Polar robot arm

projekt POiGK

Krzysztof Karłowicz 184628, Jakub Poćwiardowski 184827

ACiR2

1) Interfejs

Program został napisany w Java3D. Po uruchomieniu programu w interfejsie oprócz ekranu z naszym środowiskiem, ujrzymy instrukcję sterowania oraz panel przycisków po prawej stronie. Użytkownik za jego pośrednictwem może między innymi włączyć lub wyłączyć muzykę, przywrócić pierwotne ustawienie kamery, a także sterować uczeniem ruchów robota (nagrywaniem). Przycisk nagrywania zmienia kolor na czerwony gdy jest ono aktywne, niektóre przyciski aktywują się lub deaktywują w zależności od sytuacji (np. nagranie nie może być odtworzone gdy nie istnieje, lub jesteśmy w trakcie nagrywania). Niżej wyświetlane są na bieżąco aktualizowane koordynaty robota, które można edytować i tym samym zaprogramować jego ruch, który zostanie wykonany niezwłocznie po wciśnięciu przycisku ustawienia koordynatów. Jeśli użytkownik wpisze wielkość spoza zakresu, ruch wykonany zostanie na wartość równą reszcie z dzielenia tej wartości przez zakres. Ruch wykonywany jest w pętli z funkcją zamrożenia ruchu na 20 ms, celem symulacji samego procesu ustawiania.

Przykładowy zrzut ekranu (w czasie nagrywania):

2) Konstrukcja robota

Cały robot stworzony jest z 11 brył. Robot posiada dwa obroty: pierwszy wokół osi własnej (pionowej), i drugi w osi do niej prostopadłej, sterujący wychyleniem w ustalonym zakresie (-45°,45°). Kolejnym elementem jest wysuwane ramię z końcówką magnetyczną, pozwalającą na chwytanie prymitywu (kulki). Wysuwanie tego elementu również zostało ograniczone, a samo

wysunięcie zostało w interfejsie przeliczone na wartości od 0 do 100% dla lepszej czytelności. Robot operuje na kulistej przestrzeni roboczej. Sterowanie robotem odbywa się za pomocą klawiatury, lub wpisywania koordynatów wspomnianych wcześniej, ustawienie kamery możemy zaś modyfikować za pomocą myszki.

3) Kolizja

Do egzekwowania kolizji służy dodatkowa klasa „detektorKolizji", która na podstawie grupy transformacji obiektu oraz jego więzów, sprawdza czy obiekt jest w kolizji z innym obiektem. Aby sprawdzić czy kolizja powinna mieć wpływ na ruch robota, stworzono, już w głównej klasie dodatkową funkcję „sprawdzanieKolizji()", która bierze pod uwagę wszystkie kluczowe czynniki, np. czy piłka jest chwycona. Gdy taki warunek zajdzie, na ruch robota ma wpływ kolizja kulki z podłożem (np. gdyby użytkownik trzymając ją chciał wcisnąć ją w podłoże). Gdy kulka nie jest chwycona, sprawdzamy kolizję ramienia z podłożem i ramienia z kulką. W przypadku drugiej z wymienionych kombinacji kolizji aktywuje się również możliwość chwycenia kulki. Ponieważ blokowanie ruchu kolizją blokowało każdy rodzaj ruchu, stworzono zmienną char „ostatni_klawisz" dzięki której program sprawdza który klawisz został wciśnięty jako ostatni i w razie kolizji blokuje tylko ten ruch, za który dany klawisz odpowiada.

4) Uczenie sekwencji ruchu robota (nagrywanie)

Do realizacji nagrywania służą wspomniane już wcześniej przyciski w interfejsie. Po wciśnięciu przycisku, zaświeci się on na czerwono i od teraz wciskając jakiekolwiek klawisze zostaną one zapisane do wektora „nagrane_przyciski". Zapisujemy także transformacje wszystkich elementów robota i kulki w momencie wciśnięcia przycisku. Gdy zakończymy i odtworzymy nagranie, nastąpi ustawienie elementów do zapisanych transformacji i realizacja symulacji wciskania zapisanych klawiszy z wektora. Ponieważ gdy przytrzymujemy przycisk, zapisywany jest on wielokrotnie, nie potrzebujemy mierzyć ruchów, ani czasu ich wykonania, a robot wykona ruch dokładnie taki jak gdy my nim sterowaliśmy. System nagrywania jest również odporny na różne scenariusze trzymania kulki, np. gdy nagrywanie zostało rozpoczęte lub zakończone w czasie jej trzymania. W programie rozważone są wszystkie warianty i odpowiednio zaimplementowane zmiany zaczepienia (rodzica) kulki, a także zmiany jej transformacji.

5) Dźwięk i elementy estetyczne

Oprócz wspomnianej już wcześniej muzyki, w programie został również zaimplementowany industrialny dźwięk grający w momencie poruszenia się ramienia przez użytkownika oznaczony jako obiekt „sfx" klasy java.sound.sampled.Clip. W programie zaimplementowany jest warunek sprawdzający wciśnięcie przycisków odpowiadających za ruch robota oraz sprawdzenie czy dźwięk już nie gra, aby uniknąć jego dublowania. Po jego spełnieniu dźwięk jest uruchamiany. Gdy puścimy przycisk (implementacja w funkcji keyReleased()), dźwięk zatrzymuje się. Ponieważ taka konstrukcja powodowała bardzo szybkie puszczanie i zatrzymywanie dźwięku przy odtwarzaniu nagrania oraz sterowaniu numerycznym, funkcję „keyReleased()" uruchamiamy dopiero po zakończeniu pętli danego ruchu w sterowaniu numerycznym oraz przy wykryciu zmiany KeyEvent przy odtwarzaniu nagrania.

Otoczenie robota inspirowane jest budową hali produkcyjnej. W tym celu dokonano importu odpowiednich tekstur zewnętrznych. Ściany wokół robota zbudowane są w oparciu o klasę Box. Trzy z nich posiadają zwyczajną teksturę ściany, czwarta zaś reprezentuje dalszą część hali dodając scenerii głębi. Również z klasy Box zbudowany jest sufit, pod którym umieszczono dwa źródła światła punktowego. Podłoga pod robotem jest betonowa i nieco mniejsza dla lepszej widoczności. Dodatkowo robot stoi na murowanej podstawie.