Politechnika Wrocławska
Wydział Informatyki i Telekomunikacji

Kierunek: **Informatyka algorytmiczna (INA)**

PRACA DYPLOMOWA MAGISTERSKA

Mechanizm multilateracji w rozproszonej sieci sensorów audio

Multilateration mechanism in distributed net of audio sensors

Gabriel Budziński

Opiekun pracy
dr inż. Przemysław Błaśkiewicz

Słowa kluczowe: multilateracja, sensory audio, synchronizacja czasu

WROCŁAW, 2024

Streszczenie

Problem pozycjonowania w przestrzeni na podstawie emitowanego dźwięku obiektu pozycjonowanego wiąże się z wykorzystaniem możliwie zsynchronizowanych w czasie węzłów (mikrofonów) i pomiarze różnic czasu odbioru dźwięku przez czujniki. W pracy zostanie zbudowana sieć (co najmniej 4 sztuki) sensorów audio połączonych bezprzewodowo między sobą i ze stacją główną. Zadaniem sieci będzie wskazanie lokalizacji w przestrzeni punkowego przedmiotu emitującego dźwięk. Oprócz wyboru i implementacji algorytmu multilateracji zaproponowane zostanie rozwiązanie problemu synchronizacji czasu między sensorami, minimalizacji opóźnienia w komunikacji oraz kalibracji systemu.

Słowa kluczowe: multilateracja, sensory audio, synchronizacja czasu

Abstract

The problem of positioning in space based on the emitted sound of the positioned object involves the use of as closely synchronized nodes (microphones) as possible in time and measuring the differences in the time of sound reception by sensors. In the work, a network (of at least 4 units) of audio sensors connected wirelessly to each other and to the main station will be built. The network's task will be to indicate the location in space of a point-like object emitting sound. In addition to selecting and implementing the multilateration algorithm, a solution to the problem of time synchronization between sensors, minimizing communication delay, and system calibration will be proposed.

Keywords: multilateration, WASN, clock synchronization

Spis treści

Wstęp

Multilateracja jest metodą lokalizacji opartą na odległościach od punktów o znanych współrzędnych do szukanego punktu aby określić jego położenie. Zazwyczaj odległości te uzyskuje się na podstawie czasu, w którym sygnał dotarł do odbiornika (*ang. time of arrival, TOA*). Problem tego typu jest podstawowym w wielu współczesnych serwisach lokalizacyjnych, takich jak GPS (*ang. Global Positioning System*) i był od badany przynajmniej od lat '60, ze znacznymi postępami poczynionymi w latach '70 podczas opracowywania systemu GPS, a w latach '90 w związku z błyskawicznym wzrostem liczby telefonów komórkowych i potrzebą lokalizacji osób komunikujących się przy ich użyciu z służbami ratunkowymi [3]. W ostatnich latach coraz większym zainteresowaniem cieszy są zastosowanie metod multilateracji w sieciach urządzeń IoT [1] w celu autonomicznego uzyskiwania informacji lokalizacji przestrzennej bez potrzeby użycia dedykowanej do tego celu aparatury.

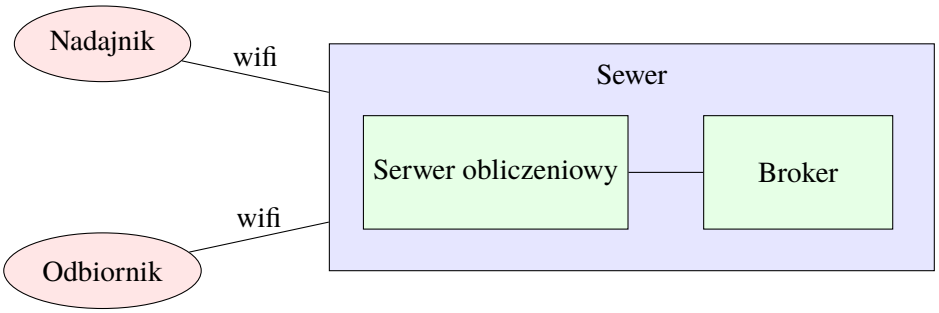
Na przestrzeni lat rozwijano wiele wariantów rozwiązań problemu lokalizacji na podstawie czasu otrzymania sygnału. Pierwszym były metody iteracyjne, dzięki którym w prosty sposób otrzymywano rozwiązanie równań nieliniowych, ale nie gwarantowały ani zbiegania do poprawnego rozwiązania, ani krótkiego czasu otrzymania go. W latach '80 wprowadzono metody manipulacji układami równań tak by otrzymać równanie kwadratowe jednej niewiadomej dające dwa rozwiązanie, z których jedno było poprawne. Wreszcie w połowie lat '90 zaczęto używać układów równań liniowych, których rozwiązanie otrzymywano dzięki programowaniu liniowemu. Zaletą tej metody jest jej wszechstronność, ponieważ daje możliwość łatwego rozszerzenia problemu o więcej pomiarów. Te dodatkowe informacje mogą być dodatkowymi pomiarami TOA lub każdymi innymi związanymi z problemem przy pomocy równań liniowych. Dzięki temu mogły rozwinąć się algorytmy lokalizacji wykorzystujące kąt przybycia (*ang. angle of arrival, AOA*), różnicę częstotliwości (*ang. frequency difference of arrival, FDOA*), moc odebranego sygnału (*ang. received signal strength, RSS*) oraz kierunek przybycia[1] (*ang. direction of arrival, DOA*). W poniższej pracy skupimy się na podstawowym wariancie multilateracji, czyli TOA.

[1] Różnicą między DOA a AOA jest odległość, z której oodbierany jest sygnał. DOA zakłada, że źródło sygnału jest na tyle daleko, że AOA jest wspólny dla wszystkich odbiorników. W AOA różnice pomiarów są na tyle duże, aby móc użyć ich do triangulacji.

Rozdział 1

Sprzęt systemowy

Niemal wszystkie prace adresujace temat multilateracji opierają się na systemach urządzeń działających w zakresie fal elektromagnetycznych. W tej pracy poświęcimy uwagę systemowi działającemu w domenie fal dźwiękowych, jak ten aspekt wpływa na skuteczność i dokładność rozwiązania problemu multilateracji.



1.1. Broker MQTT

Urządzenia systemowe porozumiewają się przy uzyciu protokołu MQTT. Każdy z węzłów oraz serwer łączą się z centralnym brokerem, który przekierowuje wiadomości opatrzone tematem do klientów subsrybujących dany temat. Broker jest oparty o otwarty projekt *Mosquitto*.

1.2. Serwer obliczeniowy

Centralnym urządzeniem systemu jest serwer obliczeniowy kumulujący dane otrzymane z sensorów do rozwiązania problemu multilateracji. Do implementacji serwera zdecydowano się użyć języka `python` ze względu na dostępność bibiotek oferujących narzędzia potrzebne do działania serwera, takie jak pakiet `paho` oferujący klienta MQTT czy pakiet matematyczny `numpy`. Ponadto prostota składni tego języka pozwoliła na szybkie wdrażanie modyfikacji działania kolejnych aspektów serwera. Serwer hostowany jest na tej samej maszynie co broker MQTT.

1.3. Węzeł

Każdy z węzłów oparty jest o mikrokontroler ESP8266-01s zaprogramowany przy użyciu Arduino IDE w języku `C++`. W systemie występują dwa rodzaje węzłów:

- nadajnik,
- odbiornik.

Nadajnik jest wyposażony jest w przełącznik cewkowy sterowany przez mikrokontroler, który służy do kontrolowania brzęczyka zasilanego napięciem $12V$. Wybrano brzęczyk o głośności $90dB$ w celu zmaksymalizowania zasięgu działania systemu. Odbiornik jest natomiast wyposażony w mikrofon elektretowy,

którego sygnał wzmacniany jest przez wzmacniacz operacyjny. Sygnał analogowy jest ostatecznie zmieniany na sygnał binarny na podstawie odniesienia to określonego poziomu napięcia. Czułość mikrofonu dostrajana jest ręcznie poprzez potencjometr.

Rozdział 2

Eksperyment zerowy

Po przygotowaniu komponentów systemu wstępnie zaimplementowano program rozwiązujący problem multilateracji, aby zbadać czy problem nie jest zbyt trywialny, aby opisać go w pracy, lub przeciwnym razie, na podstawie wyników eksperymentu, zastanowić się jakie przeszkody stoją na drodze do rozwiązania o zadowalającej precyzji.

2.1. Opis działania

Program zaimplementowano na podstawie rozwiązania aproksymacyjnego **??** postaci

$$\hat{\mathbf{x}} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b} \tag{2.1}$$

Współrzędne odbiorników są znane.

Programy węzłów

Program 1 Program nadajnika

```
1:  buzz ← False
2:  buzzTime ← 0
3:  lastBuzzTime ← 0
4:  syncTime ← 0
5:  function onMessage(message, topic)
6:      if topic is TOPIC then
7:          buzz ← message
8:      end if
9:      if topic is TIME_TOPIC then
10:         syncTime ← micros()
11:     end if
12: end function
13: loop
14:     if buzz and micros() − lastBuzzTime > BUZZ_INTERVAL then
15:         buzzTime ← micros() − syncTime
16:         publish(buzzTime)
17:         lastBuzzTime ← buzzTime
18:         buzzer()
19:     end if
20: end loop
```

- $micros()$ - funkcja zwracająca liczbę mikrosekund od uruchomienia urządzenia,
- $TOPIC$ - nazwa kanału MQTT korespondującego do danego węzła,

Program 2 Program odbiornika

```
1:  micState ← LOW
2:  micTime ← False
3:  lastMicTime ← 0
4:  syncTime ← 0
5:  function onMessage(topic)
6:      if topic is TIME_TOPIC then
7:          syncTime ← micros()
8:      end if
9:  end function
10: function micInterrupt
11:     micState ← HIGH
12: end function
13: loop
14:     if micState is HIGH and micros() − lastMicTime > MIC_INTERVAL then
15:         micTime ← micros() − syncTime
16:         publish(micTime)
17:         lastMicTime ← micTime
18:         micState ← LOW
19:     end if
20: end loop
```

- $TIME\_TOPIC$ - nazwa kanału MQTT służącego do przesyłania wiadomości o synchronizacji czasu.
- $publish()$ - funkcja publikująca wiadomość na zadanym kanale MQTT

Program serwera

Program 3 Program serwera

```
1:  M ← (A^T A)^{-1} A^T
2:  function calc_position
3:      for node in NODES do
4:          d ← (SS/10^6) · time(node) − time(source)
5:          b_node ← d − Σ coords(node)^2
6:      end for
7:      return M · b
8:  end function
9:  loop main
10:     sleep(t_1)
11:     calc_position()
12: end loop
13: loop sync_clock
14:     publish(TIME_TOPIC)
15:     sleep(t_2)
16: end loop
```

- A - macierz zawierająca współrzędne obiorników z równania **??**
- SS - prędkość dźwięku $[\frac{m}{s}]$,
- $time()$ - czas dostarczony w ostatniej wiadomości od węzła,
- $coords()$ - współrzędne węzła,
- t_1 - czas pomiędzy obliczeniami pozycji,
- t_2 - czas pomiędzy wiadomościami synchronizującymi zegary.

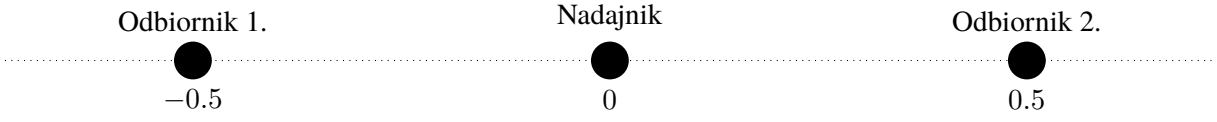
Opis algorytmu

W programie serwera zaimplementowano funkcję rozwiązującą równanie **??** przy pomocy funkcji bibliotecznych dostępnych w pakiecie `numpy`. Wywołania następują co interwał t_1. Co interwał t_2 serwer wysyła wiadomość synchronizującą z założeniem, że węzły odbierają ją równocześnie i w ten sposób synchronizują zegary.

2.2. Ewaluacja działania systemu

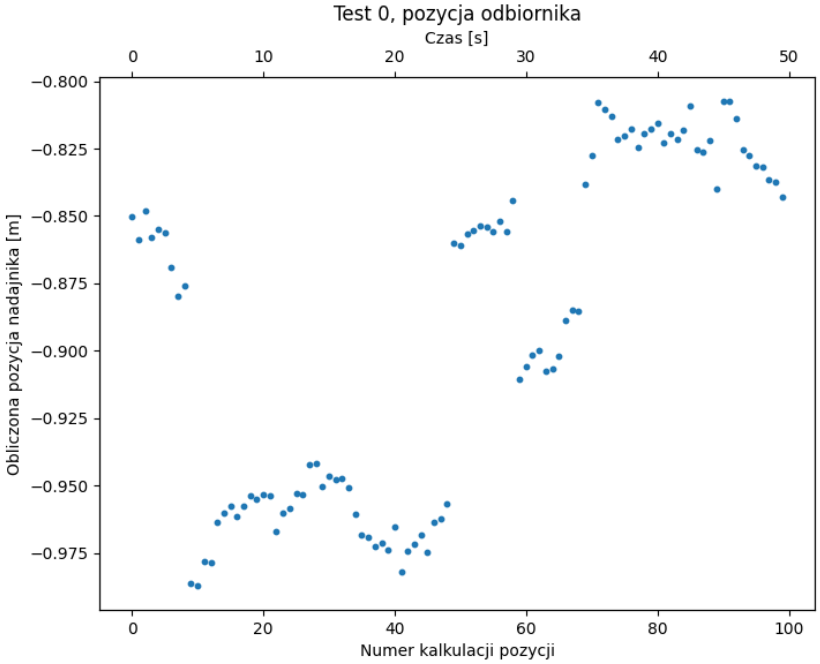
Pierwsze testy przeprowadzone zostały na systemie operującym w jednym wymiarze (to jest wszystkie węzły są współliniowe) w celu jak największego uproszczenia, pozwalającego na szybsze wykrywanie i rektyfikację błędów. Wszystkie odległości będą podawane w metrach.

Struktura systemu testowego

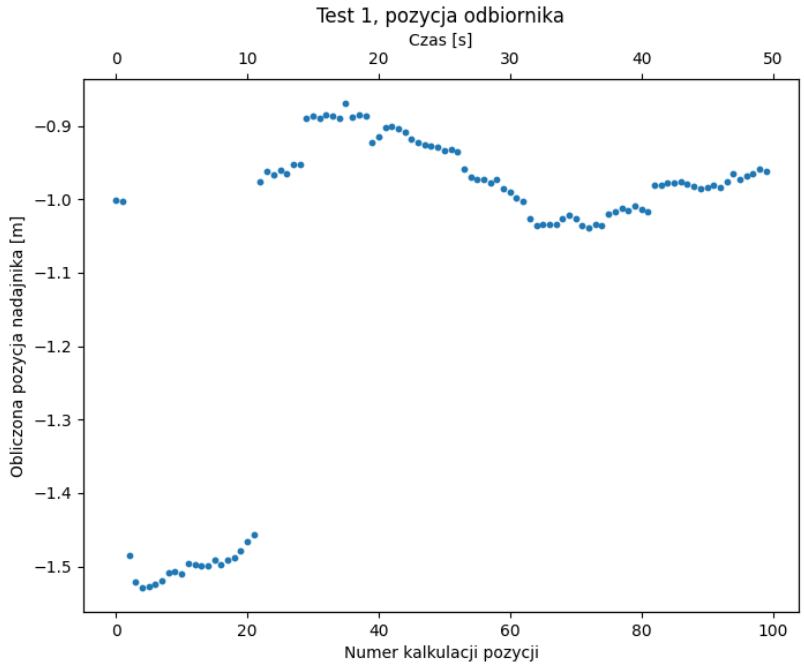


Rys. 2.1: Układ systemu testowego

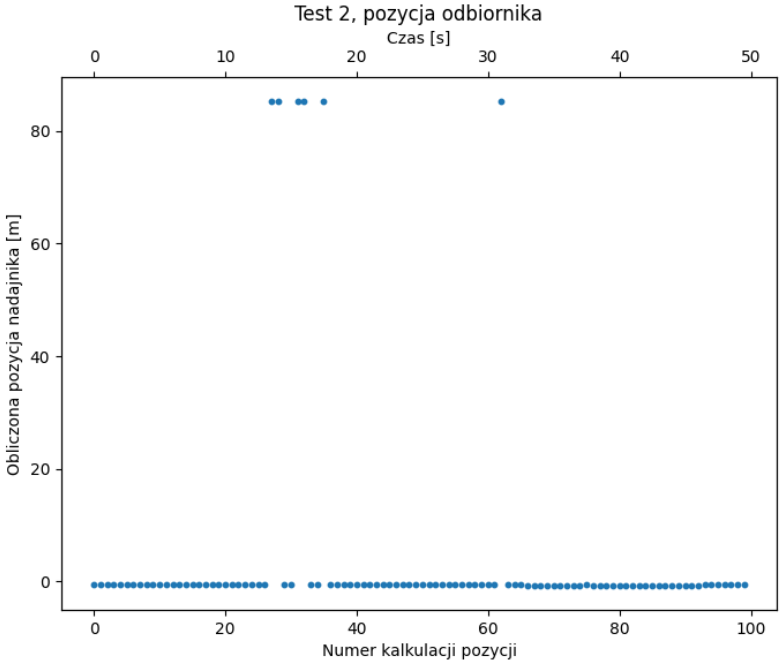
Nadajnik umiejscowiono w punkcie (0), natomiast dwa odbiorniki w punktach odpowiednio (-0.5) i (0.5), wszystkie węzły były stacjonarne. Nadajnik co $0.5\ s$ nadawał sygnał o długości $10\ ms$, a serwer co $0.5\ s$ zwracał wynik zagadnienia multilateracji na podstawie ostatnio otrzymanych danych. Wykonano 5 następujących bezpośrednio po sobie eksprymentów, poniżej przedstawiono wyniki trzech pierwszych, ponieważ są wystarczające do ukazania zachodzącego trendu.



Rys. 2.2: Wykres obliczonej pozycji odbiornika w zależności od czasu



Rys. 2.3: Wykres obliczonej pozycji odbiornika w zależności od czasu



Rys. 2.4: Wykres obliczonej pozycji odbiornika w zależności od czasu

2.3. Interpretacja wyników i wnioski

Już na pierwszym z wykresów widać, że wartości wyjściowe algorytmu są niestabilne, współrzędna nadajnika wacha się w przedziale $(-0.98, -0.8)$, a także co jakiś czas następują nagłe przeskoki między następującymi po sobie wartościami. Coraz większa rozbieżność następuje wraz z upływem czasu. Już w drugim teście różnica ekstremalnych odchyleń jest większa niż odległość między nadajnikami, a dokładność jest absolutnie niezadowalająca. Kolejne testy jedynie utwierdzają te obserwacje.

Na niską dokładność i stabilność wyników może mieć wpływ wiele czynników takich jak:

- niepoprawna synchronizacja zegarów węzłów,
- niepoprawna kalibracja czułości mikrofonów,
- fałszywe odczyty wynikające z odbić fali dźwiękowej,
- niska jakość rozwiązania aproksymacyjnego.
- TODO

Obserwując skoki pomiędzy sąsiadującymi obserwacjami położenia rozdzielającymi okresy względnej stabilności i mając na uwadze statyczność otoczenia systemu możemy śmiało wysnuć wniosek, że tak nagła zmiana średniej obliczanej wartości może wynikać najprawdopodobniej z błędnej synchronizacji czasu. Wygląda na to, że sygnał wysyłany przez serwer obliczeniowy nie jest odbierany we wszystkich węzłach równocześnie. Następny rozdział będzie poświęcony badaniu i tworzeniu algorytmu synchronizacji zegarów w węzłach.

Rozdział 3

Synchronizacja czasu

Synchronizacja czasu jest jednym z podstawowych problemów systemów opartych o wiele urządzeń posiadających własne zegary. Wiedza o dokładnym czasie zachodzących w sieci zdarzeń jest niezbędna do szybkiego przesyłu danych, koordynacji procesów czy aktualizacji systemu plików. W przypadku niniejszej pracy dokładne określenie czasu zachodzących zdarzeń jest kluczowe do wiarygodnego określenia odległości pomiędzy węzłami na podstawie interwału czasowego pomiędzy nadaniem a odbiorem sygnału dźwiękowego.

3.1. Synchronizacja programowa

Pierwszym podejściem do rozwiązania problemu synchronizacji zegarów było zastosowanie synchronizacji programowej [4], w której urządzenie-host utrzymuje wysokiej rozdzielczości licznik czasowy i wysyła sygnały o jego wartości pozostałym węzłom w sieci. Na podstawie tych informacji każdy z węzłów oblicza różnicę w zegarach i wprowadza odpowiednie przesunięcie własnego licznika.

Algorytm synchronizacji NTP

Jednym z najbardziej rozpowszechnionych protokołów synchronizacji programowej jest *Network Time Protocol* opisany w pracy [2]. Zachowując oznaczenia w niej zastosowane opiszmy w skrócie zasadę działania tego protokołu.

Na rysunku 3.1 przestawiono schemat działania protokołu NTP. Urządzenia *A* i *B* wymieniają wiadomości zawierające sygnatury czasowe. Niech $T_i,\ T_{i-1},\ T_{i-2},\ T_{i-3}$ będą czterema ostatnimi wiadomościami oraz niech $a = T_{i-2} - T_{i-3}$ oraz $b = T_{i-1} - T_i$. Wtedy całkowity czas transmisji δ_i i przesunięcie zegara θ_i urządzenia *B* względem urządzenia *A* to

$$\delta_i = a - b \quad \text{oraz} \quad \theta_i = \frac{a+b}{2}$$

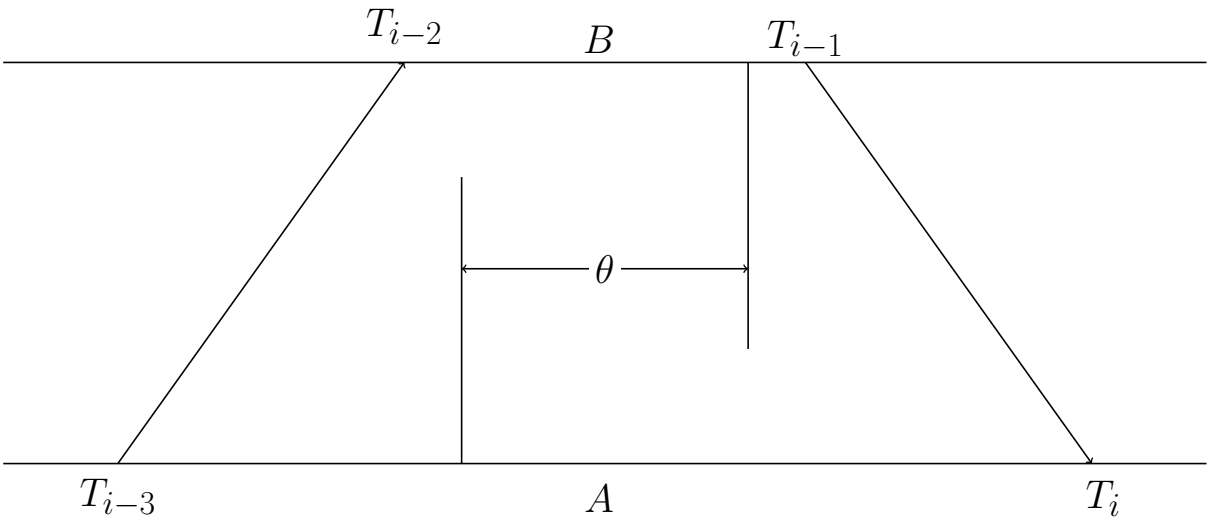
Dodatkowym wnioskiem przedstawionym w powyższej pracy jest własność prawdziwego przesunięcia względem aktualnie obliczonego:

$$\theta_i - \frac{\delta_i}{2} \leqslant \theta \leqslant \theta_i + \frac{\delta_i}{2}$$

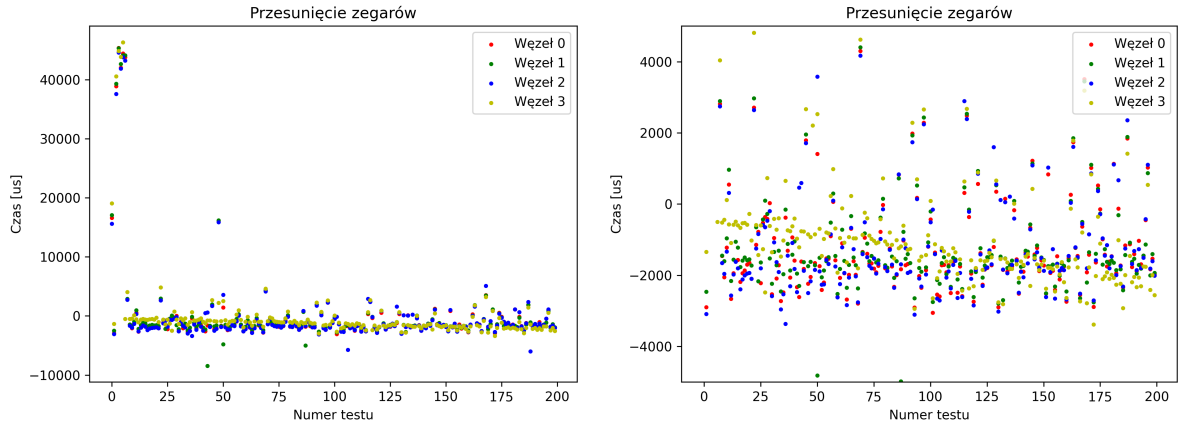
Jak łatwo zauważyć im krótszy jest czas propagacji tym lepsze przybliżenie dostajemy.

Na postawie tych informacji podjęto próbę synchronizacji programowej zegarów węzłów w systemie multilateracyjnym. Przy użyciu czterech węzłów przeprowadzono eksperymenty mające na celu próbę ustalenia przesunięć zegarów każdego z urządzeń względem centralnego serwera. Opisany wyżej schemat wymiany czterech wiadomości powtarzano n razy wspólnie dla wszystkich węzłów zapisując obliczone przesunięcia oraz czasy propagacji.

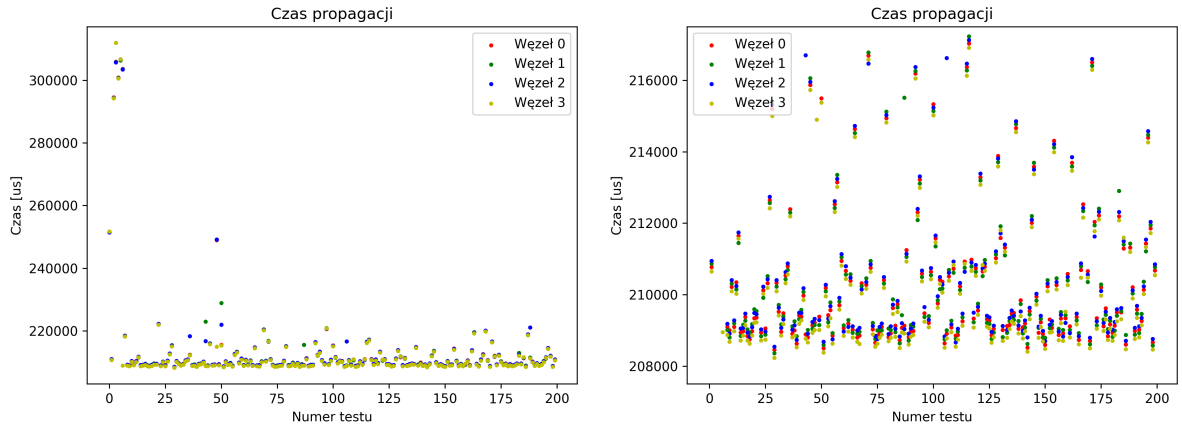
W celu poprawy czytelności wyniki przesunięć zostały znormalizowane poprzez przesunięcie o wartość średnią dla każdego z węzłów.



Rys. 3.1: Pomiar opóźnienia transmisji i przesunięcia zegara

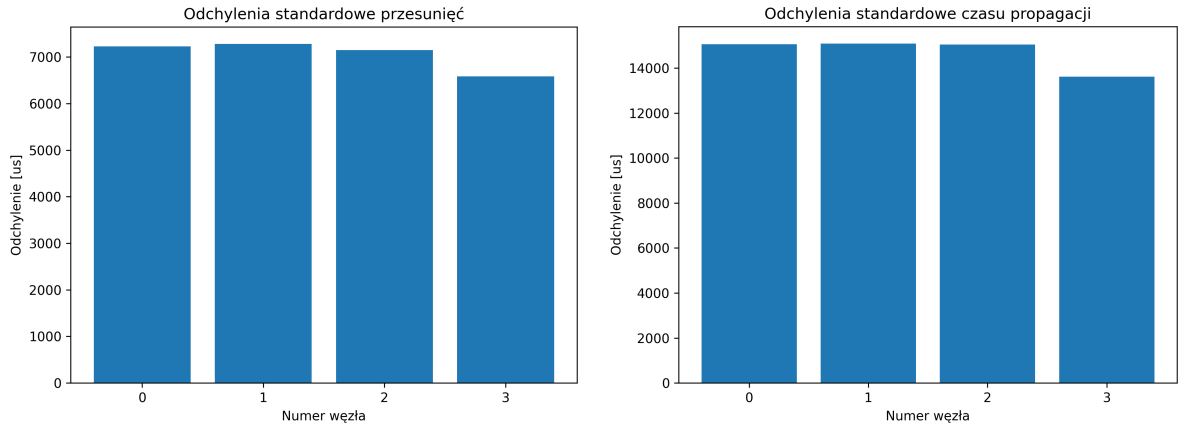


Rys. 3.2: Wyniki pomiarów przesunięć zegarów



Rys. 3.3: Wyniki pomiarów czasów propagacji

Na podstawie otrzymanych wykresów łatwo zauważyć, że czas propagacji wiadomości w systemie jest nieprzewidywalny i zmienia sie znacząco z pomiaru na pomiar. Interesującą nas statystyką są jednak zaobserwowane przesunięcia zegarów, bez których nie jesteśmy w stanie poprawnie ocenić odległości od źródła dźwięku. Tutaj wartości wyglądają na bardziej skoncentrowane, jednak nie widać wyraźnych tendencji koncentracji wokół wartości średnich dla żadnego z węzłów. Ponadto zaobserwowane odchyle-

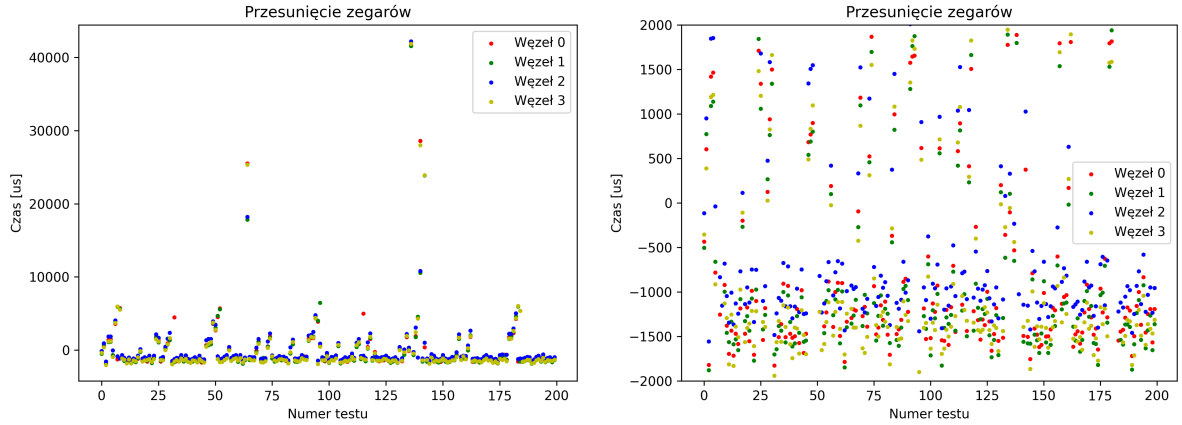


Rys. 3.4: Odchylenia standardowe pomiarów

nia standardowe $\sigma_i,\ i \in \{0,1,2,3\}$ wielkości $\approx 7000\mu s$ są nieakceptowalne, ponieważ w takim czasie dźwięk w powietrzu pokonuje $\frac{7000}{1000000}s \cdot 343\frac{m}{s} = 2.401m$. Możliwe jest, że wielokrotne powtarzanie pomiarów da zadowalającą wartość średnią, pozwalającą na centymetrową precyzję obliczanych odległości. W porównaniu z pozostałymi sposobami brane będą pod uwagę wartości uśrednione.

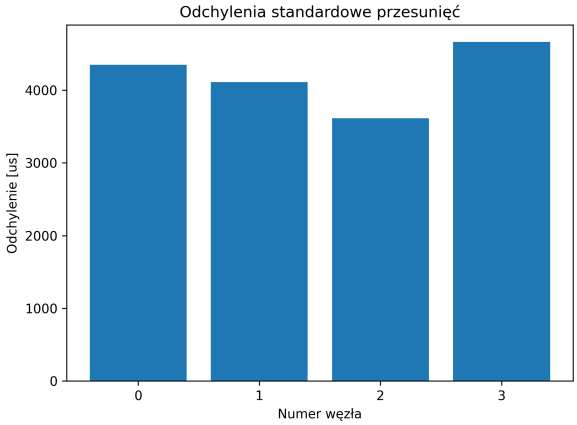
Bezpośredni pomiar przesunięć zegarów

Biorąc pod uwagę zauważoną nieprzewidywalność i rozrzut czasów propagacji (spowodowanych najprawdopodobniej użyciem protokołu MQTT do przesyłu wiadomości pomiędzy urządzeniami) następnym pomysłem schematu synchronizacji jest bezpośrednie badanie względnego przesunięcia zegarów. Węzeł wysyła n wiadomości zawierających aktualną wartość zegara, która po odebraniu przez serwer jest porównywana z zegarem w nim dostępnym.



Rys. 3.5: Wyniki pomiarów przesunięć zegarów

Wykresy przesunięć wygenerowane na podstawie tych testów na pierwszy rzut oka są skoncentrowane podobnie jak poprzednie, jednakże odchylenie standardowe są prawie dwukrotnie mniejsze niż uprzednio, co daje nadzieje na bardziej wiarygodne wyniki. W porównaniu z pozostałymi sposobami brane będą pod uwagę wartości uśrednione.



Rys. 3.6: Odchylenia standardowe pomiarów

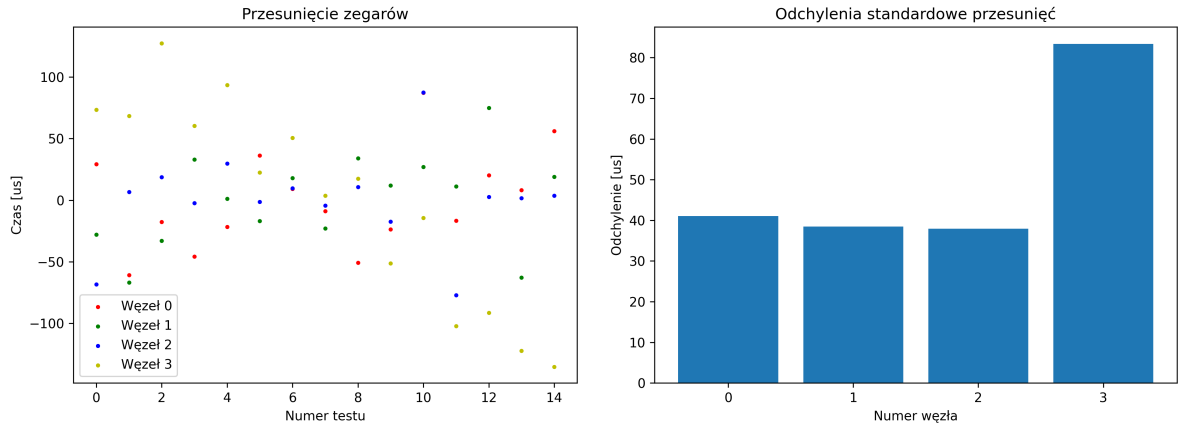
3.2. Synchronizacja sprzętowa

Synchronizacja z użyciem mikrofonów

Innym pomysłem na synchronizację zegarów w węzłach było użycie mikrofonów, w które węzły odbiorcze są wyposażone. Potrzebujmy znać jedynie przesunięcie naszego zegara względem zegara w węźle nadawczym dlatego wystarczającym będzie porównanie czasu nadania i odebrania sygnału dźwiękowego. Ponadto ten rodzaj synchronizacji w przeciwieństwie do synchronizacji programowej, która uzgadniała ze sobą jedynie zegary na podstawie wymienianych wiadomości, wlicza w czas transmisji wszelkie nie wzięte wcześniej pod uwagę opóźnienia, takie jak:

- Czas pomiędzy wysłaniem wiadomości o nadaniu sygnału a zamknięciem kontaktora i poruszeniem membraną brzęczyka,
- Czas pomiędzy odebraniem sygnału przez mikrofon a zmianą stanu zmiennej na to wskazującej.

Przeprowadzono testy tego typu synchronizacji, których wyniki przedstawiono na wykresach 3.7. W celu zwiększenia czytelności odrzucono pierwszy z pomiarów oraz przesunięto wyniki o wartość średnią.

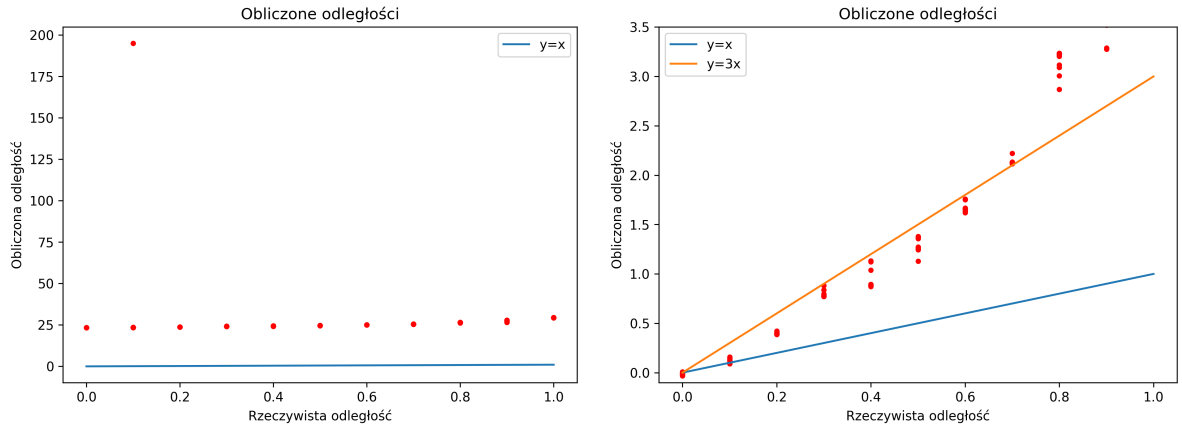


Rys. 3.7: Wyniki pomiarów przesunięć zegarów

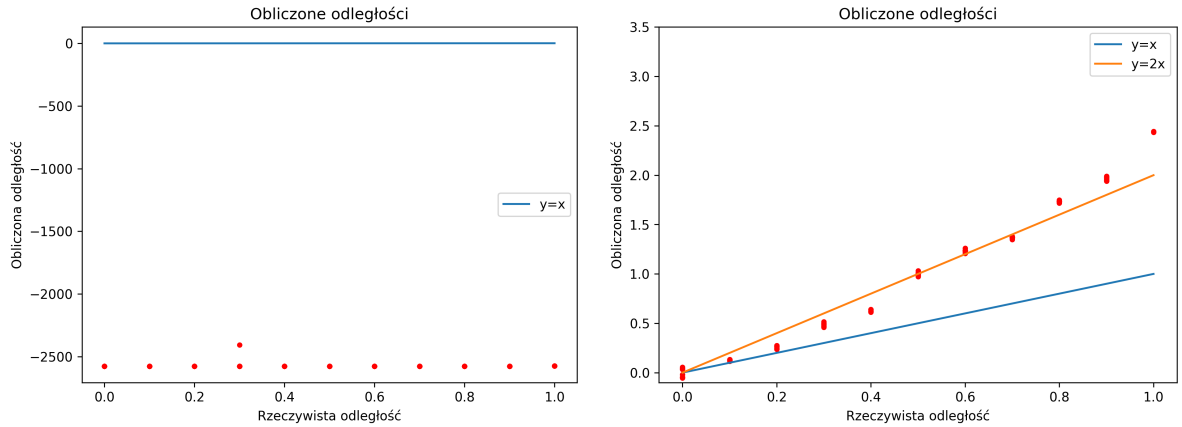
Łatwo zauważyć, ża udało się zmniejszyć odchylenia obliczonych przesunięć zegara aż o dwa rzędy wielkości. Taka dokładność daje znacznie lepsze przybliżenie rzeczywistej odległości między węzłami, ponieważ w czasie $40\mu s$ dźwięk pokona jedynie $\frac{40}{10000000} s \cdot 343 \frac{m}{s} \approx 0,014m$. Mając tak dokładne odległości będziemy mogli wprowadzić je do modelu multilateracyjnego.

Porównanie metod

Porównajmy teraz dokładność obliczanych odległości na podstawie interwału czasowego pomiędzy nadaniem dźwięku a jego odbiorem w różnych wariantach synchronizacji zegarów. Wykresy uzyskane przy zastosowaniu synchronizacji czasowej zostały przedstawione przed i po znormalizowaniu poprzez przesunięcie tak, by średnia pomiarów rozpoczynała się od 0. Pomiary oparte o synchronizację z użyciem mikrofonów nie wymagały tego dodatkowego kroku.

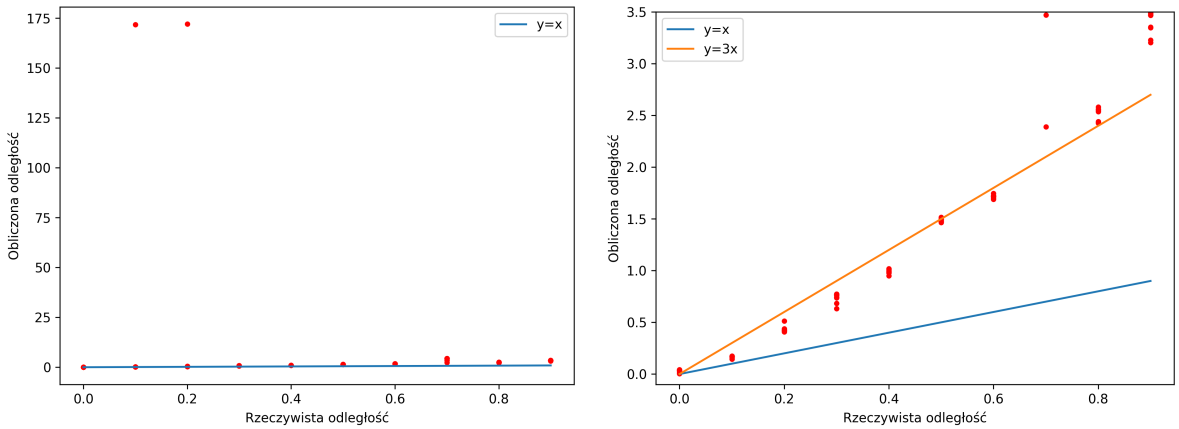


Rys. 3.8: Wyniki pomiarów odległości z użyciem synchronizacji 3.1



Rys. 3.9: Wyniki pomiarów odległości z użyciem synchronizacji 3.1

Na wykresach widać, że wszystkie trzy algorytmy synchronizacji po normalizacji dają wyniki o podobnej dokładności adekwatnej do użycia w danych wejściowych multilateracji. Ciekawą jest natomiast obserwacja skalowania obliczonych odległości. Widzimy, że w przypadku 3.8 oraz 3.10 otrzymane wartości leżą blisko trzykrotności rzeczywistej badanej odległości, natomiast w 3.9 blisko dwukrotności. W kolejnym rozdziale pochylimy się nad możliwymi przyczynami i rozwiązaniem tego zachowania.



Rys. 3.10: Wyniki pomiarów odległości z użyciem synchronizacji 3.2

Rozdział 4

Multilateriacja

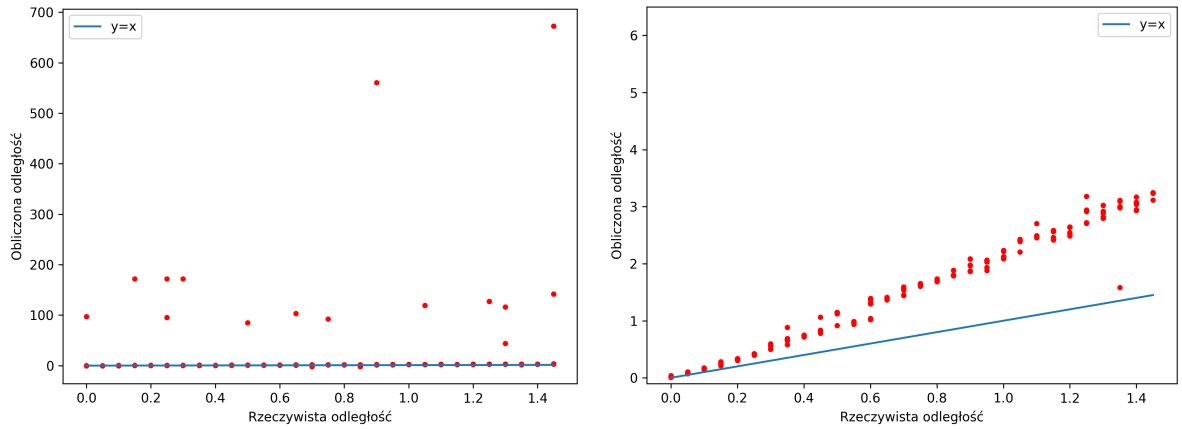
4.1. Przygotowanie danych wejściowych

Aby otrzymać poprawne wyniki algorytmu multilateracji dane wejściowe, w naszym przypadku odległości między węzłami odbiorczymi a nadajnikiem powinny być jak najbliższe rzeczywistym odległościom z możliwie małymi odchyleniami. Będziemy kontynuować usprawnianie metod uzyskiwania poprawnych wyników zapoczątkowane w rozdziale poprzednim.

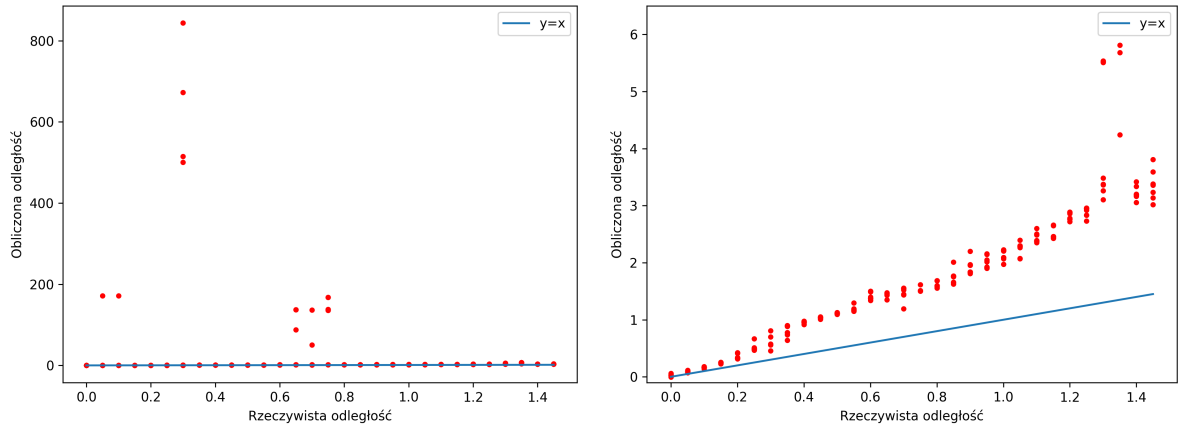
Korekcja odległości

W wynikach eksperymentów porównawczych metod synchronizacji czasu węzłów przeprowadzonych w rozdziale 3. zaobserwowaliśmy skalowanie wyników na pierwszy rzut oka zachowujące się liniowo. Przyjrzyjmy się teraz dokładnie temu zjawisku. W tym i kolejnych przypadkach będziemy używać już jedynie synchronizacji sprzętowej z użyciem mikrofonów ze względu na brak konieczności dodatkowej kalibracji przesunięcia punktu 0.

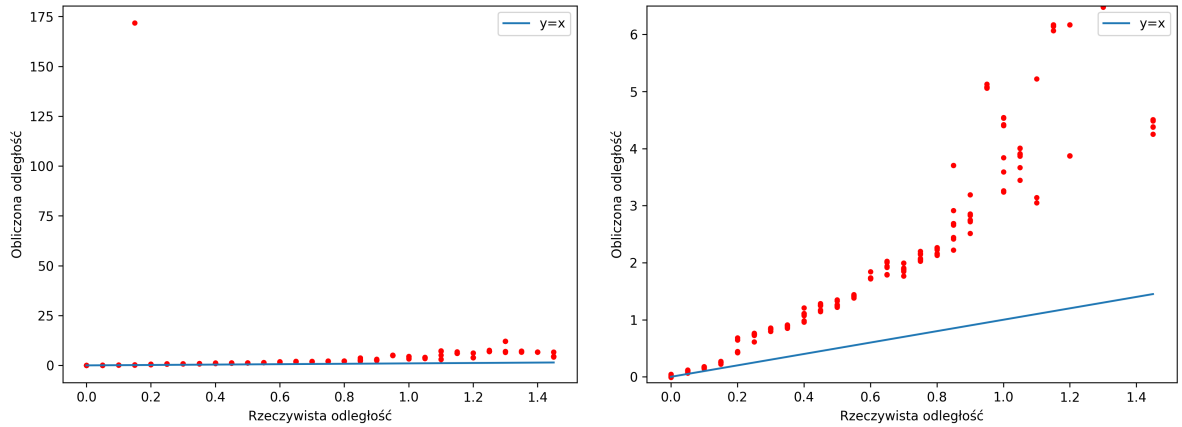
Prawdopodobnym powodem skalowania obliczanych odległości może być odczyt zmiany sygnału mikrofonowego odczytywanego przez mikrokontroler. Poniższe wykresy są wynikiem czterech kolejnych eksperymentów, w których jedyną zmienną była czułość zintegrowanego wzmacniacza mikrofonu. Wzmacniacz ten nie pozwala na precyzyjną regulację, a jedynie na zmianę rezystancji wbudowanego potencjometru. Ponieważ przedział czułości odpowiadający wykrywaniu sygnału węzła nadającego przy jednoczesnym zminimalizowaniu fałszywych aktywacji jest niewielki (około $\frac{1}{8}$ obrotu potencjometru) cztery zbadane przypadki nie dzielą równo badanego zakresu. Rozpoczynając od największej możliwej czułości przy każdym kolejnym eksperymencie zmniejszano ją póki pozwalała wciąż na wykrywanie sygnału z badanych odległości.



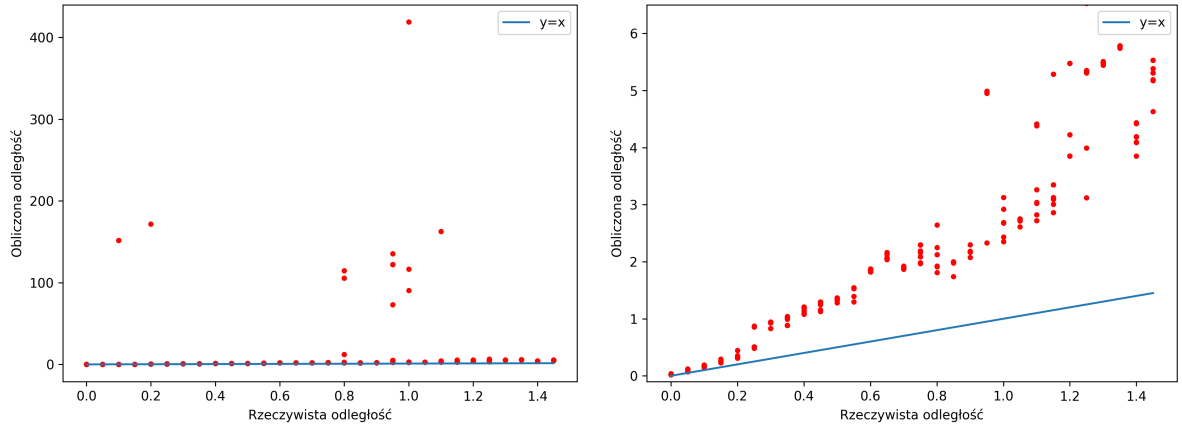
Rys. 4.1: Pomiar obliczanych odległości 1.



Rys. 4.2: Pomiar obliczanych odległości 2.



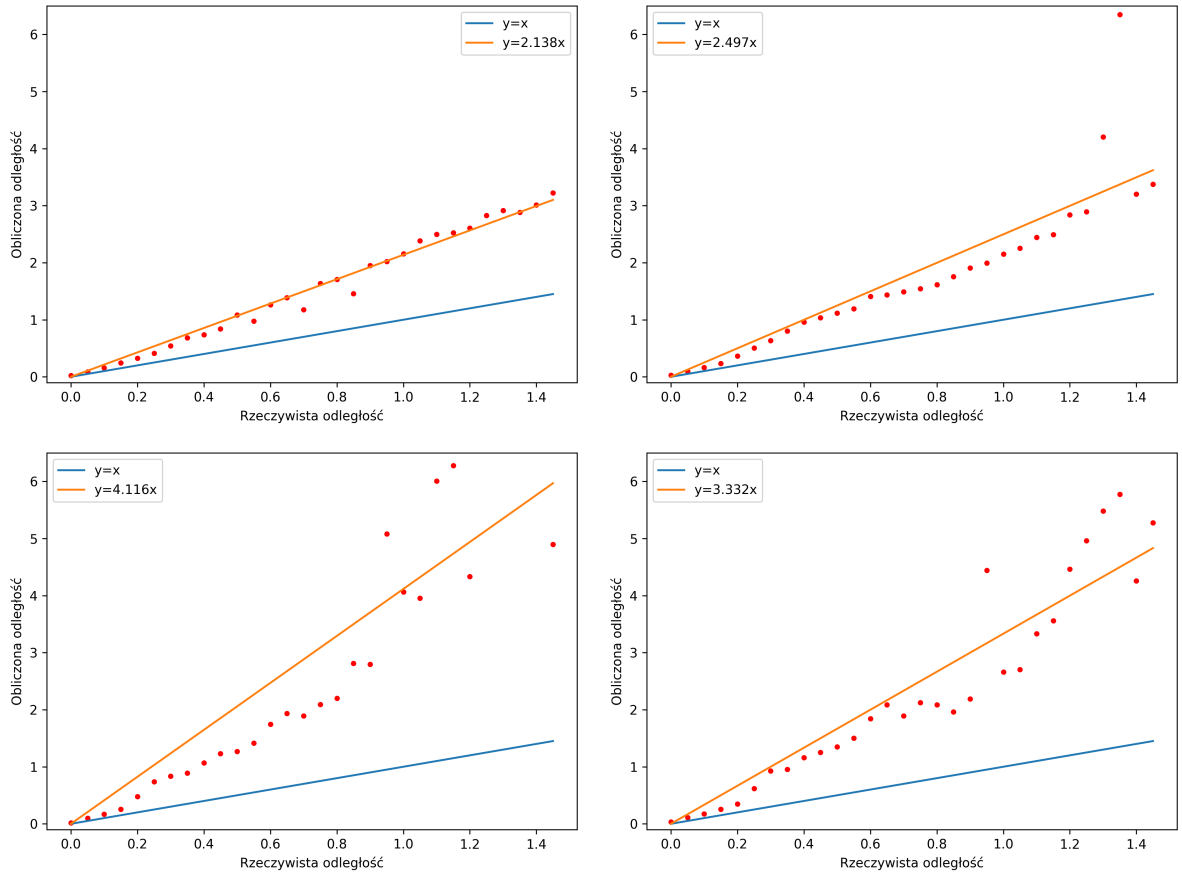
Rys. 4.3: Pomiar obliczanych odległości 3.



Rys. 4.4: Pomiar obliczanych odległości 4.

Aby lepiej odczytać informacje z wykresu uśrednijmy pomiary dla każdej z badanych odległości i dodajmy do nich funkcje liniowe o współczynniku otrzymanym przy pomocy regresji liniowej z tychże uśrednionych punktów.

Na wykresach można zauważyć trendy związane ze zmniejszającą się czułością wzmacniacza mikrofonu:



Rys. 4.5: Średnie obliczonych odległości

- zmniejszanie się odległości punktu, w którym czułość jest zbyt mała by niezawodnie wyrywać nadawane sygnały,
- współczynnik prostej aproksymującej skalowane odległości rośnie.

Podobne efekty zaobserwowano kiedy brzęczyk nie był kierowany bezpośrednio w kierunku odbiornika.

4.2. Ewaluacja działania systemu

4.3. Wyniki

Interpretacja

Wnioski

Podsumowanie

Literatura

[1] Y. Li, Y. Zhuang, X. Hu, Z. Gao, J. Hu, L. Chen, Z. He, L. Pei, K. Chen, M. Wang, X. Niu, R. Chen, J. Thompson, F. M. Ghannouchi, N. El-Sheimy. Toward location-enabled iot (le-iot): Iot positioning techniques, error sources, and error mitigation. *IEEE Internet of Things Journal*, 8(6):4035–4062, 2021.

[2] D. Mills. Internet time synchronization: the network time protocol. *IEEE Transactions on Communications*, 39(10):1482–1493, 1991.

[3] U. G. P. Office. govinfo.gov. `https://www.govinfo.gov/content/pkg/PLAW-106publ81/pdf/PLAW-106publ81.pdf`, 1999. [Dostęp 14-05-2024].

[4] Z. Yong-xing, Z. Chao, Y. Yu-lei. Research on computer time synchronization by software method. *2011 International Conference on Electronics, Communications and Control (ICECC)*, strony 998–1001, 2011.