Politechnika Wrocławska

Wydział Elektroniki, Fotoniki i Mikrosystemów

KIERUNEK: Automatyka i Robotyka (AIR)

PRACA DYPLOMOWA MAGISTERSKA

TYTUŁ PRACY:

Zadanie sterowania robota mobilnego klasy (1,2)

AUTOR:

Wojciech Chojnowski

PROMOTOR:

Dr inż. Robert Muszyński,
Katedra Cybernetyki i Robotyki

WROCŁAW 2023

Spis treści

Rozdział 1

Wstęp

Niniejsza praca jest kontynuacją badań nad tematyką dotyczącą robotów mobilnych, a w szczególności robotów klasy (1,2). W poprzednich pracach dotyczących robotów mobilnych studenci skupiali się nad skonstruowaniem modelu takiego robota oraz zaimplementowaniem algorytmów sterowania. Początkowym zamysłem było wyprowadzenie modeli pełnego i uproszczonego kinematyki robota napędzanego za pośrednictwem półsfery oraz zbadanie zachowań tych modeli w środowisku symulacyjnym [Ryb13]. W kolejnej pracy postanowiono, aby robot był napędzany już za pomocą dwóch półsfer. Wyprowadzono model kinematyki takiego robota w wersji pełnej oraz uproszczonej, która zakładała, że robot porusza się na dwóch skrętnych kołach. Wszystkie badania były prowadzone w środowisku symulacyjnym [Jon14]. Następnie postanowiono, mając dane symulacyjne stworzyć prototyp takiego robota napędzanego dwoma półsferami, który miał posłużyć do kolejnych badań nad algorytmami sterowania takim fizycznym modelem [Gó17]. Ostatnie dwie prace dotyczyły konstrukcji platformy mobilnej klasy (1,2), jednakże nie napędzały ją dwie wirujące półsfery, lecz zwykłe koła [Mej17] [Lub23]. W tej pracy zostaną opisane wybrane algorytmy sterowania robotem klasy (1,2). Jednakże wszystkie badania zostały wykonane w warunkach symulacyjnych, a nie na rzeczywistym robocie. Dlatego wszelkie wnioski wyciągnięte z tej pracy są czysto teoretyczne i niesprawdzone na rzeczywistej, poruszającej się platformie mobilnej ale mogą posłużyć do wykorzystania na fizycznym modelu, który już został wykonany przez studentów.

Celem pracy dyplomowej jest dokonanie przeglądu algorytmów sterowania możliwych do zastosowania w zadaniu sterowania robota mobilnego klasy (1,2) oraz porównanie ich własności w badaniach symulacyjnych. Cel pracy zrealizowano uwzględniając dwa aspekty. Pierwszym był aspekt inżynierski, który zakłada implementację modelu kinematyki robota mobilnego klasy (1,2), a także dwóch algorytmów pozwalających na śledzenie zadanej trajektorii. Implementacja została wykonana w środowisku programistycznym *MATLAB* [Matb] z zastosowaniem dodatkowego pakietu *SIMULINK* [Matc]. Drugi z aspektów to badawczy, który zakładał porównanie obu algorytmów poprzez symulacje. Badano je pod względem dokładności i szybkości działania, a także pod względem analizy poszczególnych parametrów robota w środowisku programistycznym.

Układ pracy jest następujący. W rozdziale 2 przedstawiono model kinematyki robota. Rozdział 3 dotyczy algorytmów sterowania, które pozwalają na rozwiązanie zadania śledzenia zadanej trajektorii. Przedstawiono wyprowadzenie wzorów dla dwóch algorytmów wraz z opisem poszczególnych elementów. Są to algorytm linearyzacji statycznej oraz algorytm linearyzacji dynamicznej. W rozdziale 4 skupiono się na implementacji algorytmów

w środowisku programistycznym w celu przeprowadzenia późniejszych badań. W rozdziale 5 dokonano dogłębnej analizy porównawczej dwóch algorytmów. Badano śledzenie zadawanej trajektorii przez algorytm. Poddano również analizie zachowanie się w czasie kątów skręcenia i obrotu kół, a także bezwzględnej orientacji korpusu robota i jego kół. Analizie podlegały jeszcze prędkości kół. Rozdział 6 zawiera podsumowanie oraz wnioski dotyczące realizowanej pracy.

Rozdział 2

Model kinematyki robota

Pierwszą czynnością, którą należy wykonać określając model kinematyczny robota mobilnego, jest wybranie wektora współrzędnych uogólnionych $q(t) \in \mathbb{R}^n$, a także prędkości $\dot{q}(t) \in \mathbb{R}^n$ [TMD+00]. Uwzględnia się również ograniczenia wynikające z założenia o braku poślizgu wzdłużnego oraz poprzecznego kół, którym poddawany jest robot mobilny. Te ograniczenia uniemożliwiają obranie dowolnej trajektorii oraz zmniejszają liczbę dostępnych sterowań. Zapisuje się je w postaci Pfaffa

$$A(q)\dot{q} = 0,$$

gdzie $A(q) \in \mathbb{R}^{l \times n}$ jest macierzą ograniczeń. Liczba ograniczeń $l = \operatorname{rank} A(q)$, natomiast liczba sterowań $m = n - l$ [TMD+00]. Po zdefiniowaniu ograniczeń można przystąpić do przedstawienia modelu za pomocą bezdryfowego układu sterowania w postaci

$$\dot{q} = G(q)\eta, \tag{2.1}$$

gdzie $G(q) \in \mathbb{R}^{n \times m}$ jest macierzą sterowań, natomiast $\eta \in \mathbb{R}^m$ jest wektorem sterowań.

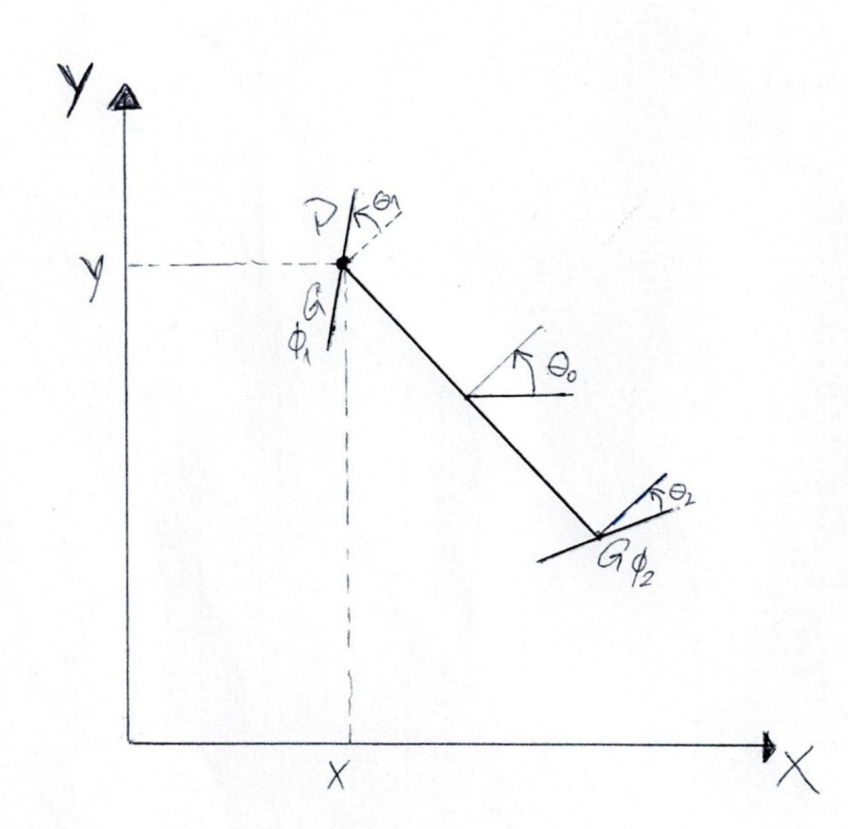
Na rysunku 2.1 przedstawiono schemat modelu robota mobilnego [Jon17]. Wektor współrzędnych uogólnionych badanego robota mobilnego klasy (1,2) jest siedmioelementowy

$$q = (x, y, \theta_0, \theta_1, \varphi_{1,}, \theta_2, \varphi_2)^T,$$

gdzie x, y oznaczają położenie środka wybranego koła, θ_i to kąt skręcenia i-tego koła, a φ_i jest kątem jego obrotu, natomiast θ_0 jest orientacją korpusu robota. Wszystkie elementy wektora zostały zaznaczone na schemacie robota mobilnego. Przy tak wybranym wektorze współrzędnych, bezdryfowy układ sterowania (2.1) przyjmuje postać [Jon14]

$$\begin{cases} \dot{x} = r\cos(\theta_1 + \theta_0)\eta_2 \\ \dot{y} = r\sin(\theta_1 + \theta_0)\eta_2 \\ \dot{\theta}_0 = \frac{r}{2l}\csc\theta_2 s_{1-2}\eta_2 \\ \dot{\theta}_1 = \eta_1 \\ \dot{\varphi}_1 = \eta_2 \\ \dot{\theta}_2 = \eta_3 \\ \dot{\varphi}_2 = \csc\theta_2 \sin\theta_1 \eta_2 \end{cases}, \tag{2.2}$$

gdzie $2l$ jest odległością między kołami robota, a r jest promieniem koła robota mobilnego. Podany model degeneruje się w przypadku kiedy kąt $\theta_2 = 0$.



Rysunek 2.1 Model robota mobilnego klasy (1,2)

Rozdział 3

Algorytmy sterowania robota mobilnego

W badaniu własności sterowania robotem mobilnym klasy (1,2) zastosowano algorytmy pozwalające rozwiązywać zadania sterowania śledzenia zadanej trajektorii na poziomie kinematycznym. Grupa ta charakteryzuje się dobraniem takiego sterowania η, aby robot potrafił podążać po zadanej trajektorii $q_d(t)$, a przy tym błąd $e(t) = q(t) - q_d(t)$ dążył do zera [Jon17]. Celem jest minimalizacja różnicy między rzeczywistą trajektorią robota, a zadaną trajektorią, co prowadzi do poruszania się robota po określonej trasie. W rozdziale zostaną zaprezentowane dwa algorytmy. Pierwszym będzie algorytm linearyzacji statycznej. Drugim będzie algorytm linearyzacji dynamicznej. Dla obydwu algorytmów zostanie przedstawione ich wyprowadzenie, wraz z wytłumaczeniem poszczególnych kroków.

3.1 Linearyzacja statyczna

Pierwszym użytym w badaniach algorytmem pozwalającym na śledzenie zadanej trajektorii jest algorytm linearyzacji statycznej. Polega on na doborze takich wyjść linearyzujących, aby przez statyczne sprzężenie zwrotne uzyskać sterowania umożliwiające ich śledzenie. W tym celu dla układu (2.1) należy wybrać współrzędne linearyzujące w postaci odwzorowania

$$h(q) : \mathbb{R}^n \to \mathbb{R}^m \tag{3.1}$$

nazywanego wyjściem linearyzującym, charakteryzującym się tym, że zawiera ona tyle składowych co sterowanie. Po obliczeniu pochodnej

$$\dot{h}(q) = \frac{\partial h}{\partial q}\dot{q} \tag{3.2}$$

i podstawieniu w niej wartości $\dot{q}$ z bezdryfowego układu sterowania (2.1) otrzymujemy równanie postaci

$$\dot{h}(q) = \frac{\partial h}{\partial q}G\eta. \tag{3.3}$$

Następnie przyjmujemy, że η jest równe

$$\eta = \left(\frac{\partial h}{\partial q}G\right)^{-1} u, \tag{3.4}$$

gdzie $u \in \mathbb{R}^m$ jest nowym wektorem sterowania, a występująca w równaniu macierz $\frac{\partial h}{\partial q}G = R$ jest macierzą odsprzęgania. Podstawienie (3.4) do (3.9) pod warunkiem, że wyznacznik macierzy R jest niezerowy, prowadzi do układu

$$\dot{h} = u \tag{3.5}$$

dla którego sprzężenie zwrotne postaci

$$u = \dot{h}_d - K(h - h_d), \tag{3.6}$$

gdzie h_d jest wartością zadaną wyjść linearyzujących h, a $K \in \mathbb{R}^{n\times m}$ jest dodatnio określoną macierzą wzmocnień, zapewniającą eksponencjalną stabilność układu. Sumarycznie algorytm sterowania opisany jest wzorem

$$\eta = \left(\frac{\partial h}{\partial q}G\right)^{-1} (\dot{h}_d - K(h - h_d)). \tag{3.7}$$

3.2 Linearyzacja dynamiczna

Podobnie jak w przypadku linearyzacji statycznej najpierw należy dokonać wyboru odpowiednich współrzędnych linearyzujących, tak jak zostało to przedstawione w równaniu (3.1). Teraz, podobnie jak w przypadku linearyzacji statycznej liczymy pochodne wyjścia h, przy czym nie poprzestajemy na pierwszej. W najprostszym przypadku, po obliczeniu drugiej pochodnej

$$\ddot{h}(q) = \frac{\partial \dot{h}}{\partial q}\dot{q} \tag{3.8}$$

rozszerzamy przestrzeń stanu układu o nowe współrzędne, które najczęściej przyjmują postać $\dot{\eta}$ [dWSB96]. Kolejno zapiszemy $\ddot{h}$ w postaci

$$\ddot{h} = K_{dd}u + P, \tag{3.9}$$

gdzie u oznacza wektor nowych sterowań (można również używać $\dot{\eta}$), K_{dd} jest macierzą współczynników stojących przy wartościach sterowań u, natomiast P jest wektorem pozostałych elementów powstałych podczas wyliczania drugiej pochodnej $\ddot{h}$. Jeśli wyznacznik macierzy det K_{dd} jest niezerowy, przyjmuje się za u

$$u = K_{dd}^{-1}(-P + v). \tag{3.10}$$

Podstawienie równania (3.9) do (3.10) pozwala na uzyskanie układu w postaci podwójnego integratora

$$\ddot{h} = u. \tag{3.11}$$

Znając własności podwójnego integratora można przyjąć sterowanie u jako

$$u = \ddot{h}_d - K_1(\dot{h} - \dot{h}_d) - K_2(h - h_d), \tag{3.12}$$

gdzie h_d jest zadanym sterowaniem, a $\ddot{h}_d, \dot{h}_d$ jego pochodnymi, natomiast $K_1, K_2 \in \mathbb{R}^{n\times m}$ są dodatnio określonymi macierzami wzmocnień. Ostatecznie zmienne sterujące przyjmują postać

$$\dot{\eta} = u = K_{dd}^{-1}(-P + \ddot{h}_d - K_1(\dot{h} - \dot{h}_d) - K_2(h - h_d)). \tag{3.13}$$

Zestawiając powyższe równanie z pierwotnym równaniem kinematyki robota otrzymujemy model w rozszerzonej przestrzeni stanu o postaci

$$\begin{pmatrix} \dot{q} \\ \dot{\eta} \end{pmatrix} = \begin{pmatrix} G\eta \\ K_{dd}^{-1}(-P + \ddot{h}_d - K_1(\dot{h} - \dot{h}_d) - K_2(h - h_d)) \end{pmatrix}. \tag{3.14}$$

Jeżeli w wyliczonej drugiej pochodnej wyjścia linearyzującego $\ddot{h}$ nie występuje pochodna danej składowej wejść sterujących η, wówczas rozszerzamy wektor stanu o te składowe η w miejsce ich pochodnych $\dot{\eta}$ [dWSB96].

Rozdział 4

Implementacja algorytmów sterowania robota

W tym rozdziale zastosujemy opisane w rozdziale 3 algorytmy sterowania do robota mobilnego klasy (1,2). W implementacji algorytmów linearyzacji statycznej oraz dynamicznej zastosowano następujące skróty: $c_{10} = \cos(\theta_1 + \theta_0)$, $s_{10} = \sin(\theta_1 + \theta_0)$, $s_{1-2} = \sin(\theta_1 - \theta_2)$, $s_2 = \sin\theta_2$.

4.1 Linearyzacja statyczna

Dla rozpatrywanego robota, który został przedstawiony na rysunku 2.1 wektor współrzędnych linearyzujących, należy dobrać w taki sposób, by móc zapewnić niezerową wartość wyznacznika macierzy odsprzęgania R [Jon17]. W tym celu jako współrzędne linearyzujace wybrano

$$h(q) = \begin{pmatrix} x + dc_{10} \\ y + ds_{10} \\ \theta_2 \end{pmatrix}, \tag{4.1}$$

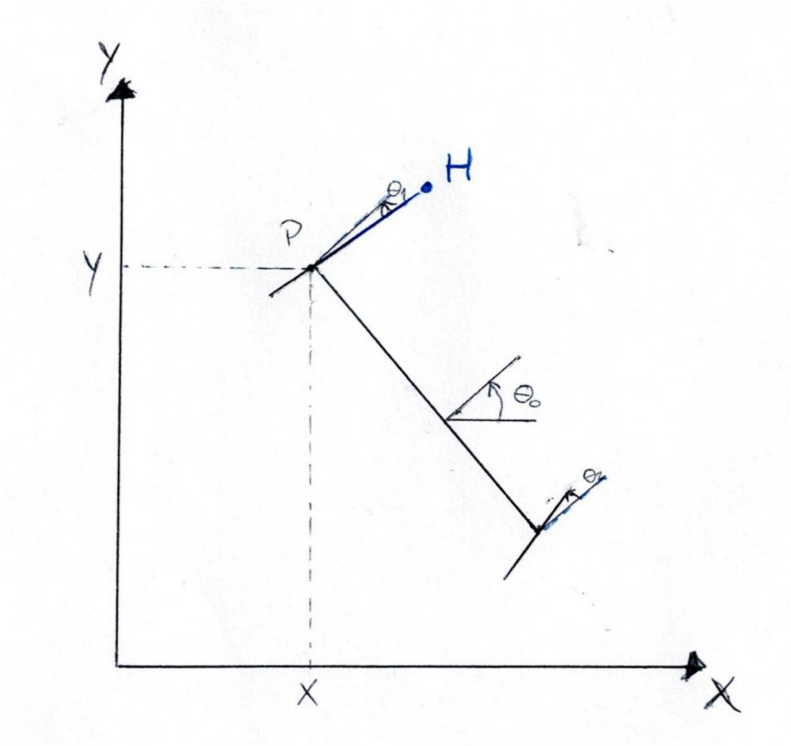
gdzie dwie pierwsze składowe są współrzędnymi punktu H pokazanego na rysunku 4.1. Jest to podyktowane faktem, że wybranie naturalnie nasuwających się współrzędnych punktu P ($d = 0$ w równaniu (4.1)) prowadzi do zerowania się wyznacznika macierzy odsprzęgania R. Co więcej w celu uniknięcia wyzerowania się wyznacznika macierzy odsprzęgania R jako trzecie wyjście linearyzujące należy wybrać θ_2. Wybór θ_1 lub θ_0 jako trzecie wyjście linearyzujące prowadzi do $\det R = 0$.

Kolejnym krokiem jest wyliczenie pochodnej wektora $h(q)$. Przyjmuje ona postać

$$\frac{\partial h(q)}{\partial q} = \begin{bmatrix} 1 & 0 & -ds_{10} & -ds_{10} & 0 & 0 & 0 \\ 0 & 1 & dc_{10} & dc_{10} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \tag{4.2}$$

co pomnożone przez macierz G z równania (2.1) w rezultacie daje

$$R = \frac{\partial h}{\partial q} G = \begin{bmatrix} -ds_{10} & rc_{10} + \frac{drs_{1-2}s_{10}}{2ls_2} & 0 \\ dc_{10} & rs_{10} - \frac{drs_{1-2}c_{10}}{2ls_2} & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.3}$$



Rysunek 4.1 Model robota mobilnego dla wyjść linearyzujących h

której wyznacznik det $R = -dr$. Mając niezerowy wyznacznik macierzy R jest możliwe wyznaczenie macierzy odwrotnej [Mata] o postaci

$$R^{-1} = \begin{bmatrix} \frac{-s_{10}}{d} - \frac{s_{1-2}c_{10}}{2ls_2} & \frac{c_{10}}{d} - \frac{s_{1-2}s_{10}}{2ls_2} & 0 \\ \frac{c_{10}}{r} & \frac{s_{10}}{r} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.4}$$

Podstawiając wyliczoną macierz odwrotną R^{-1} do równania (3.7) otrzymujemy

$$\begin{pmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \end{pmatrix} = \begin{bmatrix} \frac{-s_{10}}{d} - \frac{s_{1-2}c_{10}}{2ls_2} & \frac{c_{10}}{d} - \frac{s_{1-2}s_{10}}{2ls_2} & 0 \\ \frac{c_{10}}{r} & \frac{s_{10}}{r} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} \dot{x}_d - Ke_x \\ \dot{y}_d - Ke_y \\ \dot{\theta}_{2d} - Ke_{\theta_2} \end{pmatrix}, \tag{4.5}$$

gdzie e_x, e_y, e_{θ_2} oznaczają kolejno błędy $(x - x_d), (y - y_d), (\theta_2 - \theta_{2d})$, zaś K jest dodatnim wzmocnieniem. W rezultacie poszczególne sterowania mają wartości

$$\begin{cases} \eta_1 = \left(\frac{-s_{10}}{d} - \frac{s_{1-2}c_{10}}{2ls_2}\right)(\dot{x}_d - Ke_x) + \left(\frac{c_{10}}{d} - \frac{s_{1-2}s_{10}}{2ls_2}\right)(\dot{y}_d - Ke_y) \\ \eta_2 = \left(\frac{c_{10}}{r}\right)(\dot{x}_d - Ke_x) + \left(\frac{s_{10}}{r}\right)(\dot{y}_d - Ke_y) \\ \eta_3 = \dot{\theta}_{2d} - Ke_{\theta_2} \end{cases}. \tag{4.6}$$

4.2 Linearyzacja dynamiczna

Stosując algorytm linearyzacji dynamicznej dla robota klasy (1,2) przy wyborze wyjścia linearyzującego h w prostej postaci nie napotyka się na problem związany z zerującym

się wyznacznikiem macierzy odsprzęgania, stąd wybierzemy

$$h(q) = \begin{pmatrix} x \\ y \\ \theta_2 \end{pmatrix}. \tag{4.7}$$

Wyliczając drugą pochodną $\ddot{h}$ z modelu (2.2) dostajemy

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\theta}_2 \end{pmatrix} = \begin{pmatrix} \dot{\eta}_2 r c_{10} - \dot{\theta}_1 \eta_2 r s_{10} - \dot{\theta}_0 \eta_2 r s_{10} \\ \dot{\eta}_2 r s_{10} - \dot{\theta}_1 \eta_2 r c_{10} - \dot{\theta}_0 \eta_2 r c_{10} \\ \dot{\eta}_3 \end{pmatrix}, \tag{4.8}$$

a podstawienie wartości $\dot{\theta}_1$ oraz $\dot{\theta}_0$ z równania (2.2) oraz wykorzystanie równania (3.9) prowadzi do

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\theta}_2 \end{pmatrix} = \begin{bmatrix} -r\eta_2 s_{10} & r c_{10} & 0 \\ r\eta_2 c_{10} & r s_{10} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} \eta_1 \\ \dot{\eta}_2 \\ \dot{\eta}_3 \end{pmatrix} + \begin{pmatrix} -\frac{r^2\eta_2^2 s_{1-2} s_{10}}{2l s_2} \\ \frac{r^2\eta_2^2 s_{1-2} c_{10}}{2l s_2} \\ 0 \end{pmatrix}. \tag{4.9}$$

Kolejnym krokiem jest wyznaczenie macierzy odwrotnej K_{dd}^{-1}. W tym celu wyliczono wyznacznik macierzy $\det K_{dd} = -r^2\eta_2$, który nie zeruje się pod warunkiem, że $\eta_2 \neq 0$. Macierz odwrotna K_{dd}^{-1} przyjmuje postać

$$K_{dd}^{-1} = \begin{bmatrix} \frac{-s_{10}}{r\eta_2} & \frac{c_{10}}{r\eta_2} & 0 \\ \frac{c_{10}}{r} & \frac{s_{10}}{r} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.10}$$

Podstawiając wyliczoną [Mata] macierz odwrotną K_{dd}^{-1} do równania (3.13) otrzymano

$$\begin{pmatrix} \eta_1 \\ \dot{\eta}_2 \\ \dot{\eta}_3 \end{pmatrix} = \begin{bmatrix} \frac{-s_{10}}{r\eta_2} & \frac{c_{10}}{r\eta_2} & 0 \\ \frac{c_{10}}{r} & \frac{s_{10}}{r} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} \frac{r^2\eta_2^2 s_{1-2} s_{10}}{2l s_2} + \ddot{x}_d - K_1 e_{\dot{x}} - K_2 e_x \\ -\frac{r^2\eta_2^2 s_{1-2} c_{10}}{2l s_2} + \ddot{y}_d - K_1 e_{\dot{y}} - K_2 e_y \\ \ddot{\theta}_{2d} - K_1 e_{\dot{\theta}_2} - K_2 e_{\theta_2} \end{pmatrix}, \tag{4.11}$$

gdzie e_x, e_y itd. oznaczają kolejno $e_x = (x - x_d)$ itd, natomiast K_1 oraz K_2 są dodatnimi wzmocnieniami. W efekcie poszczególne sterowania mają wartość

$$\begin{cases} \eta_1 = -\frac{s_{10}}{r\eta_2}\left(\frac{r^2\eta_2^2 s_{1-2} s_{10}}{2l s_2} + \ddot{x}_d - K_1 e_{\dot{x}} - K_2 e_x\right) - \frac{c_{10}}{r\eta_2}\left(\frac{r^2\eta_2^2 s_{1-2} c_{10}}{2l s_2} - \ddot{y}_d + K_1 e_{\dot{y}} + K_2 e_y\right) \\ \dot{\eta}_2 = \frac{c_{10}}{r}\left(\frac{r^2\eta_2^2 s_{1-2} s_{10}}{2l s_2} + \ddot{x}_d - K_1 e_{\dot{x}} - K_2 e_x\right) - \frac{s_{10}}{r}\left(\frac{r^2\eta_2^2 s_{1-2} c_{10}}{2l s_2} - \ddot{y}_d + K_1 e_{\dot{y}} + K_2 e_y\right) \\ \dot{\eta}_3 = \ddot{\theta}_{2d} - K_1 e_{\dot{\theta}_2} - K_2 e_{\theta_2} \end{cases}. \tag{4.12}$$

Rozdział 5

Badania symulacyjne

W rozdziale zaprezentowano wyniki badań, jakie zostały przeprowadzone dla modelu robota mobilnego klasy (1,2), opisanego w rozdziale 2. Do badań użyto dwa algorytmy pozwalające na rozwiązanie zadania śledzenia zadanej trajektorii. Są to: algorytm linearyzacji statycznej oraz algorytm linearyzacji dynamicznej.

Badania zostały przeprowadzone w środowisku programistycznym *MATLAB* [Matb], wraz z zastosowaniem dodatkowego pakietu numerycznego *SIMULINK* [Matc]. Pakiet ten zapewnia użytkownikowi tworzenie modeli symulacyjnych w postaci schematu blokowego przy pomocy interfejsu graficznego.

We wszystkich symulacjach przyjęto takie same wartości parametrów robota: promień koła $r = 0.03$ oraz połowa odległości miedzy kołami robota $l = 0.1$. Przyjęto dla wszystkich symulacji również te same warunki początkowe

$$q(0) = \left(0, 0.1, 0, \frac{\pi}{4}, 0, \frac{\pi}{4}, 0\right)^T.$$

Wyjście linearyzujące (3.1), $h(q) \in \mathbb{R}^3$ zawierają dwie składowe opisujące położenie oraz jedną składową opisującą orientację. W badaniach jako trajektorię zadaną dla składowych położenia wyjścia linearyzującego przyjęto kolejno:

- linię prostą przebiegającą pod kątem $\frac{\pi}{4}$ do osi globalnego układu współrzędnych

$$\begin{cases} x_d = t \\ y_d = t \end{cases}, \tag{5.1}$$

- okrąg

$$\begin{cases} x_d = \cos t \\ y_d = \sin t \end{cases}, \tag{5.2}$$

- krzywą przypominającą swoim kształtem ósemkę

$$\begin{cases} x_d = \sin \frac{t}{2} \\ y_d = \cos\left(t + \frac{\pi}{4}\right) \end{cases}. \tag{5.3}$$

Jako trajektorię zadaną dla składowych orientacji przyjęto:

- dwie trajektorie punktowe

$$\theta_{2d} = \frac{\pi}{4}, \tag{5.4}$$

$$\theta_{2d} = \frac{\pi}{2}, \tag{5.5}$$

- jedną w postaci prostej

$$\theta_{2d} = 0.1t, \tag{5.6}$$

- jedną okresową

$$\theta_{2d} = \sin 0.2t. \tag{5.7}$$

Każdorazowo wyniki symulacji zilustrowano zestawem 6 wykresów, na które składają się :

- ścieżka zadana i rzeczywista dla składowych położenia wyjścia linearyzującego h,
- trajektorie rzeczywiste dla składowych wektora współrzędnych uogólnionych q: θ_1, θ_2,
- trajektorie rzeczywiste θ_0, $\theta_0 + \theta_1$, $\theta_0 + \theta_2$*,
- błędu położenia $\sqrt{e_x^2 + e_y^2}$ i orientacji $|e_{\theta_2}|$,
- trajektorie rzeczywiste φ_1, φ_2,
- prędkości rzeczywiste $\dot{\varphi}_1$, $\dot{\varphi}_2$.

Warto zauważyć, że θ_1 i θ_2 opisują skręcenia kół względem korpusu robota, zaś θ_0, $\theta_0+\theta_1$, $\theta_0+\theta_2$ są bezwzględną orientacją korpusu robota i jego kół wyrażoną w globalnym układzie współrzędnych.

5.1 Linearyzacja statyczna

Wykorzystując algorytm linearyzacji statycznej jako wyjście linearyzujące wybrano współrzędne x oraz y przesunięte o odległość d i współrzędną θ_2. Na potrzeby symulacji wartość przesunięcia ustawiono na $d = 0.15$ zaś wzmocnienie $K = I_3$. W celu zbadania działania algorytmu linearyzacji statycznej przeanalizowano możliwie jak najwięcej przykładów. Pozwoliło to na lepsze zobrazowanie zachowań algorytmu dla prostych przypadków oraz tych trudniejszych.

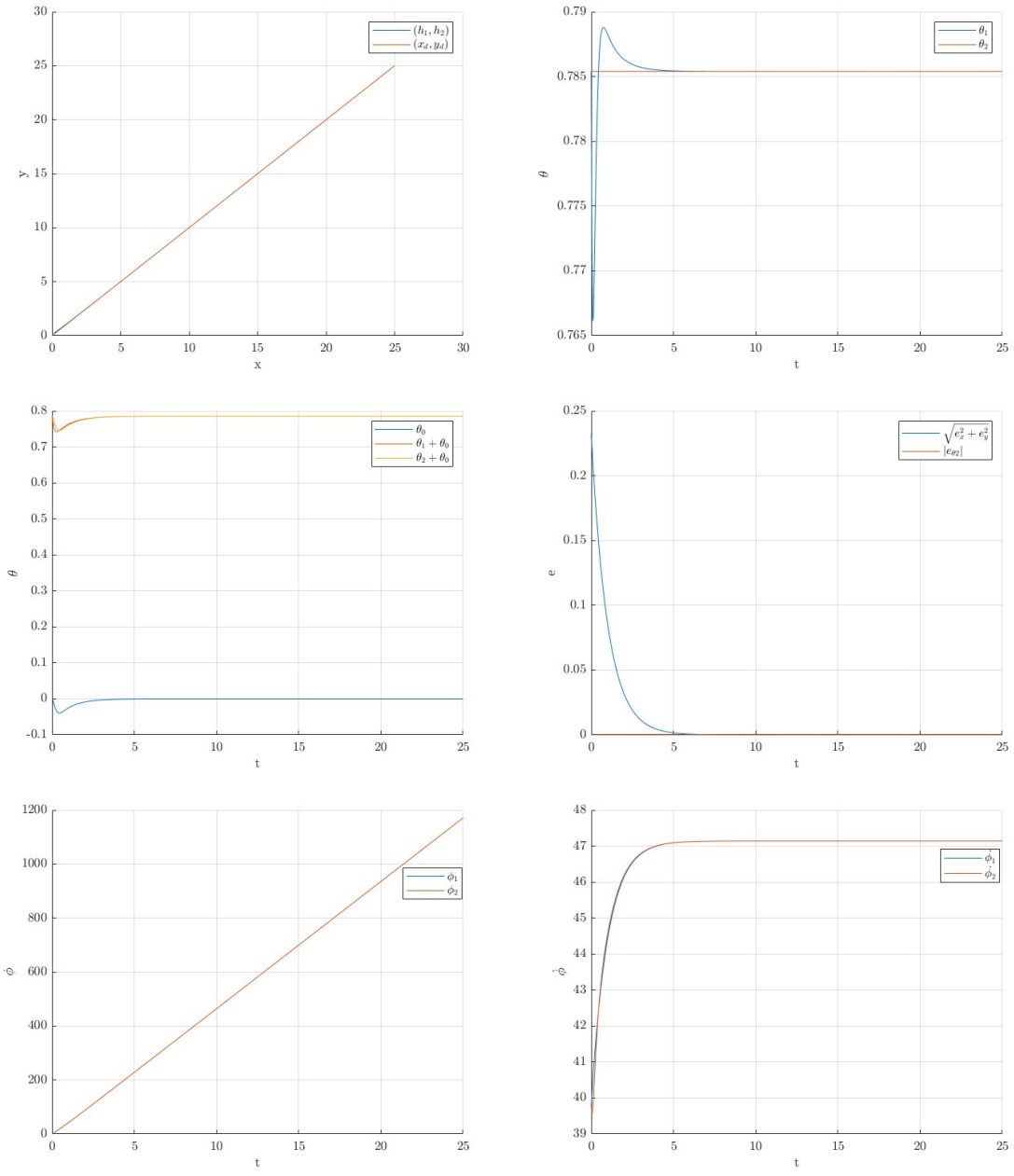
5.1.1 Trajektoria zadana w postaci linii prostej

Badania rozpoczęto od trywialnego przypadku zadania śledzenia trajektorii w postaci przebiegającej pod kątem $\frac{\pi}{4}$ linii prostej (5.1) dla składowych pozycyjnych wyjścia linearyzującego oraz trajektorii (5.4)-(5.7) dla składowej orientacji.

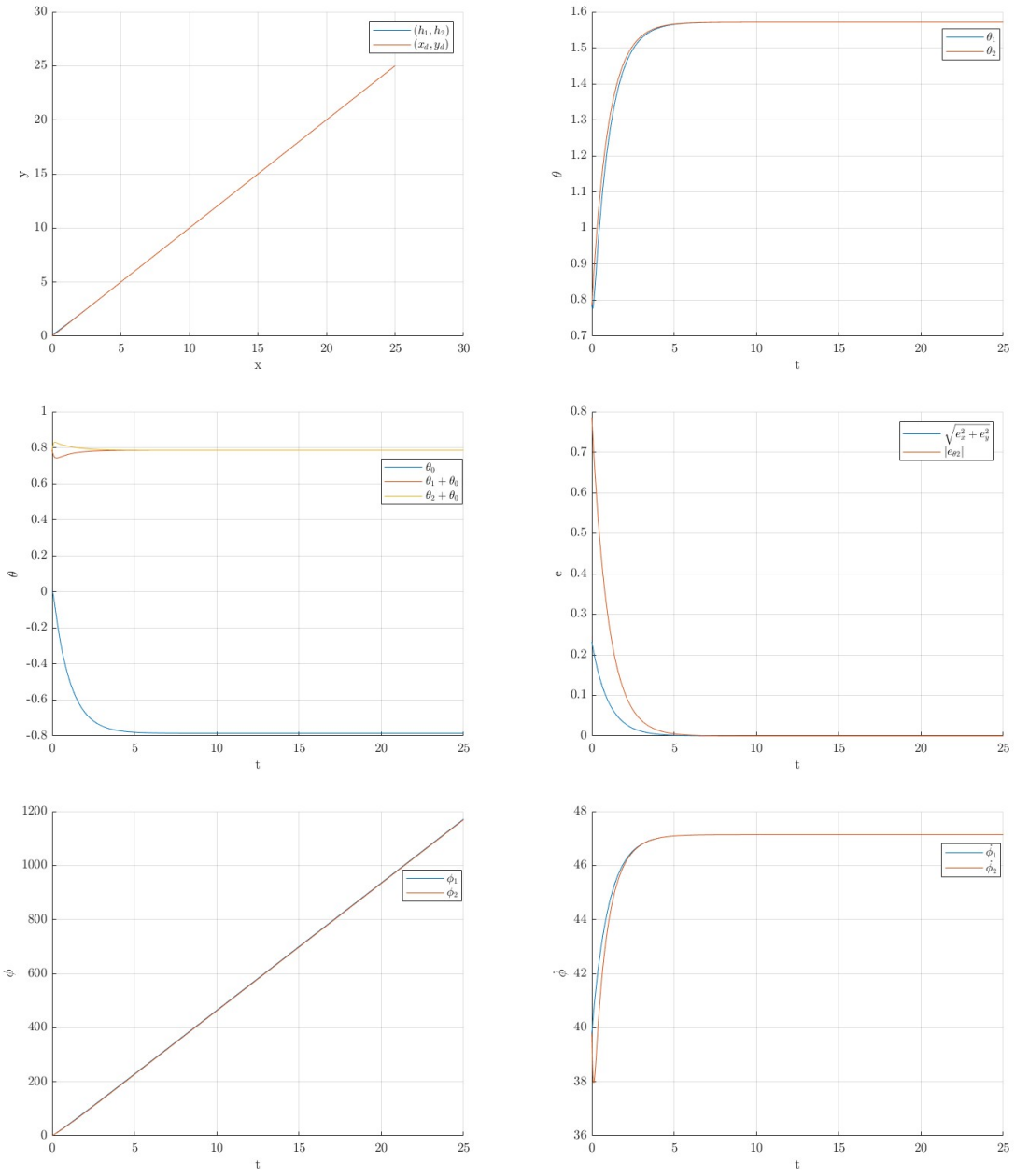
W przypadku stałej trajektorii zadanej dla orientacji równej $\frac{\pi}{4}$†, błędy, zobacz rysunek 5.1, zbiegają eksponencjalnie do zera, a rzeczywista ścieżka ruchu praktycznie pokrywa się ze ścieżką zadaną. Z wykresu prędkości i położenia kół widać, że w stanie ustalonym

*θ_0 opisuje orientację względem korpusu, natomiast $\theta_0 + \theta_1$, $\theta_0 + \theta_2$ względem poziomu

†przy warunkach początkowych dla orientacji również równej $\frac{\pi}{4}$ występuje początkowa wartość błędu orientacji



Rysunek 5.1 Linearyzacja statyczna dla prostej i $\theta_{2d} = \frac{\pi}{4}$

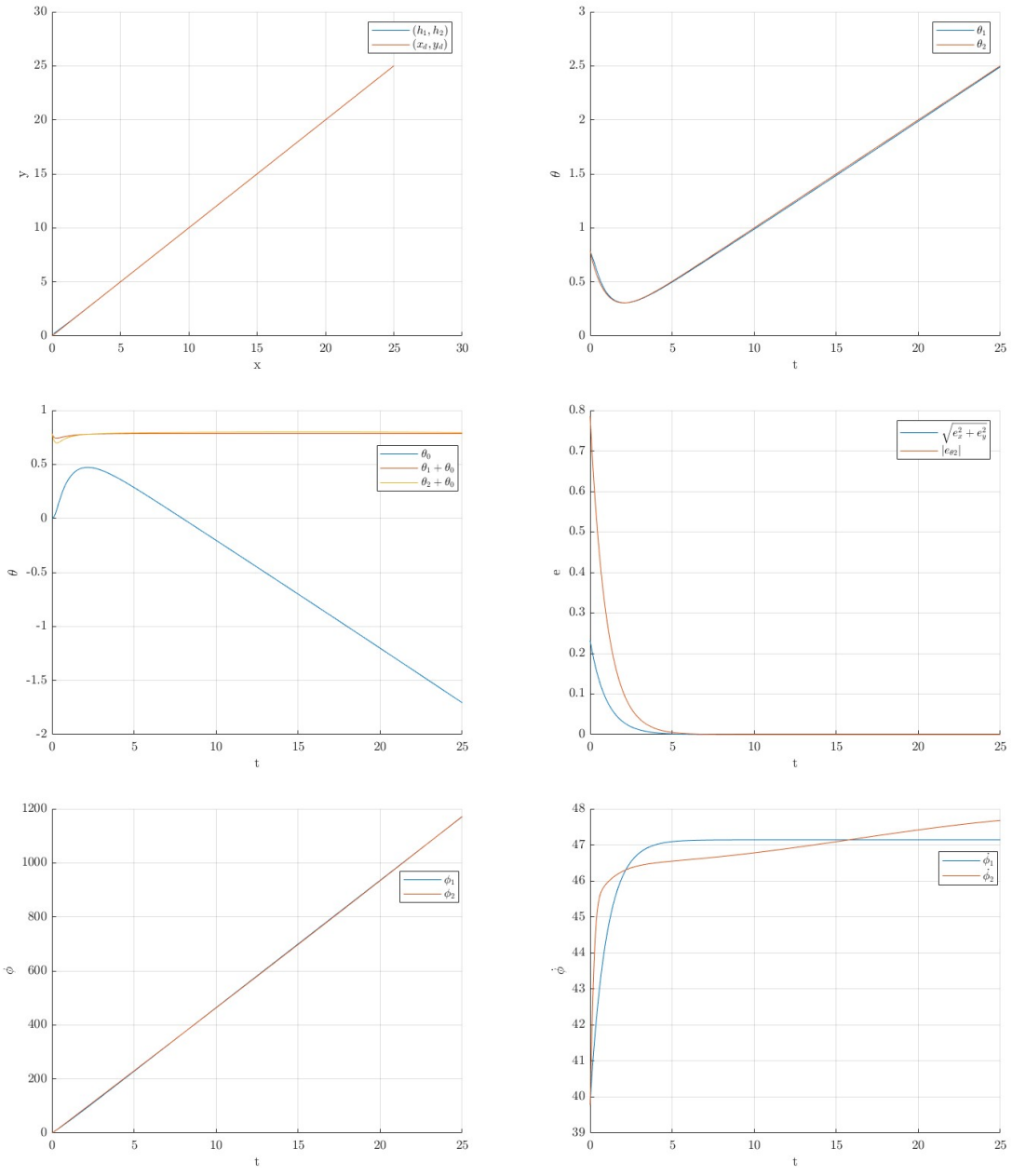


Rysunek 5.2 Linearyzacja statyczna dla prostej i $\theta_{2d} = \frac{\pi}{2}$

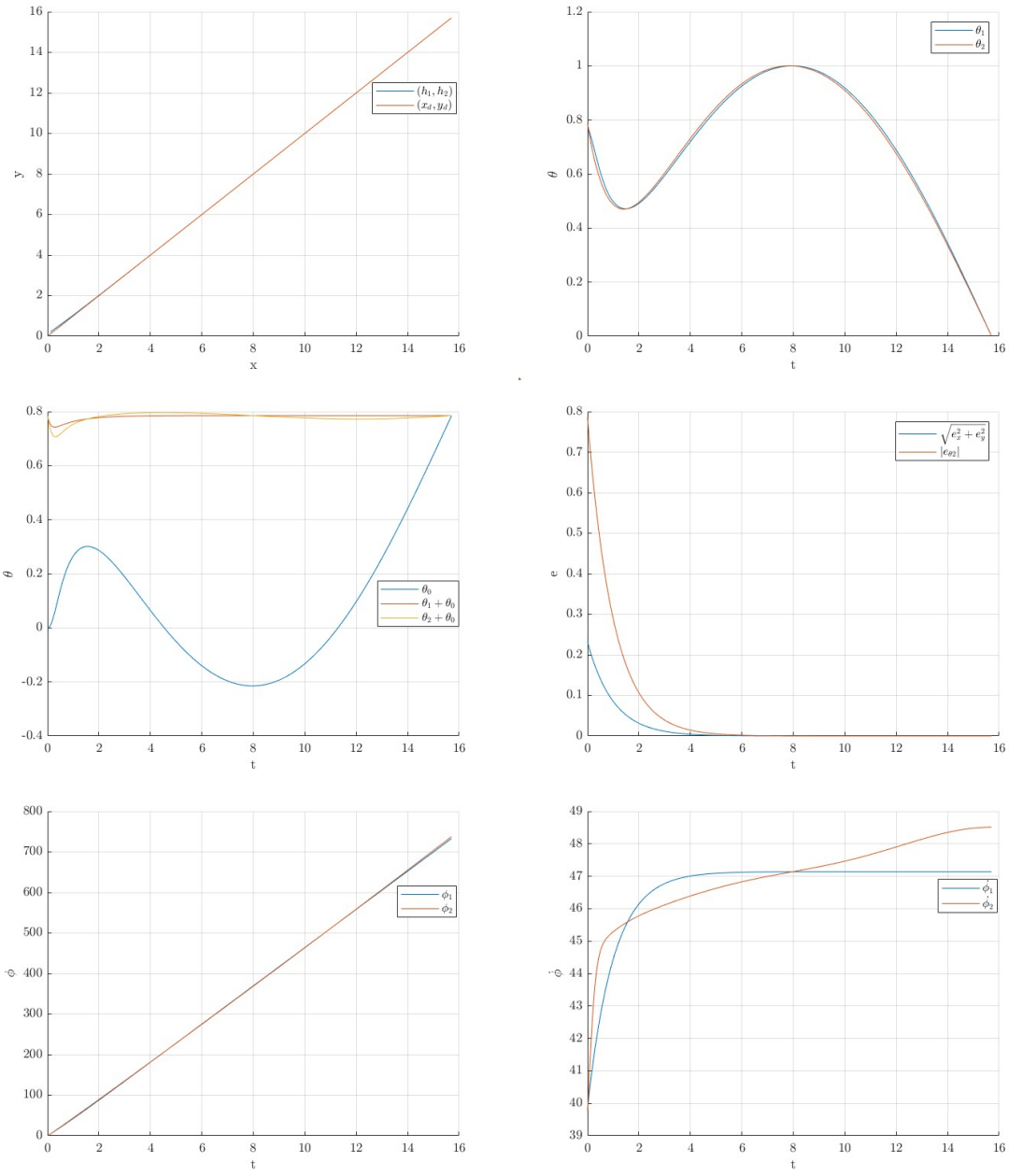
koła robota obracają się ze stałymi, jednakowymi prędkościami. Układ zachowuje się bardzo podobnie również dla przypadku z niezerowym początkowym błędem orientacji dla trajektorii zadanej składowej orientacji (5.5) równej $\frac{\pi}{2}$, co zilustrowano na rysunku 5.2.

Podobne zachowanie obserwujemy przy zmieniającej się liniowo trajektorii zadanej dla orientacji (5.6). I teraz błędy eksponencjalnie zbiegają do zera, rzeczywista i zadana ścieżka praktycznie pokrywają się, jednakże teraz prędkość koła, którego orientację zmieniamy rośnie. Zachowanie to widoczne jest na rysunku 5.3.

W przypadku zmieniającej się okresowo trajektorii zadanej dla orientacji (5.7) rzeczywista ścieżka również praktycznie pokrywa się z zadaną, błędy eksponencjalnie zbiegają



Rysunek 5.3 Linearyzacja statyczna dla prostej i $\theta_{2d} = 0.1t$



Rysunek 5.4 Linearyzacja statyczna dla prostej i $\theta_{2d} = \sin 0.2t$

do zera, a prędkość koła, którego orientacja jest zmieniana rośnie podobnie jak to miało miejsce w poprzednim przypadku. Jednakże teraz przeprowadzenie symulacji było możliwe jedynie dla pierwszych 16 sekund gdyż, dokładnie dla $t = 15.7s$ trajektoria zadana $\theta_{2d} = \sin 0.2t$ osiąga wartość π. Biorąc pod uwagę, że w modelu (2.2) występuje dzielenie przez $\sin\theta_2$, osiągnięcie wartości przez $\theta_{2d} = \sin\pi$ spowoduje dzielenie przez zero. Jest to sytuacja, która prowadzi do przerwania działania algorytmu i pojawia się ona we wszystkich przypadkach trajektorii zadanej dla obu algorytmów, gdy $\theta_{2d} = \sin 0.2t$. Zapewnienie niezerowania się wartości kąta θ_2 (na przykład poprzez dodanie do zadawanego kąta θ_{2d} stałej wartości większej od 1) zapobiega przerwaniu działania algorytmu.

5.1.2 Trajektoria zadana w postaci okręgu

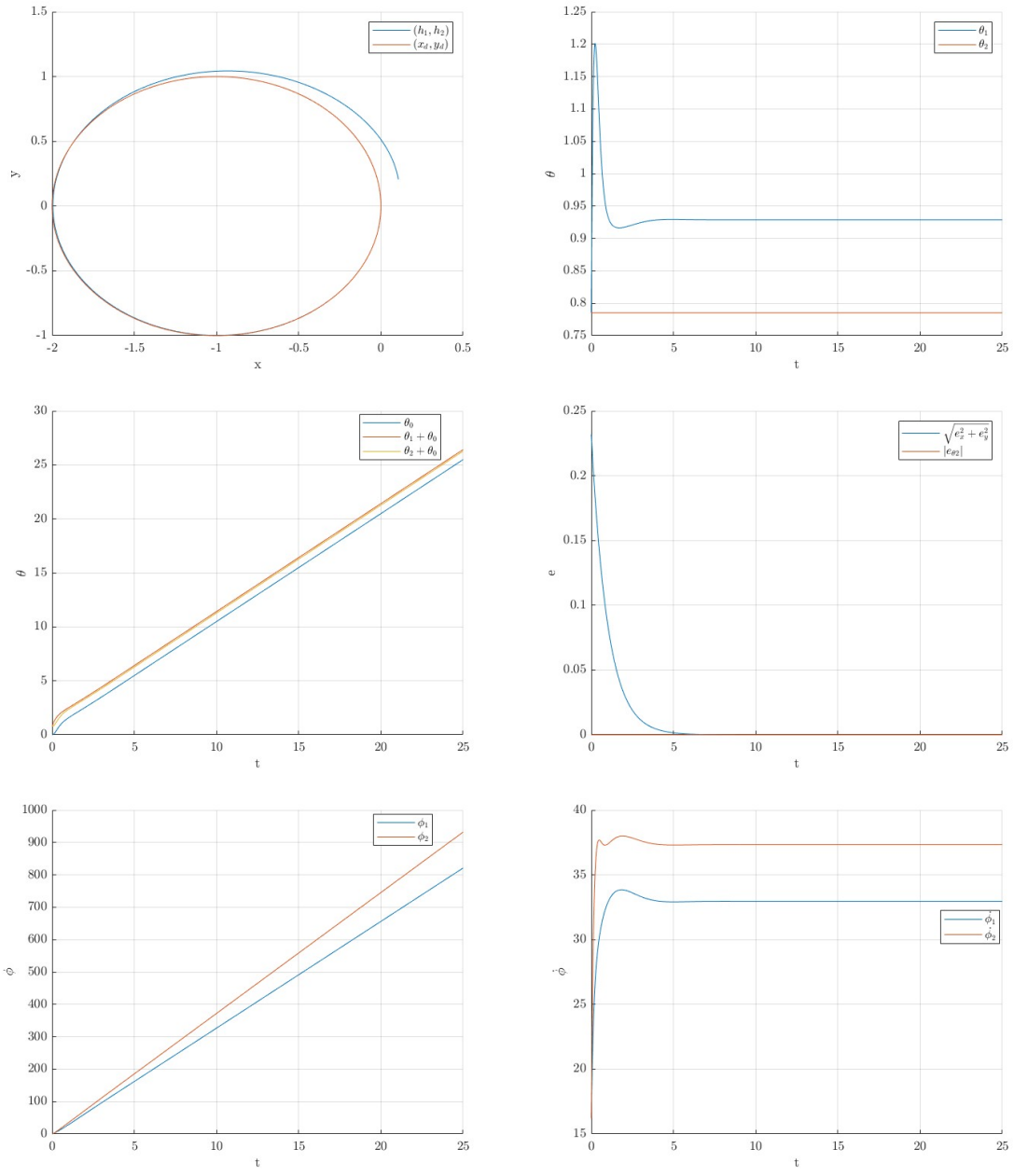
W poprzednim podrozdziale przeprowadzono badania na trywialnym przykładzie prostej jako trajektorii zadanej, która pozwalała na pokazanie jedynie podstawowych cech działania algorytmu linearyzacji statycznej. W tym podrozdziale jako trajektorię zadaną przyjęto dla składowej orientacji wyjścia linearyzującego okrąg opisany równaniem (5.2).

W przypadku trajektorii zadanej dla orientacji równej $\frac{\pi}{4}$, zobacz rysunek 5.5, rzeczywista ścieżka ruchu robota nie pokrywa się aż tak dokładnie z zadaną ścieżką jak to miało miejsce w przypadku trajektorii zadanej (5.1). W tym przypadku efekt przesunięcia współrzędnych linearyzujących o odległość d jest zdecydowanie lepiej widoczny, aczkolwiek błędy zbiegają eksponencjalnie do zera w praktycznie tym samym czasie co na rysunku 5.1. Analizując ten przypadek można dostrzec, że bezwzględna orientacja kół zmienia się w taki sposób, by koła były cały czas zorientowane w kierunku stycznej do okręgu. Można to stwierdzić na podstawie wykresu trajektorii rzeczywistych θ_0, $\theta_0 + \theta_1$, $\theta_0 + \theta_2$, gdzie jak widać trajektorie te są do siebie równoległe. Z wykresu prędkości kół widać, że teraz w stanie ustalonym koła robota obracają się ze stałymi ale różnymi prędkościami. Jednakże prędkości i położenie kół nie są jednakowe, gdyż jedno koło jedzie po wewnętrznej stronie okręgu, a drugie po zewnętrznej.

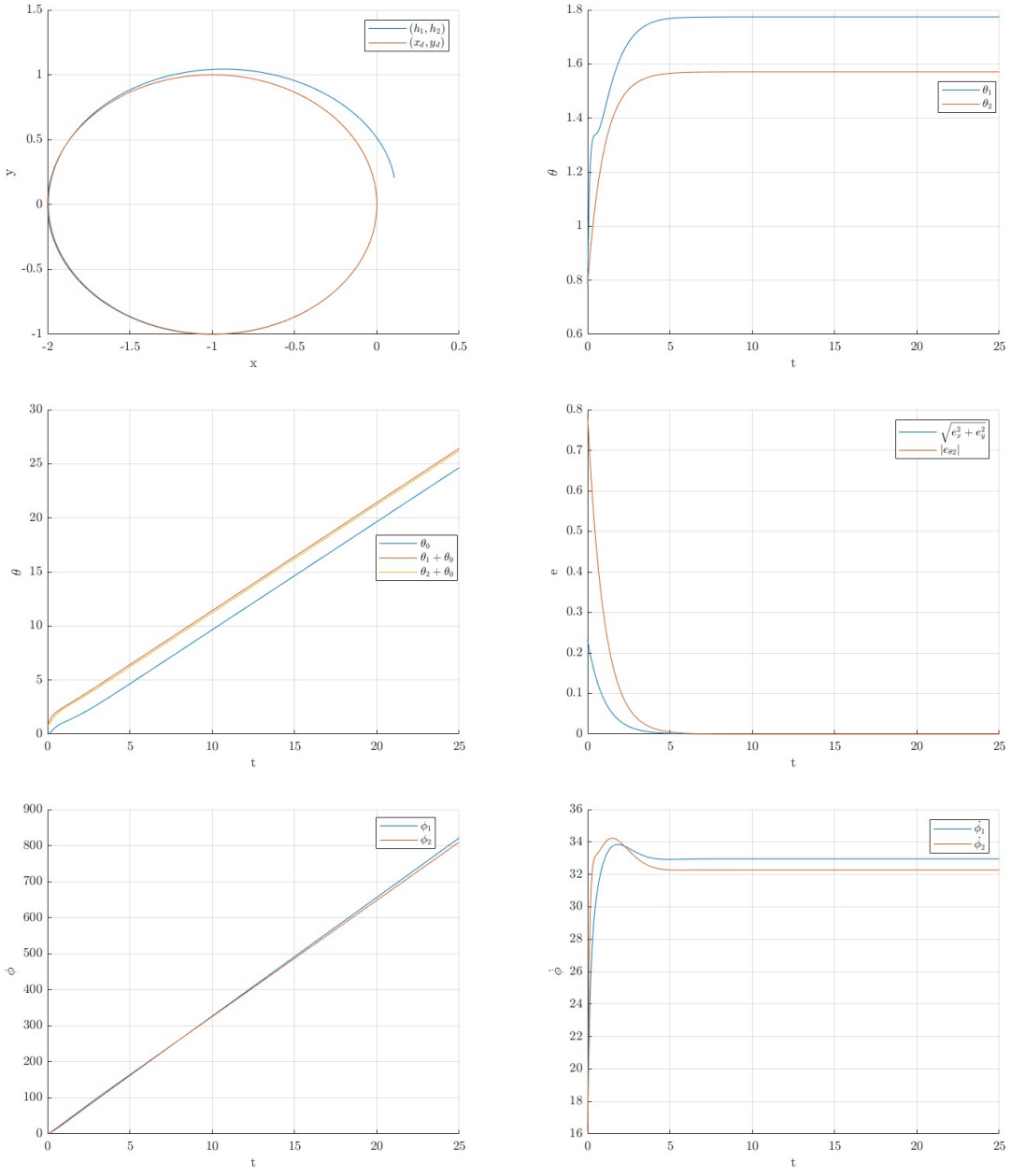
W przypadku stałej trajektorii zadanej składowej orientacji równej $\frac{\pi}{2}$, zobacz rysunek 5.6, na wykresie błędów zbiegających eksponencjalnie do zera uwidacznia się przebieg składowej orientacji co jest naturalnie związane z warunkami początkowymi. Rzeczywista ścieżka również pokrywa się po pewnym czasie z zadaną ścieżką, a bezwzględna orientacja kół zmienia się w taki sposób, by koła były zorientowane w kierunku stycznej do okręgu. Różnica wartości prędkości i położenia jest mniejsza pomiędzy dwoma kołami, bardziej zbliżona do tej na rysunku 5.2 niż tej na rysunku 5.5.

W przypadku trajektorii zadanej dla składowej orientacji w postaci prostej (5.6), zobacz rysunek 5.7, obserwujemy takie same zachowanie przebiegu ścieżki rzeczywistej i zadanej jak w poprzednich dwóch przypadkach. Wykresy błędów dla składowych położenia i orientacji również zbiegają eksponencjalnie do zera. Na wykresach przedstawiających wartości kątów obrotu i skręcenia kół można dostrzec pewną zależność. Widać, że kąt skręcenia koła pierwszego jest większy niż koła drugiego, natomiast kąt obrotu koła pierwszego jest mniejszy niż drugiego. Jednakże iloczyn tych dwóch wartości dla obu kół jest taki sam. Wykresy prędkości obrazują, że koło pierwsze bardzo szybko się stabilizuje na stałej wartości, natomiast koło drugie bardzo szybko osiąga lokalne maksimum po czym maleje.

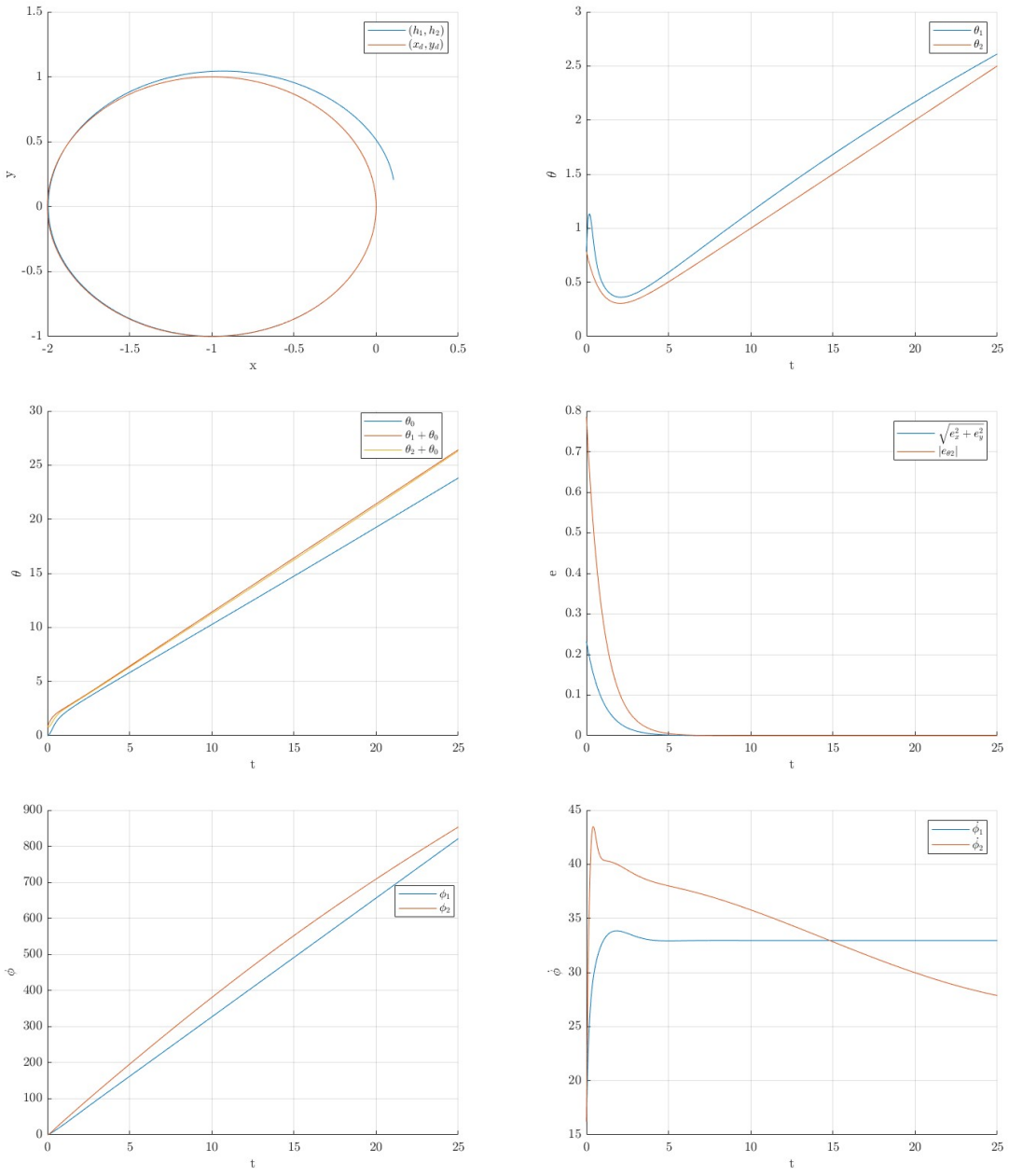
Dla okresowo zmieniającej się trajektorii zadanej składowej orientacji (5.7), patrz rysunek 5.8, rzeczywista ścieżka pokrywa się z zadaną w taki sam sposób jak miało to miejsce na rysunkach 5.6 i 5.7. To samo dotyczy przebiegu błędów. Wykresy położenia i prędkości kół pokazują, że koło drugie osiąga większe wartości niż koło pierwsze, podobnie jak to zostało przedstawione na rysunku 5.5. Jednakże prędkość koła drugiego w stanie ustalonym nie jest stała. Wartości θ_1 i θ_2 mają przebieg sinusoidalny. Podobnie jak dla przypadku trajektorii zadanej w postaci linii prostej algorytm przerywa działanie w okolicach 16 sekundy.



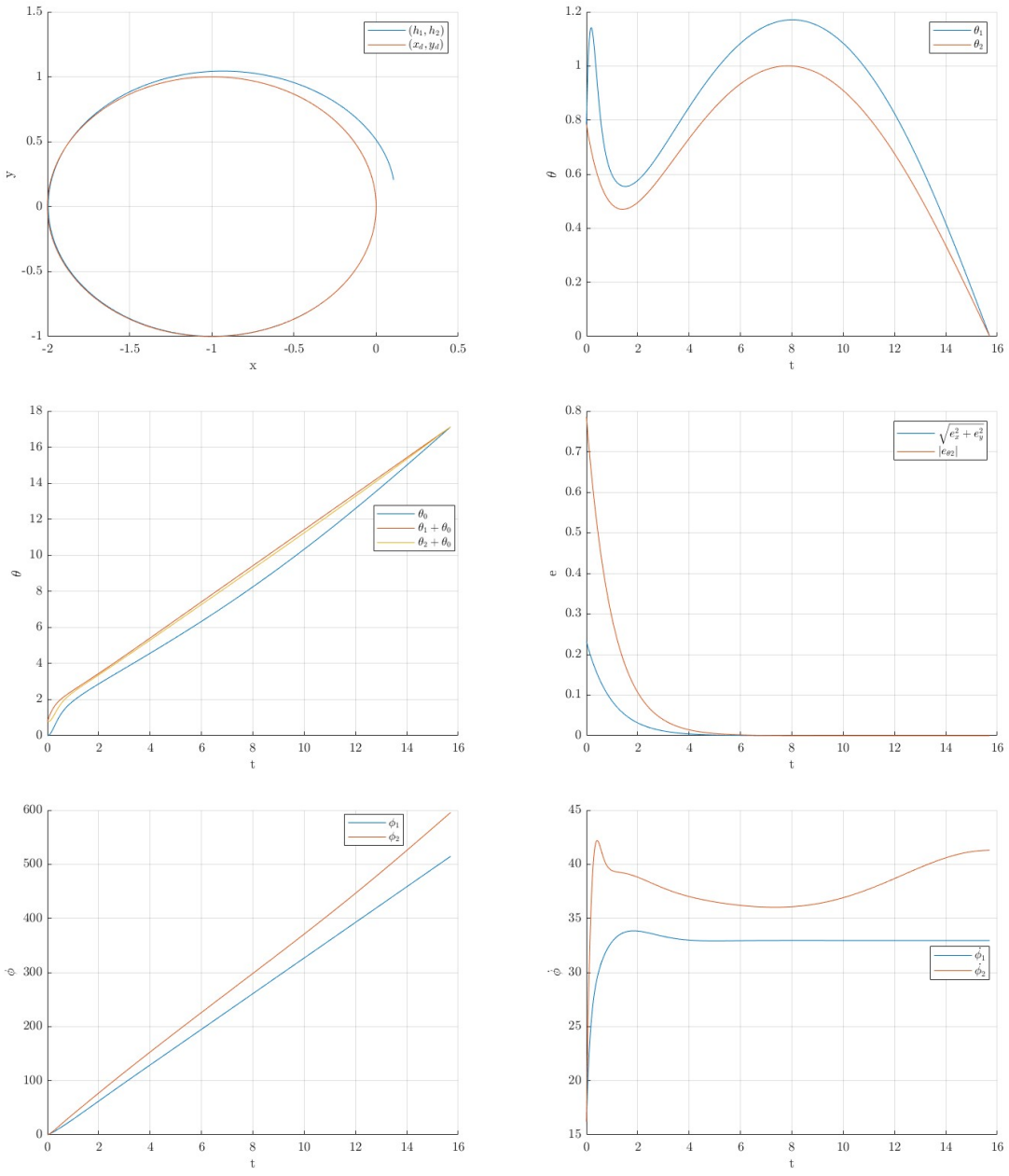
Rysunek 5.5 Linearyzacja statyczna dla okręgu i $\theta_{2d} = \frac{\pi}{4}$



Rysunek 5.6 Linearyzacja statyczna dla okręgu i $\theta_{2d} = \frac{\pi}{2}$



Rysunek 5.7 Linearyzacja statyczna dla okręgu i $\theta_{2d} = 0.1t$



Rysunek 5.8 Linearyzacja statyczna dla okręgu i $\theta_{2d} = \sin 0.2t$

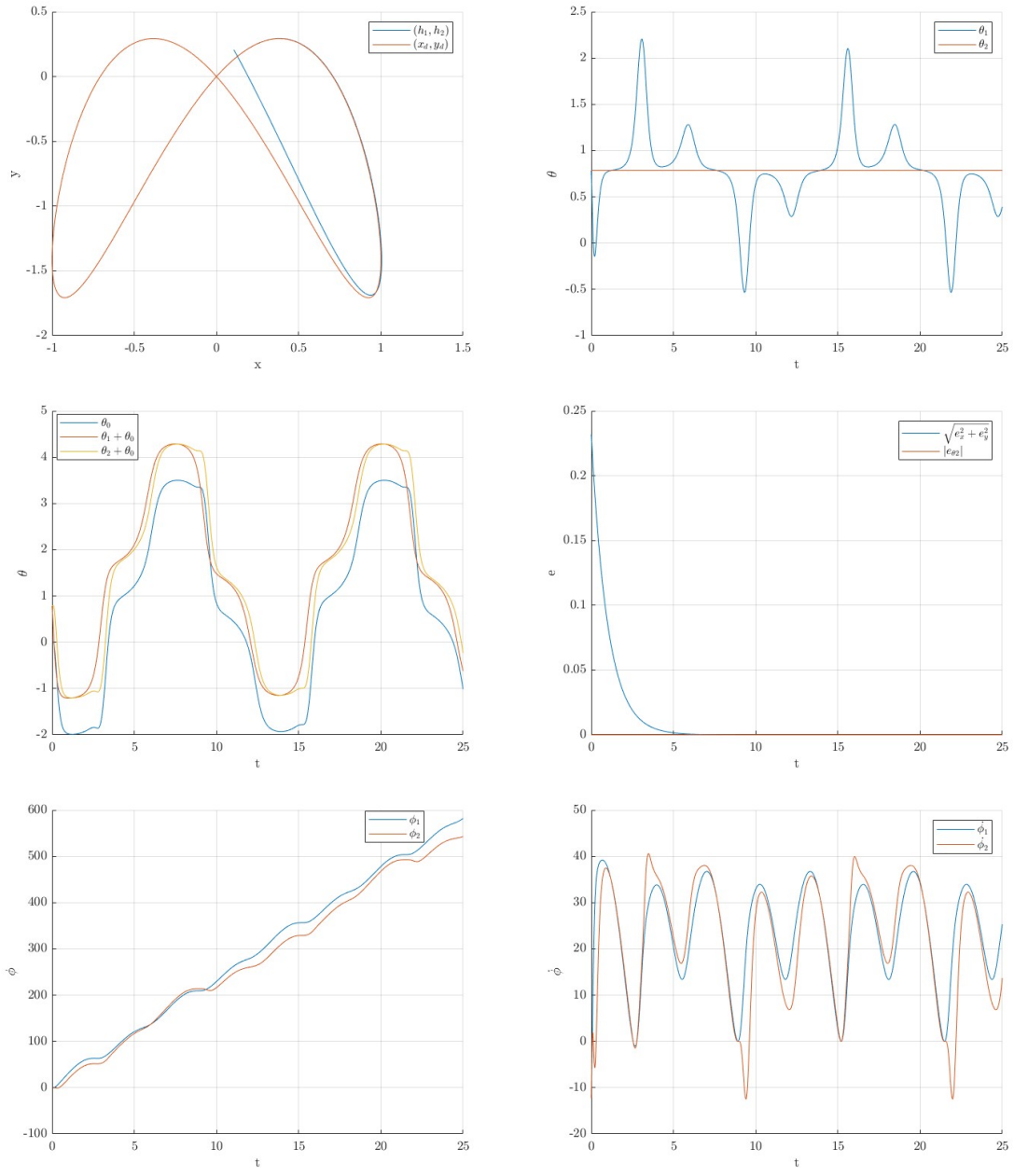
5.1.3 Trajektoria zadana w kształcie ósemki

Ostatnim analizowanym przykładem dla algorytmu linearyzacji statycznej jest trajektoria zadana opisana równaniem (5.3), która swoim kształtem przypomina ósemkę. Ten przypadek jest zarazem najmniej trywialny.

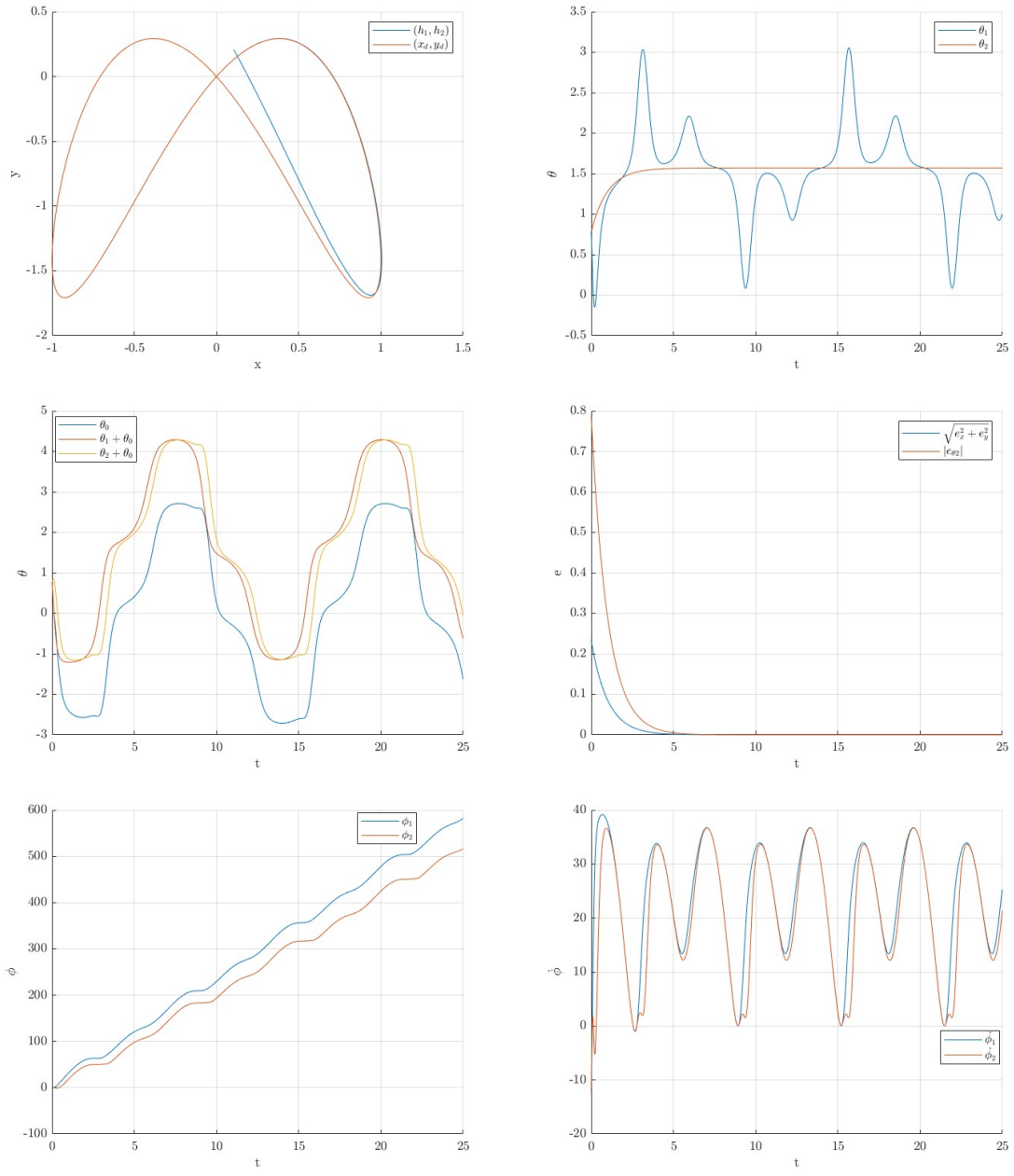
Dla trajektorii zadanej składowej orientacji równej $\frac{\pi}{4}$, zobacz rysunek 5.9, rzeczywista ścieżka ruchu robota nie pokrywa się aż tak dokładnie z zadaną ścieżką podobnie jak to miało miejsce na rysunku 5.5, pomimo, że wykresy błędów zbiegają eksponencjalnie do zera w tym samym czasie co na rysunku 5.1. Widać, że przebieg θ_1 zdecydowanie odbiega swoim wyglądem od wszystkich dotychczas prezentowanych przypadków. Do tego na wykresach prędkości obrotowej kół są widoczne znaczące oscylacje, a na wykresach położenia kół można zaobserwować wcześniej niespotykane zmiany wartości. Na początku koło pierwsze osiąga wyższe wartości niż koło drugie, by po około 7 sekundach nastąpiła zmiana i w okolicach dziesiątej sekundy koło pierwsze znów miało większe wartości. Podobnie jak na rysunku 5.5 bezwzględna orientacja kół jest zmienna w taki sposób, by koła były zorientowane w kierunku stycznej do ósemki. Widoczne jest to na wykresie trajektorii rzeczywistych θ_0, $\theta_0 + \theta_1$, $\theta_0 + \theta_2$, gdzie te trajektorie są do siebie równoległe.

Porównując przypadek trajektorii zadanej dla orientacji równej $\frac{\pi}{2}$, zobacz rysunek 5.10, z rysunkiem 5.9 można zauważyć różnice na wykresach położenia kół. W tym przypadku koło pierwsze osiąga większe wartości niż koło drugie na całym przedziale czasowym. Dodatkowo na wykresie błędów zbiegających eksponencjalnie do zera uwidacznia się przebieg składowej orientacji co jest naturalnie związane z warunkami początkowymi. Z reszty wykresów dla analizowanego przypadku nie wyciągnięto żadnych nowych wniosków, gdyż praktycznie są takie same.

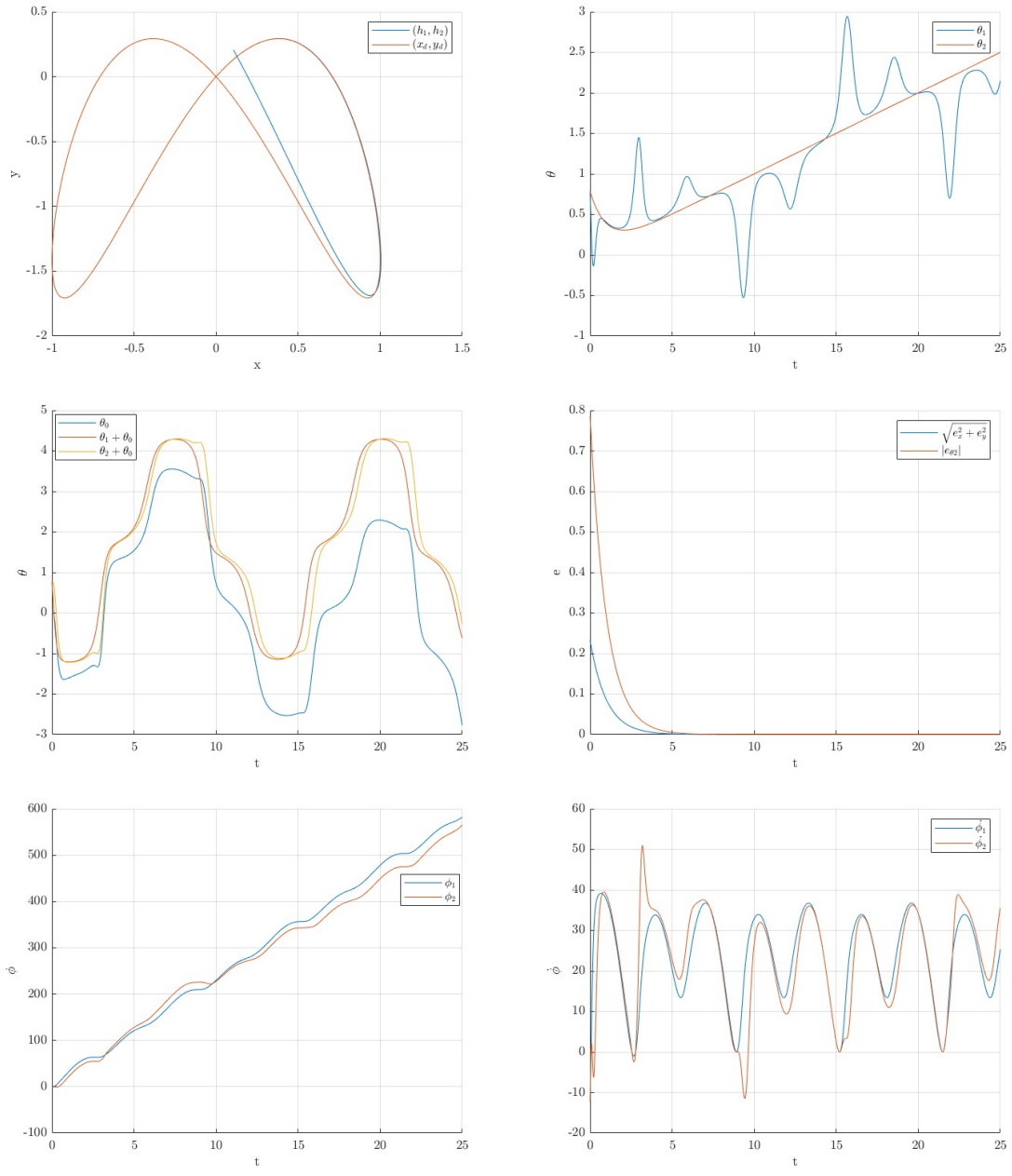
Z analizy wykresów przedstawionych na rysunku 5.11 oraz 5.12 można wyciągnąć niemalże identyczne wnioski do tych opisanych dla rysunku 5.9. Jedynymi zauważalnymi różnicami są: niezerowy przebieg składowej orientacji, który eksponencjalnie zbiega do zera oraz różnice wykresów θ_1 i θ_2, które wynikają z różnego rodzaju zadawanych trajektorii dla składowej orientacji. Ponownie dla sinusoidalnej składowej orientacji algorytm przerywa działanie w okolicach 16 sekundy. Przykłady te miały na celu pokazanie, że algorytm radzi sobie ze skomplikowanymi zadawanymi trajektoriami i daje poprawne wyniki.



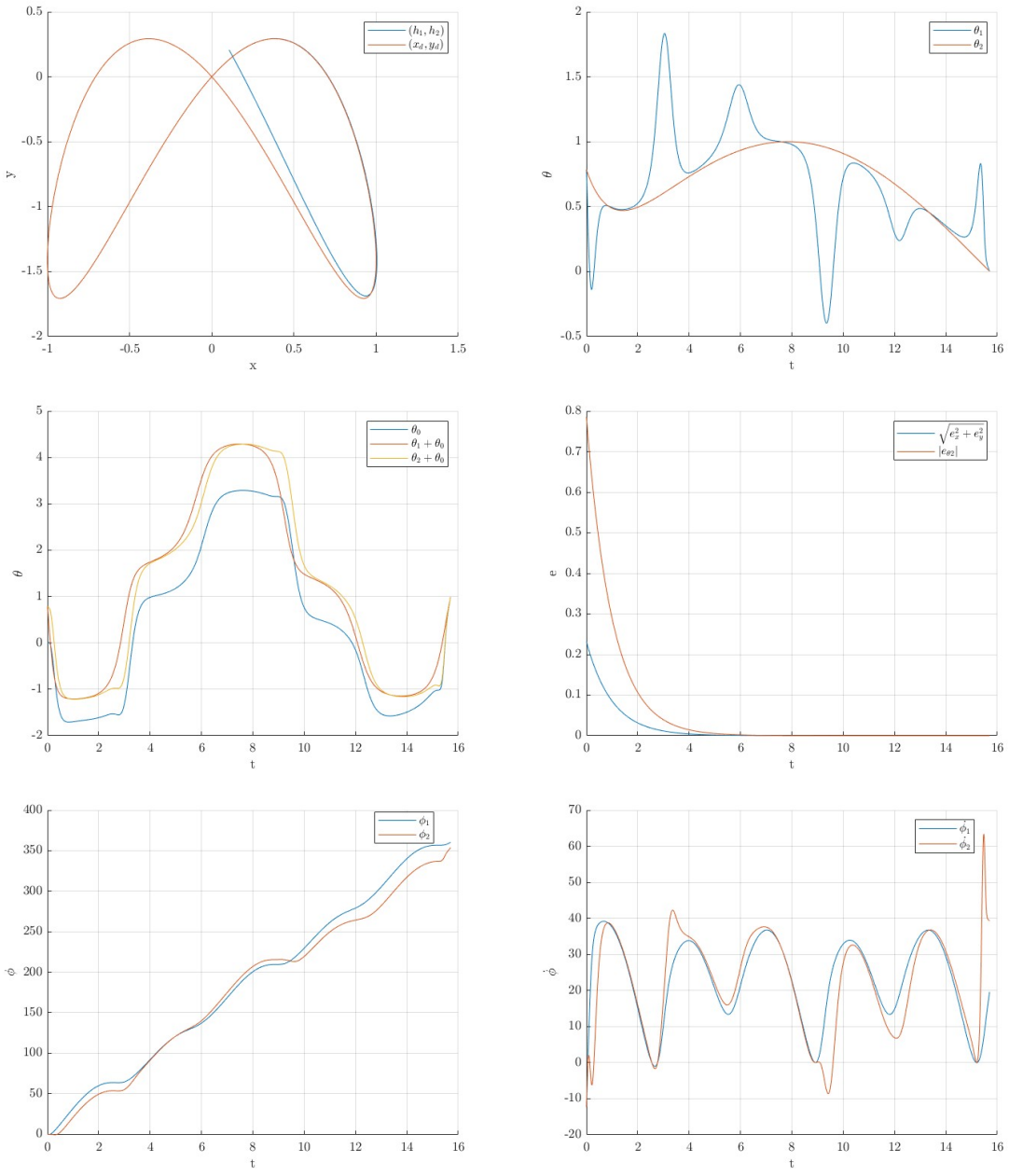
Rysunek 5.9 Linearyzacja statyczna dla ósemki i $\theta_{2d} = \frac{\pi}{4}$



Rysunek 5.10 Linearyzacja statyczna dla ósemki i $\theta_{2d} = \frac{\pi}{2}$



Rysunek 5.11 Linearyzacja statyczna dla ósemki i $\theta_{2d} = 0.1t$



Rysunek 5.12 Linearyzacja statyczna dla ósemki i $\theta_{2d} = \sin 0.2t$

5.2 Linearyzacja dynamiczna

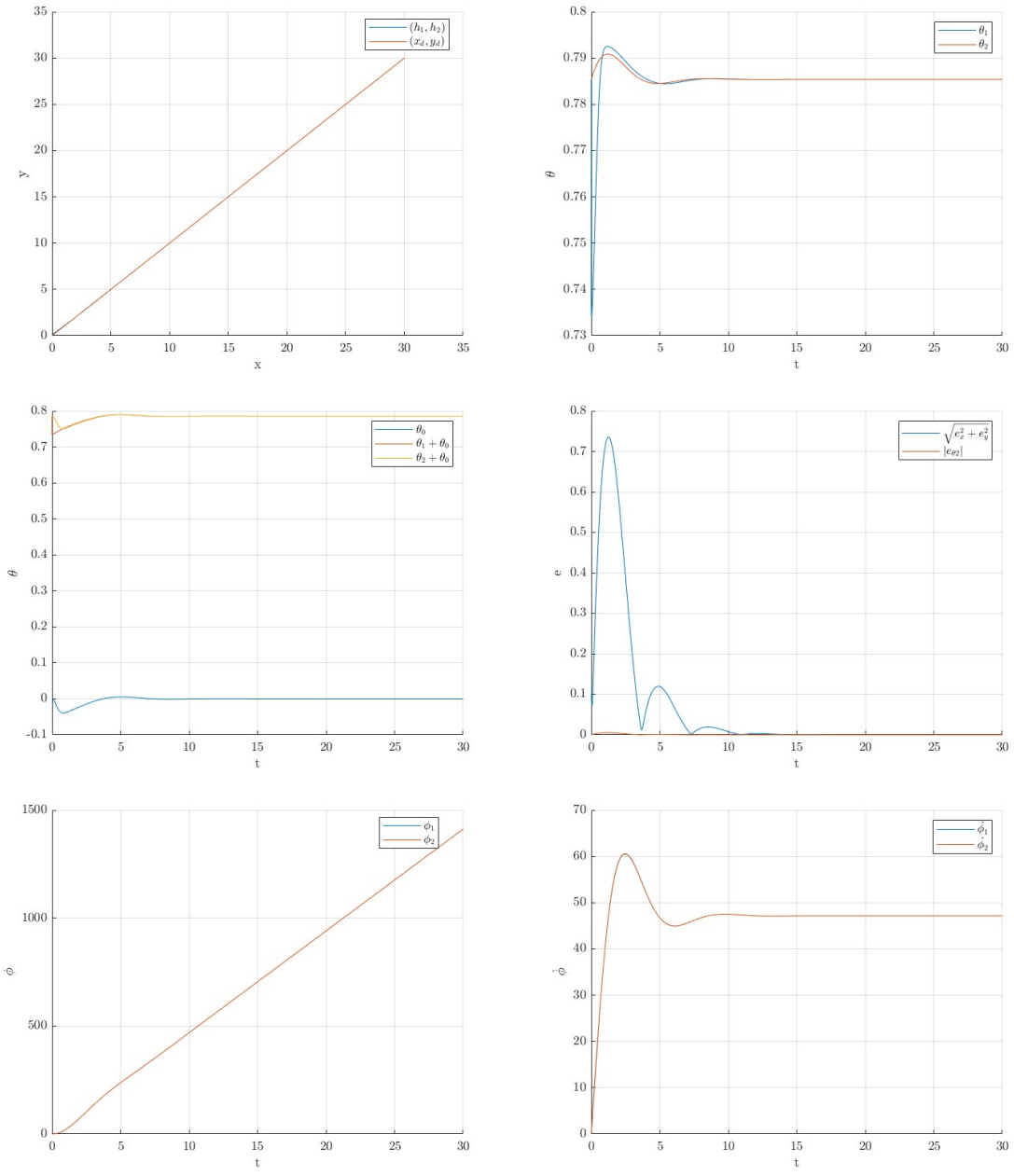
Badając algorytm linearyzacji dynamicznej wykorzystano dwie macierze wzmocnień $K_1 = K_2 = I_3$. Dla przypadku tego algorytmu nie używano przesunięcia współrzędnych x, y o odległość d. Dodatkowo w skład wartości początkowych w algorytmie linearyzacji dynamicznej weszły dwie współrzędne η_2 oraz η_3, dla których przyjęto wartości równe 0.01. Przed takim wyborem przeprowadzono testy podając na wejściu algorytmu różne wartości początkowe η_2, η_3 i zauważono, że algorytm działa lepiej, a także wykresy linearyzacji dynamicznej są bardziej zbliżone do wykresów linearyzacji statycznej, gdy te parametry mają możliwie jak najmniejszą dodatnią ale niezerową wartość.

5.2.1 Trajektoria zadana w postaci linii prostej

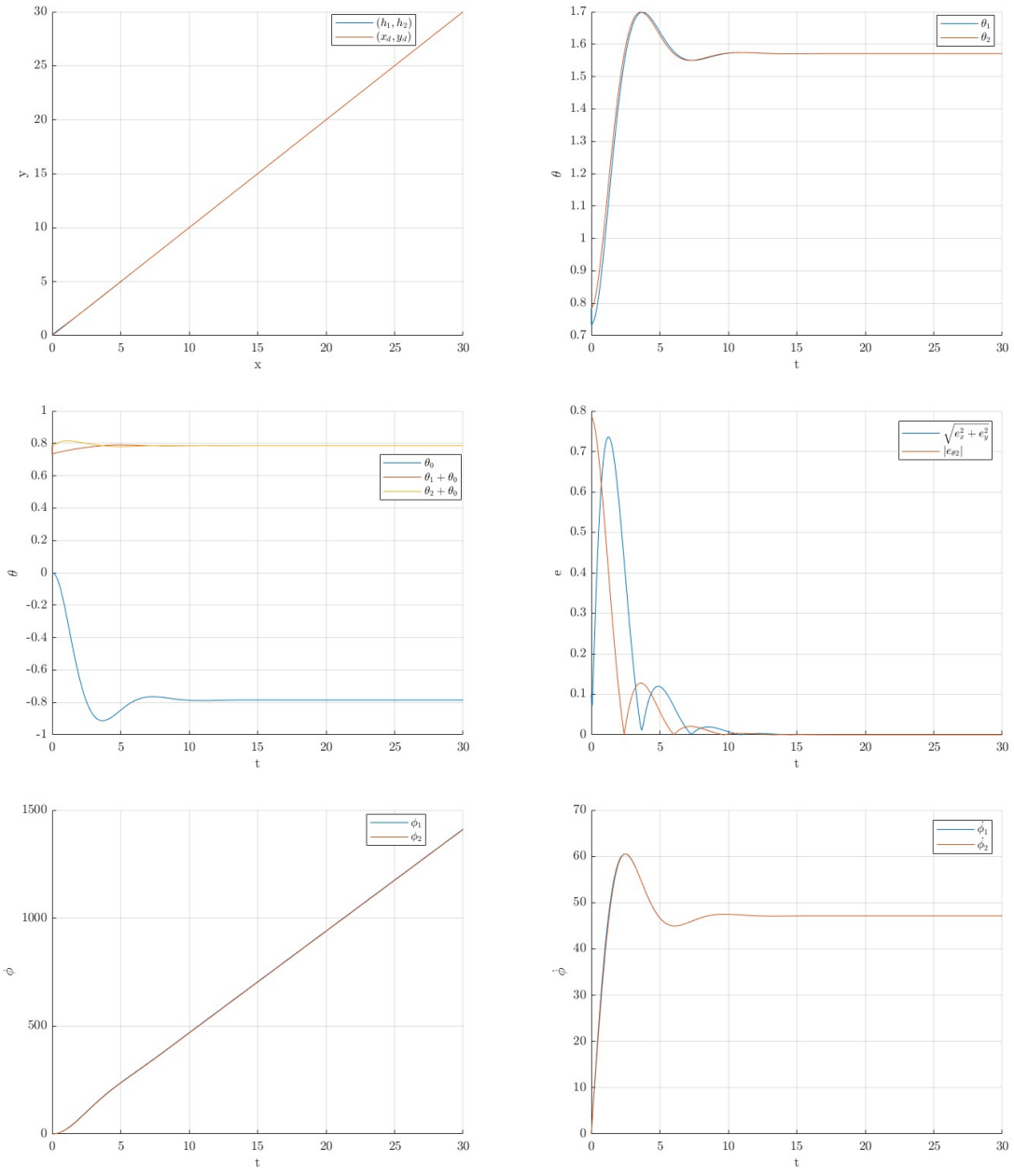
Podobnie jak dla przypadku algorytmu linearyzacji statycznej użyto tych samych przykładów w badaniu linearyzacji dynamicznej. Rozpoczęto od trywialnego przykładu linii prostej przebiegającej pod kątem $\frac{\pi}{4}$ do osi globalnego układu współrzędnych jako trajektorii zadanej dla składowej położenia wyjścia linearyzującego opisanej równaniem (5.1), zobacz rysunek 5.13. Analizując ten przypadek można zauważyć pewne podobieństwa na wykresach pomiędzy dwoma algorytmami. Na pierwszym wykresie ścieżka rzeczywista praktycznie również pokrywa się z zadaną. Z wykresu prędkości i położenia kół widać, że w stanie ustalonym koła robota obracają się ze stałymi, jednakowymi prędkościami, lecz stabilizują się o kilka sekund później. Dosyć znaczącą różnicą pomiędzy algorytmem linearyzacji statycznej, a dynamicznej jest to, że dla linearyzacji dynamicznej, wykres błędu stabilizuje się prawie dwa razy dłużej i występują na nim znaczące oscylacje. Jest to negatywna cecha algorytmu, która jeszcze w tym przykładzie nie wpływa na wykresy ścieżek, lecz dla bardziej skomplikowanych przykładów zadawanych trajektorii zostanie mocniej uwidoczniona. Na wykresie przedstawiającym θ_1 i θ_2 widać, że ta pierwsza wartość nie jest stała, lecz potrzebuje kliku sekund na ustabilizowanie się na wartości $\frac{\pi}{4}$. Rożnica między wartością z której startuje θ_1, a tą na której się stabilizuje jest minimalna, lecz zauważalna. Pozytywną różnicą pomiędzy algorytmami jest, że już dla trywialnego przypadku widać, że bezwzględna orientacja kół jest zmienna w taki sposób, by koła były zorientowane w kierunku stycznej do ścieżki. Widoczne jest to na wykresie dotyczącym trajektorii rzeczywistych θ_0, $\theta_0 + \theta_1$, $\theta_0 + \theta_2$, gdzie te trajektorie są do siebie równoległe.

Układ zachowuje się bardzo podobnie również dla przypadku z niezerowym początkowym błędem orientacji dla trajektorii zadanej (5.5) co zilustrowano na rysunku 5.14. Ścieżka rzeczywista praktycznie pokrywa się z zadaną. Wykresy prędkości i położenia kół są identyczne jak na rysunku 5.13. Wartości θ_1 i θ_2 również potrzebują kilka sekund na ustabilizowanie się na wartości $\frac{\pi}{2}$.

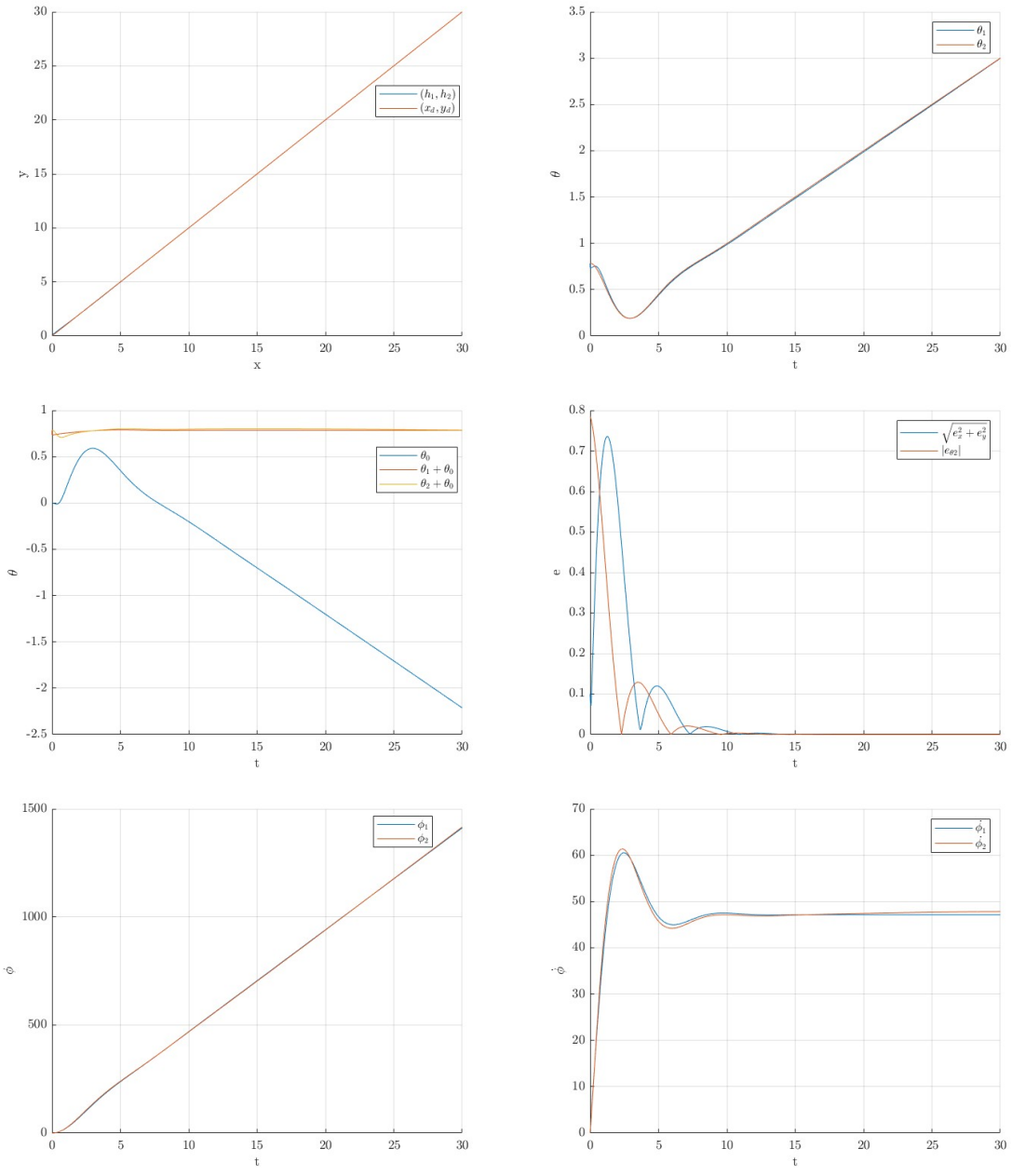
Podobne zachowanie obserwujemy przy zmieniającej się liniowo trajektorii zadanej dla orientacji (5.6). I teraz błędy zbiegają do zera, lecz zawierają oscylacje, rzeczywista i zadana ścieżka praktycznie pokrywają się. Wartości θ_1 i θ_2 rosną liniowo zgodnie z zadaną trajektorią. Na wykresie położenia i prędkości kół, przedstawionych na rysunku 5.15, można już zauważyć minimalne różnice między kołami, co dla zadawanych stałych trajektorii było wcześniej niezauważalne.



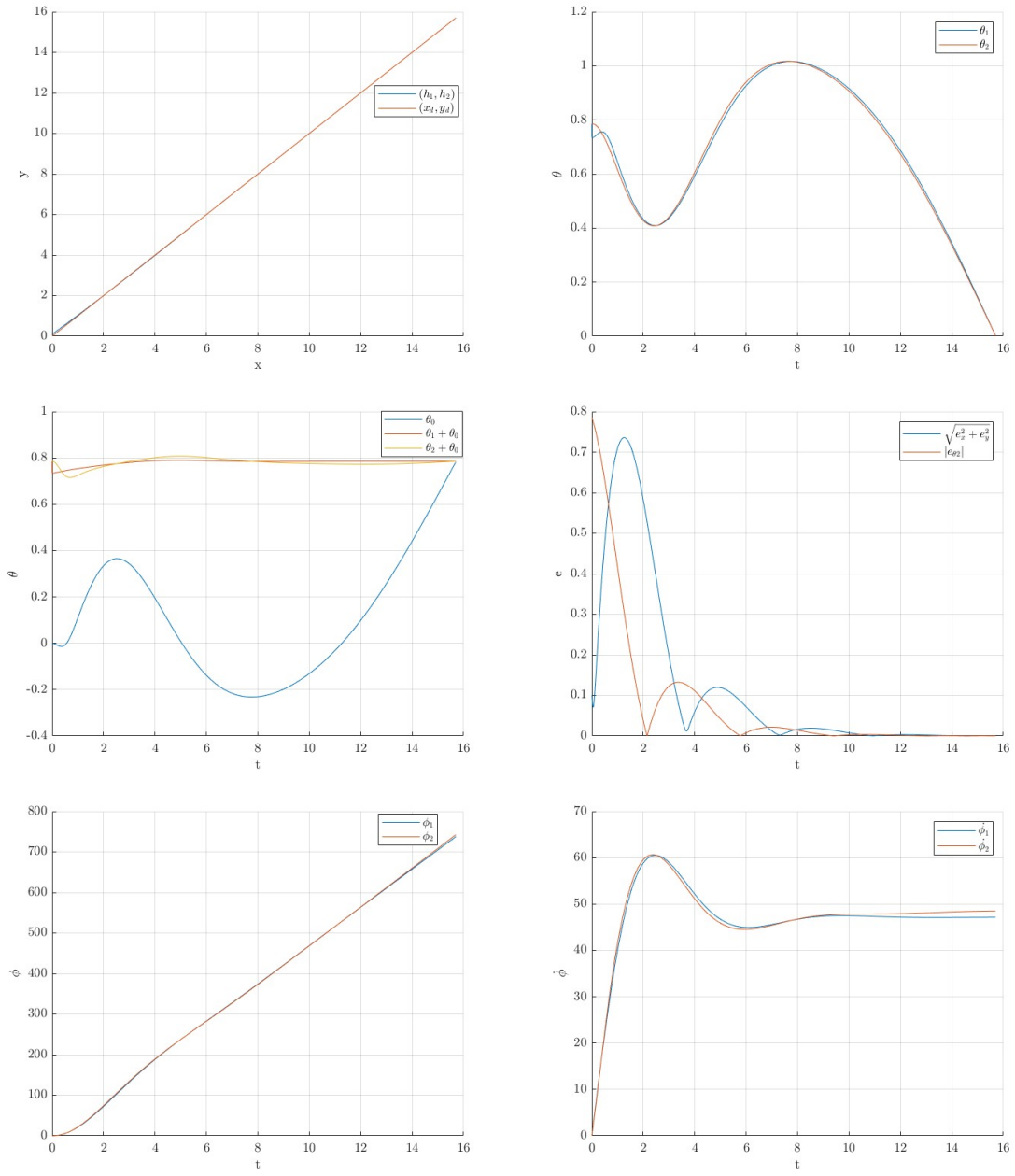
Rysunek 5.13 Linearyzacja dynamiczna dla prostej i $\theta_{2d} = \frac{\pi}{4}$



Rysunek 5.14 Linearyzacja dynamiczna dla prostej i $\theta_{2d} = \frac{\pi}{2}$



Rysunek 5.15 Linearyzacja dynamiczna dla prostej i $\theta_{2d} = 0.1t$



Rysunek 5.16 Linearyzacja dynamiczna dla prostej i $\theta_{2d} = \sin 0.2t$

W przypadku okresowej zmiany trajektorii zadanej dla składowej orientacji przedstawionym na rysunku 5.16 obserwujemy analogiczne zachowania jak wcześniej. Algorytm linearyzacji dynamicznej również cierpi na przypadek dzielenia przez 0, co zostało już wcześniej dokładnie opisane w podrozdziale 5.1.1 pod rysunkiem 5.4.

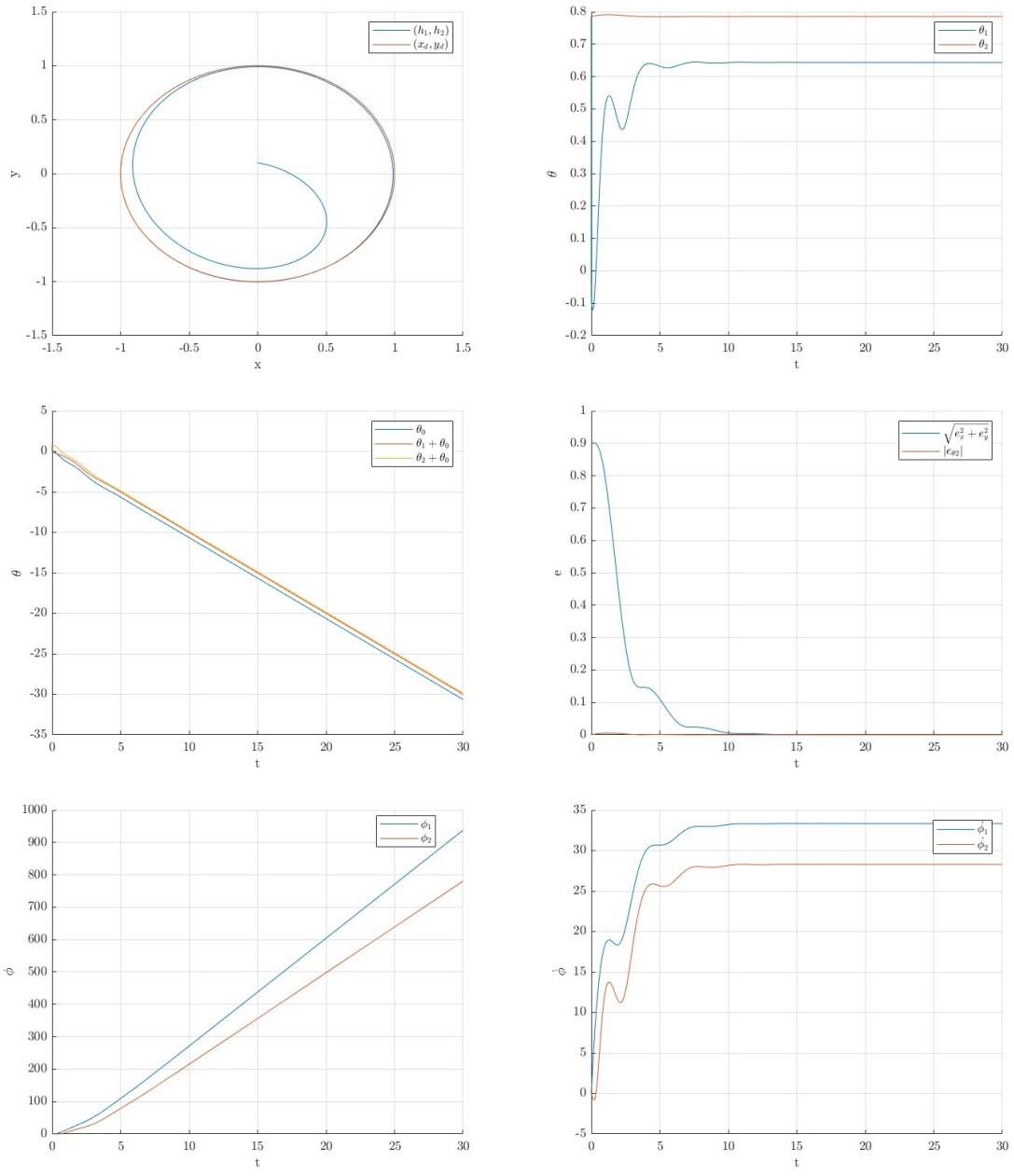
5.2.2 Trajektoria zadana w postaci okręgu

W tym podrozdziale jako trajektorię zadaną dla składowej położenia wyjścia linearyzującego przyjęto okrąg opisany równaniem (5.2). Pierwszą analizowaną trajektorią zadaną dla składowej orientacji jest $\frac{\pi}{4}$, zobacz rysunek 5.17. W porównaniu z rysunkiem 5.13 widać wyraźnie, że ścieżka rzeczywista nie pokrywa się idealnie ze ścieżką zadaną. Z wykresu błędów wynika, że algorytm osiąga zadaną trajektorię w okolicach dopiero 13 sekundy. Ten przypadek bardzo dobrze odzwierciedla to co było wspomniane w podrozdziale 5.2.1 czyli, że algorytm linearyzacji dynamicznej działa gorzej niż linearyzacji statycznej. Z wykresów trajektorii rzeczywistych θ_0, $\theta_0+\theta_1$, $\theta_0+\theta_2$, można zaobserwować, że trajektorie są do siebie równoległe. Bezwzględna orientacja kół jest zmienna w taki sposób, by koła były zorientowane w kierunku stycznej do okręgu. Z wykresów prędkości i położenia kół widać, że koło pierwsze pokonuje większą odległość niż koło drugie. Związane jest to z tym, że koło pierwsze porusza się po zewnętrznej stornie okręgu. Osiąga ono również po około 10 sekundach wyższą i stałą prędkość od koła drugiego.

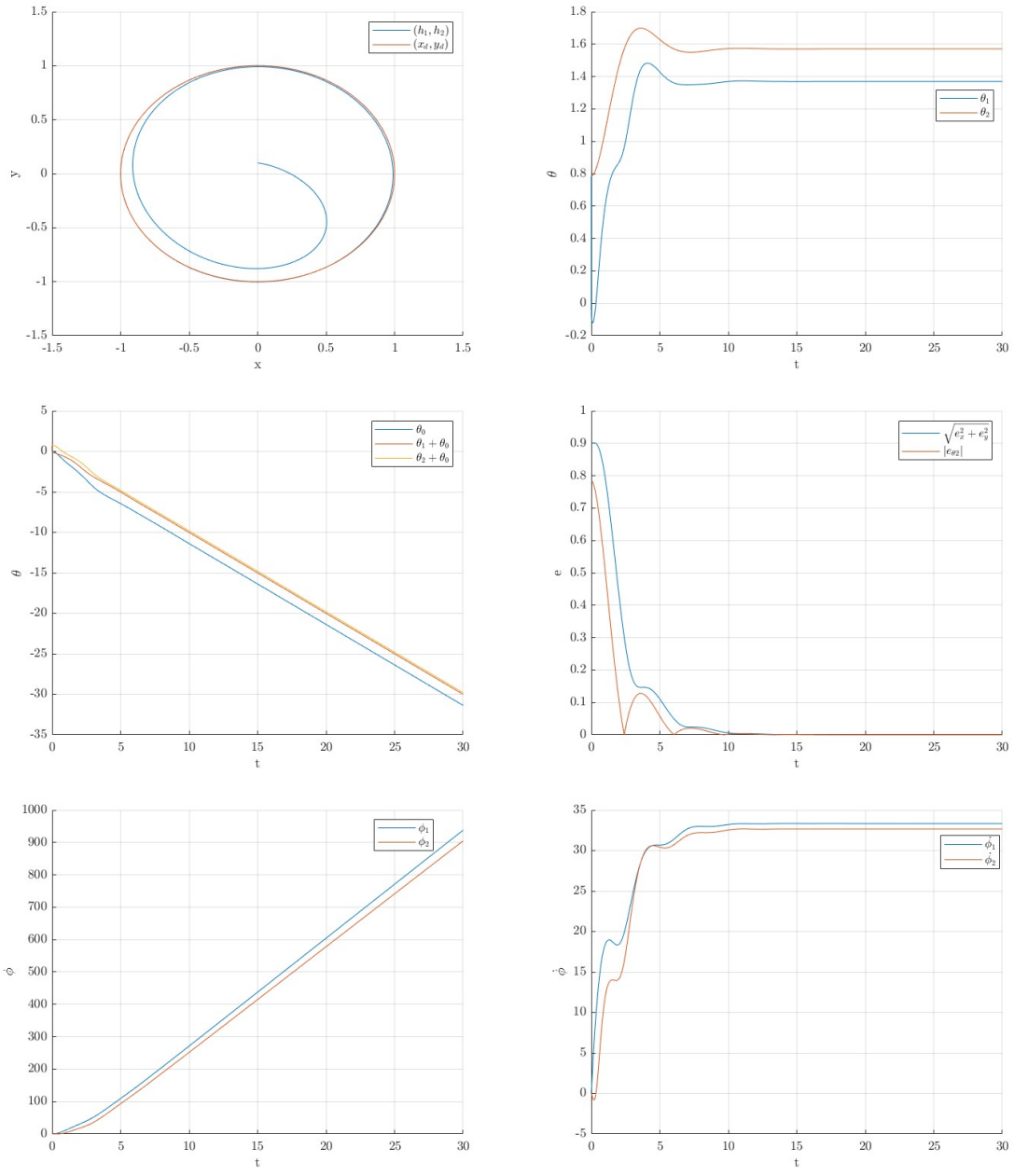
Porównując przypadek trajektorii zadanej dla składowej orientacji równej $\frac{\pi}{2}$ można wyciągnąć podobne wnioski co dla poprzedniego przypadku. Na wykresach położenia i prędkości kół widać mniejsze różnice wartości między nimi. Dodatkowo na wykresie błędów dochodzi składowa orientacji, która zbiega do zera z oscylacjami. Wykresy te przedstawiono na rysunku 5.18.

W przypadku trajektorii zadanej dla składowej orientacji w postaci prostej (5.6), zobacz rysunek 5.19, wykres ścieżki rzeczywistej i zadanej pokazują tą samą zależność co we wcześniejszych dwóch rysunkach 5.17 i 5.18. Wykresy błędów zbiegają do zera wraz z oscylacjami. Na wykresach przedstawiających wartości kątów obrotu i skręcenia kół można dostrzec zależność, którą opisano już dla linearyzacji statycznej zilustrowanej na rysunku 5.7. Widać, że kąt skręcenia koła pierwszego jest mniejszy niż koła drugiego, natomiast kąt obrotu koła pierwszego jest większy niż drugiego. Jednakże iloczyn tych dwóch wartości dla obu kół jest równy. Prędkość koła pierwszego stabilizuje się na stałej wartości po około 7 sekundach, natomiast dla koła drugiego powoli rośnie po początkowych oscylacjach.

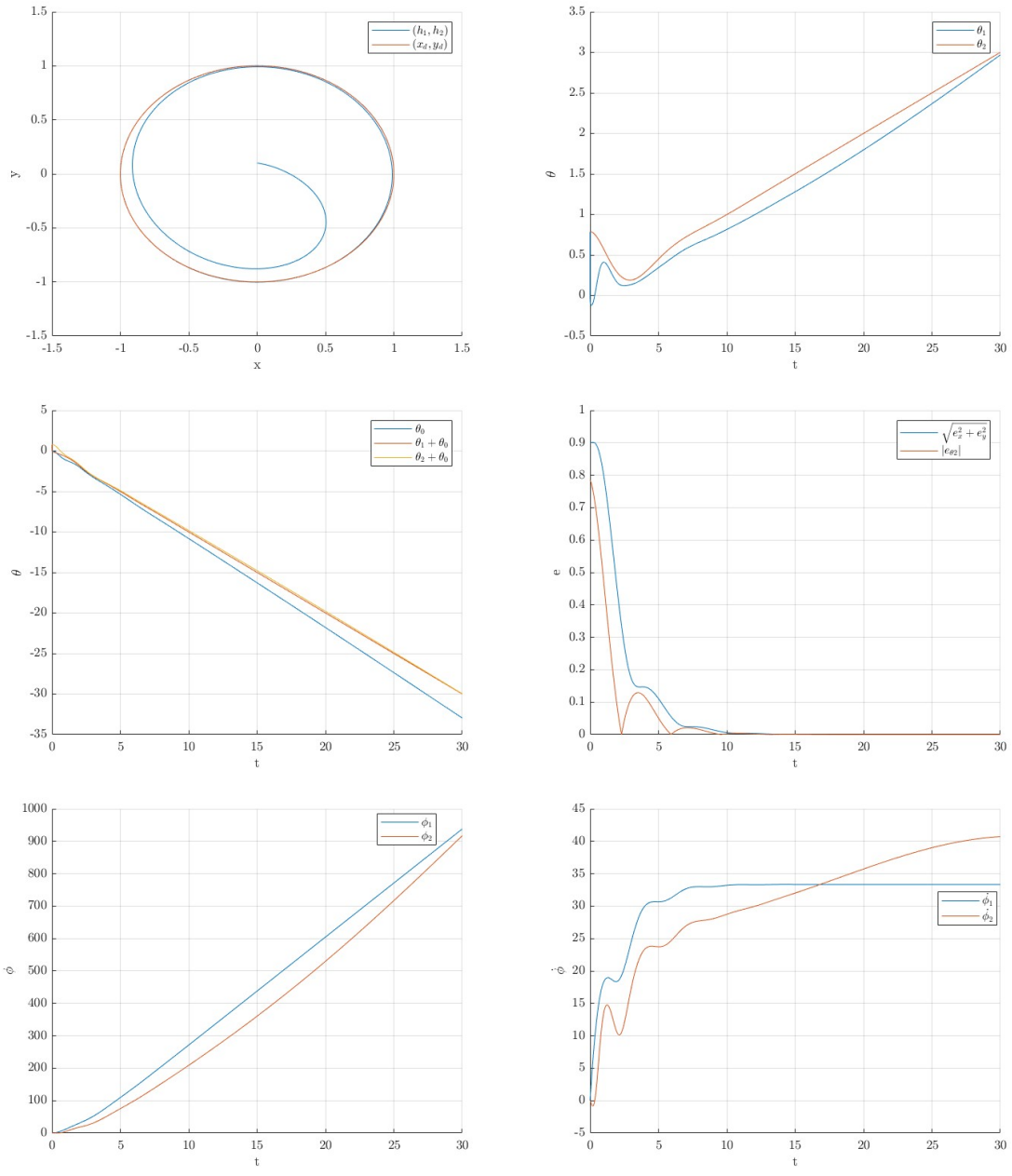
Dla okresowo zmieniającej się trajektorii zadanej składowej orientacji (5.7), patrz rysunek 5.20, wykresy przedstawiają bardzo podobne rezultaty co dla wcześniejszych trzech badanych trajektorii. Jedyną różnicą, która została już wytłumaczona kilka razy, jest przebieg wykresów kończący się w okolicach 16 sekundy.



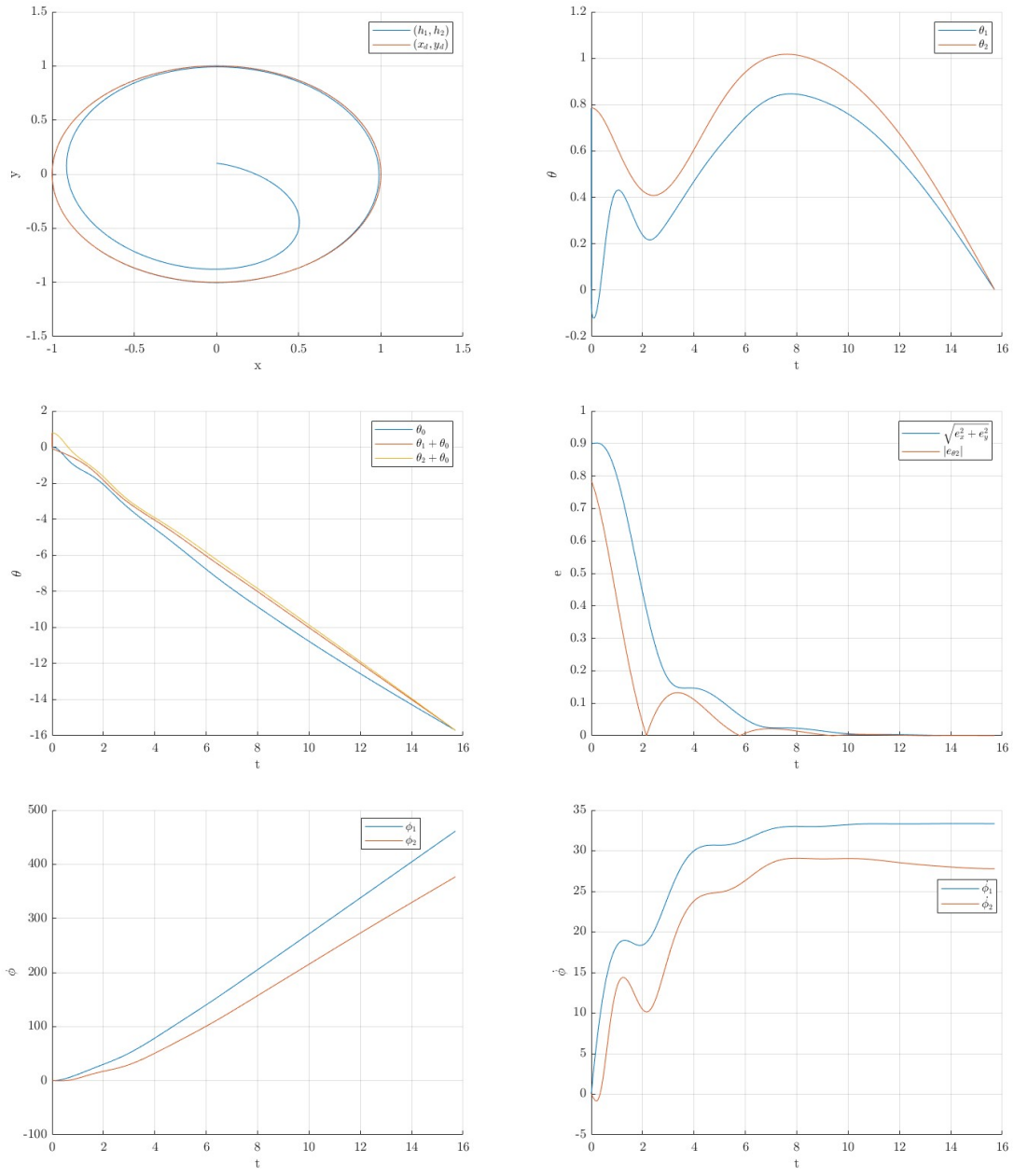
Rysunek 5.17 Linearyzacja dynamiczna dla okręgu i $\theta_{2d} = \frac{\pi}{4}$



Rysunek 5.18 Linearyzacja dynamiczna dla okręgu i $\theta_{2d} = \frac{\pi}{2}$



Rysunek 5.19 Linearyzacja dynamiczna dla okręgu i $\theta_{2d} = 0.1t$



Rysunek 5.20 Linearyzacja dynamiczna dla okręgu i $\theta_{2d} = \sin 0.2t$

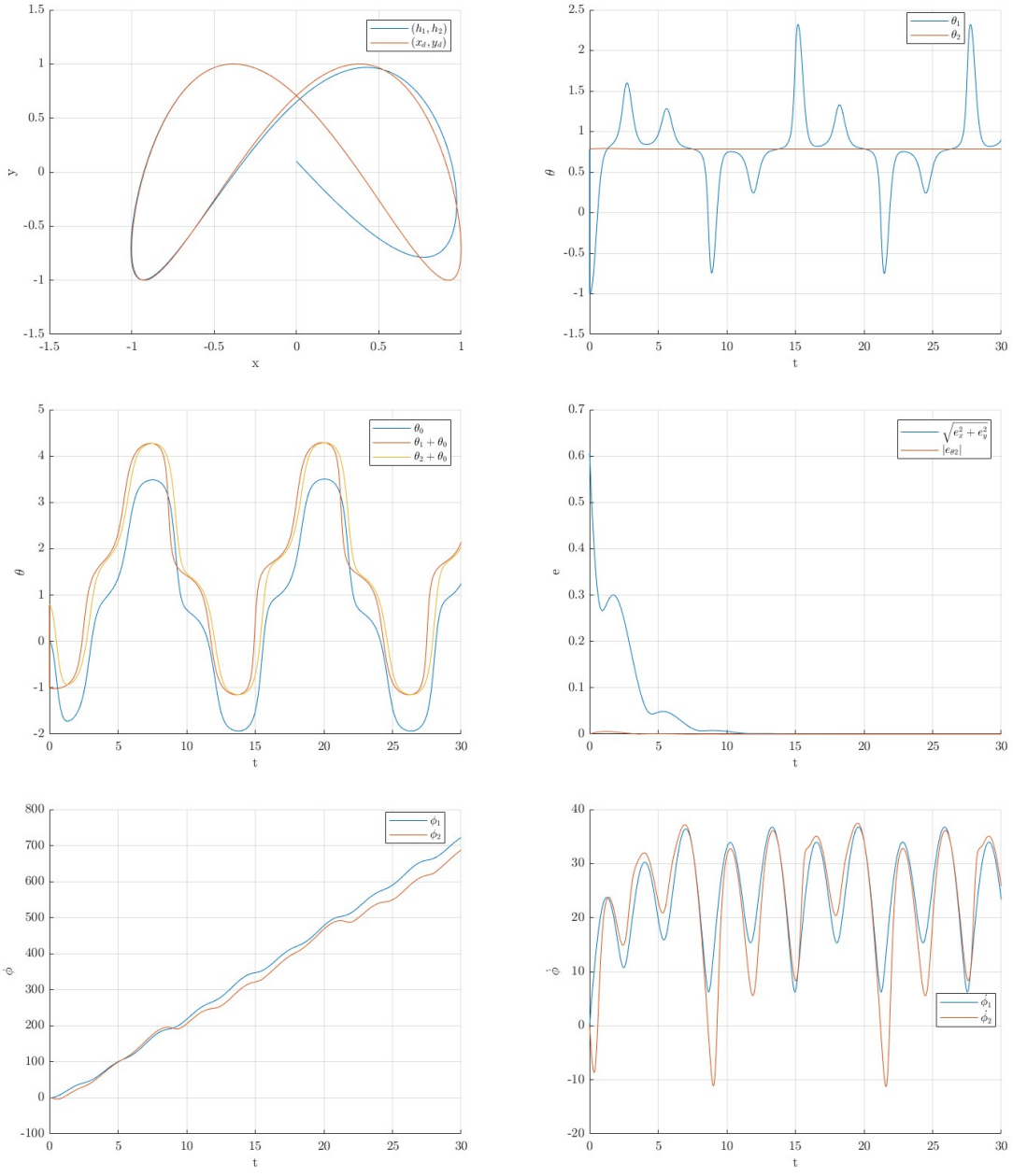
5.2.3 Trajektoria zadana w kształcie ósemki

Ostatnim analizowanym przykładem w badaniach jest trajektoria zadana opisana równaniem (5.3), która swoim kształtem przypomina ósemkę, zobacz rysunek 5.21. Dla trajektorii zadanej składowej orientacji równej $\frac{\pi}{4}$, wykres ścieżki rzeczywistej i zadanej ukazuje nam coś dotychczas niespotykanego. Ścieżki różnią się od siebie na początku zasadniczo. Algorytm porusza się po ścieżce dosyć wybrzuszonej. Aby zniwelować ten efekt można manipulować wartościami wzmocnień K_1 i K_2. Na przykład wybierając dodatnie wartości $K_1 = 5$ i $K_2 = 7$ efekt ten zniknie i ścieżki będą podobne do tych pokazanych dla przypadku linearyzacji statycznej na rysunku 5.9. Porównując wykres błędów algorytmu linearyzacji dynamicznej z błędami algorytmu linearyzacji statycznej widać, że stabilizują się one dopiero po 10 sekundach oraz charakteryzują je delikatne oscylacje. Wykresy prędkości i położenia kół robota, wartości θ_1 i θ_2 są niemalże identyczne jak dla linearyzacji statycznej i przedstawiają te same zależności.

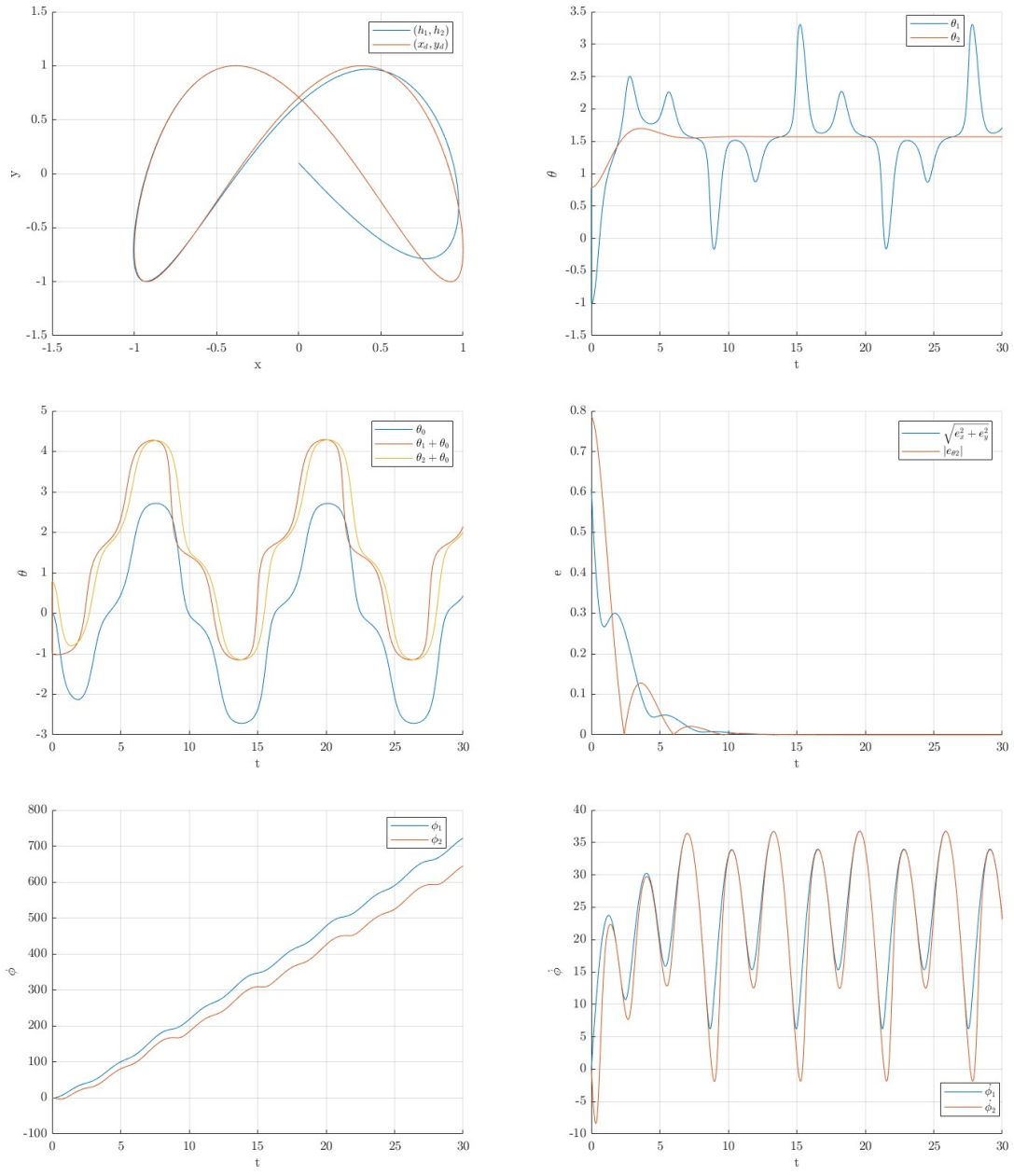
Układ zachowuje się bardzo podobnie również dla przypadku z niezerowym początkowym błędem orientacji dla trajektorii zadanej (5.5) co zilustrowano na rysunku 5.22. Analizując ten sam przypadek dla obu algorytmów widać, że poza wykresami ścieżek i błędów wykresy są niemalże identyczne i można z nich wyciągnąć te same wnioski.

Dla przypadku trajektorii zadanej składowej orientacji w postaci (5.6), zobacz rysunek 5.23, dla wykresów θ_1 i θ_2, położeń i prędkości kół, a także trajektorii rzeczywistych θ_0, $\theta_0+\theta_1$, $\theta_0+\theta_2$ widzimy, że te wykresy są identyczne jak te przedstawione na rysunku 5.11 dla linearyzacji statycznej. Jedyne różnice dotyczą wcześniej opisanych wykresów ścieżek rzeczywistej i zadanej. Problem ten dogłębnie wyjaśniono na początku podrozdziału 5.2.3. Natomiast wykresy błędów są identyczne jak te na rysunku 5.22. Wykresy θ_0, $\theta_0+\theta_1$, $\theta_0+\theta_2$ są do siebie równoległe jak w poprzednich przypadkach.

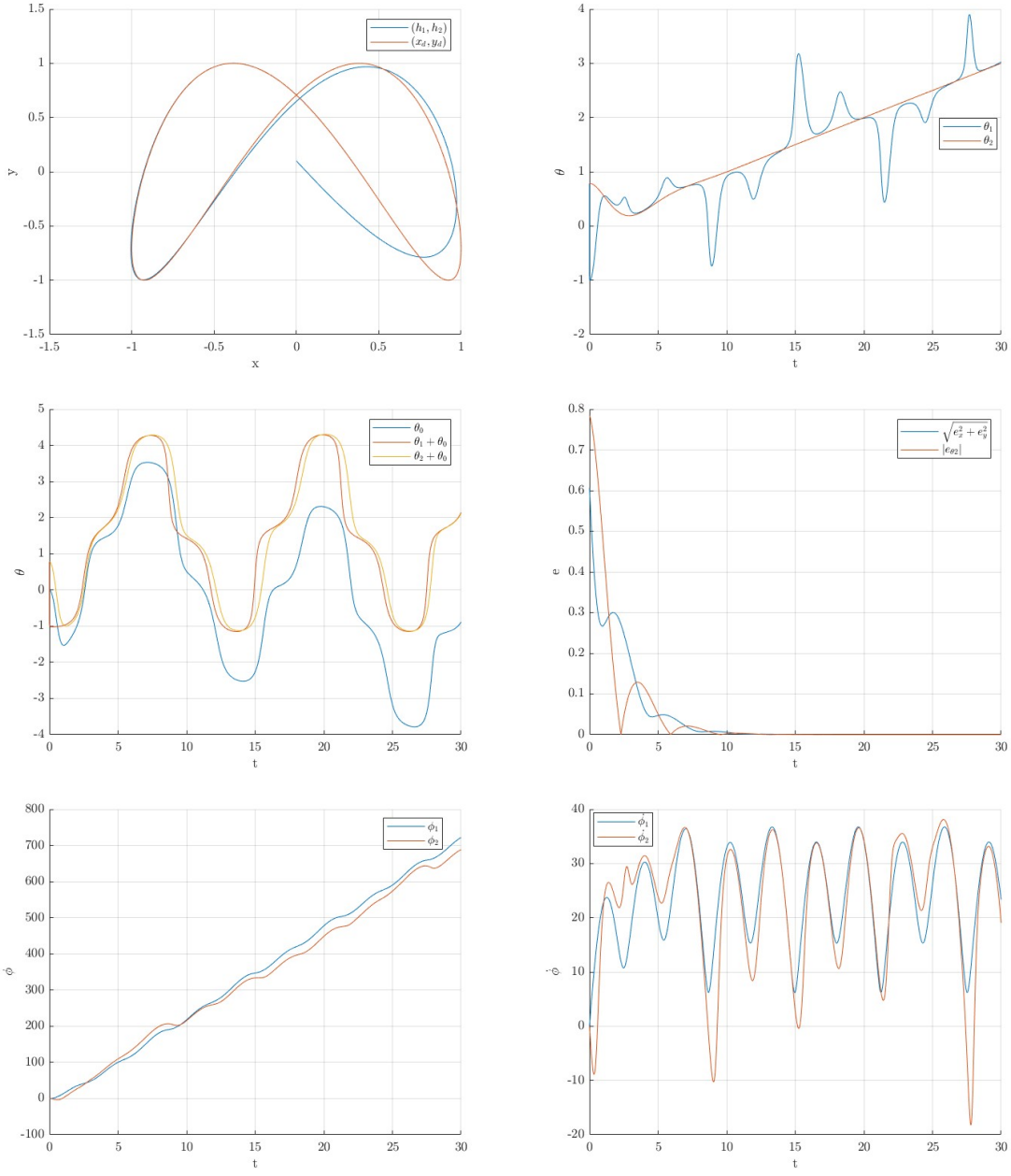
Ostatni rysunek 5.24 dotyczący trajektorii zadanej dla składowej orientacji 5.7 nie przynosi żadnych nowych wniosków, jednakże pokazuje, że algorytm działa poprawnie dla skomplikowanych przypadków. Ścieżki mają taki sam przebieg jak dla poprzednich trzech przypadków składowej orientacji. Na wykresach błędów występują oscylacje i zbiegają do zera po około 10 sekundach. Z wykresu prędkości i położeń kół widzimy, że ich wartości są zmienne w czasie. To oznacza, że dla pewnych chwil raz pierwsze koło osiąga większą wartość prędkości czy położenia, a dla innych drugie koło. Wykres θ_1 ma przebieg oscylacyjny, natomiast θ_2 przebiega sinusoidalnie zgodnie z trajektorią zadaną.



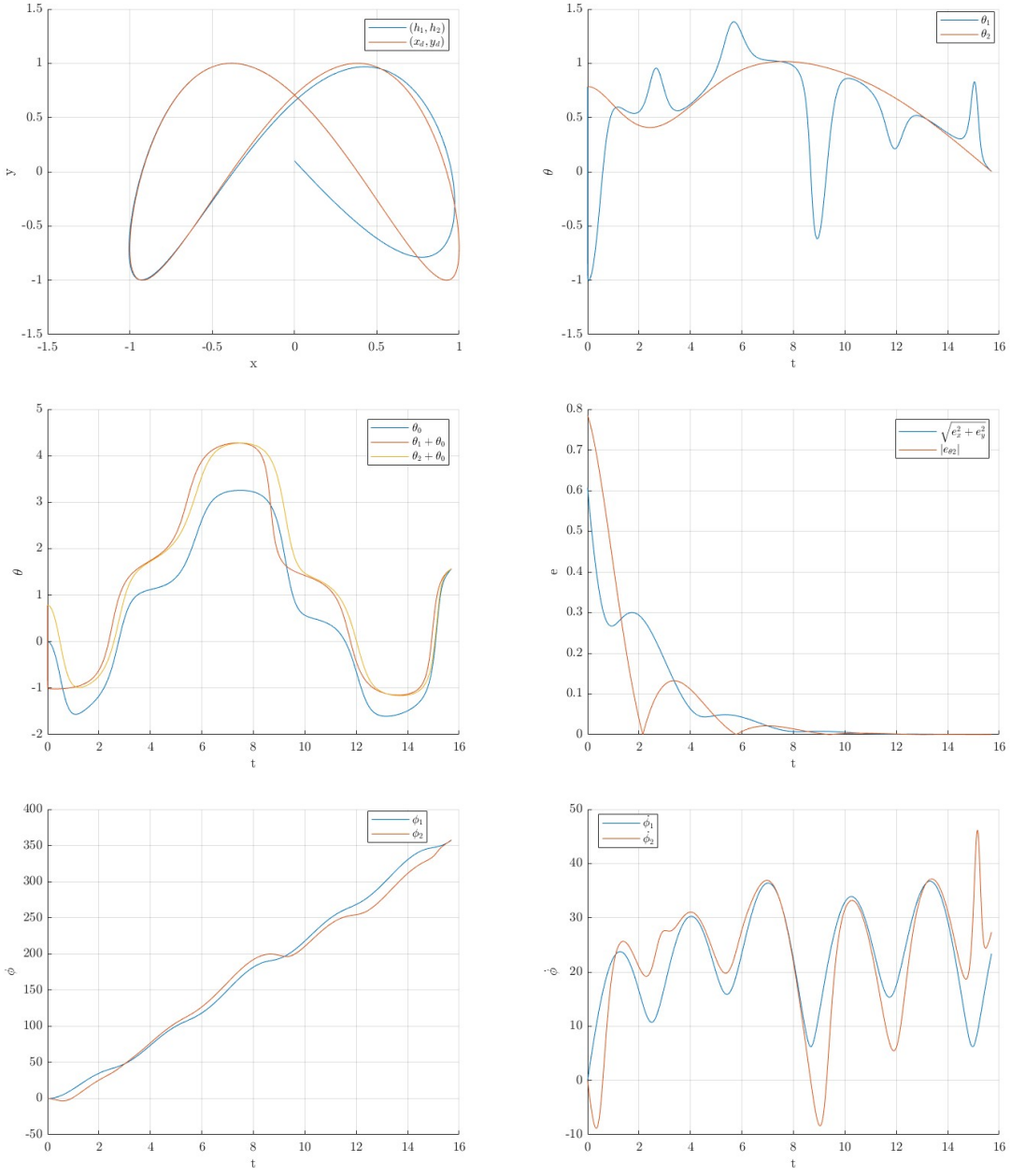
Rysunek 5.21 Linearyzacja dynamiczna dla ósemki i $\theta_{2d} = \frac{\pi}{4}$



Rysunek 5.22 Linearyzacja dynamiczna dla ósemki i $\theta_{2d} = \frac{\pi}{2}$



Rysunek 5.23 Linearyzacja dynamiczna dla ósemki i $\theta_{2d} = 0.1t$



Rysunek 5.24 Linearyzacja dynamiczna dla ósemki i $\theta_{2d} = \sin 0.2t$

Rozdział 6

Podsumowanie

Celem pracy było dokonanie przeglądu algorytmów sterowania możliwych do zastosowania w zadaniu sterowania robota mobilnego klasy (1,2) oraz porównanie ich własności w badaniach symulacyjnych, co zostało zrealizowane. Po zaznajomieniu się z właściwościami robotów mobilnych w szczególności klasy (1,2), zaimplementowano model kinematyki takiego robota. Kolejno po zapoznaniu się z dostępnymi algorytmami sterowania robotów mobilnych, wybrano dwa algorytmy i przeprowadzono ich implementację. Były to algorytm linearyzacji statycznej, a także algorytm linearyzacji dynamicznej. Następnie dla robota wybrano również trajektorie zadane, dla których zostały zbadane własności algorytmów. Ostatnim zadaniem była analiza porównawcza obydwu algorytmów. Badano je pod względem szybkości oraz dokładności w osiąganiu zadanej trajektorii. Sprawdzano jak zachowują się wartości błędów składowych położenia i orientacji. Poddano również analizie zachowanie się w czasie kątów skręcenia i obrotu kół, a także bezwzględnej orientacji korpusu robota i jego kół. Analizie podlegały jeszcze prędkości kół.

Pierwszym wnioskiem, który wynika z matematycznej analizy algorytmów jest to, aby algorytm linearyzacji statycznej działał w sposób prawidłowy należy współrzędne położenia punktu robota przesunąć o pewną odległość d. W badanym przypadku punktem, który został przesunięty był środek pierwszego koła robota. Natomiast dla algorytmu linearyzacji dynamicznej nie było konieczne stosowanie przesunięcia punktu robota o pewną odległość. Dla obu algorytmów badany robot klasy (1,2) potrafi podążać po zadanej trajektorii, zarówno jak dla tej trywialnej, jak i tej bardziej skomplikowanej. Dla algorytmu linearyzacji statycznej osiąga on w prawie dwukrotnie krótszym czasie zadaną ścieżkę, co potwierdzają wykresy wartości błędów wyjść linearyzujących. Dla obydwu algorytmów robot po zadaniu trajektorii dla trzeciej składowej wyjścia linearyzującego osiąga ją po krótkim czasie i stabilizuje się na tej wartości. Wykresy przedstawiające położenia i prędkości kół również prezentują oczekiwane dane. W zależności od tego czy robot porusza się po okręgu czy po prostej, jedno koło może w tym czasie poruszać się z większą prędkością po dłuższym torze od drugiego lub mogą mieć takie same wartości co wydaję się być dosyć logiczne. Dodatnie wzmocnienia K_1 oraz K_2 dla algorytmu linearyzacji dynamicznej mają spore znaczenie dla dokładności w podążaniu po zadanej ścieżce. Stosując większe wartości wzmocnień niż 1 pozwalamy na szybsze i dokładniejsze śledzenie zadanej trajektorii. Dla algorytmu linearyzacji statycznej przyjęto wartość wzmocnienia równą 1 i nie badano algorytmu dla innych wartości wzmocnień, gdyż zwracane wyniki już były satysfakcjonujące. Dodatkowo w algorytmie linearyzacji dynamicznej ważne jest, aby stosować dodatnie i niezerowe wartości początkowe η_2 i η_3 jeśli chcemy, aby robot się poruszał. Ciekawym

i równocześnie oczywistym wnioskiem przeprowadzonych badań jest to, że bezwzględna orientacja kół jest zmienna w taki sposób by koła były zorientowane w kierunku stycznej do ścieżki.

Po przeanalizowaniu zasymulowanych przypadków stwierdzono, że szybszym oraz dokładniejszym algorytmem dla robota mobilnego klasy (1,2) jest algorytm linearyzacji statycznej, jednakże stosując ten algorytm niezbędne jest przesunięcie punktu robota o pewną odległość d co oznacza, że zamiast badać trajektorię zadaną dla wyróżnionego punktu robota, badany jest wirtualny punkt znajdujący się poza nim. Możliwe, że dla nie specjalnie skomplikowanego modelu robota nie jest konieczne stosowanie trudniejszych w implementacji algorytmów.

Dla osoby, która byłaby zainteresowana kontynuacją badań nad tematyką dotyczącą sterowania robotem mobilnym klasy (1,2) można polecić dokładniejsze sprawdzenie zachowania się robota dla różnych wartości wzmocnień, co w tej pracy nie zostało wykonane. Przydatne byłoby też wyeliminowanie problemu z dzieleniem przez zero. Ciekawym zadaniem, które pozwoliłoby na lepsze zobrazowanie tematyki zadania sterowania robotem mobilnym byłoby stworzenie aplikacji pokazującej animację, na której można śledzić zachowanie się robota, podczas sterowania wirtualnym punktem.

Literatura

[dWSB96] C. C. de Wit, B. Siciliano, G. Bastin, *Theory of Robot Control*. Springer – Verlag New York, 1996.

[Gó17] D. Góral, *Konstrukcja robota mobilnego napędzanego dwiema półsferami*. Praca inżynierska, Wydział Mechaniczny, Politechnika Wrocławska, https://kcir.pwr.edu.pl/~mucha/#studenckie, 2017.

[Jon14] P. Joniak, *Badania symulacyjne zachowania robota mobilnego napędzanego dwiema półsferami*. Projekt inżynierski, Wydział Elektroniki, Politechnika Wrocławska, https://kcir.pwr.edu.pl/~mucha/#studenckie, 2014.

[Jon17] P. Joniak, *Zadanie sterowania robota mobilnego napędzanego dwiema półsferami*. Praca magisterska, Wydział Elektroniki, Politechnika Wrocławska, https://kcir.pwr.edu.pl/~mucha/#studenckie, 2017.

[Lub23] T. Lubelski, *Budowa małego mobilnego robota laboratoryjnego klasy (1,2)*. Praca inżynierska, Wydział Elektroniki, Fotoniki i Mikrosystemów, Politechnika Wrocławska, https://kcir.pwr.edu.pl/~mucha/#studenckie, 2023.

[Mata] MathDF. Kalkulator macierzy. https://mathdf.com/pl/.

[Matb] MathWorks. Matlab. https://www.mathworks.com/products/matlab.html.

[Matc] MathWorks. Simulink. https://www.mathworks.com/products/simulink.html.

[Mej17] S. Mejer, *Budowa i sterowanie robota mobilnego klasy (1,2)*. Praca inżynierska, Wydział Elektroniki, Politechnika Wrocławska, https://kcir.pwr.edu.pl/~mucha/#studenckie, 2017.

[Ryb13] M. Rybaczyński, *Model robota mobilnego napędzanego za pośrednictwem półsfery*. Praca inżynierska, Wydział Elektroniki, Politechnika Wrocławska, https://kcir.pwr.edu.pl/~mucha/#studenckie, 2013.

[TMD+00] K. Tchoń, A. Mazur, I. Dulęba, R. Hossa, R. Muszyński, *Manipulatory i roboty mobilne: modele, planowanie ruchu, sterowanie*. Akademicka Oficyna Wydawnicza PLJ, 2000.

Spis rysunków