Informatic tool for autogenerating elements of the pipe geometry checking station

Narzędzie informatyczne do autogenerowania elementów stanowiska do sprawdzania geometrii rur

DAWID MROCZKOWSKI
MAREK WYLEŻOŁ *

DOI: https://doi.org/10.17814/mechanik.2023.1.2

The aim of the scientific research was to accelerate the process of creating pipe geometry checking stations by creating a tool for auto-generating its elements.
KEYWORDS: metrology, geometry of bent pipes, CATIA v5

Celem badania było przyspieszenie procesu tworzenia stacji kontroli geometrii rur poprzez stworzenie narzędzia do automatycznego generowania jej elementów.
SŁOWA KLUCZOWE: metrologia, geometria rur giętych, CATIA v5

Wprowadzenie

Celem artykułu było przedstawienie możliwości przyspieszenia procesu tworzenia sprawdzianów do weryfikacji geometrii rur poprzez utworzenie narzędzia informatycznego do autogenerowania [3, 4] elementów tych sprawdzianów. Potrzeba posiadania takiego narzędzia wynikła nie tylko z dążenia do przyspieszenia procesu konstruowania sprawdzianu, lecz także z chęci redukcji kosztów jego docelowego wytworzenia poprzez standaryzację elementów składowych i wykorzystanie istniejących już elementów.

Sprawdziany geometrii rur odgrywają ważną rolę, ponieważ umożliwiają weryfikację istotnych wymiarów produktu, aby zapewnić, że sprawdzany element zostanie poprawnie zamontowany i zmieści się np. w dostępnej przestrzeni podwozia samochodu. Z powodu ograniczonego miejsca i potrzeby dopasowania rury do pozostałych elementów podwozia dostarczonych przez klienta, geometria rur przybiera nieraz skomplikowane kształty, które wymagają docelowej weryfikacji.

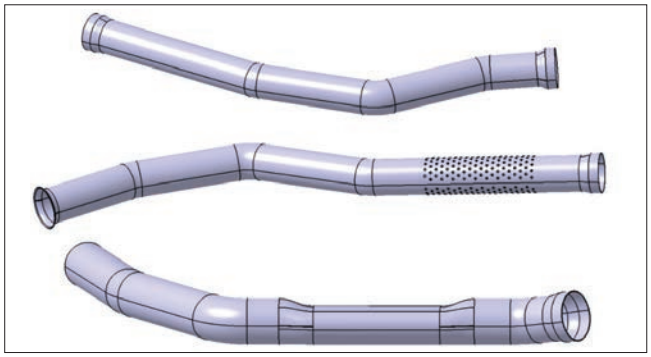

Fig. 1. Example models of tested pipes
Rys. 1. Przykładowe modele sprawdzanych rur

Na rys. 1 przedstawiono przykładowe modele geometrii rur. Modele opracowano w środowisku systemu CATIA v5 [1].

Do utworzenia narzędzia informatycznego zastosowano język programowania VBA (Visual Basic for Applications), który – dzięki swojej składni – daje większe możliwości niż operacje dostępne standardowo z poziomu interfejsu użytkownika. Zastosowanie języka VBA pozwala zatem na:

- zaprojektowanie interfejsu, za pomocą którego będzie możliwa walidacja wprowadzanych przez użytkownika wartości,
- połączenie z pozostałymi aplikacjami Microsoft Office oraz z bazą danych,

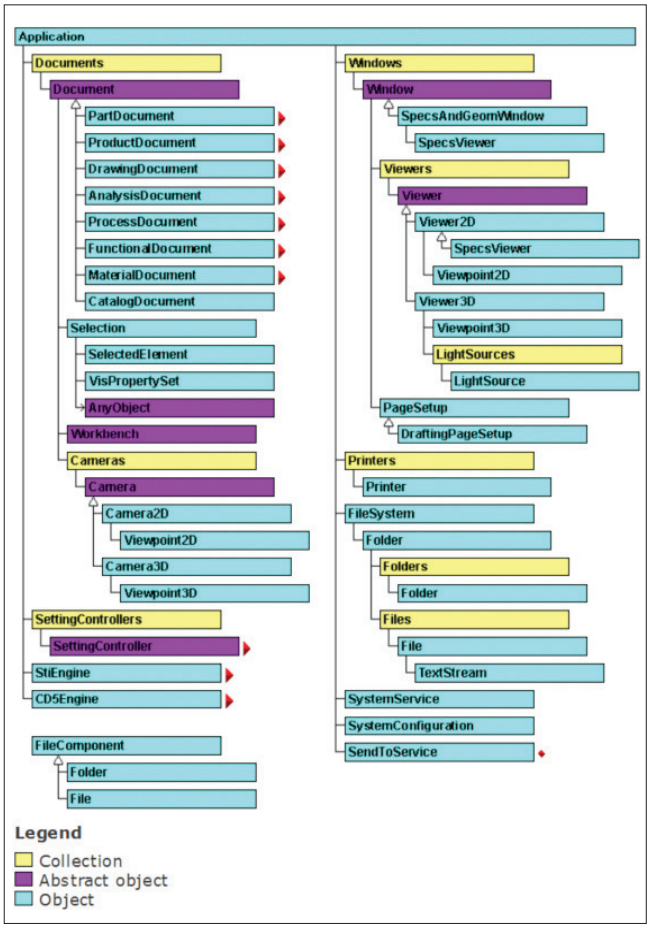

Fig. 2. The structure of the main object of the application
Rys. 2. Struktura obiektu głównego aplikacji

* Mgr inż. Dawid Mroczkowski – dawid96.12mroczkowski@gmail.com, Polska
Dr hab. inż. Marek Wyleżoł, prof. PŚ – marek.wylezol@polsl.pl, https://orcid.org/0000-0001-6324-510X – Katedra Podstaw Konstrukcji Maszyn, Wydział Mechaniczny Technologiczny, Politechnika Śląska, Gliwice, Polska

- implementację skomplikowanych algorytmów, np. algorytmów sztucznej inteligencji lub metod numerycznych,
- dostęp do większej liczby funkcji programu poprzez możliwość odwołania się do danej aplikacji i jej dokumentów (np. tworzenie dokumentów i zmiana ich nazw, zmiana cech elementów interfejsu systemu CATIA v5).

Oprogramowanie narzędzia zaczyna się od zdefiniowania obiektu głównego aplikacji. Obiekt ten zawiera kolekcje, które pozwalają na dostęp do dokumentów, okien aplikacji, drukarek, obsługi kamery itd. Przykładowo: tworząc obiekt, który dziedziczy z abstrakcyjnej klasy „Document", można się odwołać do dowolnego modułu oprogramowania CATIA v5 poprzez zdefiniowanie obiektu tej klasy i korzystać z jego właściwości i zdefiniowanych metod (rys. 2). Działania te są tożsame ze sposobem programowania obiektowego.

Schemat działania

W artykule przedstawiono cztery etapy prowadzące do utworzenia narzędzia, które docelowo umożliwiają sprawne wygenerowanie sprawdzianu do weryfikacji geometrii rur (rys. 3). Są to:

- operacja wygenerowania modułu,
- operacja wygenerowania osi przewodniej rury,
- operacja wygenerowania podstawy modułu,
- utworzenie interfejsu użytkownika scalającego wszystkie operacje.

Przyjęte założenia

Podczas projektowania narzędzia informatycznego wzięto pod uwagę następujące założenia:

- ograniczenie do minimum liczby działań wykonywanych przez użytkownika,
- wygenerowane elementy powinny być elementami, których charakterystyczne wymiary są pozyskiwane z tabeli projektowej (*design table*) [1, 2, 5], stanowiącej pewnego rodzaju bazę danych często używanych elementów,
- narzędzie będzie samodzielnie dobierało konfigurację projektową „żebra" na podstawie zmierzonej wysokości „widelca",
- każdy z elementów powinien mieć zdefiniowane parametry umożliwiające łatwą modyfikację charakterystycznych wymiarów, takich jak średnica otworów czy wcięcie w adapterze o wymiarze równym średnicy rury,
- każdy moduł będzie nowo utworzonym produktem (w sensie systemu CATIA v5), składającym się z czterech elementów,
- narzędzie powinno umożliwiać generowanie modułów na zdefiniowanym odcinku wzdłuż osi przewodniej rury,
- narzędzie powinno zawierać opcję generowania podstawy modułu,
- narzędzie powinno również zawierać opcję generowania osi przewodniej rury,
- interfejs użytkownika powinien być intuicyjny i niezłożony.

Fig. 3. Tool implementation stages
Rys. 3. Etapy wykonania narzędzia

Generowanie modułu do weryfikacji geometrii sprawdzianu

Pierwsza operacja narzędzia generuje moduł służący do sprawdzania geometrii. Moduł składa się z następujących elementów (rys. 4): podstawki (1), żebra (2), widelca (3) i adaptera (4).

Podstawka, żebro oraz adapter generowane są na podstawie tabeli projektowej, ponieważ są to często używane elementy, które znajdują się już w magazynie w różnych wariantach wymiarowych (w przypadku wybranego przedsiębiorstwa). Charakterystyczne wymiary tych elementów zostały zamieszczone we wspomnianej tabeli projektowej. Ich wartości należą do zdefiniowanych parametrów, które są powiązane z modelem za pomocą określonych relacji (parametryzacja relacyjna [2, 4]).

Narzędzie generuje model widelca modułu, którego wysokość i kąt gięcia są zależne od osi przewodniej rury, a grubość jest definiowana przez użytkownika. Następnie generowane jest żebro, którego wymiary są zawarte w tablicy projektowej (rys. 3). Program samodzielnie mierzy wysokość widelca i na podstawie tego wymiaru dobiera odpowiednią konfigurację widelca z tablicy projektowej. Specjalna funkcja – iterując przez tablice – poszukuje konfiguracji o wysokości zbliżonej do zmierzonej wysokości widelca, ale mniejszej niż ta wartość (rys. 5).

Na rys. 5 przedstawiono kod programowy funkcji umożliwiającej wyszukiwanie indeksu wariantu, który ma największą wartość wysokości żebra, z przedziału wartości mniejszych niż zmierzona wysokość widelca (rys. 5).

Fragment kodu pokazany na rys. 7 definiuje tablicę projektową, a także definiuje obiekt w postaci skoroszytu arkusza kalkulacyjnego MS Excel, tworząc połączenie pomiędzy nimi. Następnie tworzy wewnętrzne powiązanie pomiędzy kolumnami skoroszytu ze zdefiniowanymi parametrami modelu (rys. 7).

W dalszej kolejności program przeszukuje wszystkie wartości powstałej tablicy projektowej i za pomocą wcześniej zdefiniowanej funkcji znajdującej indeks wariantu ustawia określoną konfigurację tablicy projektowej (rys. 8).

Program generuje moduły na całej długości wybranej osi przewodniej rury, która jest następnie dzielona na fragmenty. Oś rury w rzeczywistości jest generowana za pomocą funkcji „Polyline" [1], zawierającej w sobie informacje o współrzędnych punktów gięcia oraz wartościach promieni gięcia (rys. 9).

Moduły są generowane na obu końcach każdego odcinka rury z określonym przez użytkownika

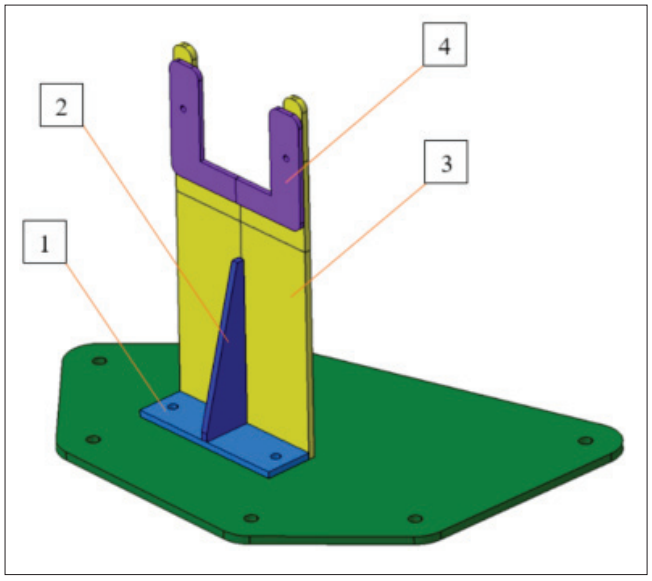

Fig. 4. Test module with its base
Rys. 4. Moduł sprawdzianu wraz z podstawą

```
Public Function findBestIndex(fArr)
    Dim valHolder
    valHolder = fArr(2)
        For i = 2 To UBound(fArr)
            If fArr(i) > valHolder Then
                valHolder = fArr(i)
            End If
        Next

    For i = 2 To UBound(fArr)
        If fArr(i) = valHolder Then
            findBestIndex = i - 1
            Exit For
        End If
    Next i
End Function
```

Fig. 5. The code of the looking function for the index of the highest value in the table
Rys. 5. Kod funkcji poszukującej indeksu największej wartości w tablicy

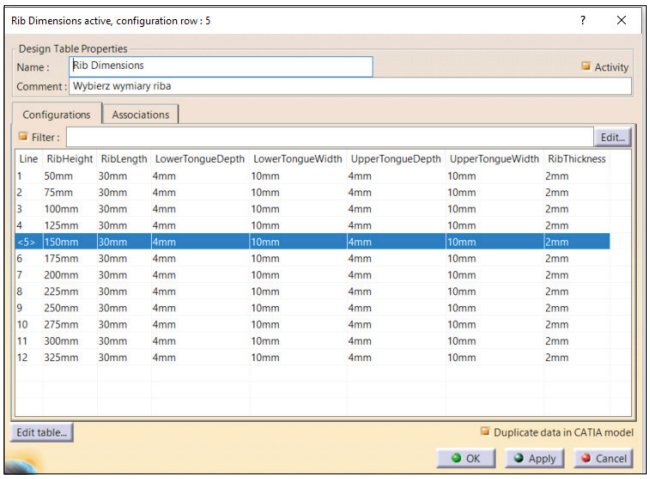

Fig. 6. Design table containing the dimensional variants of the rib
Rys. 6. Tablica projektowa zawierająca warianty wymiarowe żebra

```
Dim sFolderPath As String
sFolderPath = "C:\Users\dawimroc\Desktop\DesignTableTri.xls"

Dim oDesignTable As DesignTable
Set oDesignTable = oRelations.CreateDesignTable("Rib Dimensions",
                   "Wybierz wymiary riba", False, sFolderPath)

With oDesignTable
    .AddAssociation RibHeight, "RibHeight"
    .AddAssociation RibLength, "RibLength"
    .AddAssociation LowerTongueDepth, "LowerTongueDepth"
    .AddAssociation LowerTongueWidth, "LowerTongueWidth"
    .AddAssociation UpperTongueDepth, "UpperTongueDepth"
    .AddAssociation UpperTongueWidth, "UpperTongueWidth"
    .AddAssociation RibThickness, "RibThickness"
End With
```

Fig. 7. The code snippet responsible for defining the design table
Rys. 7. Fragment kodu odpowiedzialny za definiowanie tablicy projektowej

odsunięciem od krawędzi zagięć. W celu uzyskania zdefiniowanego odsunięcia określono w programie równanie, które na podstawie: punktów rozmieszczonych w przestrzeni, kąta pomiędzy odcinkami oraz wartości promieni gięcia rury oblicza wartość, jaką dodaje do wartości wpisanej przez użytkownika, aby uzyskać określone odsunięcie od zagięcia rury. Dystans, jaki zostaje dodany jako wynik równania, zmienia się w zależności od wartości promienia rury i kąta pomiędzy odcinkami (rys. 10).

```
Dim NumberOfConfigurations
NumberOfConfigurations = oDesignTable.ConfigurationsNb
Dim heightsFromDesignTable() As Double
ReDim heightsFromDesignTable(2 To NumberOfConfigurations + 1)
For i = 2 To oDesignTable.ConfigurationsNb + 1
    heightsFromDesignTable(i) = CDbl(Left(oDesignTable.CellAsString(i, 1),
                                Len(oDesignTable.CellAsString(i, 1)) - 2))
Next i

Dim DistanceFiltered() As Double
For i = 2 To oDesignTable.ConfigurationsNb + 1
    If heightsFromDesignTable(i) < Distance - 20 Then
        ReDim Preserve DistanceFiltered(2 To i)
        DistanceFiltered(i) = heightsFromDesignTable(i)
    End If
Next i

oDesignTable.Configuration = findBestIndex(DistanceFiltered)
oDesignTable.CopyMode = True
```

Fig. 8. The code snippet responsible for setting the appropriate configuration in the design table
Rys. 8. Fragment kodu odpowiedzialny za ustawienie odpowiedniej konfiguracji w tabeli projektowej

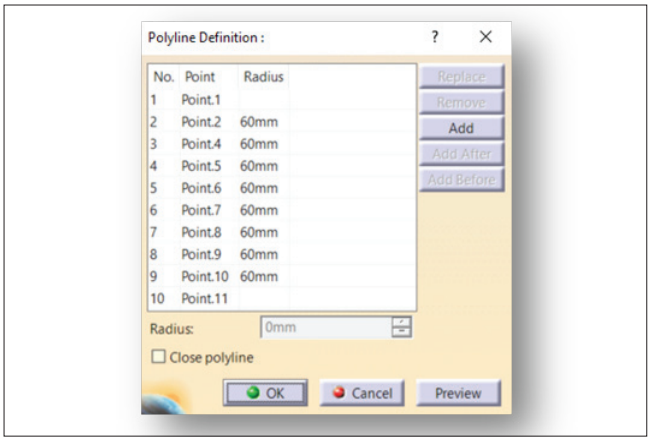

Fig. 9. Tube guiding axis as CATIA v5 "Polyline" operation
Rys. 9. Oś przewodnia rury jako efekt użycia funkcji „Polyline" systemu CATIA v5

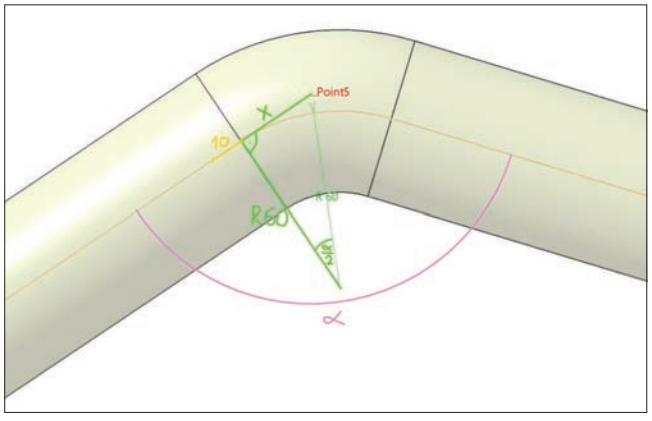

Fig. 10. Geometric derivation of the equation used in the program
Rys. 10. Geometryczne wyprowadzenie równania zastosowanego w programie

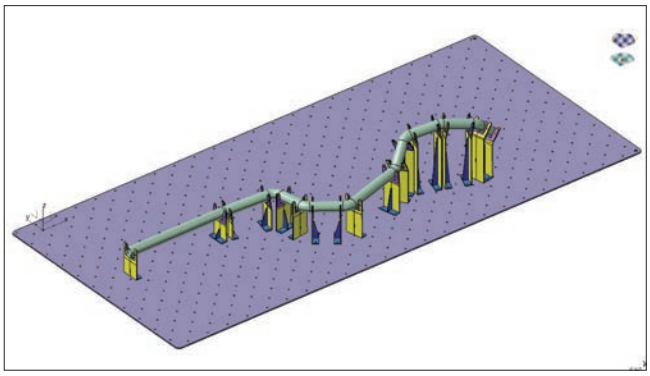

Fig. 11. Tool use for the entire guiding axis of the pipe
Rys. 11. Zastosowanie narzędzia do całej osi przewodniej rury

$$x = R \cdot Tg\left(\frac{90 - \frac{180 - \propto}{2} \cdot \pi}{180}\right)$$

Gdzie:
x – wartość dodawana do odległości wpisanej przez użytkownika,
R – promień gięcia,
$\propto$ – kąt pomiędzy odcinkami osi.

Narzędzie tworzy każdy moduł jako produkt zawierający cztery elementy składowe. Wymiary każdego z elementów można modyfikować poprzez wybór odpowiedniej konfiguracji z tablicy projektowej. Na rys. 11 przedstawiono przykład zastosowania narzędzia.

Operacja generowania osi przewodniej rury

Wybierając operację generowania osi przewodniej rury, użytkownik tworzy oś przewodnią całej rury (lub jej wybranego fragmentu), która jest potrzebna do wygenerowania modułów. Operacja ta ma zastosowanie, gdy model rury jest dostarczony bez „historii" i nie zawiera zdefiniowanych punktów gięcia (nie znamy współrzędnych punktów) oraz nie znamy wartości promieni gięcia rury.

Narzędzie tworzy oś przewodnią (za pomocą wspomnianej funkcji „Polyline" [1]) rury na wybranym jej fragmencie. Różnica w porównaniu z bezpośrednim zastosowaniem operacji polega na tym, że użytkownik nie musi znać punktów gięcia rury ani jej promieni gięcia. Wybiera z modelu 3D krawędzie gięcia (miejsca zakończenia gięcia rury), a następnie program z punktów centralnych krawędzi tworzy odcinki, których intersekcje w przestrzeni są punktami gięcia rury (rys. 12).

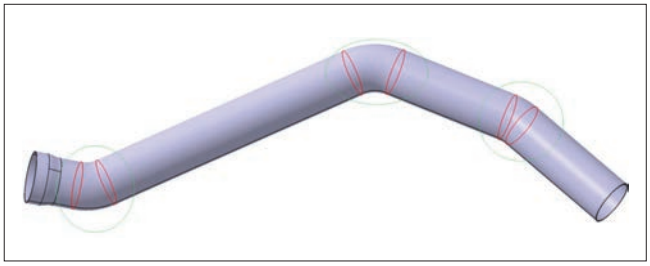

Fig. 12. User selection of bend edges
Rys. 12. Wybór krawędzi gięcia przez użytkownika

Następnie mierzone są: kąty pomiędzy utworzonymi odcinkami, odległości pomiędzy punktami centralnymi wybranych krawędzi i punktami gięcia rury. Ostatecznie program tworzy oś przewodnią rury jako wynik funkcji „Polyline" [1], przypisując jej utworzone punkty gięcia i obliczone z równania promienie gięcia (rys. 13).

$$R = x \cdot Tg\left(\frac{90 - \frac{180-\propto}{2} \cdot \pi}{180}\right)$$

Gdzie:
x – odległość pomiędzy punktem centralnym wybranej krawędzi a punktem gięcia,
R – promień gięcia,
$\propto$ – kąt pomiędzy odcinkami osi.

Operacja ta generuje w drzewie struktury modelu zbiór geometryczny, który zawiera odizolowane punkty gięcia (*pipe inputs*), co jest dobrą praktyką, ponieważ zapobiega niezamierzonym modyfikacjom współrzędnych (rys. 14).

Operacja generowania podstawy modułu

W wyniku tej operacji zostaje wygenerowana postać konstrukcyjna podstawy modułu poprzez zdefiniowanie jej grubości, promieni narożników, średnicy otworów oraz (podobnie jak w przypadku narzędzia do generowania osi przewodniej) wybór krawędzi otworów na płycie sprawdzianu przez użytkownika (rys. 15). Wygenerowany element jest w pełni parametryczny i ma wyizolowane punkty środkowe otworów, co zapobiega niezaplanowanym modyfikacjom i ułatwia zmianę kształtu podstawy poprzez modyfikację współrzędnych punktów (rys. 16).

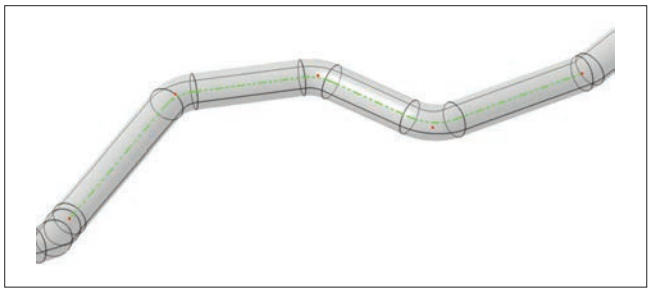

Fig. 13. Application of the pipe guiding axis generation operation
Rys. 13. Zastosowanie operacji generowania osi przewodniej rury

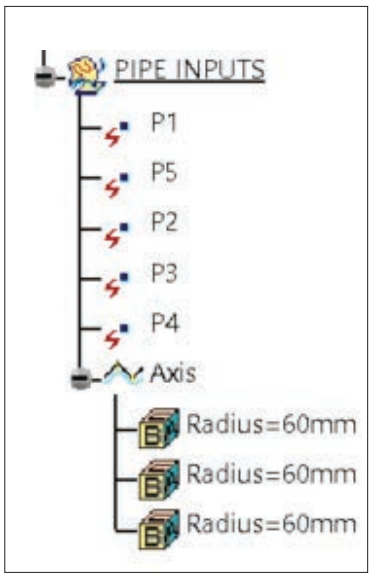

Fig. 14. Set of points created by the program
Rys. 14. Zbiór punktów utworzony przez program

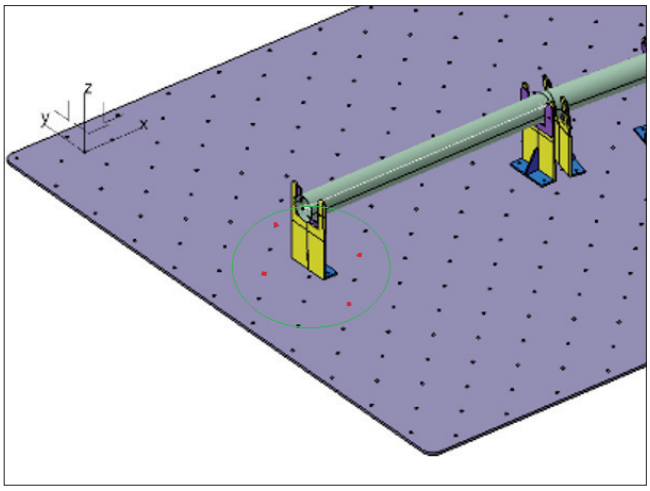

Fig. 15. User selection of hole edges on the gage plate
Rys. 15. Wybór krawędzi otworów na płycie sprawdzianu przez użytkownika

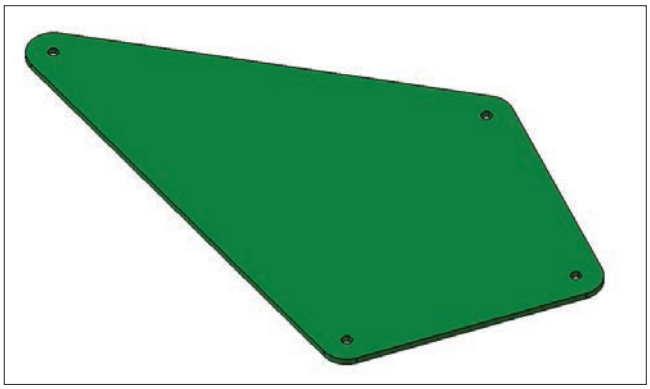

Fig. 16. Application of the base generation operation of the module
Rys. 16. Zastosowanie operacji generowania podstawy modułu

Interfejs narzędzia

W kolejnym kroku wykonano intuicyjny interfejs użytkownika zawierający ilustracje generowanych elementów. W każdej zakładce interfejsu użytkownika zawarte są pola, w których użytkownik wpisuje wymiary charakterystyczne dla elementu. W przypadku karty do generowania modułu wpisuje się wymiary, według których program dobierze element z tablicy projektowej.

Utworzone pola zostały wyposażone w funkcje umożliwiające pełną weryfikację wprowadzanych danych, co zapobiega:

- pozostawieniu pustego pola,
- wprowadzeniu wartości typu tekstowego,
- wprowadzeniu innych niepożądanych znaków,
- wprowadzeniu wartości uniemożliwiających wygenerowanie elementu (np. skrajnie duży wymiar w porównaniu z pozostałymi wymiarami) (rys. 17).

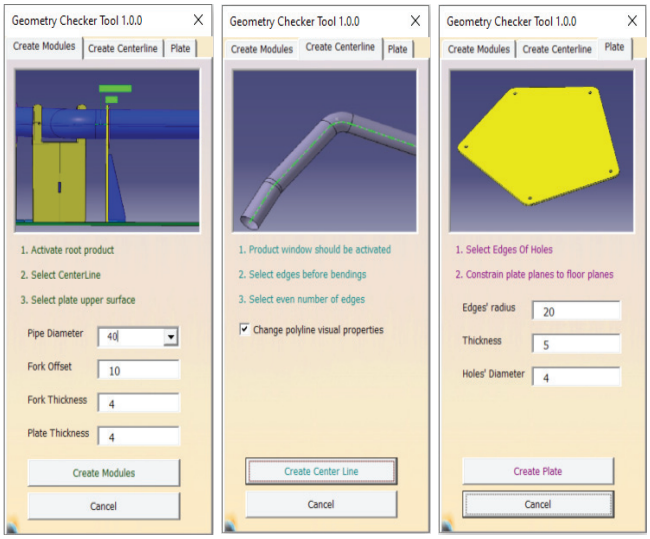

Fig. 17. User interface
Rys. 17. Interfejs użytkownika

Podsumowanie

Wykonane narzędzie programowe jest intuicyjne i łatwe w użytkowaniu. Pozwala na wygenerowanie elementów potrzebnych do wytworzenia sprawdzianu geometrii obiektów rurowych. Przyspiesza pracę konstruktora i zapewnia użycie elementów, które zostały już wcześniej wytworzone, redukując w ten sposób koszty i standaryzując zastosowane elementy. Generowanie modeli elementów zawsze następuje z użyciem wartości parametrów, które zostały zawarte w swego rodzaju bazie danych, wraz ze zdefiniowanymi relacjami pomiędzy nimi. Wykonane narzędzie spełnia zdefiniowane założenia.

LITERATURA

[1] *CATIA v5*, https://www.3ds.com/products-services/catia/ (dostęp: 01.10.2022).
[2] Skarka W., Mazurek A. „*CATIA. Podstawy modelowania i zapisu konstrukcji*". Gliwice: Wydawnictwo Helion (2005).
[3] Skarka W. „*CATIA V5. Podstawy budowy modeli autogenerujących*". Gliwice: Wydawnictwo Helion (2005).
[4] Wełyczko A. „*CATIA V5. Przykłady efektywnego zastosowania systemu w projektowaniu mechanicznym*". Gliwice: Wydawnictwo Helion (2005).
[5] Wyleżoł M. „*CATIA: Podstawy modelowania powierzchniowego i hybrydowego*". Gliwice: Wydawnictwo Helion (2003).