się bardziej rozbiegają z oczekiwanym - propagowanie błędu w kolejnych iteracjach. Tak jak oczekiwaliśmy, błąd skumulowany rośnie cały czas. Na drugim ze wskazanych wykresów widoczne są od pewnego momentu proste rosnące, które oznaczają nic innego, jak dopisywanie do końca wektora błędów ostatniej wartości już po zakończeniu działania programu.

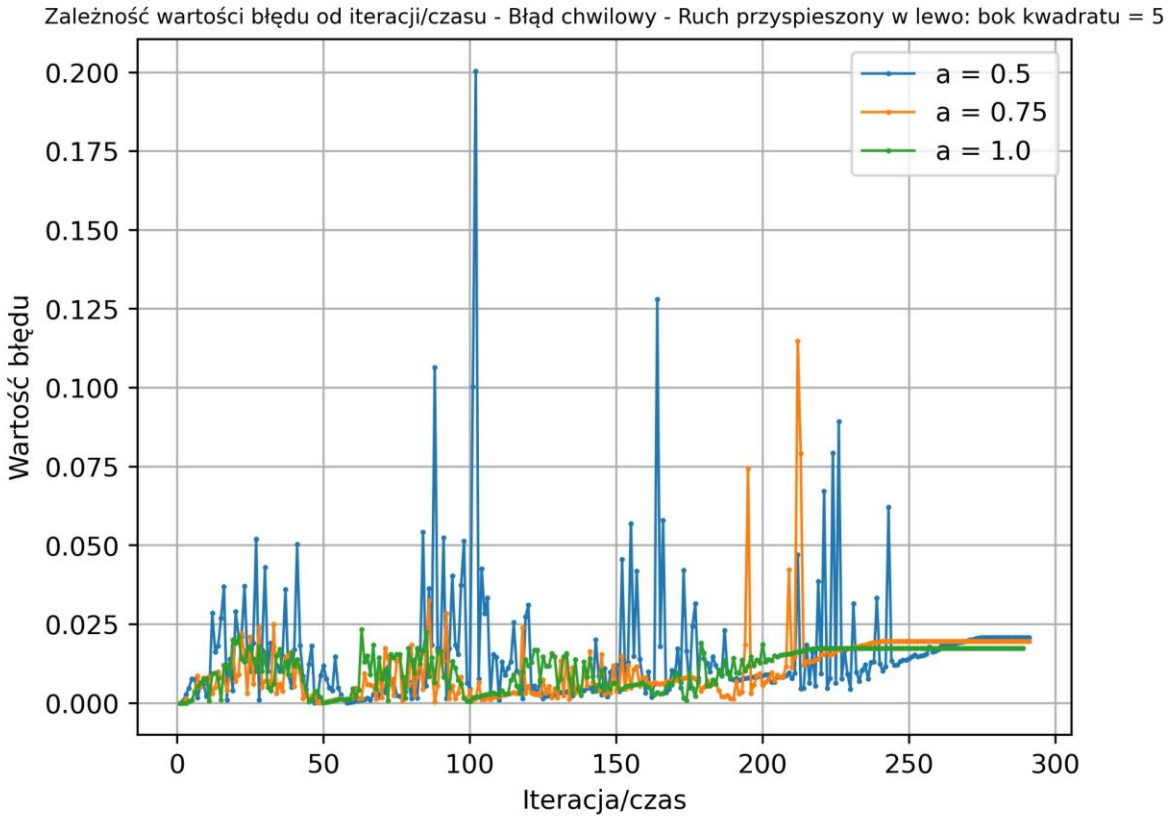
Znaczenie ma także opóźnienie *Gazebo*, gdyż nie do końca odzwierciedla to, co się rzeczywiście dzieje i możliwe, że porównuje wyniki z poprzedniej iteracji. Od rozpoczęcia jazdy błąd dość szybko rośnie, co pewnie jest spowodowane opisaną rozbieżnością między *Gazebo* a rzeczywistym położeniem.

2. Jak wpływa zmiana przyspieszenia przy trapezowym profilu prędkości (zakładając te same parametry kwadratu)?

Testy wykonane zostały dla trzech różnych wartości przyspieszeń z przedziału [0.5, 1.0] m/s^2 - dokładnie dla 0.5, 0.75 oraz 1.0 dla wartości *lewo* - kierunku wykonywania kwadratu. Jeżeli zadamy mniejsze przyspieszenie, to robot dokładniej wykonuje kwadrat oraz obraca się o 90 stopni. Przy najwyższym przyspieszeniu wartości gwałtowniej się zmieniają, co powoduje, że robot czasami nie zdąża wykonać pełnego obrotu o 90 stopni, bądź wykona trochę niedokładny bok kwadratu. Wartości gwałtowniej się zmieniają, gdyż jest to symulacja dyskretna i nie istnieje pojęcie prędkości jako funkcji ciągłej. W każdej dyskretnej chwili czasu prędkość jest modyfikowana przez program, co powoduje pewną nieciągłość w funkcji prędkości od czasu. Zmniejszenie okresu próbkowania (zwiększenie częstotliwości działania węzła) spowodowałoby wygładzenie funkcji prędkości, a co z tym idzie zwiększenie dokładności, lecz dla mniejszych okresów próbkowania *ROS* zużywa zbyt wiele zasobów. Dodatkowo, tak jak wspomniane zostało już wcześniej, symulator *Gazebo* nie do końca odzwierciedla, to co się faktycznie dzieje. Daje to przestrzeń do policzenia kolejnego typu błędów.

Kluczowymi chwilami dla trajektorii w profilu trapezu są te co wyżej plus koniec przyspieszania i początek spowalniania.

Wykresy przedstawiające zależność błędu chwilowego w czasie eksperymentu dla różnych wartości badanego parametru - przyspieszenia:

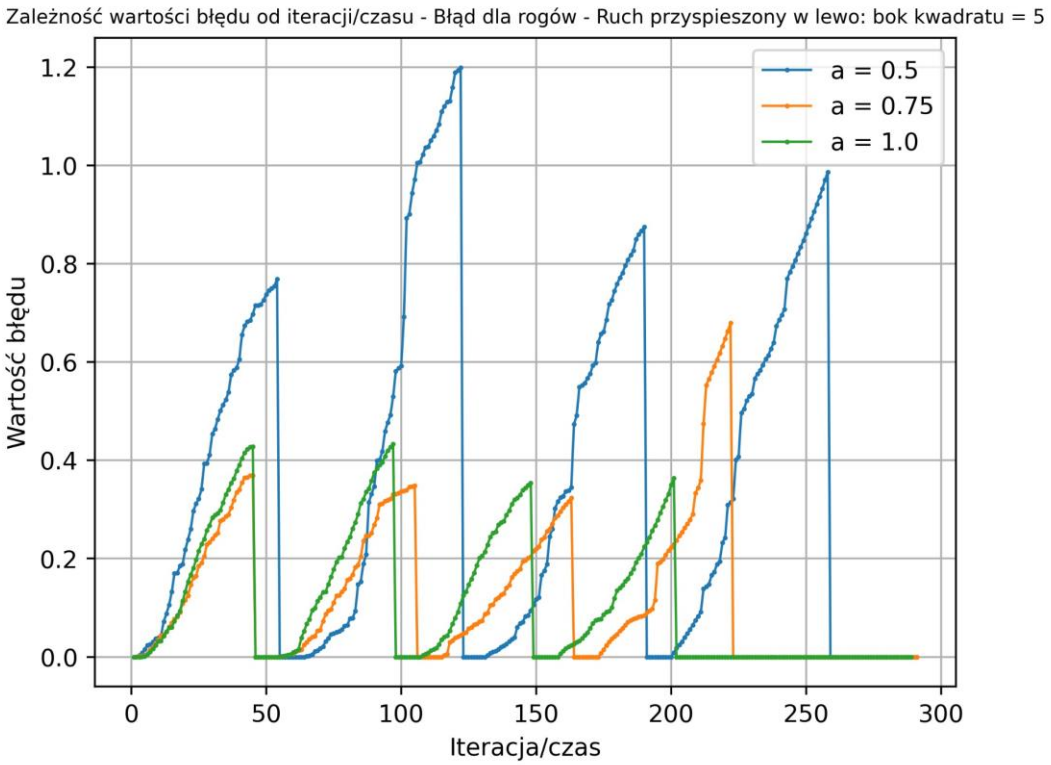


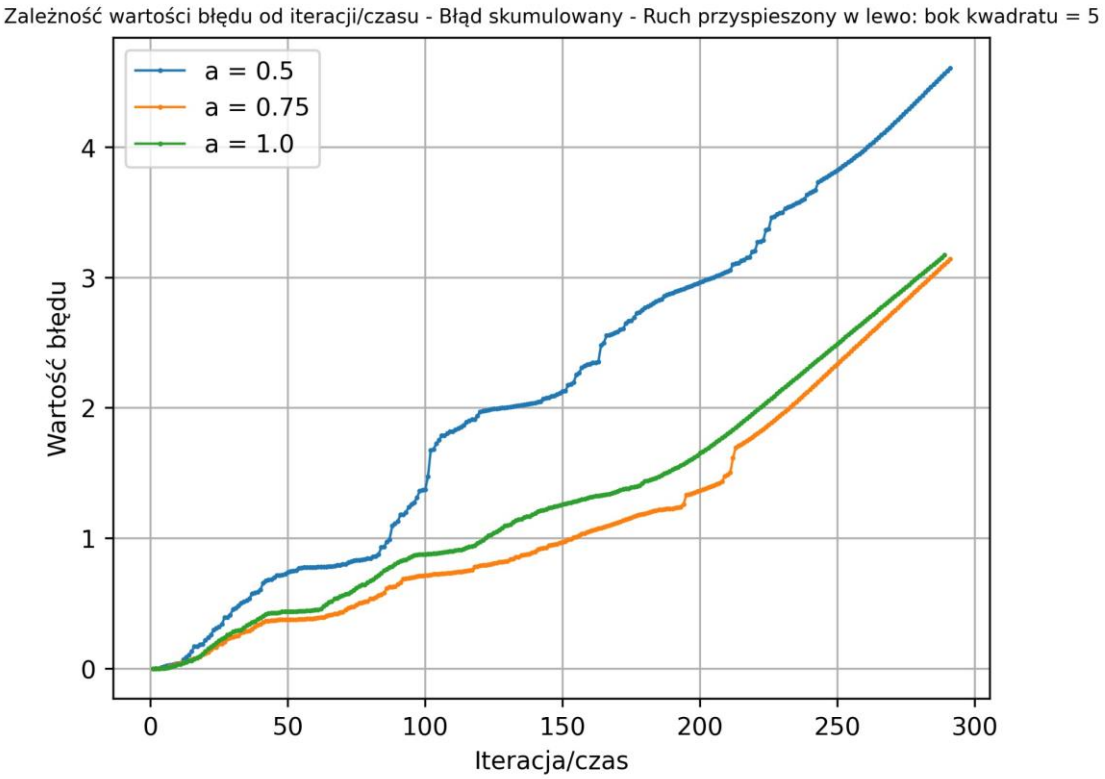
Jak widać na powyższym wykresie, symulacja mniejszego przyspieszenia powoduje większe błędy. Taka anomalia mogła zajść z powodu braku opóźnień w kodzie, to znaczy przykładowo *Gazebo* oraz węzeł nadający prędkość mogły się uruchomić kompletnie rozsynchronizowane, co mogło powodować nienaturalne błędy przy niższym przyspieszeniu. Ponadto dość widoczne jest, że najwyższe wartości błędu chwilowego są w momencie, gdy robot jedzie ruchem jednostajnym. Wtedy też osiąga on najwyższą wartość i różnica między tym aktualnym położeniem a tym w Gazebo może być także największa.

Przyspieszenie	0,5	0,75	1,0
Błąd skumulowany	3,967	1,838	1,680
Skumulowany błąd dla rogów kwadratu	3,829	1,720	1,578

Skumulowane błędy mogą wydawać się podobne, a nawet mniejsze w porównaniu do ruchu o profilu trapezowym. Gdy mamy większe przyspieszenie, błąd się zmniejsza (analogicznie jak poprzednio) z powodu krótszego czasu jazdy robota. Jeżeli czas trwa dwa rady dłużej, to błąd jest liczony dwa razy więcej, stąd też możliwe większe błędy skumulowane.

Wykresy przedstawiające błędy skumulowane oraz błędy na pojedynczych bokach dla tego typu ruchu znajdują się poniżej.

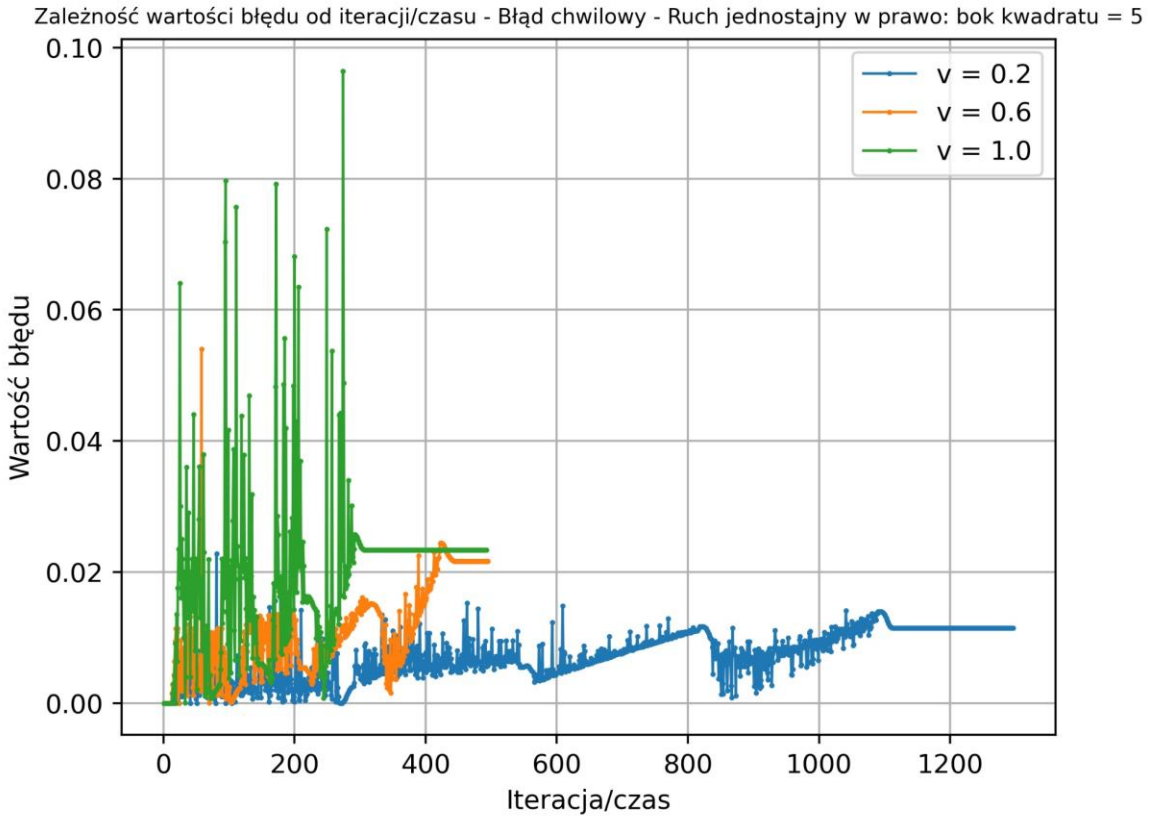




3. **Jak wpływa kierunek wykonywania kwadratu na jakość odometrii (zakładając te same prędkości ruchu)?**

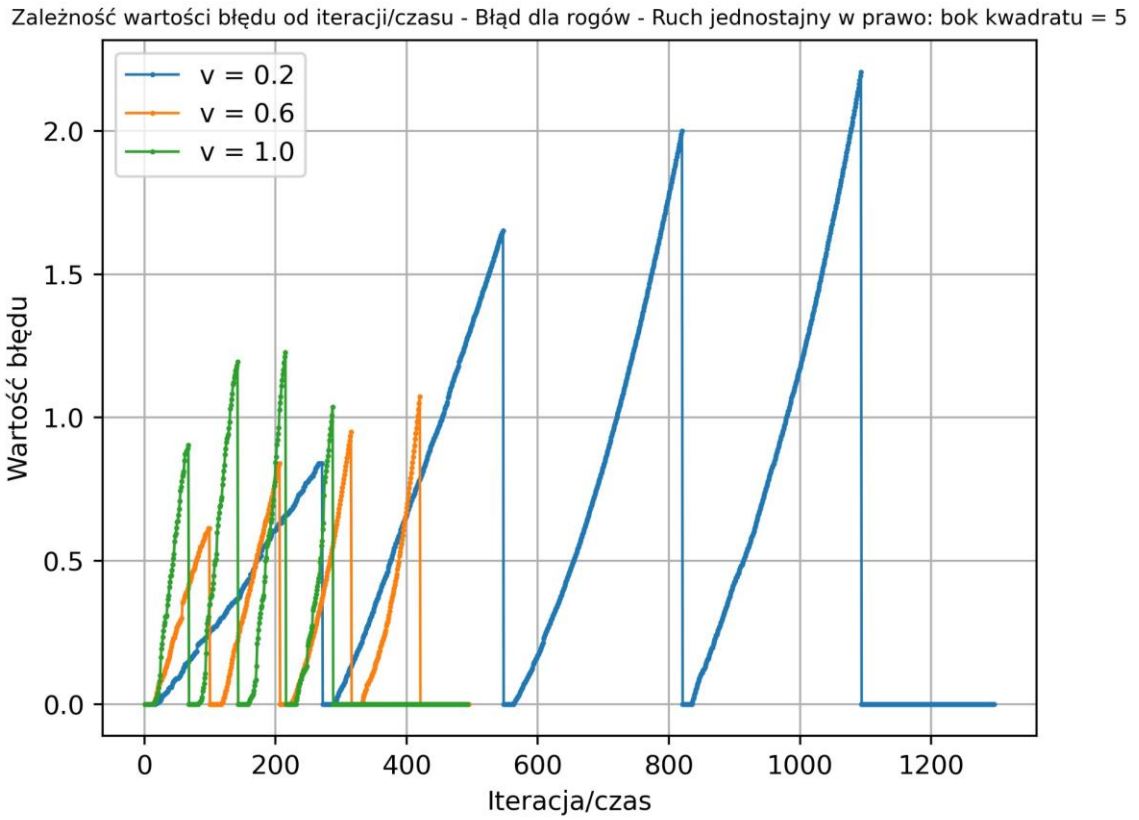
Testy zostały wykonane dla różnych wartości parametru wartości kierunku kwadratu *lewo* lub *prawo*. W związku, iż w poprzednich podpunktach zostały wykonane testy dla parametru kierunku *lewo,* postanowiliśmy w tym podpunkcie pokazać jedynie wyniki dla tych samych parametrów, co w podpunkcie 1 i 2 (dla obu kombinacji typu ruchu). Jednakże wyniki poniżej są przedstawione dla parametru kierunku *prawo.*

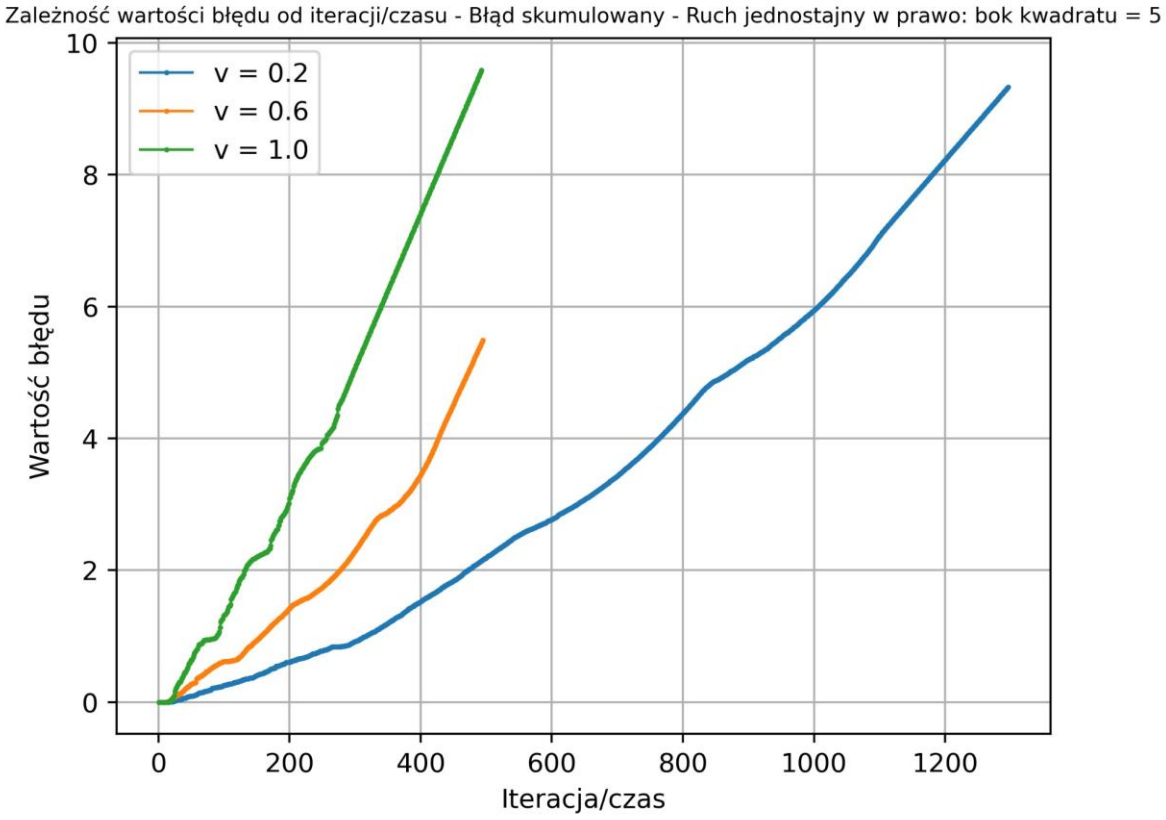
- Przy sterowaniu skokowym:



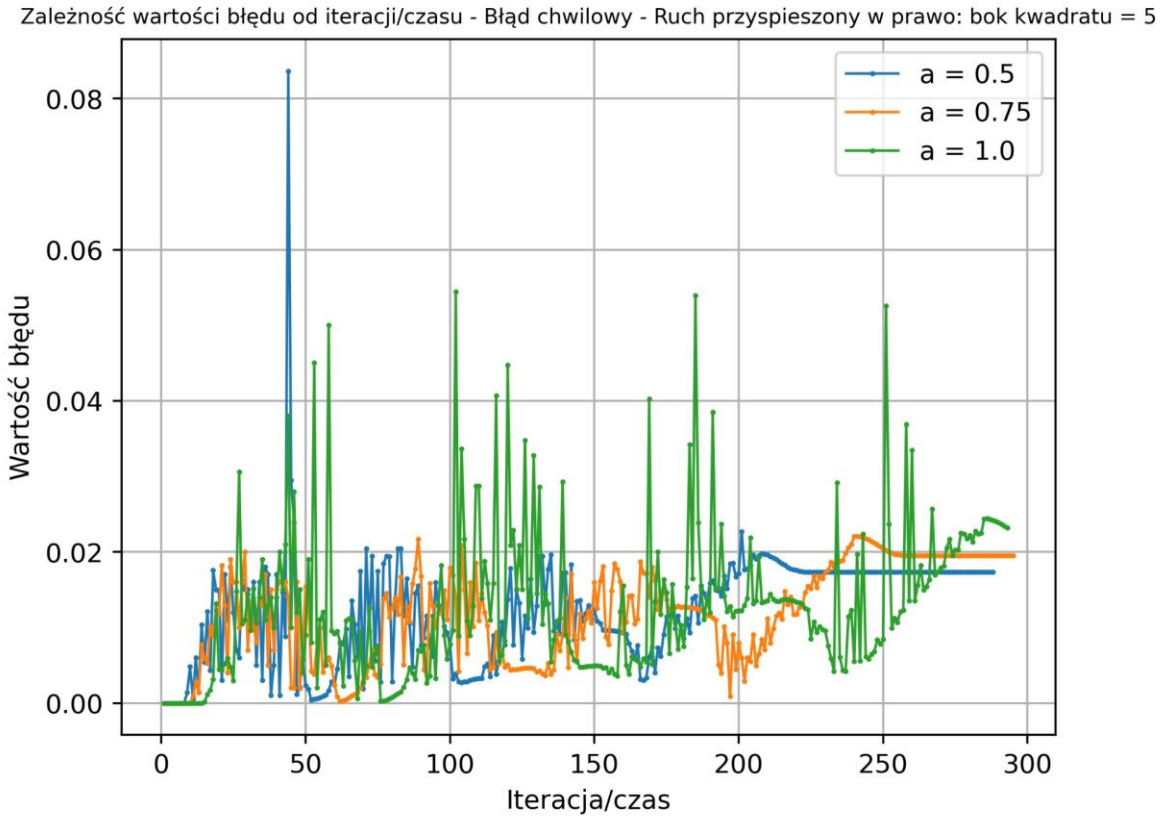
Jak widać na powyższym wykresie, większa prędkość powoduje większe błędy.

Prędkość	0,2	0,6	1,0
Błąd skumulowany	6,987	3,857	4,780
Skumulowany błąd dla rogów kwadratu	6,698	3,475	4,362


Zależność wartości błędu od iteracji/czasu - Błąd dla rogów - Ruch jednostajny w prawo: bok kwadratu = 5
v = 0.2
v = 0.6
v = 1.0
Wartość błędu
Iteracja/czas
0.0
0.5
1.0
1.5
2.0
0
200
400
600
800
1000
1200

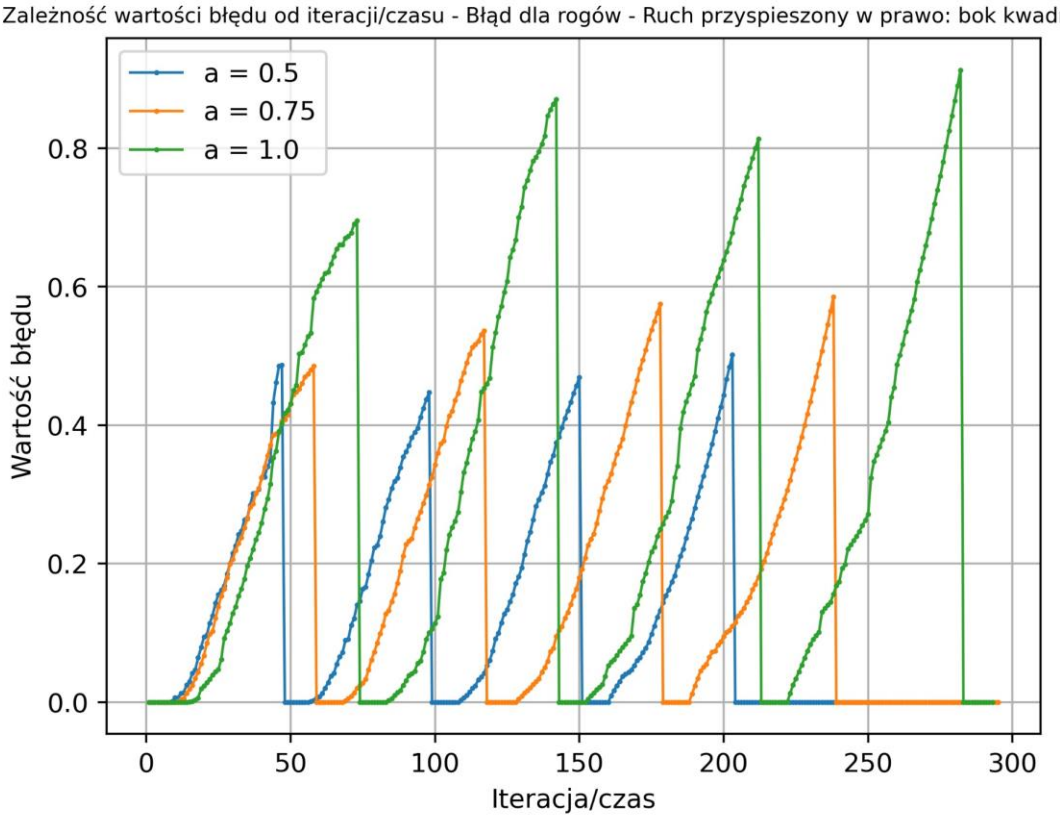

Zależność wartości błędu od iteracji/czasu - Błąd skumulowany - Ruch jednostajny w prawo: bok kwadratu = 5
v = 0.2
v = 0.6
v = 1.0
Wartość błędu
Iteracja/czas
0
2
4
6
8
10
0
200
400
600
800
1000
1200

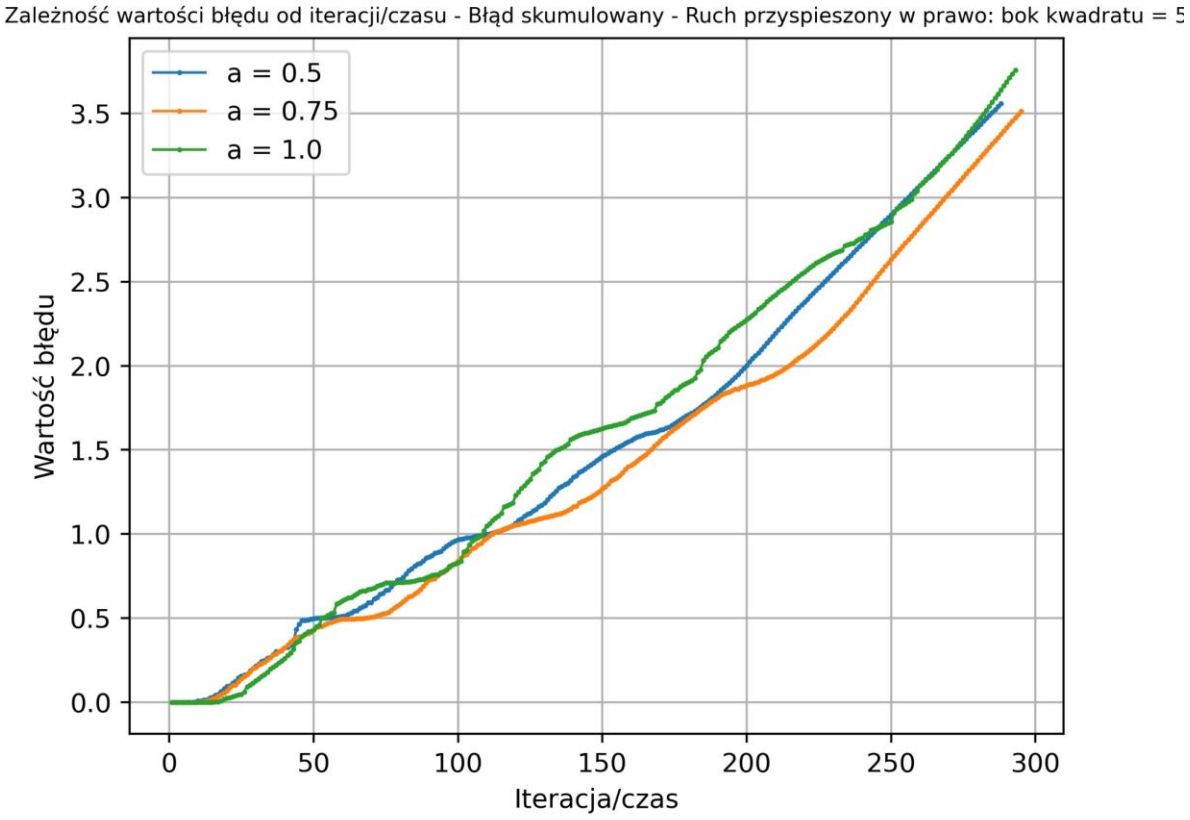
- Przy trapezowym profilu prędkości:



Jak widać na powyższym wykresie, większe przyspieszenie powoduje większe błędy.

Przyspieszenie	0,5	0,75	1,0
Błąd skumulowany	2,078	2,398	3,521
Skumulowany błąd dla rogów kwadratu	1,910	2,183	3,293


Zależność wartości błędu od iteracji/czasu - Błąd dla rogów - Ruch przyspieszony w prawo: bok kwadratu = 5
a = 0.5
a = 0.75
a = 1.0
Wartość błędu
0.0
0.2
0.4
0.6
0.8
0
50
100
150
200
250
300
Iteracja/czas


Zależność wartości błędu od iteracji/czasu - Błąd skumulowany - Ruch przyspieszony w prawo: bok kwadratu = 5
a = 0.5
a = 0.75
a = 1.0
Wartość błędu
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
0
50
100
150
200
250
300
Iteracja/czas

Wykresy oraz dane w tabelach możemy porównać z wynikami w poprzednich podpunktach. Dość zauważalnym faktem, jest to, że przy zwiększaniu wartości przyspieszenia przy skręcie w lewo, błąd maleje. Natomiast przy skręcie w prawo rośnie. Dodatkowo jazda oraz obracanie o 90 stopni są zauważalnie gorsze w skręcie w prawo. Podobnie jest przy sterowaniu skokowym. Błędy skumulowane są znacznie większe przy skręcie w prawo w porównaniu do skrętu w lewo. Może być to wina niedokładności *Gazebo* bądź obliczeń położeń wystawianych na temacie odometrii. Wniosek - jakość odometrii przy skręcie w prawo jest gorsza.

PODSUMOWANIE

Laboratorium 2. przyniosło skuteczne narzędzia do badania odometrii robota z bazą różnicową. Węzeł zadający trajektorię umożliwia sterowanie ruchem robota po kwadracie, podczas gdy węzeł liczący błąd dostarcza istotne informacje dotyczące dokładności ruchu w porównaniu z założeniami trajektorii. Otrzymane wyniki pozwalają na weryfikację i optymalizację parametrów ruchu robota. Dość ważną informacją jest to, że symulator *Gazebo* nie do końca odzwierciedla, to co się faktycznie dzieje. W związku z tym liczone były błędy, z których odpowiednio można było wyciągnąć wnioski.