ALGORYTM STERUJĄCY ROBOTA PORUSZAJĄCEGO SIĘ PO SCHODACH

Jakub Kokosiński, Karolina Michalak, Michał Banach, Zofia Decker

AGH Akademia Górniczo-Hutnicza im. Stanisława Staszica
Wydział Inżynierii Mechanicznej i Robotyki,
Katedra Robotyki i Mechatroniki,
Al. Mickiewicza 30, 30-059 Kraków

jakokosinski@student.agh.edu.pl, karolinam@student.agh.edu.pl,
mbanach@student.agh.edu.pl, decker@student.agh.edu.pl

Słowa kluczowe: LEGO Mindstorms NXT 2.0, mobilny robot, schemat blokowy, robotyka, mechatronika

STRESZCZENIE

Prace nad częścią informatyczną robota poruszającego się po schodach polegały na opracowaniu algorytmu. Założyliśmy podstawowe funkcjonalności naszego robota, takie jak kontrolowanie odległości od stopnia czy samodzielna kalibracja po wykryciu nieprawidłowości w pomiarach. Rozpatrzyliśmy sytuacje krytyczne, w których program mógłby zacząć pracę w sposób nieprawidłowy. Całość algorytmu została przedstawiona w postaci graficznej - schematu blokowego.

1. Wstęp

Z roku na rok mobilne roboty transportowe stają się coraz bardziej popularne [1]. Wiele zadań wykonywanych dotychczas przez człowieka zostaje przejęte właśnie przez nie [2]. Budowa takiego urządzenia jest dużym wyzwaniem dla zespołu inżynierów. Oprócz samego opracowania koncepcji należy uwzględnić jak największą liczbę sytuacji, w których nasz robot powinien sobie bezproblemowo poradzić. Jedną z takich sytuacji, a dokładniej przeszkód, jakie mobilny robot może spotkać na swojej drodze są schody [3].

Niniejszy artykuł powstał w ramach projektu, którego celem jest wykonanie oraz udokumentowanie algorytmu sterującego robotem pokonującym takie przeszkody. Trzysegmentowy robot porusza się po schodach w górę i w dół. Napotykając przeszkodę, robot przenosi przy pomocy przekładni liniowej poszczególne segmenty pokonując w ten sposób stopnie.

Kamieniem milowym projektu jest określenie funkcjonalności robota, parametrów pracy, stworzenie czytelnych nazw zmiennych oraz stałych, stworzenie dokumentacji algorytmu, a także wykonanie schematów blokowych procesów: wchodzenia, schodzenia, kalibracji oraz uwzględnienie w nich sytuacji prowadzących do błędów lub sytuacji niestabilnych.

Poniżej przedstawiono ogólne założenia dotyczące opracowanego schematu blokowego oraz omówiono sytuacje krytyczne – momenty w jakich robot może mieć problem ze znalezieniem właściwego rozwiązania.

Jako język algorytmu posłużono się schematem blokowym [4] oraz opisem słownym [5].

Na schemat blokowy zdecydowano się z następujących powodów. Jest to najpopularniejszy sposób opisu algorytmu. Dzięki graficznej reprezentacji nie sprawia problemu osobom, które nie miały wcześniej styczności z tego typu rozwiązaniami.

Opis słowny pełni rolę pomocniczej reprezentacji badanego zagadnienia. Jest on szczególnie użyteczny ze względu na zawarte w nim ilustracje obrazujące poszczególne stany, w jakich znajduje się robot mobilny.

Spis rozwiązań, na jakie się zdecydowano został przedstawiony przy pomocy poniższej tablicy morfologicznej. Wybrane pozycje zostały pogrubione oraz wyszczególnione czerwonym obramowaniem.

Tab. 1. Tablica morfologiczna sekcji informatyków.

MORPHOLOGICAL CHART - SEKCJA INFORMATYKÓW PROJEKTU ROBOTA PORUSZAJĄCEGO SIĘ PO SCHODACH						
	1.	2.	3.	4.	5.	6.
Opis algorytmu	SFC	**Schemat blokowy**	**Opis słowny**	Lista kroków		
Program do tworzenia algorytmu	Paint	MS PowerPoint	**Creately**	Visual-Paradigm	diagrams.net	
Sposób zapisu algorytmu	Jeden zbiorczy algorytm	**Podział na algorytm główny oraz algorytmy pomocnicze (funkcje)**	Wszystkie algorytmy w osobnych plikach			
Język programu	**Arduino**	C++	Python	Język blokowy mindstorm	LD	Scratch
Program do tworzenia kodu	**VisualCode**	**TinkerCAD**	ArduinoIDE	Notepad++	Notatnik	Word
Sposób opisu elementów robota	**Akronimy**	Małe litery	Duże litery	Numeracja	Symbol określający typ elementu + numer	
Sposób opisu stałych oraz zmiennych	**Według dokumentacji języka Python**	Według dokumentacji języa C++	Symbol określający typ + numer	Własny sposób opisu		
Sposób kalibracji	**Wykorzystanie czujników przednich**	Wykorzystanie czujników tylnych	Brak kalibracji	Wykorzystanie wszystkich czujników	**Tylko w pozycji bezpiecznej - robot nie dotknie stopnia podczas obrotu**	
Moment wykonania kalibracji	Przed każdym krokiem	**Przed każdym powtórzeniem cyklu**	Tylko na rozpoczęcie programu	Tylko na rozpoczęcie schodzenia oraz wchodzenia		
Sposób zakończenia wchodzenia	**Po wejściu na określoną liczbę stopni**	Ruch do wyczerpania się stopni	**Ruch do wystąpienia sytuacji krytycznej**	**Ruch do wystąpienia błędu**		
Sposób zakończenia schodzenia	**Po zejściu po określonej liczbie stopni**	Zejście po wszystkich stopniach	Ruch do wystąpienia sytuacji krytycznej	**Ruch do wystąpienia błędu**		

W kategorii *opis algorytmu* wybrano **schemat blokowy** oraz **opis słowny**. Kombinacja tych dwóch metod daje pełny obraz przedstawianego zagadnienia oraz stanowi przejrzystą instrukcję pod przyszłe prace związane z pisanie kodu źródłowego robota.

Algorytm typu SFC został odrzucony na samym początku ze względu na małą przystępność dla osób zaczynających z tym językiem.

Zdecydowano, że lista kroków również zostanie odrzucona ze względu na zbyt ogólnikowe przedstawienie rozpatrywanej problematyki.

Jako *program do tworzenia algorytmu* posłużono się narzędziem **Creately**. Prostota oraz możliwość pracy w chmurze z innymi użytkownikami korzystnie wpłynęła na odbiór tego oprogramowania.

W środowisku Creately algorytm został *podzielony na trzy schematy*. **Jeden algorytm główny** (podzielony na dwa osobne pliki) oraz **jedna funkcja pomocnicza**. Takie rozwiązanie zapewnia maksymalną przejrzystość prezentowanego zagadnienia.

Wybrana *platforma* oraz *język programu* to **Arduino**. Głównym czynnikiem, który wpłynął na taką decyzję jest możliwość symulacji układu w środowisku TinkerCAD.

Środowiska, z których zdecydowano się korzystać podczas tworzenia kodu to **VisualCode** oraz **TinkerCAD**.
Dzięki wykorzystaniu wymienionego oprogramowania zyskujemy możliwość synchronizacji plików z GitHubem oraz symulacji napisanego kodu bez budowania rzeczywistego układu.

Urządzenia wejścia/wyjścia zostały opisane za pomocą **akronimów**. Dokładny sposób zapisu został przedstawiony w *Dokumentacji technicznej algorytmu* [6] w rozdziale *Opis elementów wejścia/wyjścia*.

Zmienne oraz stałe zostały opisane według zasad zgodnych z dokumentacją języka Python oraz zawartych w książce *Czysty kod w Pythonie* [7]. Dokładny sposób zapisu został przedstawiony w *Dokumentacji technicznej algorytmu* [6] w rozdziałach *Stałe używane w algorytmie* oraz *Zmienne używane w algorytmie*.

Jako *sposób kalibracji* wybrano **wykorzystanie czujników przednich** w połączeniu z **wykonywaniem tego procesu wyłącznie w pozycji bezpiecznej**.
Dzięki takiemu rozwiązaniu mamy pewność, że robot będzie ustawiony prostopadle do następnego stopnia oraz nie spadnie ze schodów podczas wykonywania obrotu wchodzącego w proces kalibracji.

Proces kalibracji będzie wykonywany **przed każdym powtórzeniem cyklu**. Taka implementacja funkcji pomocniczej zapewnia optymalne

rozwiązanie pomiędzy brakiem kalibracji, a jej nadmiernym występowaniem, które to przekłada się na znaczne wydłużenie czasu trwania programu.

Program *kończy proces wchodzenia* po schodach, gdy robot **pokona z góry określoną ilość stopni**, **znajdzie się w sytuacji krytycznej** lub **napotka błąd**.

Wszystkie powyższe przypadki zostały opisane w *Dokumentacji technicznej algorytmu* [6] – rozdział *Ruch w górę po schodach – opis*.

Robot *kończy proces schodzenia po schodach* w **przypadku zejścia po określonej ilości stopni** lub w momencie **wystąpienia błędu**.

Wszystkie powyższe przypadki zostały opisane w *Dokumentacji technicznej algorytmu* [6] – rozdział *Ruch w dół po schodach – opis*.

2. Algorytm sterujący

Prace nad algorytmem sterującym rozpoczęto od zdefiniowania **zakresu pracy robota**. Z całego cyklu wyszczególniono trzy podstawowe operacje, tzn.:

a) proces wchodzenia,
b) proces schodzenia,
c) kalibrację.

Podpunkty a i b są traktowane jako pętla główna programu. Po skończeniu wchodzenia robot przechodzi do procesu schodzenia. Natomiast podpunkt c jest zaimplementowany jako funkcja pomocnicza (tzw. macro), która jest wywoływana ze szczególnymi parametrami przez pętlę główną programu. Powyższe założenia zostały podjęte za pomocą tablicy morfologicznej przedstawionej w rozdziale 1.

Uznano, że najlepszą formą do przedstawienia samej idei będzie połączenie **opisu słownego** wraz z graficzną reprezentacją w postaci **schematów blokowych** (patrz *Tab. 1*.).

Następnie przystąpiono do ustalenie zakresu pracy robota. Założenia:

a) wejście na określoną ilość stopni (wstępnie przyjęto tę liczbę równą 10) lub gdy jest to niemożliwe, do wystąpienia **warunków brzegowych**,
b) zejście po tej samej liczbie stopni, na jakie robot wszedł lub gdy jest to niemożliwe, do wystąpienia warunków brzegowych,

c) jeżeli na jakimś etapie wykonywania programu odczyt z czujników będzie się różnił robot przystępuje do procesu kalibracji,
d) kalibracja może odbyć się tylko i wyłącznie w **pozycji bezpiecznej**, tzn., gdy wszystkie koła mają zapewnioną przyczepność z podłożem, a możliwości jej stracenia podczas kalibracji nie istnieje,
e) robot może wchodzić tylko jedną stroną oraz schodzić tylko drugą (jest to spowodowane innym rozmieszczeniem czujników – dalmierzy),
f) jeżeli odległość do następnego stopnia jest bardzo duża (maksymalny dystans został określony za pomocą stałej) robot nie przystępuje do wejścia,
g) jeżeli następny stopień okazał się za wysoki robot nie przystępuje do wejścia – przechodzi do procesu schodzenia,
h) jeżeli to tylko możliwe wykorzystywać czujniki, w przypadku, gdy nie ma takiej możliwości należy posłużyć się stałymi, które zostaną wyznaczone dla zbudowanego fizycznie robota.

Pod pojęciem warunek brzegowy rozumiane poniższe sytuacje:

a) odległość do następnego stopnia jest za duża,
b) następny stopień jest za wysoki.

Dokładniejszy opis tego zagadnienia zamieszczono w *Dokumentacji technicznej algorytmu* [6] – rozdziały *Ruch w górę po schodach – opis: 3. Podnoszenie pierwszego segmentu oraz 8. Sprawdzenie odległości*.

Opis słowny algorytmu

Na samym początku prac przystąpiono do opracowania opisu słownego [6]. Przedstawienie algorytmu w postaci tekstu ma tą zaletę, że jest podatne na wszelkie modyfikacje. Tym samym, sposób ten był najbardziej odpowiedni do rozpoczęcia pracy.

Opis słowny został podzielony na następujące fragmenty:

a) **Opis elementów wejścia/wyjścia** – rozmieszczenie wykorzystywanych elementów na przykładzie rysunku poglądowego oraz sposób nazewnictwa,
b) **Stałe używane w algorytmie** – spis wszystkich stałych użytych w programie; składa się z listy zbiorczej oraz opisu każdej stałej z osobna,
c) **Zmienne używane w algorytmie** - spis wszystkich zmiennych użytych w programie; składa się z listy zbiorczej oraz opisu każdej zmiennej z osobna,

d) **Kalibracja** – opis słowny procesu kalibracji,
e) **Proces** wchodzenia - opis słowny procesu wchodzenia,
f) **Proces schodzenia** – opis słowny procesu schodzenia.

Opis sensorów oraz silników ma postać trzyliterowych akronimów nadawanych według zasad przedstawionych na rys. 1.

Rys. 1. Sposób tworzenia nazw urządzeń wejścia i wyjścia.

S/M	**F/B LUB W/T**	**R/L LUB R/L/F/B**

Dla pierwszego segmentu:
- S – sensor – czujnik ultradźwiękowu,
- M – motor – silnik elektryczny.

Dla drugiego segmentu:
- W przypadku S na pierwszej pozycji:
  - F – front – czujnik umiejscowiony z przodu,
  - B – B – czujnik umiejscowiony z przodu,
- W przypadku M na pierwszej pozycji:
  - W – wheels – silnik napędza koła,
  - T – transmission – silnik napędza przekładnię liniową odpowiedzialną za zmianę wysokości segmentów.

Dla trzeciego segmentu:
- W przypadku S na pierwszej pozycji:
  - R – right – czujnik umiejscowiony po prawej stronie,
  - L – left – czujnik umiejscowiony po lewej stronie,
- W przypadku M na pierwszej pozycji:
  - R – right – silnik napędza koła umiejscowione po prawej stronie,
  - L – left – silnik napędza koła umiejscowione po lewej stronie,
  - F – front – silnik napędza przekładnie lub koła z przodu,
  - B – back – silnik napędza przekładnie lub koła z tyłu.

Np.: MWF to M – motor, W – wheels, F – front; silnik napędzający koła znajdujące się w przednim segmencie.

Podczas tworzenia właściwej części opisu słownego, czyli procesów kalibracji oraz poruszania się po schodach na bieżąco tworzono listę potrzebnych stałych oraz zmiennych. Dla lepszej przejrzystości są one identyczne w przypadku opisu, algorytmów, jak i przypszłego progamu.

Tak jak wspomniano we wstępie, sposób nazewnictwa zapożyczono z języka Python, a dokładniej z książki *Czysty kod w Pythonie* [10].

Według tej zasady: TUTAJ_BEDZIE_PRZECHOWYWANA_STAŁA oraz tutaj_bedzie_przechowywana_zmienna.

Spis oraz szczegółowy opis stałych i zmiennych znajduje się w *Dokumentacji technicznej algorytmu* [6].

Kalibracja [8]

Proces kalibracji polega na podniesieniu segmentu pierwszego oraz trzeciego (robot styka się z podłożem tylko za pomocą kół segmentu środkowego), a następnie sprawdza, czy odczyt z czujników SFR i SFL (przednie czujniki ultradźwiękowe, patrz więcej: [6]) są sobie równe.

Jeżeli ten warunek nie jest spełniony oznacza to, że robot nie jest ustawiony prostopadle do płaszczyzny stopnia. W tym wypadku należy dokonać obrotu wokół osi robota:

a) jeżeli odczyt SFR > SFL robot musi wykonać obrót w lewą stronę,
b) jeżeli odczyt SFR < SFL robot musi wykonać obrót w prawą stronę.

Praca silników MWR oraz MWL jest podtrzymywana do zrównania się odczytów z czujników SFR i SFL.

Następnie robot opuszcza segment pierwszy oraz trzeci. Kalibracja została zakończona, a robot może przystąpić do kontynuowania pracy w pętli głównej programu.

Istnieje również możliwość, że odczyt czujników SFR oraz SFL jest taki sam od początku. W tym wypadku należy rozważyć dwie sytuacje:

a) robot ustawiony w prawidłowej pozycji do dalszej pracy,
b) robot ustawiony przodem do narożnika – jeden czujnik mierzy dystans do jednej ściany, a drugi do drugiej.

Powyższe zagadnienie (*Przypadek 3*), jak i cały proces kalibracji został przedstawiony w rozdziale *Kalibracja – opis Dokumentacji technicznej algorytmu* [6].

Implemetnacja kalibracji ma postać wywołania po przez algorytm główny. W zależności od momentu, funkcja pomocniczna (jaką jest omawiany proces)

zostaje uruchomiona z odpowiednimi parametrami, które przekładają się na uwzględnienie tylko wymaganych przypadków z ciała funkcji.

Wchodzenie po schodach [9]

Ruch w górę po schodach został podzielony na kilka etapów (patrz więcej: [6]):

1. Kalibracja,
2. Podjazd do stopnia,
3. Podnoszenie pierwszego segmentu,
4. Wjazd pierwszego segmentu na stopień,
5. Podnoszenie drugiego segmentu,
6. Wjazd drugiego segmentu na stopień,
7. Podnoszenie trzeciego segmentu,
8. Sprawdzenie odległości,
9. Wjazd trzeciego segmentu na stopień.

Po samych nazwach możemy dojść do wniosku, że proces ten jest bardzo powtarzalny i łatwo dostrzec regułę.

Na samym początku przystępujemy do skalibrowania pozycji robota. Jeżeli program wykonuje pierwszą iteracją (stopień numer 0 – podłoże wyjściowe) uwzględnione zostają wszystkie 3 przypadki (patrz Kalibracja). Podczas kolejnych powtórzeń programu sprawdzany jest tylko warunek z punktu 2.

Następnie robot przystępuje do właściwej części pracy, która polega na przetransportowaniu każdego z segmentów na następny stopień.

Cały proces zostaje zakończony, gdy liczba pokonanych stopni wyniesie 10 lub gdy wystąpi jeden z warunków brzegowych:

a) Podczas podnoszenia segmentu pierwszego okaże się, że następny stopień jest za wysoki,
b) Po całkowitym wejściu na stopień okaże się, że odległość do następnego schodka jest bardzo duża, tzn. odczyt z czujników SFR oraz SFL jest większy niż wartość zapisana w stałej LENGTH_OF_INFINITY.

Po spełnieni jednego z powyższych warunków program przechodzi do procesu schodzenia.

Schodzenie po schodach [10]

Ruch w dół po schodach został podzielony na kilka etapów (patrz więcej: [6]):

1. Podjazd do krawędzi stopnia,

    a. Odjazd w bezpieczną pozycję,
    b. Kalibracja,
    c. Podjazd do krawędzi,
2. Opuszczanie trzeciego segmentu,
3. Zjazd drugiego segmentu ze stopnia,
4. Opuszczanie drugiego segmentu,
5. Zjazd pierwszego segmentu ze stopnia,
6. Opuszczanie pierwszego segmentu,
7. Zakończenie programu.

W zależności od sposobu zakończenia części programu związanej z wejściem po schodach krok 1. może zostać wykoanany lub pominięty.

Jeżeli robot nie znajduje się przy krawędzi stopnia musi do niego podjechać. Podczas tej operacji ma miejsce wystąpienie funkcji kalibracji.

Następnie robot zjeżdża na stopień niżej. Kroki od 1. do 6. są powtarzane do momentu zejścia po wszystkich stopniach, na które wcześniej udało mu się wejść (maksymalnie 10).

Podczas kolejnych iteracji tej części programu krok 1. jest zawsze wykonywany.

Na koniec robot przechodzi do kroku 7. – odjeżdża na bezpieczną odległość od stopnia i kończy pracę.

Opis schematów

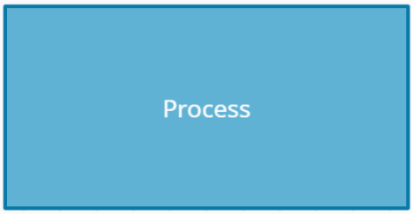
Podstawowe rodzaje bloków [11]:

strzałka (łącznik) – wskazuje kierunek przepływu danych lub kolejność

wykonywania.

Rys. 2. Wygląd bloku ,,strzałka" w schemacie blokowym.

operator – prostokąt, do którego wpisywane są wszystkie operacje z



wyjątkiem instrukcji wyboru.

Rys. 3. Wygląd bloku ,,operator" w schemacie blokowym.

predykat – romb, do którego wpisywane są wyłącznie instrukcje wyboru.

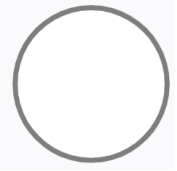
Rys. 4. Wygląd bloku ,,predykat" w schemacie blokowym.

etykieta (blok graniczny) – owal służący do oznaczania początku albo końca

sekwencji schematu (kończy, zaczyna albo przerywa lub przenosi schemat).

Rys. 5. Wygląd bloku „etykieta" w schemacie blokowym.

łącznik wewnętrzny – służy do łączenia odrębnych części schematu znajdujących



się na tej samej stronie.

Rys. 6. Wygląd bloku „łącznik wewnętrzny" w schemacie blokowym.

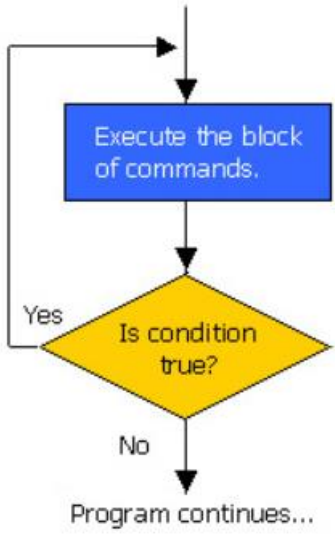
łącznik zewnętrzny – służy do łączenia odrębnych części schematu znajdujących

się na różnych stronach.

Rys. 7. Wygląd bloku „łącznik zewnętrzny" w schemacie blokowym.

Pętla do-While w schematach blokowych:

Do-While-Blok poleceń, który jest wykonywany przynajmniej raz. i jest wykonywany ponownie tak długo, jak warunek jest spełniony [12].



Rys. 8. Wygląd bloku pętli „do-while" w schemacie blokowym.[13]

a) proces wchodzenia

Przed rozpoczęciem wchodzenia po schodach robot pobiera ustalone stałe. Następnie robot przeprowadza proces kalibracji, aby poprawnie ustawić się do procesu wchodzenia (patrz więcej: [6]).Robot rozpoczyna 1) część wchodzenia od sprawdzenia odległości czujnikami SFR i SFL. Jeżeli odległość zmierzona przez czujniki SFR i SFL jest większa niż odległość (SAFE_DITSANCE_IN_FRONT_OF_STEP) (patrz więcej: [6]) robot podjeżdża do stopnia. Silniki MWF, MWB, MWR i MWL kręcą się do przodu. Jeżeli odległość zmierzona przez czujniki SFR i SFL jest równa (SAFE_DITSANCE_IN_FRONT_OF_STEP) (patrz więcej: [6]) to silniki MWF, MWB, MWR i MWL rozpoczynają obracanie się w przód do momentu wykonania (NUMBER_OF_HORIZONTAL_TURNS_FOR_APPROACH) (patrz więcej: [6]). W przypadku gdy odległość zmierzona przez czujniki SFR i SFL jest mniejsza od (SAFE_DITSANCE_IN_FRONT_OF_STEP) (patrz

więcej: [6]) program kończy pracę. W następnym etapie 2) wchodzenia robot sprawdza czy odległości czujek SFR i SFL są równe, jeżeli nie są następuje kalibrowanie aż do ich zrównania. Następnie silnik MTF porusza segment pierwszy do góry. Jeżeli czujniki SFR i SFL nie wykryją żądanej odległości (tzn. odczyt będzie mniejszy niż (LENGTH_OF_A_ROBOT + SAFE_LENGTH) (patrz więcej: [6]) przed wykonaniem przez silnik MTF liczby obrotów równej (MAX_OF_VERTICAL_TURNS - NUMBER_OF_VERTICAL_TURNS) (patrz więcej: [6]) silnik MTF zaczyna się kręcić w drugą stronę do momentu całkowitego opuszczenia pierwszego segmentu. Jest to przypadek, w którym występuje warunek krawędziowy i schodek jest za wysoki dla robota, dlatego przechodzi on w tedy do procesu schodzenia. Gdy czujniki SFR i SFL zmierzą odległość równą lub większą od (LENGTH_OF_A_ROBOT + SAFE_LENGTH) (patrz więcej: [6]). Silnik wykonuje dodatkowo odmierzoną ilość obrotów zapisaną w zmiennej (NUMBER_OF_VERTICAL_TURNS) (patrz więcej: [6]). ilość wykonanych obrotów zostaje zapisana w zmiennej (number_of_turns_for_step_height) która uwzględnia dodatkowe obroty w liczbie (NUMBER_OF_VERTICAL_TURNS) (patrz więcej: [6]). Kolejny etap 3) w którym Czujniki (podniesione) SFR i SFL mierzą odległość do następnego schodka. Teraz w etapie 3a) Jeżeli odległość jest zbyty duża(niemierzalną) oznacza to, że to ostatni schodek do pokonania, więc silniki MWF, MWR, MWL i MWB kręcą się w przód, gdy wykonają (NUMBER_OF_HORIZONTAL_TURNS_FOR_FIRST_MODULE) (patrz więcej: [6]) obrotów. Dalej silniki MTF oraz MTB wykonują (number_of_turns_for_step_height) (patrz więcej: [6]) obrotów podnosząc drugi segment a następnie silniki MWF, MWR, MWL i MWB kręcą się w przód wykonując (NUMBER_OF_HORIZONTAL_TURNS_FOR_SECOND_MODULE) (patrz więcej: [6]), po czym Silnik MTB wykonuje (number_of_turns_for_step_height) obrotów podnosząc trzeci segment. W ten sposób proces wchodzenia się zakończył i rozpoczyna się proces schodzenia (z 3 segmentem wiszącym nad poprzednim schodkiem. Teraz w etapie 3b) Jeżeli odległość jest mierzalna (mamy przed sobą kolejny schodek) robot najpierw sprawdza czy SFR i SFL są sobie równe, jeżeli nie są następuje kalibracja aż do ich zrównania. Dalej silniki MWF, MWR, MWL i MWB kręcą się w przód aż do chwili, gdy odczyt z czujników wyniesie (length_of_next_step – LENGTH_OF_FIRST_MODULE) (patrz więcej: [6]) a następnie silniki MTF oraz MTB wykonują (number_of_turns_for_step_height) podnosząc drugi segment. Idąc do etapu 4b),

Robot zaczyna podjeżdżać, silniki MWF, MWR, MWL i MWB kręcą się w przód do momentu, gdy odczyt z czujników SFR i SFL wyniesie (length_of_next_step – LENGTH_OF_FIRST_MODULE - LENGTH_OF_SECOND_MODULE) (patrz więcej: [6]) a następnie MTB wykonuje (number_of_turns_for_step_height) (patrz więcej: [6]) obrotów podnosząc trzeci segment. W tej pozycji robot zapisuje zmierzoną odległość z czujników SFR i SFL do tablicy tab_length_to_edge[i], gdzie i to numer stopnia. Na koniec, robot zaczyna podjeżdżać, silniki MWF, MWR, MWL i MWB kręcą się w przód do momentu, gdy odczyt z czujników SFR i SFL wyniesie (length_of_next_step – LENGTH_OF_A_ROBOT) (patrz więcej: [6]). Proces ten znajduje się w pętli, w której za każdym razem, gdy pokonamy schodek zostaje zwiększona stała (number_of_steps) gdy będzie ona wynosić 10 proces wchodzenia się zakończy a zacznie proces schodzenia.

b) proces schodzenia

Proces schodzenia zaczyna się od jednego z dwóch przypadków:

1) Tylna część robota nie wjechała na ostatni schodek i znajduje się nad poprzednim schodkiem.
2) Cały robot znajduje się na schodku i musi cofnąć się na bezpieczną odległość do krawędzi.

Jeżeli wystąpił przypadek 1) robot może przystąpić do kolejnego etapu schodzenia 3) ale jeżeli wystąpił przypadek 2) robot najpierw odjeżdża na bezpieczna pozycję wykonując (NUMBER_OF_HORIZONTAL_TURNS_APPOACH), następnie robot przystępuje do kalibracji po czym sprawdza czy odległości SFR i SFL są sobie równe, jeżeli są robot przystępuje do dalszej części schodzenia, a jeżeli nie są robot wyłącza silniki na 5 sekund po czym ponownie sprawdza odległość czy SFR i SFL, jeżeli są równe robot przystępuje do dalszej części schodzenia w przeciwnym wypadku występuje błąd. Po sprawdzeniu zgodności odległości SFR i SFL robot podjeżdża do krawędzi cofając się do tyłu do momentu, w którym czujniki SFR i SFL zmierzą odległość (tab_length_to_edge[i]) dla odpowiadającego schodka. W 3) etapie, robot ponownie sprawdza równości odległości SFR i SFL (tak jak za pierwszym razem), następnie silnik MTB opuszczają tylną część do momentu, gdy czujniki SBR i SBL zmierzą odległość równą (BACK_CLEARANCE) liczba obrotów zostaje zapisana w zmiennej (number_of_turns_for_step_height), jeżeli SFR i SFL nie są równe program kończy pracę. Następnie podczas etapu 4) sprawdzamy czy SFR i SFL mają równe odległości (tak jak za pierwszym razem) jeżeli mają MWF, MWR, MWL i MWB kręcą się w tył do momentu, w którym SFR i SFL zmierzą odległość równą (LENGTH_OF_SECOND_MODULE +

tab_length_to_edge[i]) w przeciwnym razie, gdy odległości SFR i SFL nie są równe program kończy pracę. W kolejnym etapie 5) Silniki MTF oraz MTB wykonują number_of_turns_for_step_height obrotów opuszczając drugi segment. Następnie w etapie 6) najpierw sprawdzamy czy odległości czujek SFR i SFL są równe, jeżeli są Silniki MWF, MWR, MWL i MWB kręcą się w tył do momentu, w którym czujniki SFR i SFL zmierzą odległość równą (LENGTH_OF_FIRST_MODULE+LENGTH_OF_SECOND_MODULE+tab _length_to_edge[i]) jeżeli nie są program kończy pracę. W dalszym etapie 7) Silnik MTF wykonuje (number_of_turns_for_step_height) opuszczając pierwszy segment. Etapy (2-7) są zapętlone i będą się powtarzać aż robot pokona „i" stopni i znajdzie się na etapie początkowym. Na sam koniec w etapie 8) robot odjeżdża od stopnia na bezpieczną odległość, Silniki MWF, MWR, MWL i MWB wykonują NUMBER_OF_HORIZONTAL_TURNS_FOR_APPROACH po czym następuje koniec programu.

c)proces kalibracji

Proces kalibracji składa się z trzech równoległych drzewek, które zawierają ciąg czynności, które trzeba wykonać, aby robot mógł poprawnie wejść i zejść ze schodów. Program decyduje który przypadek należy wykonać w danym momencie na podstawie numeru schodka, na którym się znajduje. 1) przypadek sprawdzany jest na początku wchodzenia czujniki SFR i SFL mierzą odległość do pierwszego schodka, jeżeli jest ona równa LENGTH_OF_INFINITY lub czujniki pokazują błąd program kończy działanie. 2) przypadek może występować podczas kalibracji na każdym schodku poza etapem początkowym. Czujniki SFR i SFL mierzą odległość, jeżeli odległości te są od siebie różne następuje kalibracja, jeżeli są sobie równe następuje koniec kalibracji. W przypadku gdy odległości SFR i SFL są od siebie różne silniki MTF i MTB wykonują NUMBER_OF_VERTICAL_TURNS_FOR_CLEARANCE obrotów. Segmenty pierwszy I trzeci zostają lekko podniesione. Następnie, jeżeli odległość SFL jest mniejsza od odległości SFR to robot musi się obrócić w lewą stronę. Silnik MWL kręci się w tył, a MWR w przód do momentu, w którym odległości zmierzone przez czujniki SFR i SFL będą sobie równe. Jeżeli odległość SFL jest większa od odległości SFR robot musi się obrócić w prawą stronę. Silnik MWL kręci się w przód, a MWR w tył do momentu, gdy odległości zmierzone przez czujniki SFR i SFL będą sobie równe. Po dokonanej kalibracji silniki MTF i MTB opuszczają segmenty pierwszy i trzeci. 3) warunek również jest sprawdzany na początku wchodzenia,

sprawdza on czy robot ustawiony jest w prawidłowej pozycji lub jest zwrócony w stronę narożnika. Aby wykluczyć drugą opcje robot wykonuje NUMBER_OF_VERTICAL_TURNS_FOR_CLEARANCE podnosząc w ten sposób Segment pierwszy I trzeci. Następnie MWL kręci się do tyłu, MWR do przodu i
Jeżeli odczyt czujnika SFR się zwiększa jesteśmy pewni, że robot jest zwrócony w stronę narożnika. W takim wypadku robot kontynuuje obrót w lewo do momentu, gdy z odczyty z czujnika SFR, jak i SFL będą maleć. Jeżeli odczyt czujnika SFR się zmniejsza to robot zwrócony jest w dobrą stronę. Dla obu przypadków obrót jest podtrzymywany do momentu zrównania się odczytanych wartości czujników SFR i SFL. Na koniec segmenty pierwszy i trzeci zostają opuszczone przy użyciu silników MTF, MTB i następuje koniec kalibracji.

3. **Wnioski i uwagi**

Napisany przez nas algorytm w postaci schematu blokowego opisuje ruch 3-segmentowego robota po schodach. Zdecydowaliśmy się na przedstawienie algorytmu w sposób graficzny ze względu na jego przejrzystość. Podczas prac korzystaliśmy ze strony creately.com, która pozwoliła nam na tworzenie jednolitych schematów przez wszystkich członków sekcji.

Stworzony program podzielony jest na 2 równoważne algorytmy dla ruchu robota w górę i w dół schodów oraz algorytm podrzędny, którym jest kalibracja. Podczas poruszania się robot korzysta z czujników, których pomiary definiują ruch jaki robot powinien wykonać. Już podczas początkowych prac zwróciliśmy uwagę, że istotny wpływ na algorytm ma prawidłowe rozmieszczenie poszczególnych sensorów. Największa trudność podczas analizy programu wystąpiła przy wychwytywaniu potencjalnych błędów. Ze względu na brak możliwości przetestowania algorytmu na rzeczywistym modelu robota nie mogliśmy sprawdzić i przygotować algorytmu pod niektóre potencjalne błędy.

Jako przyszły rozwój algorytmu zakładamy kolejne funkcjonalności robota. Jedną z nich jest np. funkcja pokonywania półpięter. Rozbudowa polegałaby na zamianie rozmieszczenia czujników, tak aby pole widzenia było równoległe do podłoża. W ten sposób robot mógłby oszacować, która odległość jest mniejsza. Jest to o tyle wartościowa informacja, że po obróceniu się robota o 90 stopni na szczycie schodów może on podjąć decyzję, w którą stronę kontynuować proces wchodzenia, np. do bliższej ściany. Kolejna modyfikacja algorytmu jest uzależniona od znajomości dokładnego rozkładu

masy robota. Moglibyśmy wtedy wprowadzić takie zmiany w algorytmie sterującym, aby robot mógł poruszać się po węższych stopniach. Jeżeli segment drugi będzie zdecydowanie cięższy od segmentu pierwszego i drugiego, możemy zmniejszyć wymaganą długość stopnia z LENGTH_OF_A_ROBOT do (LENGTH_OF_FIRST_MODULE + LENGTH_OF_SECOND_MODULE). W takiej sytuacji segment trzeci będzie mógł znajdować się w powietrzu podczas unoszenia segmentu pierwszego na wysokość następnego stopnia.

4. **Bibliografia**

[1] forbes.com; https://www.forbes.com/sites/stevebanker/2019/03/11/the-autonomous-mobile-robot-market-is-taking-off-like-a-rocket-ship/?sh=3a9640b01603; [dostęp: 7.05.2021]

[2] forbes.com; https://www.forbes.com/sites/jackkelly/2020/10/27/us-lost-over-60-million-jobs-now-robots-tech-and-artificial-intelligence-will-take-millions-more/?sh=a7b0eaf1a525; [dostęp: 7.05.2021]

[3] researchgate.net; https://www.researchgate.net/publication/221786654_Climbing_Robots; [dostęp: 7.05.2021]

[4] wikipedia.org; https://en.wikipedia.org/wiki/Flowchart; [dostęp: 7.05.2021]

[5] zsestaszow.eu; http://www.gf.zsestaszow.eu/psiob/algorytmy.pdf; [dostęp: 7.05.2021]

[6] Dokumentacja techniczna algorytmu sterującego; opis_v6.pdf;

[8] Schemat blokowy kalibracji; https://app.creately.com/diagram/ItfrWicNRUe/edit; [dostęp: 7.05.2021]

[9] Schemat blokowy procesu wchodzenia; https://app.creately.com/diagram/x5HJ8CcuiHh/edit; [dostęp: 7.05.2021]

[10] Schemat blokowy procesu schodzenia; https://app.creately.com/diagram/lcW6Geumjpi/edit; [dostęp: 7.05.2021]

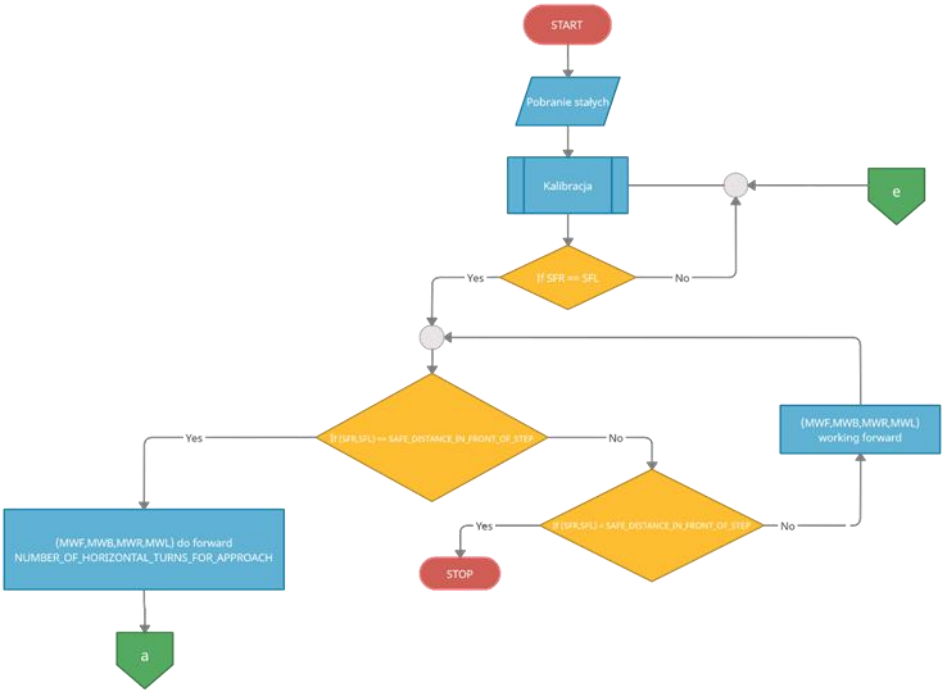
[7] Sunil Kapil; Czysty kod w Pythonie; Helion; 2020, s. 14-16.

[11] wikipedia.org; https://pl.wikipedia.org/wiki/Schemat_blokowy; [dostęp: 7.05.2021]

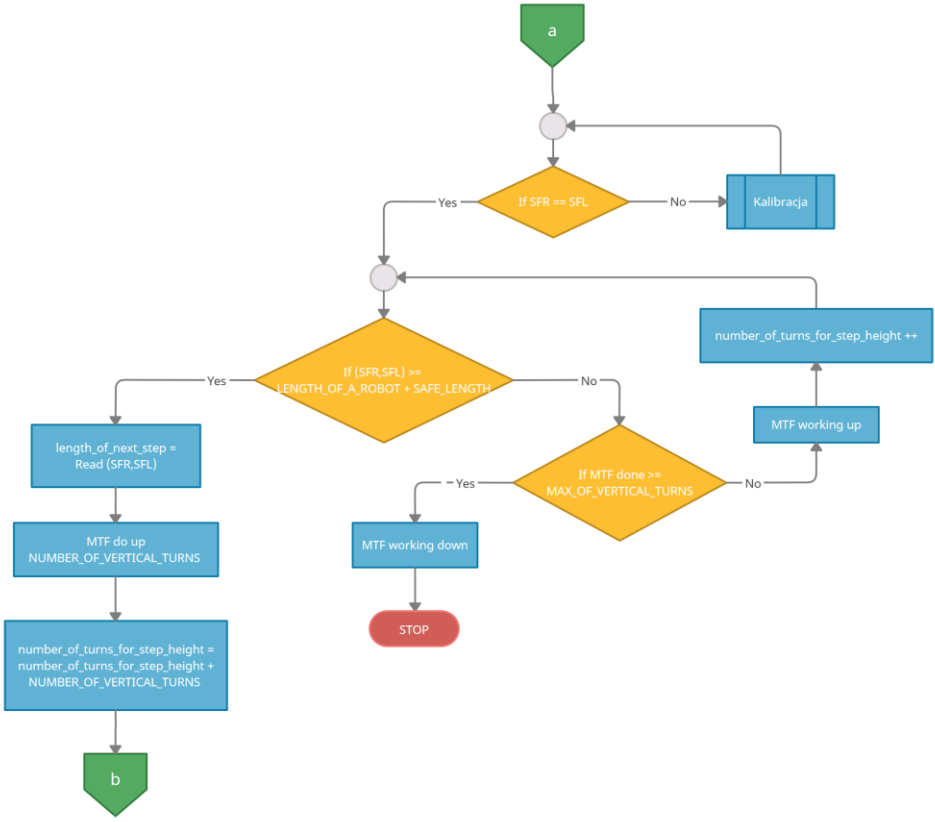
[12] developer.mozilla.org; https://developer.mozilla.org/pl/docs/Web/JavaScript/Reference/Statements/do...while; [dostęp: 7.05.2021]

[13] euler.vcsu.edu; http;//euler.vcsu.edu:7000/13693/; [dostęp: 7.05.2021]

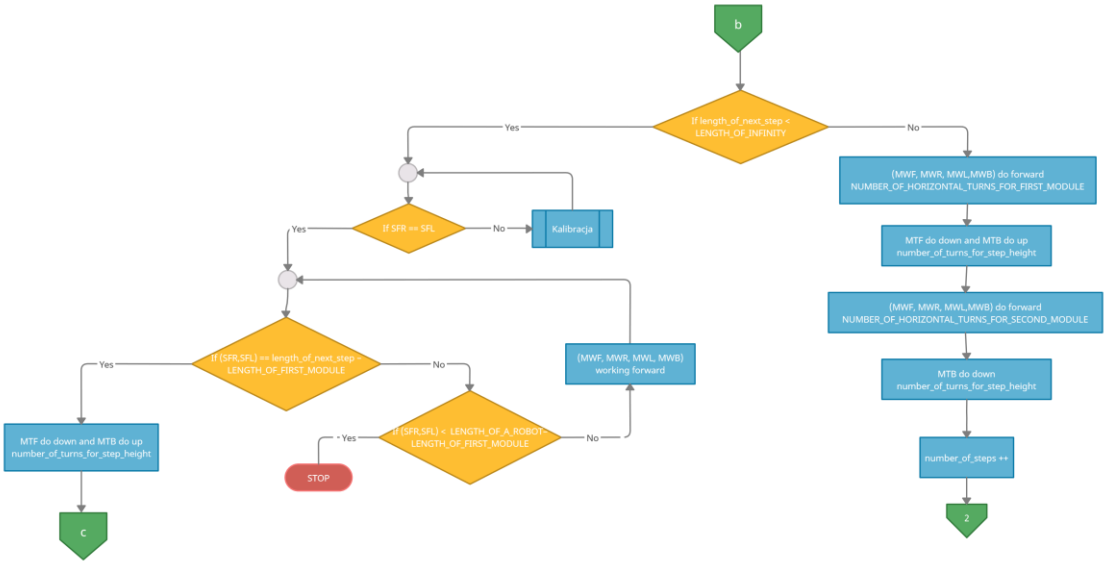
5. Załączniki:



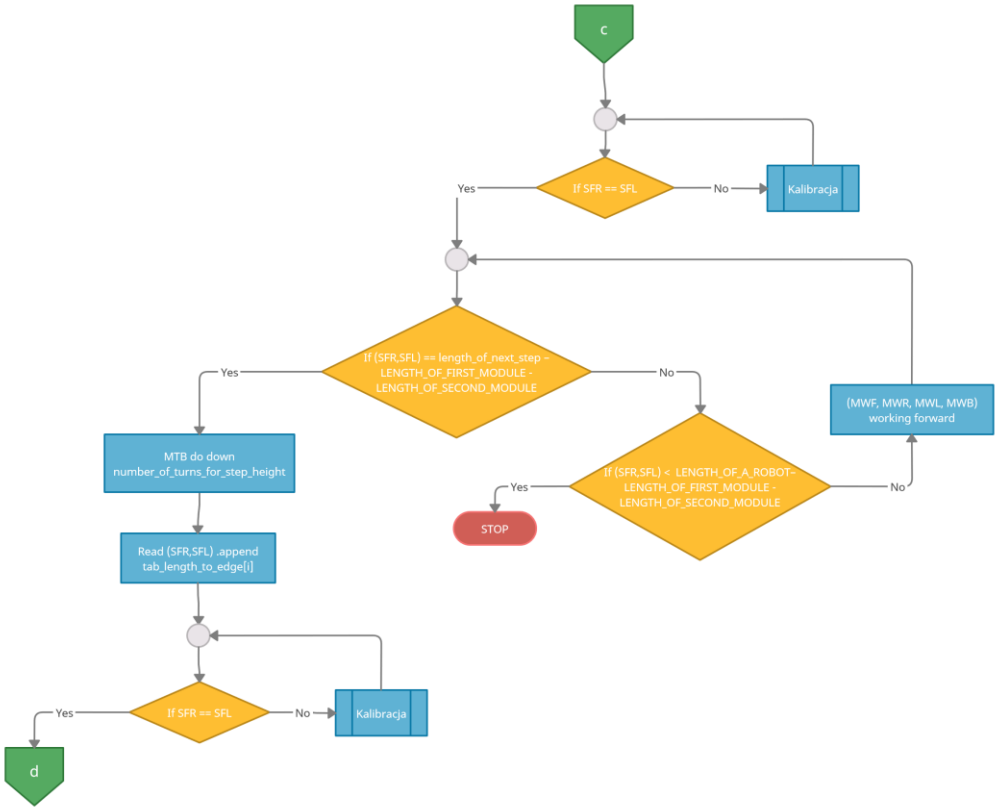
Rys. 8. Schemat blokowy procesu wchodzenia (część 1)



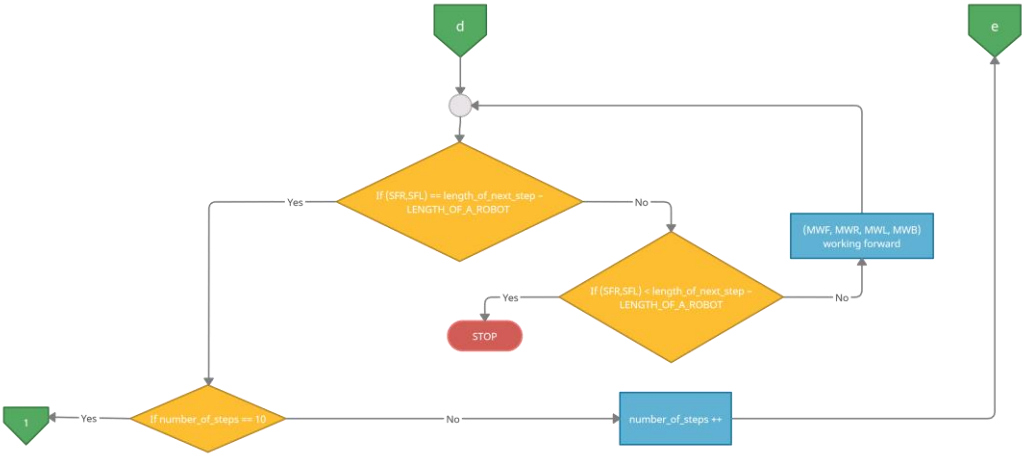
Rys. 9. Schemat blokowy procesu wchodzenia (część 2)



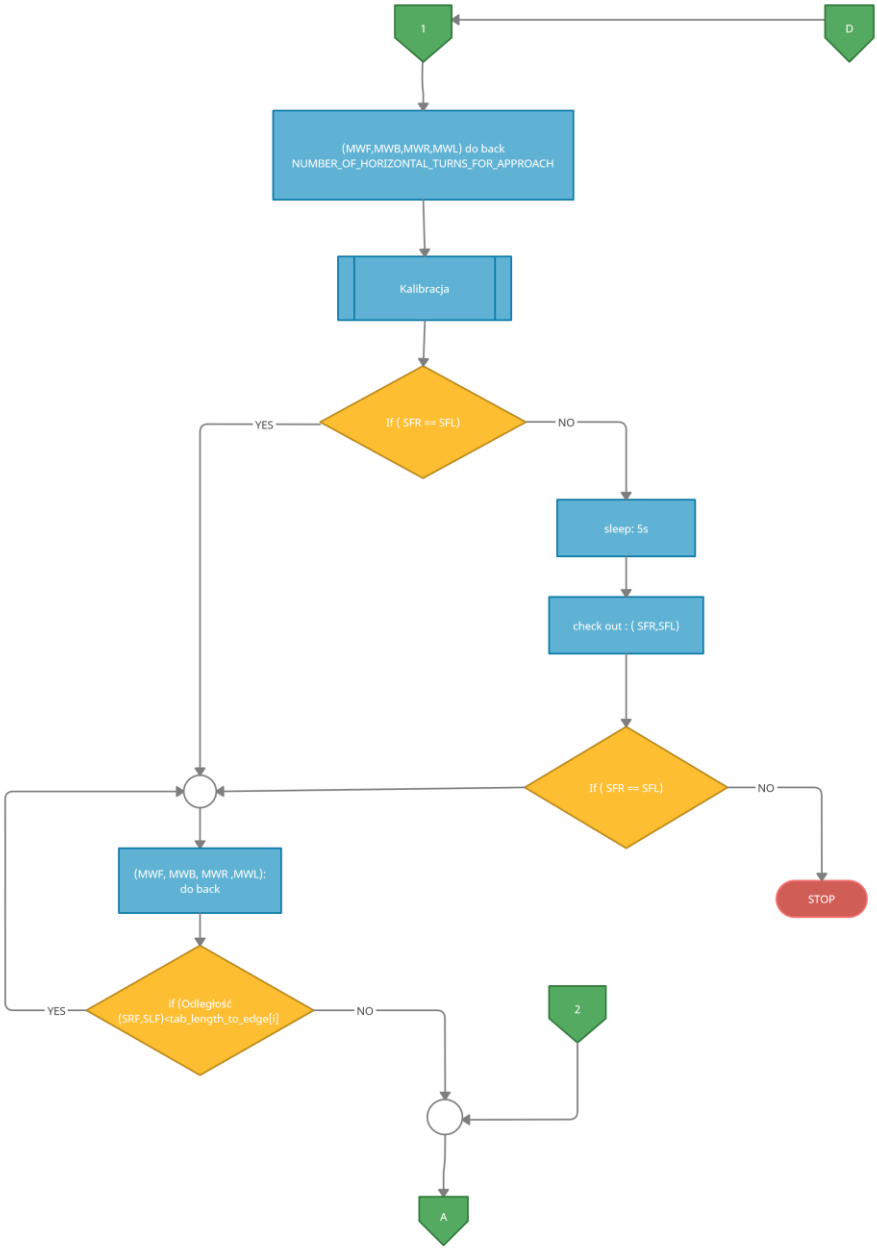
Rys. 10. Schemat blokowy procesu wchodzenia (część 3)



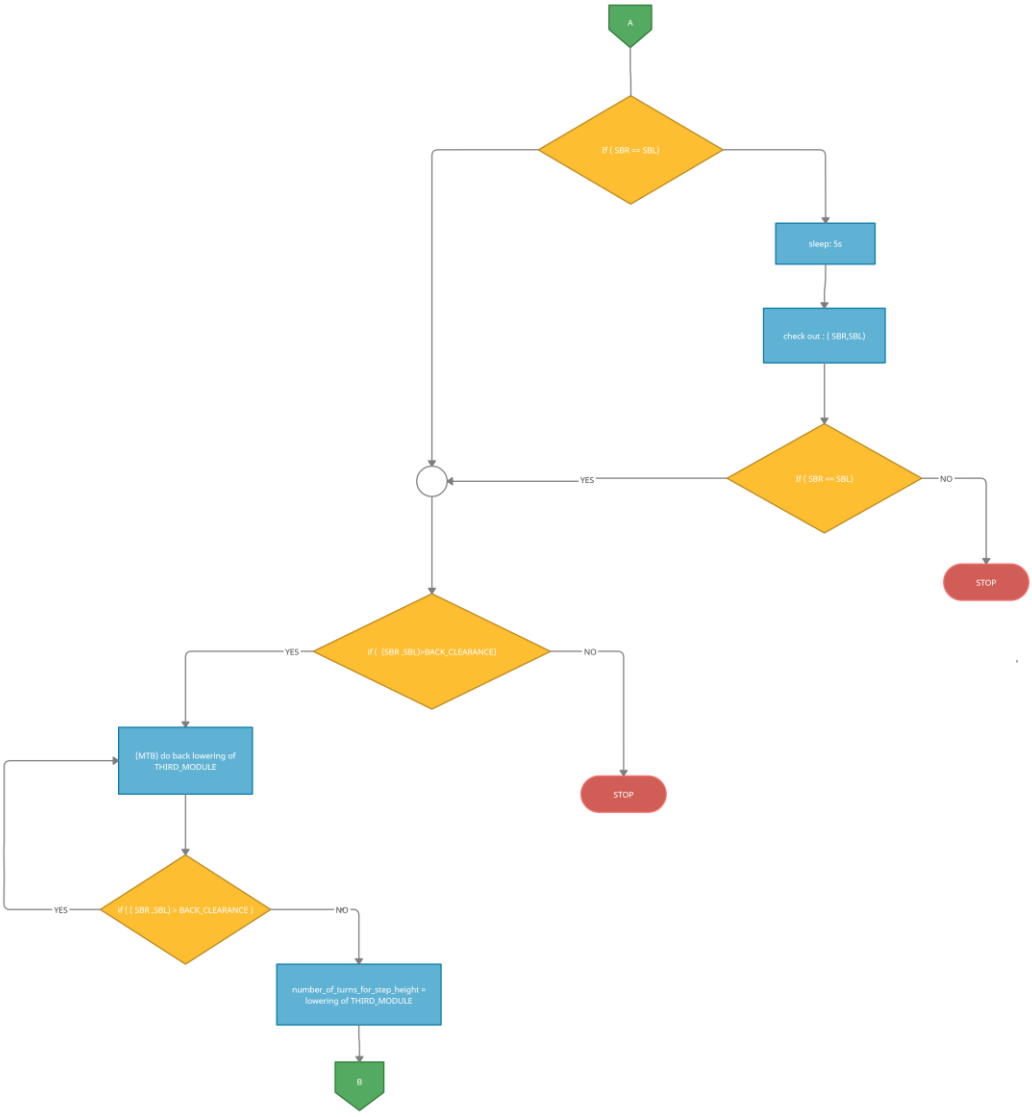
Rys. 11. Schemat blokowy procesu wchodzenia (część 4)



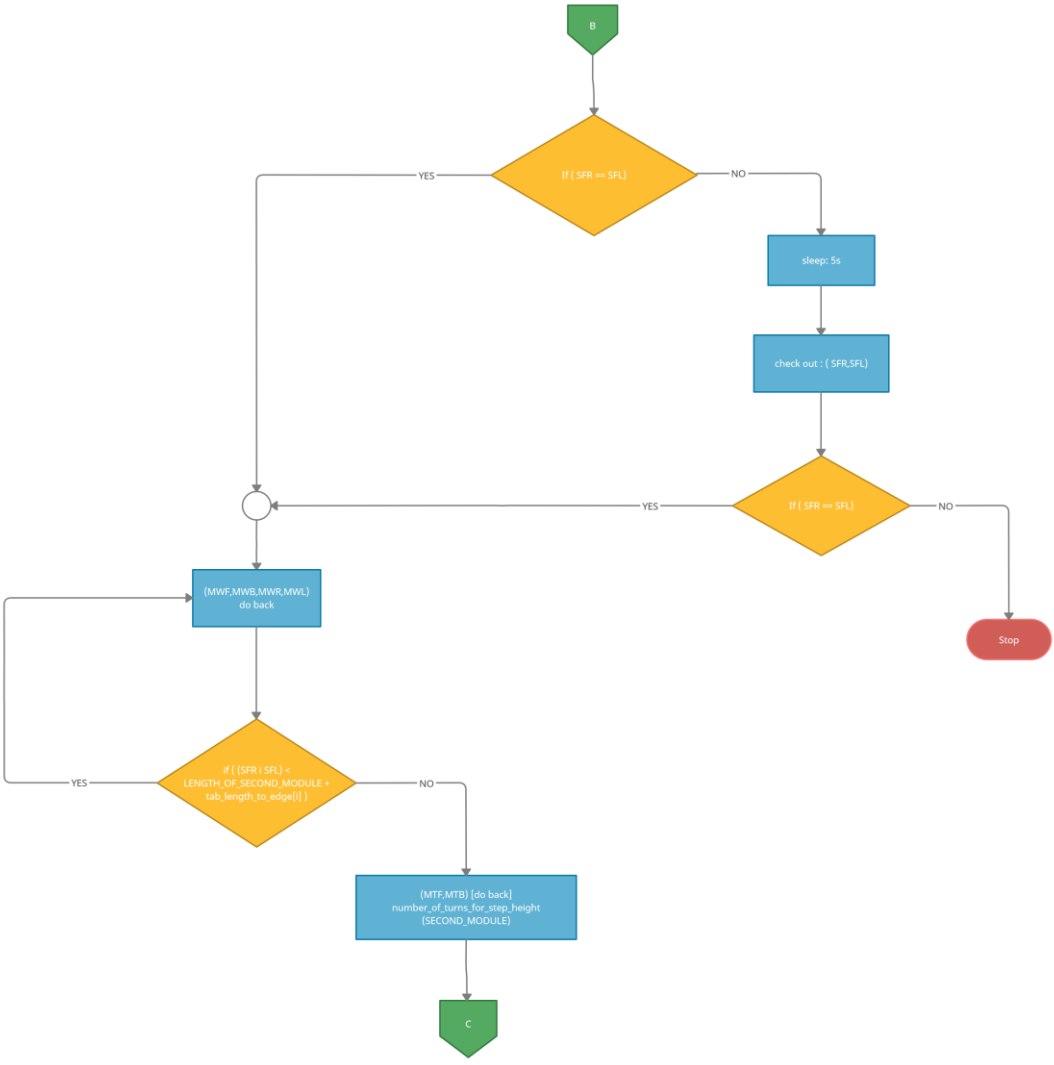
Rys. 12. Schemat blokowy procesu wchodzenia (część 5)



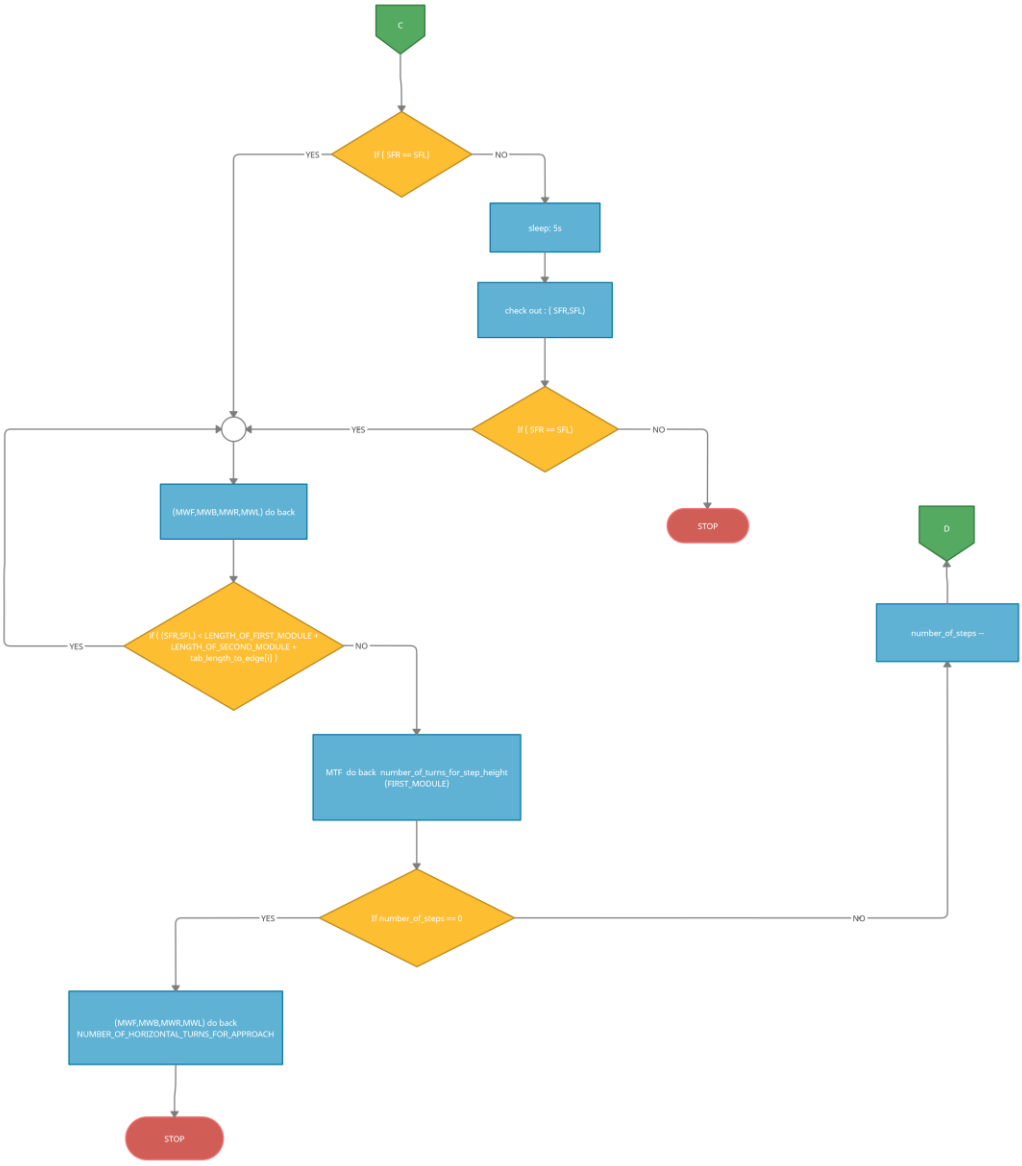
Rys. 13. Schemat blokowy procesu schodzenia (część 1)



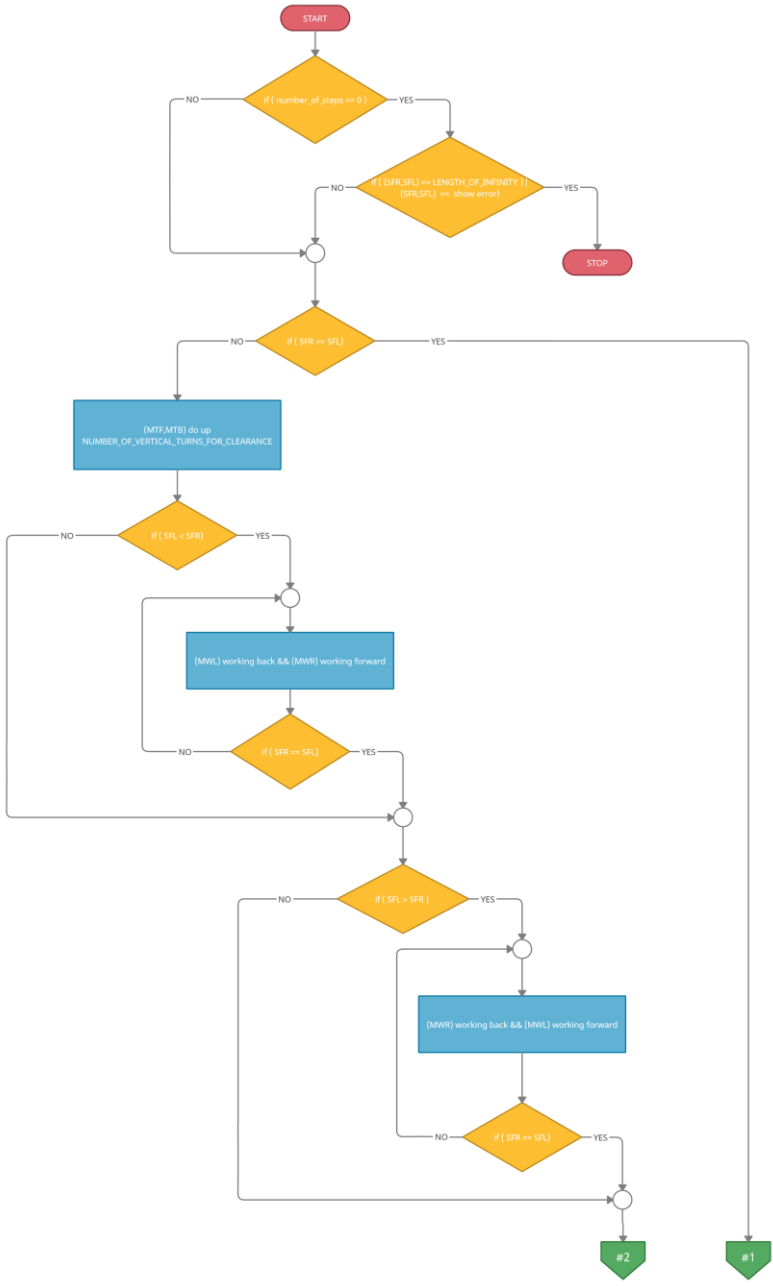
Rys. 14. Schemat blokowy procesu schodzenia (część 2)



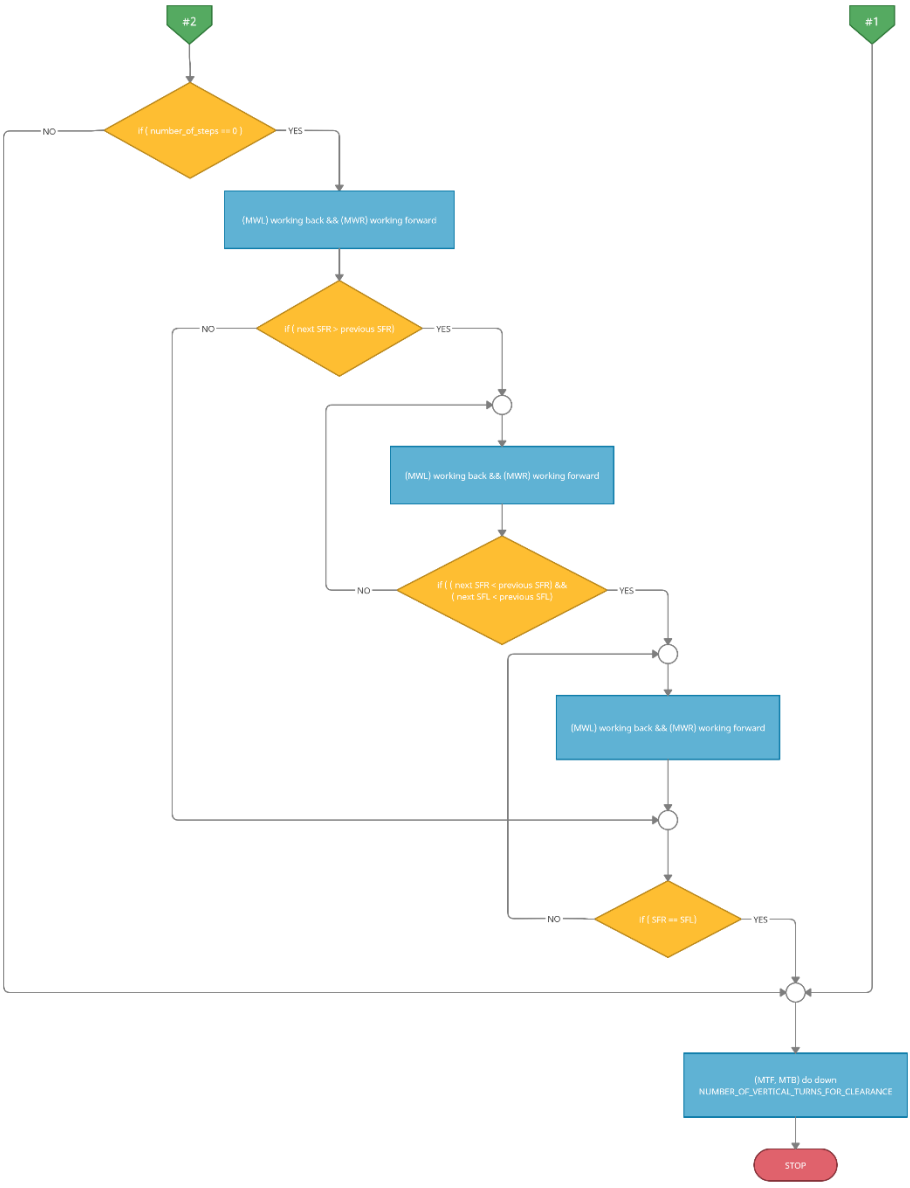
Rys. 15. Schemat blokowy procesu schodzenia (część 3)



Rys. 16. Schemat blokowy procesu schodzenia (część 4)



Rys. 17. Schemat blokowy procesu kalibracji (część 1)



Rys. 18. Schemat blokowy procesu kalibracji (część 2)