

Spis treści

[1. Wstęp 3](#_Toc89618554)

[2. Sterowanie robotem 4](#_Toc89618555)

[1. Gcode 4](#_Toc89618556)

[3. Kinematyka odwrotna robotów 6-cio osiowych 4](#_Toc89618557)

[4. Generowanie modelu obrabianego przedmiotu 4](#_Toc89618558)

[Rys.4.1 Graficzna wizualizacja chmury punktów 5](#_Toc89618559)

[Rys.4.2 Fragment tablicy wierzchołków 5](#_Toc89618560)

[Rys.4.3 Indywidualne konfiguracje punktów 6](#_Toc89618561)

[Rys.4.4 Kolejność przetważania danych przez algorytm Marching Cubes 6](#_Toc89618562)

[5. Podsumowanie wyników 7](#_Toc89618563)

[6. Bibliografia 7](#_Toc89618564)

# Wstęp

Obecnie obrabiarki CNC są w stanie sprostać większości powierzonych im zadań. Ich największym ograniczeniem jest niewielkie pole robocze. Istnieją 5-cio osiowe frezarki bramowe, których pole robocze jest wynosi nawet xxxxx. Są one jednak bardzo kosztowne.

Skrawanie dużych obiektów można zrealizować montując wrzeciono na robocie przemysłowym. Mają one duży zasięg, który przekłada się na duże pole robocze. Są jednak rzadziej stosowane w obróbce skrawaniem niż konwencjonalne centra obróbcze. Składa się na to kilka czynników, m.in. niewystarczająca sztywność konstrukcji spowodowana długim ramieniem robota, skutkując błędami pozycjonowania. Podczas skrawania dokładność pozycji narzędzia wynosi 0.5 - 2mm oraz jest zależna od aktualnej odległości narzędzia od bazy. Dodatkowym atutem, poza duży polem roboczym, jest szósta oś, zapewniająca dodatkowy stopień swobody. Umożliwia to dotarcie narzędziem do miejsc do których dotarcie byłoby niemożliwe przy konwencjonalnych obrabiarkach CNC. Możliwe jest również zastosowanie obrotnika lub prowadnic liniowych, które jeszcze bardziej zwiększają elastyczność systemu. Pomimo wad roboty wykorzystywane są do obróbki łatwiej skrawalnych materiałów, gdy uzyskiwana dokładność mieści się w tolerancji wymiarów.

Problematyczne jest również zaprogramowanie robota. Zapisane punkty zostaną odtworzone przez maszynę, ale ustalenie ich na rzeczywistym robicie jest czasochłonne oraz wymaga uczestniczenia fizycznego robota w procesie programowania. Zastosowanie programowania online byłoby szczególnie trudne w kontekście obróbki skrawaniem. Lepszym rozwiązaniem jest programowanie offline. Nie wymaga ono fizycznego robota, a oprogramowanie CAM generuje g-code, znacząco ułatwiając i przyśpieszając proces programowania.

Uruchomienie programu na rzeczywistym robocie bez wcześniejszego sprawdzenia jest ryzykowne i może prowadzić do kosztownych uszkodzeń. W takiej sytuacji pomocne są symulatory pozwalające na odtworzenie programu bez obaw o robota. Podczas symulacji wirtualny robot zachowuje się jak fizyczny, możliwe jest wykrycie np. kolizji ramienia robota z blokiem materiału. Po zakończeniu symulacji otrzymujemy również podgląd finalnego produktu.

# Sterowanie robotem

Rzeczywistym robotem można sterować na kilka sposobów. Wykorzystując panel sterowania można na żywo przemieszczać TCP w globalnym układzie współrzędnych robota lub lokalnym TCP. Sterować można również bezpośrednio przegubami robota. Funkcje te zostały przeniesione do symulatora, pozwalając na ręczne sterowanie wirtualnym robotem.

## Przemieszczanie TCP

## Obrót przegubów robota

Sterowanie osiami polega na nadaniu każdej z nich określonego kąta do jakiego ma dążyć. Wirtualny robot bez problemu może osiągnąć każdy zadany kąt na każdej z osi. Rzeczywisty robot posiada ograniczenia kątów które każda z osi może przyjąć. Jest to dobre miejsce, aby upewnić się że ograniczenia fizycznego robota zostaną odwzorowane. Zadana wartość kąta jest korygowana. Odpowiedzialna jest za to funkcja przedstawiona na rysx.



* 1. Funkcja ustalająca kąt na osi robota

Na początku należy sprawdzić czy przekazany kąt jest liczbą. Będzie to przydatne później podczas obliczeń kinematyki odwrotnej. Niektóre osie mogą obracać się w 360° dlatego wprowadzono możliwość wyłączenia korekcji ustawiając oraz . Następnie funkcja nadaje już poprawny kąt dla osi robota głównego oraz jego ducha.

Aby uzyskać ruch ramienia kąt jest interpolowany. Nie może to być jednak zwykła interpolacja liniowa ponieważ prowadziłoby to do nie optymalnej drogi. Na przykład ruch z do wymagałby 300 sekund przy prędkości obrotowej dla zwykłej interpolacji liniowej kąta. Lepszym rozwiązaniem byłby ruch w przeciwną stronę. Przy tej samej prędkości zająłby tylko 30 sekund. W silniku Unity istnieje funkcja realizująca interpolację liniową dla kątów, która eliminuje przytoczony problem. Na rysXXX przedstawiono funkcję która jest wykonywana w każdą klatkę działania programu Zmienia ona kąt „Z” który jest równoznaczny z kątem . Jednak nie można zmienić pojedynczego składnika, konieczne jest stworzenie nowego wektora 3D przechowywującego informację o rotacji w postaci kątów Eulera. Jest on następnie konwertowany na kwaternion. Tej 4-ro wymiarowej struktury algebraicznej silnik używa do manipulacji obrotem obiektów.



## Gcode

G-code to język zapisu poleceń dla obrabiarek sterowanych numerycznie. Zapisana jest w nim dokładna ścieżka którą musi pokonać narzędzie, aby poprawnie wykonać przedmiot. Funkcji g-code jest bardzo dużo, do symulatora zostały zaimplementowanie tylko podstawowe:

* G0 – ruch szybki,
* G1 – ruch liniowy interpolowany,
* G2 – ruch kołowy zgodnie z ruchem wskazówek zegara,
* G3 – ruch kołowy przeciwnie do ruchu wskazówek zegara.

Oraz parametry oraz osie:

* X, Y, Z – położenie punktu,
* A, B, C – orientacja punktu,
* F – prędkość liniowa,
* S – prędkość obrotowa wrzeciona.

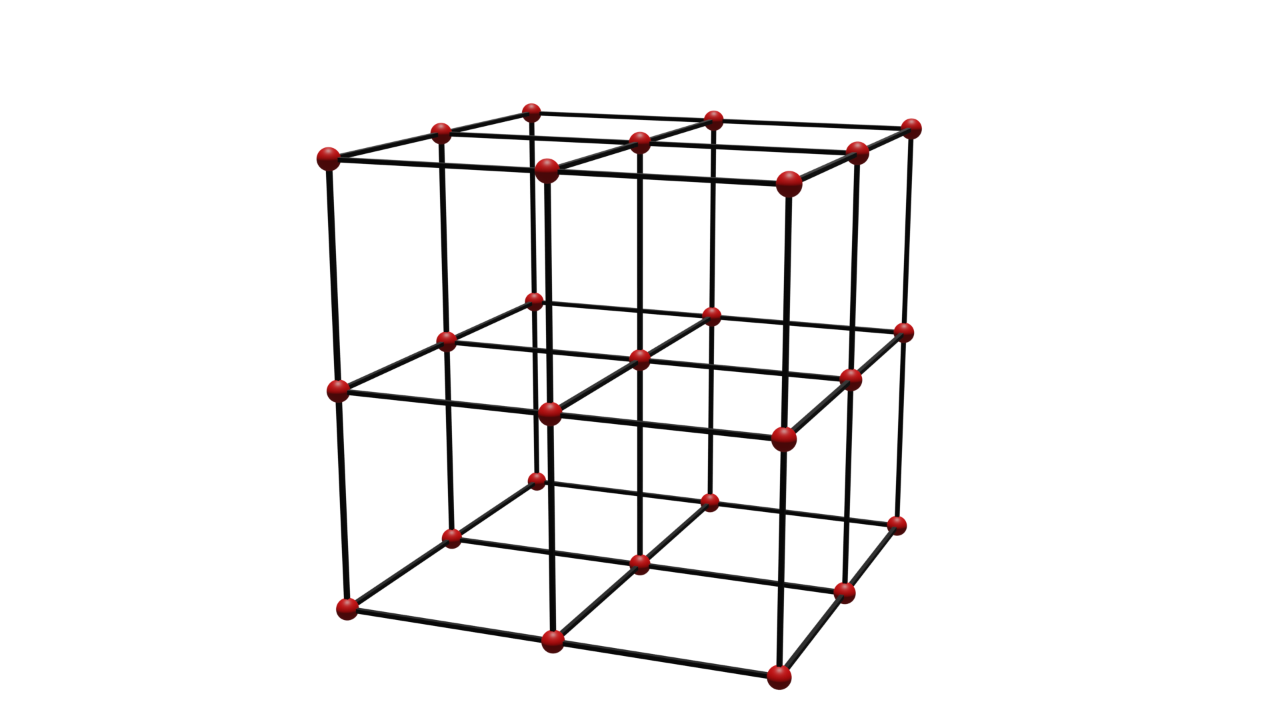
W przeciwieństwie do pozostałych parametrów, podczas symulacji prędkość wrzeciona nie ma większego znaczenia. Istotna jest informacja czy wrzeciono jest uruchomione – .

# Kinematyka odwrotna robotów 6-cio osiowych

# Generowanie modelu obrabianego przedmiotu

Sam ruch robota jest niewystarczający do weryfikacji napisanego programu. Wygenerowanie modelu 3D obrabianego przedmiotu pozwoli na wizualną ocenę wykonanej pracy narzędzia.

Kształcie bryły jest definiowany za pomocą trójwymiarowej chmury punktów (Rys.4.1). Są one rozłożone w równej odległości od siebie, tworząc sześcian. Ich ilość jest ustalona przy uruchomieniu programu. Informacja przechowywana jest w postaci „0” lub ”1”.



* 1. Graficzna wizualizacja chmury punktów

Proces skrawania jest symulowany poprzez sprawdzenie, które punkty znajdują się wewnątrz narzędzia. Jest to realizowane za pomocą serii obliczeń oraz instrukcji warunkowych.

WYJAŚNIENIE SPRAWDZANIA PUNKTU

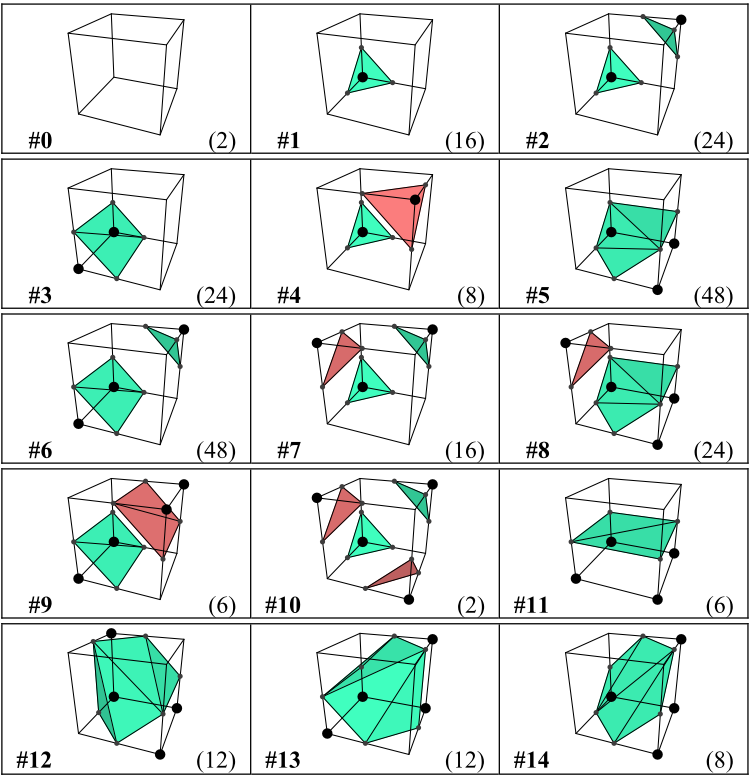
Aby poprawnie wyświetlić obrabiany przedmiot konieczne jest dodanie algorytmu generującego siatkę trójkątów na podstawie chmury punktów. Do tego celu został użyty algorytm „Marching Cubes”.

Algorytm generuje siatkę trójkątów zależną od 8 punktów znajdujących się na wierzchołkach sześcianu. Zatem na jeden sześcian przypada możliwych kombinacji. Informacja jak powinien wyglądać dana konfiguracja zakodowana jest w tablicy, której fragment przedstawiony jest na Rys.4.2.



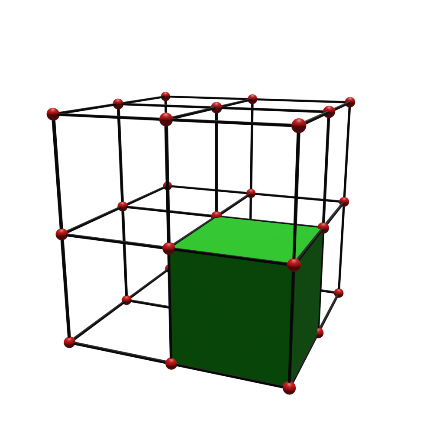
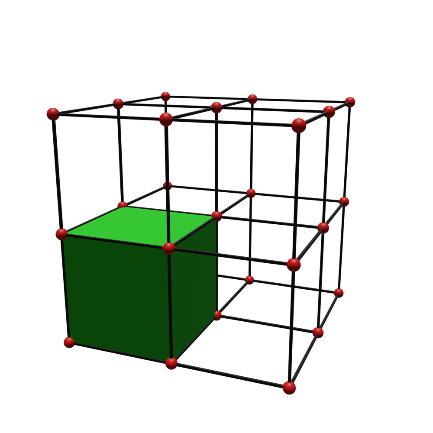
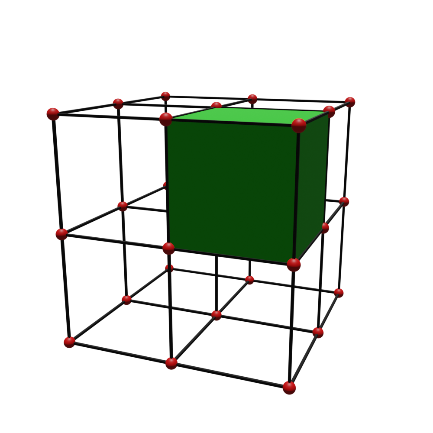
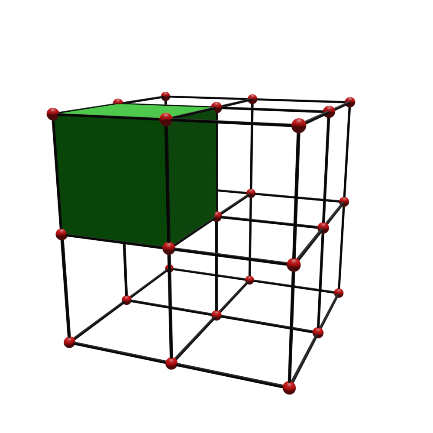
* 1. Fragment tablicy wierzchołków

Większość z nich jest rotacją lub negacją innych. Indywidualnych kombinacji jest 15 i zostały przedstawione zostały na Rys.4.3. Czarną kropką oznaczono wierzchołki które są w materiale, wierzchołki bez kropki znajdują się poza. Zielona ścianka widziana jest od przodu, czerwona od tyłu. W nawiasie podano liczbę podobnych kombinacji – obrócone lub/i zanegowane.



* 1. Indywidualne konfiguracje punktów

Algorytm jest powtażany dla każdej grupy punktów, jak przedstawiono na Rys.4.4. Generowana siatka trójkątów dla każdej grupy jest ułożona w taki sposób aby pasowała do sąsiadujących siatek, tworząc tym samym pełny model 3D obiektu.



* 1. Kolejność przetważania danych przez algorytm Marching Cubes

Dokładność modelu zależna jest od ilości punktów. Zwiększenie ilości punktów zwiększy również ilość obliczeń do wykonania, które muszą być wykonywane w każdej klatce działania programu. Wiec aby przyśpieszyć program przetwarzanie punktów oraz generowanie modelu odbywa się za pośrednictwem karty graficznej, która została stworzona do przeprowadzania wielu równoległych, ale prostych obliczeń.

PORÓWNANIE BLOKU O MAŁEJ ROZDZIELCZOŚCI ORAZ TEGO O DUŻEJ – FAJNIE BY BYŁO WYKONAĆ TEN SAM G-KOD

Ustalono wymiary chmury punktów na 100x100x100 punktów. W sumie jest to punktów, które trzeba sprawdzić oraz wygenerować na ich podstawie siatki trójkątów. Daje to akceptowalną rozdzielczość przy stabilnych 10 klatkach na sekundę.

Chmura punktów przyjmuje formę sześcianu, więc odległość między punktami zależna jest od najdłuższego boku półfabrykatu.

# Podsumowanie wyników

RZECZY KTÓRE MOŻNABY ZROBIĆ LEPIEJ:

- OPTYMALIZACJA GENEROWANIA MODELU 3D

# Bibliografia