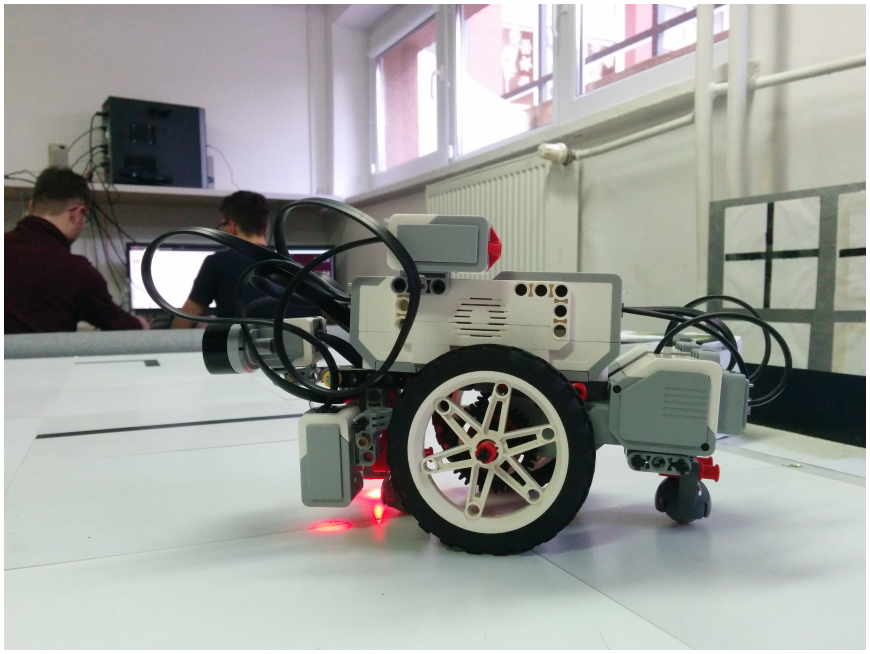
Dokumentacja
Robot na labolatoria z WR

Paweł Rybak
Aleksander Brzozowski

1 Budowa

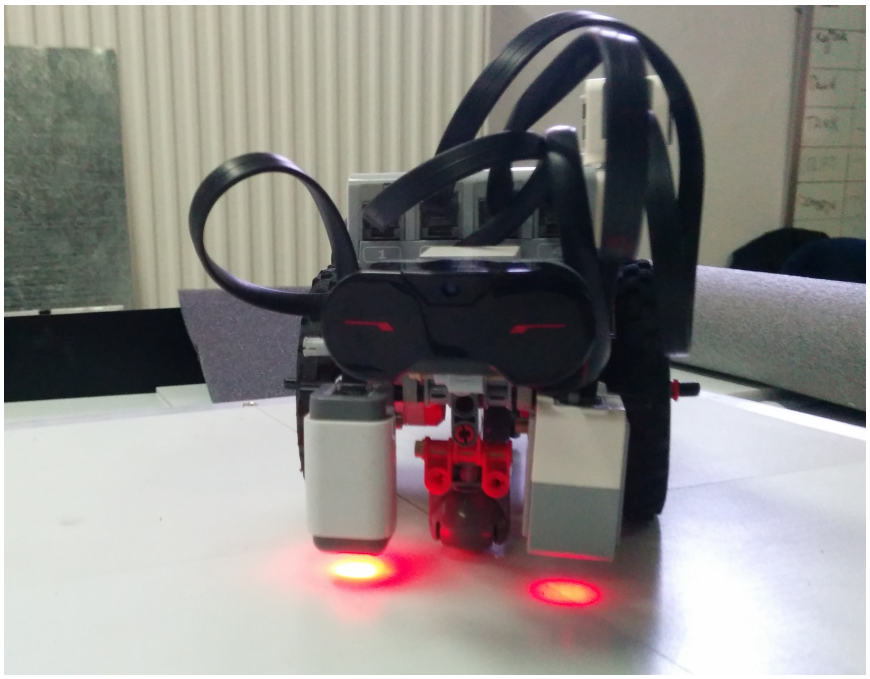
Robot jest osadzony na czterech kołach z czego dwa są przymocowane bezpośrednio do dwóch niezależnych silników, natomiast dwa koła sferyczne są bierne i służą jedynie utrzymaniu stabilności konstrukcji. Robot był skonstruowany tak, by rzut środka ciężkości robota na płaszczyznę wyznaczaną przez koła był jak najbliżej osi kół napędowych. Założenie to zostało zrealizowane z wystarczającą precyzją. Niezależny napęd każdego z kół pozwala



Rysunek 1: Widok z boku

na zmianę orientacji robota poprzez napęd różnicowy. Istotne dla umożliwienia skrętu konstrukcji było również zastosowanie kół podporowych, które nie mają więzów ruchu. Funkcję tą z dobrym wynikiem spełniają koła sferyczne. Nad silnikami została umieszczona jednostka obliczeniowa, której

funkcję pełni kostka LEGO® EV3. Jest ona lekko wysunięta do przodu względem silników, tak aby przesunąć środek masy konstrukcji do przodu. Efekt doskonale widać na obrazku 1. Czujniki zostały umieszczone przed kostką. W konstrukcji zostały wykorzystane dwa czujniki. Czujnik koloru z zestawu LEGO® EV3, który był wykorzystany w trybie mierzenia światła odbitego, oraz czujnik światła z zestawu LEGO® NXT. Czujniki te zostały przystosowane do wykrywania tego czy robot najedzie na linię, więc zostały ustawione w odległości ok 3,5cm od siebie, tak, aby linia mieściła się pomiędzy nimi, chociaż konstrukcja pozwalała na szybką modyfikację tej wartości, gdyż mocowanie tych czujników było ruchome. Nad wcześniej wspomnianymi czujnikami został umieszczony czujnik podczerwieni, służący do wykrywania przeszkód. Ułożenie czujników widać na poniższym obrazku, oraz obrazku 1.



Rysunek 2: Widok z przodu

2 Regulator PID

W naszym robocie został zastosowany lekko zmodyfikowany algorym regulatora PID. Regulator PID służy do utrzymywania wartości wyjściowej na poziomie wartości zadanej. Uzyskuje on to przy pomocy pętli sprzężenia zwrotnego - licząc róznicę między wartością aktualną i wartością zadaną, a następnie z pomocą efektorów dąży do wyzerowania owego uchybu. Posiada

on trzy człony, od których pochodzi jego nazwa - człon proporcjonalny (P), całkujący (I) i różniczkójący (D). Człon proporcjonalny służy do kompensacji aktualnego uchybu, człon całkujący kompensuje akumulację uchybów z przeszłości, natomiast człon różniczkójący kompensuje przewidywany w przyszłości uchyb. Cały regulator opisuje się jednym prostym wzorem:

$$u(t) = k_r[e(t) + 1/T_i \int_0^t e(\tau)d\tau + T_d \frac{de(t)}{dt}] \quad (1)$$

gdzie $u(t)$ to wartość sterowania, a $e(t)$ - wartość uchybu. Transmitancja takiego regulatora, przy założeniu idealnego członu różniczkującego wygląda następująco:

$$G(s) = k_r[1 + 1/sT_i + sT_d] \quad (2)$$

Aby wykorzystać ów regulator w praktyce należy jeszcze dobrać odpowiednie nastawy regulatora, czyli wartości współczynników $K_p = k_r$, $K_i = \frac{k_r}{T_i}$, oraz $K_d = k_r T_d$.

3 Implementacja algorytmu

3.1 Algorytm PID

Jak już wspomniano w sekcji 2 algorytm zastosowany w opisywanym robocie był zmodyfikowaną wersją regulatora PID. Podstawą do sterowania silnikami był odczyt z czujników, które zostały programowo wyskalowane, aby zwracać wartości w zakresie od 0 do 100, gdzie 0 to kolor biały, a 100 to czarna linia. Uchyb w naszym algorymie był określany jako różnica odczytu z prawego czunika oraz lewego czujnika. Stąd z pomocą znaku uchybu można określić w którą stronę robot zjeżdża z trasy i z pomocą odpowiedniego sterowania silnikami wracać na trasę. W czasie gdy robot był obydwoma czujnikami na kolorze białym błąd jest zerowy więc robot jedzie prosto - zgodnie z założeniem. Człon proporcjonalny wyliczany jest w oczywisty sposób - poprzez wymnożenie błędu chwilowego razy współczynnik K_p. Człon różniczkujący w dziedzinie dyskretnej jest liczony jako różnica między błędem chwilowym a błędem w poprzednim pomiarze i pomnożony razy współczynnik K_d. Człon całkujący nie jest używany w naszym algorytmie, gdyż po empirycznym sprawdzeniu jego wyników ustaliliśmy, iż robot działa lepiej bez niego. Tak wyliczone sterowanie jest dodawane do nominalnej wartości mocy silników i podawane na wejście silnika lewego, oraz odejmowane od nominalnej wartości mocy dla silnika prawego. W programie został również zaimplementowany system omijania przeszkody. Ponieważ przeszkoda w jego zadaniu zawsze miała takie same wymiary, więc omijanie jej zostało wbudowane jako trajektoria zadana. Robot po wykryciu przeszkody w odpowiedniej odległości skręca i objeżdża przeszkodę równolegle

do jej krawędzi. Powrót na trasę polega na jezdzie przed siebie aż do wykrycia linii przed sobą. Gdy ją wykryje robot podąża jeszcze kawałek, tak by po obrocie o 90° linia znów była pomiędzy jego czujnikami.

3.2 Modyfikacje w algorytmie

Algorytm przy sczytywaniu wartości z czujnika prawego, czyli czujnika LEGO® NXT zwiększa odczytaną wartość o 10%, tak, by czujnik szybciej reagował, gdyż nie byliśmy zadowoleni z jego oryginalnej szybkości reakcji w zadaniu wykrywania linii. Sama prosta błędu również została zmodyfikowana dla skrajnych wielkości. Gdy wartość błędu jest większa niż 90, z dokładnością do znaku, błąd jest zwiększany o 40%. Umożliwia to szybki obrót robota w przypadku wjazdu na zakręt o kącie 90°, przy jednoczesnym braku oscylacji w trakcie jazdy prosto, lub po łagodnych zakrętach. W kodzie zostało również zaimplementowane ograniczenie na wartości podawane na silniki, gdyż te ze względu na swoje oprogramowanie zatrzymywały wykonanie programu gdy ta wartość jest większa od 1000 z dokładnością do znaku. Z tego powodu nastawy dla silników, zanim były do nich podane były sprawdzane pod względem spełnienia powyższego warunku i w przypadku, gdy go nie spełniały były zmniejszane do odpowiednio 999 lub -999 w zależności od znaku nastawy.

3.3 Dobór nastaw dla regulatora PID

Nastawy regulatora zostały dobrane w dużej mierze metodą eksperymentów, przy niewielkiej pomocy szacowania wartości wyjściowych, oraz świadomości w czym pomagają poszczególne człony regulatora. Efektem takiego dobierania są nastawy $K_p = 2$, $K_d = 8$, oraz moc maksymalna równa 250. Wartość współczynnika członu proporcjonalnego została dobrana by robot dobrze radził sobie przy pozostawaniu na prostych, oraz łagodnych zakrętach nie wpadając w oscylacje, oraz zjeżdżał czujnikiem z linii nim włączy się funkcja zwiększania błędu. Natomiast wartość współczynnika członu różniczkującego jest ustawiona tak, by robot dobrze radził sobie z kątami ostrymi i prostymi mając do dyspozycji nasze wcześniej opisane modyfikacje.

4 Podsumowanie

Przedstawione wyżej rozwiązanie ma wiele zalet. Główną zaletą jest prostota algorytmu, oraz jego zwięzłość. Takie ustawienie czujników daje duży próg błędu gdyż aby wypaść z trasy robot musi przejechać czujnikiem całą szerokość linii. Doświadczenie pokazało jednak, że nie jest to niemożliwe w przypadku gwałtownie zmieniającej się w różne strony trasy. Z drugiej strony czujniki są odpowienio blisko, by robot jadąc po linii prostej szybko ustawiał się i poruszał wektorem równoległym do trasy, dzięki czemu nie ma

problemu z odpowiednim ustawieniem się do przeszkody, co bywało problemem u naszej konkurencji. Kolejną zaletą konstrukcji, oraz algorytmu jest możliwość przejechania trasy utrzymując względnie wysoką szybkość średnią i uzyskując dobry czas przejazdu. Wadą jest to, iż robot nie radzi sobie z konkretnym ułożeniem gwałtownie zmieniającej się trasy, oraz brak możliwości ominięcia przeszkody o wymiarach większych niż przeszkoda ustawiona w trakcie zawodów.