ROBOTYKA 2 Roboguide - projekt zaawansowany

Autor: Emilia Szymańska 248975 śrTP 8:00

07.05.2021r.

1 Cel ćwiczenia

Celem ćwiczeń jest stworzenie zaawansowanego projektu współpracujących robotów manipulacyjnych firmy FANUC z wykorzystaniem środowiska Roboguide. Moim projektem było skoordynowanie pracy czterech robotów LR Mate i200C. Pierwszy robot przekłada elementy ze stołu na transporter, na którego drugim końcu jest umiejscowiony drugi robot. Robot ten naprzemiennie przenosi elementy na transportery prowadzące do robotów odkładających przedmioty na stoły.

2 Wykonanie zadania

Filmik pokazujący działanie programu znajduje się pod linkiem: https://www.youtube.com/watch?v=vRVqwmWg6-Q.

2.1 Stworzenie środowiska

Pierwszym etapem wykonania zadanie było stworzenie własnego CELL'a z potrzebnymi maszynami. Na początku dodałam cztery roboty FANUC LR Mate 200iC (razem z chwytakami), trzy stoliki oraz trzy transportery. Cechą charakterystyczną transporterów w Roboguide jest to, że same z siebie nie przenoszą elementów i należało dodać palety, a następnie stworzyć symulację ich ruchu na transporterach. Ruch palety jest wyzwalany przez wystawienie wartości ON odpowiedniej zmiennej przypisanej do danego robota. Konfiguracja ruchu palet na transporterach została przedstawiona na rys. 1, 2 i 3.

Motion Control Type				
Device VO Controlled			\sim	
Axis Type Spe O Rotary Time Linear	ed • ✓ -→+ -++	1.000 :	sec	
Inputs				
Output Dev	IO Tag	Value	Location	
Robot Controller2	DO[1]	ON	-900	the second s
Robot Controller1	DO[1]	ON	0	
[none]	[none]	[none]	0	
			Test	
Outputs				
Input Dev	IO Tag	Value	Location	
Robot Controller2	DI[1]	ON	0	PI2
Robot Controller1	DI[1]	ON	-900	Com sets
[none]	[none]	[none]	0	

Rysunek 1: Konfiguracja ruchu palety na transporterze nr 1.

Link11, Mach	nine2				8
General Motion	Calibrat	ion Link C	AD Parts	Simulation	
Motion Control	Гуре				
Device VO Cont	trolled				\sim
Axis Type	Speed				
Rotary	Time	→+	1.000	sec	
Linear		-++	1.000	sec	
Inputs		L			
Output Dev		IO Tag	Value	Location	
Robot Controller	r1	DO[2]	ON	-900	
Robot Controller	r4	DO[1]	ON	0	
[none]		[none]	[none]	0	
				Т	est
Outputs					
Input Dev		IO Tag	Value	Location	
Robot Controller	r1	D[2]	ON	0	
Robot Controller	r4	D[1]	ON	-900	
[none]		[none]	[none]	0	
	_	_	_		_
1	01			nak H	lala
	OK	Car		фых	eib

Rysunek 2: Konfiguracja ruchu palety na transporterze nr 2.

Motion Control Type	UN LINK C	AD Fulta	Sindiation		
Device VO Controlled			~		
Axis Type Speed O Rotary Time Innuts	→+[1.000 s	ec ec		Ş
Output Dev	IO Tag	Value	Location	1 L / 🖊	- î 1
Robot Controller1 ~	DO[3]	ON	-900		11
Robot Controller3	DO[1]	ON	0	1 / /	- i 👖
[none]	[none]	[none]	0] / /	1 📕
			Test	」 / / ,	1,000
Outputs			Test		
Outputs Input Dev	IO Tag	Value	Location	1	
Outputs Input Dev Robot Controller1	IO Tag DI(3)	Value	Location 0		
Outputs Input Dev Robot Controller1 Robot Controller3	IO Tag D[[3] D[[1]	Value ON ON	Location 0 -900		had .
Outputs Input Dev Robot Controller1 Robot Controller3 [none]	IO Tag DI(3) DI(1) [none]	Value ON ON [none]	Location 0 -900 <th< td=""><td></td><td></td></th<>		

Rysunek 3: Konfiguracja ruchu palety na transporterze nr 3.

Kolejnym krokiem było dodanie elementu (pudełka), które miałoby być transportowane przez maszyny. Pudełko (box) musiało zostać powiązane z chwytakami, stołami i paletami jak przedstawiono na rys. 4 w przypadku powiązania części ze stołem.



Rysunek 4: Ustawienie pudełka na stole.

W rezultacie otrzymałam środowisko jak na rys. 5.



Rysunek 5: Środowisko symulacyjne.

2.2 Programowanie

Na początku musiałam zdefiniować funkcje chwytania i odkładania elementów na stoły/transportery dla poszczególnych robotów. Przykładowe ustawienie definicji Simulation Program dla chwytania elementu z transportera jest przedstawione na rys. 6.

🗟 Simula	tion Program Ed	itor - Robot Controller4 - GRAB2	×
ہے۔ Record	▼ xx ¹² Touchup ▼	MoveTo Forward Backward Inst None None	- 🚄 Path
· •		🛛 🗙 🛛 🛃 🖆 🔞 🖻 Robot Controller4 🗸 🕌 GRAB2	~
- 🔍	1: Pickup (💋 box	~)
	From (C Machine2:Link11	~)
	With (🕈 GP: 1 - UT: 1 (Eoat1)	~)

Rysunek 6: Program do chwytania elementu z transportera.

Program trzech robotów (rys.7, 8, 9) przenoszących elementy ze stołu na transporter lub z transportera na stół jest niemal taki sam, jedynie z drobnymi poprawkami. Robot (w nieskończonej pętli stworzonej przy pomocy opcji LABEL nr 2) po otrzymaniu sygnału od transportera (w postaci ON w zmiennej DI [digital input]) pobiera/odkłada element. Dodatkowo, by paleta mogła się przemieścić, wystawiany jest krótki sygnał ON w zmiennej DO [digital output], po 100 ms zmieniany na OFF.



Rysunek 7: Program robota nr 2.



Rysunek 8: Program robota nr 3.



Rysunek 9: Program robota nr 4.

Robot nr 1, który ma za zadanie przenosić elementy z transportera na transporter, ma trochę bardziej zaawansowany program. Funkcja DROP (wykorzystująca metodę Drop) w Simulation Program musiała zostać zdefiniowana na dwukrotnie, by obsłużyć odkładanie elementów na oba transportery. Dodatkowo koniecznym było zdefiniowanie zmiennej CHOICE w rejestrze robota, która definiuje wybór transportera spomiędzy dwóch dostępnych. Po otrzymaniu sygnału, że transporter nr 1 przywióżł element, robot pobiera go, wysyła sygnał do tego transportera (w związku z czym paleta wraca on do pozycji początkowej) i w zależności od wartości w rejestrze CHOICE odkłada element naprzemiennie na transporter nr 2 i nr 3. Po odłożeniu elementu wysyłany jest sygnał do odpowiedniego transportera, dzięki czemu możliwy jest dalszy etap pracy robotów. Prezentacja działania robota wraz z kodem jest rzedstawiona na rysunkach 10, 11.



Rysunek 10: Praca robota nr 1 z fragmentem kodu.



Rysunek 11: Kod programu robota nr 1.

3 Wnioski

- Przy wykorzystaniu transporterów należy pamiętać, że nie mają one wbudowanej funkcji przenoszenia elementów trzeba wykorzystać paletę i w jej ustawieniach skonfigurować ruch w reakcji na odpowiednie sygnały otrzymywane od robotów.
- Dodając element do środowiska, chcąc go później wykorzystać w symulacji, należy zdefiniować jego położenia w kolejnych etapach symulacji (na stołach, paletach, w zaciśniętym chwytaku). Roboguide sam z siebie nie zapewnia obsługi symulacji całej drogi elementu, programista sam musi zadbać, by symulacja wyglądała jak w rzeczywistości, z zachowaniem praw fizyki.
- Stworzony w niniejszym projekcie system spełnia swoją rolę, jednak dałoby się go zoptymalizować poprzez zmniejszenie czasów transportowania elementów i zwiększenie prędkości ruchów J i L pomiędzy wyznaczonymi punktami.
- Roboguide pozwala na kompleksowe prototypowanie systemów składających się z wielu robotów. Pozwala to uniknąć późniejszych błędów, przewidzieć pewne problemy czy zoptymalizować produkcję istniejącego systemu.