Comprehensive Documentation of the FourBarLinkage Class in Python

October 15, 2024

1 Introduction

The `FourBarLinkage` class in Python models a four-bar linkage mechanism, commonly used in mechanical systems. A four-bar linkage consists of four rigid bars connected by rotary joints, and depending on the bar lengths and angles, it can exhibit different motion types like crank-rocker, double-rocker, or double-crank. This class allows for the simulation of the mechanism and provides users the ability to input parameters such as link lengths, initial angles, and angular velocities.

User Inputs

Users can directly input the following parameters:

- `AB`, `BC`, `CD`, `DA`: Lengths of the four bars in the linkage.
- `alpha`, `theta`: Angles of the input and fixed links (in degrees).
- `alpha_velocity`: Angular velocity of the input link (in degrees per second).
- `coupler_position`, `coupler_offset`: Position and offset of the coupler link relative to link CD.
- `timeinterval`: The time interval for simulation updates.

These inputs allow users to experiment with various configurations of the linkage and observe how the system behaves under different conditions. Additionally, the graphical user interface (GUI) provided allows real-time interaction with the linkage, offering sliders for dynamic control over these parameters.

2 Class Attributes

The class contains attributes that define the fundamental parameters and the current state of the four-bar linkage mechanism.

2.1 Geometrical Parameters

- `AB`, `BC`, `CD`, `DA`: These represent the lengths of the four bars in the linkage (User Input).

2.2 Angles and Angular Velocity

- `alpha`: The angle of the input link (User Input).
- `theta`: The angle of the fixed link (User Input).
- `alpha_rad`, `theta_rad`: These are the `alpha` and `theta` angles converted from degrees to radians (automatically converted after input).
- `alpha_velocity`: Angular velocity of the input link (User Input).

2.3 Time Parameters and Mode Switches

- `t`: The time interval for the simulation to progress (User Input).
- `C_mode`: Determines which configuration of point C is used in the system (either `C1` or `C2`).

3 Methods of the FourBarLinkage Class

The following methods implement the core functionality of the `FourBarLinkage` class. They manage everything from initializing the system's parameters to calculating the positions of each joint and ensuring the mechanism's geometric validity.

3.1 __init__() - Constructor

- **Location:** Defined at the beginning of the class.
- **Description:** Initializes all parameters of the four-bar linkage, including link lengths, angles, angular velocities, and time intervals (User Inputs). It also converts the input angles from degrees to radians for further calculations and sets the initial state for simulation.

3.2 init_default_values() - Initialize Default Values

- **Location:** Defined immediately after the constructor.
- **Description:** Sets up default values for the system, such as the geometric validity, the motion type classification (e.g., crank-rocker), and any mode switches (e.g., `C1` or `C2`).

3.3 run() - Start Simulation

- **Location:** Defined after `init_default_values`.
- **Description:** Begins the simulation by calculating the classification values for the linkage and verifying the geometric validity of the system. It also computes the positions of all the points (A, B, C, D, and P) based on the current configuration of lengths and angles.

3.4 calculate_Classification_Value() - Calculate Classification Values

- **Location:** Called within `run`.
- **Description:** Computes the classification values T_1, T_2, and T_3, which determine the type of linkage mechanism (crank-rocker, double-rocker, etc.).

3.5 check_Parameter() - Check Parameters

- **Location:** Called within `run`.
- **Description:** Verifies that the linkage configuration satisfies the Grashof criteria and other geometric constraints, determining if the linkage can function as expected.

3.6 find_Linkage_Type() - Find Linkage Type

- **Location:** Called within `run`.
- **Description:** Determines the type of linkage mechanism (e.g., crank-rocker or double-crank) based on the computed classification values.

3.7 calculate_Point_Position() - Calculate Point Positions

- **Location:** Called within `run`.
- **Description:** Computes the positions of all the major points in the system (A, B, C, D, and P) using the current lengths and angles.

3.8 calculate_alpha_lims() - Calculate Alpha Limits

- **Location:** Called within `run`.
- **Description:** Determines the valid range for the input link's angle to ensure that the linkage forms a valid closed configuration.

3.9 calculate_C_Position() - Calculate Position of C

- **Location:** Called within `calculate_Point_Position`.
- **Description:** Determines the two possible positions for point C (C1 and C2) and selects the appropriate one based on the current configuration and mode.

3.10 calculate_P_Position() - Calculate Position of P

- **Location:** Called within `calculate_Point_Position`.
- **Description:** Computes the position of the coupler point P, using the coupler position and offset relative to link CD (User Input).

3.11 animation_alpha() - Animate Alpha Angle

- **Location:** Defined near the end of the class.
- **Description:** Updates the input angle (`alpha`) at each time step during the animation. The motion reverses when the angle reaches its limits to provide continuous movement.

3.12 switch_C2_C1() - Switch Between C1 and C2

- **Location:** Defined near the end of the class.
- **Description:** Switches between the two possible positions for point C (`C1` and `C2`). This is done based on the mechanism's state as it transitions between different configurations.

4 GUI Implementation

The `FourBarLinkage` class is integrated with a graphical user interface (GUI), allowing users to interact with the linkage system dynamically. Through the GUI, users can adjust parameters like link lengths and angles in real-time and observe the system's behavior.

4.1 init_linkage_display() - Initialize Linkage Display

- **Location:** In the `GUI` class.
- **Description:** Sets up the visual elements needed to display the linkage on the canvas, such as drawing the bars and marking the points A, B, C, D, and P.

4.2 refresh() - Refresh GUI

- **Location:** In the `GUI` class.
- **Description:** Updates the visual representation of the linkage every time a user changes the parameters, ensuring the display remains synchronized with the system's current state.

4.3 run_animation() - Run Animation

- **Location:** In the `GUI` class.
- **Description:** Runs the continuous animation of the four-bar linkage mechanism on the canvas. The points and bars update in real-time as the system moves, based on the input parameters.