### Documentation for `MachineComponent` Module

#### \*\*Module Overview\*\*

The `MachineComponent` module provides a class that represents a mechanical component within a machine. This class models various physical properties, enables simulation of mechanical behaviors, and integrates optimization algorithms for material selection.

#### \*\*Dependencies\*\*

- \*\*`numpy`\*\*: Used for numerical calculations.

- \*\*`time`\*\*: Utilized for time-based simulations.

- \*\*`threading`\*\*: Allows for concurrent execution of sensor simulations.

- \*\*`MaterialOptimizerGA`\*\*: A genetic algorithm class used for optimizing material properties based on given criteria.

### \*\*Class: MachineComponent\*\*

#### \*\*Constructor (`\_\_init\_\_`)\*\*

```python

Def \_\_init\_\_(self, name):

```

- \*\*Parameters\*\*:

- `name` (str): The name of the component.

- \*\*Description\*\*: Initializes a `MachineComponent` object with the specified name. The constructor sets default values for various physical properties, such as mass, position, velocity, forces, temperature, and material properties.

#### \*\*Attributes\*\*

- \*\*`name`\*\*: Name of the component.

- \*\*`cost`\*\*: Cost per unit (default is 0).

- \*\*`mass`\*\*: Mass of the component in kilograms (default is 0).

- \*\*`position`\*\*: 3D coordinates (x, y, z) in meters (default is (0, 0, 0)).

- \*\*`velocity`\*\*: Velocity vector (vx, vy, vz) in m/s (default is (0, 0, 0)).

- \*\*`forces`\*\*: Dictionary of forces applied to the component.

- \*\*`acceleration`\*\*: Acceleration vector (ax, ay, az) in m/s² (default is (0, 0, 0)).

- \*\*`volume`\*\*: Volume of the component in cubic meters (default is 0).

- \*\*`momentum`\*\*: Momentum vector in kg·m/s (default is (0, 0, 0)).

- \*\*`temperature`\*\*: Temperature in Celsius (default is 25.0).

- \*\*`pressure`\*\*: Pressure in kPa (default is 101.3).

- \*\*`vibration\_intensity`\*\*: Intensity of vibrations (default is 0.0).

- \*\*`temperature\_sensor`, `pressure\_sensor`, `vibration\_sensor`, `force\_sensor`, `rotation\_speed\_sensor`, `current\_sensor`\*\*: Lists for recording sensor data.

- \*\*`surface\_area`\*\*: Surface area in square meters (default is 0).

- \*\*`stress`\*\*: Stress value (default is 0).

- \*\*`strain`\*\*: Strain value (default is 0).

- \*\*`wear`\*\*: Wear value (default is 0).

- \*\*`material`\*\*: Material of the component (default is None).

- \*\*`friction`\*\*: Friction coefficient (default is 0).

- \*\*`cycles`\*\*: Number of cycles for fatigue analysis (default is 1e6).

- \*\*`material\_properties`\*\*: Dictionary containing material properties such as density, resistance, wear, and friction.

- \*\*`simulation\_active`\*\*: Boolean flag indicating whether sensor simulation is active.

- \*\*`mesh`\*\*: Mesh data for visualization (default is None).

#### \*\*Methods\*\*

1. \*\*`set\_mesh(self, mesh)`\*\*

- \*\*Parameters\*\*: `mesh` (any type) – Mesh data to be associated with the component.

- \*\*Description\*\*: Assigns mesh data for visualization purposes.

2. \*\*`simulate\_temperature(self)`\*\*

- \*\*Description\*\*: Simulates temperature changes in the component over time, influenced by friction and random noise. Appends simulated data to `temperature\_sensor`.

3. \*\*`simulate\_pressure(self)`\*\*

- \*\*Description\*\*: Simulates pressure changes based on applied forces and surface area. Appends simulated data to `pressure\_sensor`.

4. \*\*`simulate\_vibration(self)`\*\*

- \*\*Description\*\*: Simulates vibration intensity based on applied forces and a damping coefficient. Appends simulated data to `vibration\_sensor`.

5. \*\*`start\_sensors(self)`\*\*

- \*\*Description\*\*: Activates sensor simulation by starting threads for temperature, pressure, and vibration simulations.

6. \*\*`stop\_sensors(self)`\*\*

- \*\*Description\*\*: Deactivates sensor simulation by stopping the active threads.

7. \*\*`calculate\_net\_force(self)`\*\*

- \*\*Returns\*\*: `net\_force` (list of floats) – The resultant force vector.

- \*\*Description\*\*: Computes the sum of all forces acting on the component.

8. \*\*`simulate\_vibration\_response(self, force, damping\_coefficient)`\*\*

- \*\*Parameters\*\*:

- `force` (float): The applied force.

- `damping\_coefficient` (float): The damping coefficient.

- \*\*Returns\*\*: `amplitude` (float) – The vibration amplitude.

- \*\*Description\*\*: Simulates the vibration response of the component based on the applied force and damping coefficient.

9. \*\*`calculate\_natural\_frequency(self)`\*\*

- \*\*Returns\*\*: `natural\_frequency` (float) – The natural frequency of the component.

- \*\*Raises\*\*: `ValueError` if the mass is zero.

- \*\*Description\*\*: Calculates the natural frequency of the component based on its mass and a predefined stiffness value.

10. \*\*`apply\_force(self, name, force\_vector, contact\_point, duration)`\*\*

- \*\*Parameters\*\*:

- `name` (str): Unique identifier for the force.

- `force\_vector` (tuple of floats): Force vector (fx, fy, fz) in Newtons.

- `contact\_point` (tuple of floats): Point of force application (x, y, z) in meters.

- `duration` (int): Duration for which the force is applied.

- \*\*Description\*\*: Applies a specified force to the component for a given duration.

11. \*\*`update\_position(self)`\*\*

- \*\*Description\*\*: Updates the component’s position based on its velocity.

12. \*\*`update\_velocity(self)`\*\*

- \*\*Description\*\*: Updates the component’s velocity based on its acceleration.

13. \*\*`update(self)`\*\*

- \*\*Description\*\*: Updates both the position and velocity of the component.

14. \*\*`calculate\_momentum(self)`\*\*

- \*\*Description\*\*: Calculates and updates the momentum of the component based on its mass and velocity.

15. \*\*`calculate\_friction\_force(self, normal\_force)`\*\*

- \*\*Parameters\*\*: `normal\_force` (float) – The normal force acting on the component.

- \*\*Returns\*\*: `friction\_force` (float) – The friction force.

- \*\*Raises\*\*: `ValueError` if the material is not found in the material library.

- \*\*Description\*\*: Calculates the friction force based on the normal force and the material’s friction coefficient.

16. \*\*`detect\_collision(self, other\_component)`\*\*

- \*\*Parameters\*\*: `other\_component` (MachineComponent) – Another component to check for collisions.

- \*\*Returns\*\*: `collision\_detected` (bool) – Whether a collision was detected.

- \*\*Description\*\*: Detects collisions with another component using simple bounding box checks.

17. \*\*`resolve\_collision(self, other\_component)`\*\*

- \*\*Parameters\*\*: `other\_component` (MachineComponent) – Another component with which a collision might occur.

- \*\*Description\*\*: Resolves a detected collision by exchanging velocities along the collision normal.

18. \*\*`update\_thermal\_expansion(self)`\*\*

- \*\*Description\*\*: Updates the component’s geometry and volume based on thermal expansion due to temperature changes.

19. \*\*`calculate\_stress\_due\_to\_thermal(self)`\*\*

- \*\*Description\*\*: Calculates additional stress in the component due to thermal expansion.

20. \*\*`update\_wear(self, time\_interval)`\*\*

- \*\*Parameters\*\*: `time\_interval` (float) – The time interval over which to calculate wear.

- \*\*Description\*\*: Updates the component’s wear based on stress, strain, and material properties.

21. \*\*`calculate\_stress(self, force\_magnitud)`\*\*

- \*\*Parameters\*\*: `force\_magnitud` (float) – The magnitude of the applied force.

- \*\*Raises\*\*: `ValueError` if the surface area is zero.

- \*\*Description\*\*: Calculates the stress on the component based on the applied force and surface area.

22. \*\*`update\_temperature(self, external\_temperature, internal\_heat\_generation)`\*\*

- \*\*Parameters\*\*:

- `external\_temperature` (float): The external temperature.

- `internal\_heat\_generation` (float): Internal heat generated.

- \*\*Description\*\*: Dynamically updates the component’s temperature.

23. \*\*`update\_dynamic\_parameters(self, force\_magnitud, area, external\_temperature, internal\_heat\_generation, time\_interval)`\*\*

- \*\*Parameters\*\*:

- `force\_magnitud` (float): Force magnitude.

- `area` (float): Surface area.

- `external\_temperature` (float): External temperature.

- `internal\_heat\_generation` (float): Internal heat generation.

- `time\_interval` (float): Time interval.

- \*\*Description\*\*: Updates dynamic parameters such as stress, temperature, and wear over time.

24. \*\*`calculate\_strain(self)`\*\*

- \*\*Description\*\*: Calculates the strain on the component based on the stress and Young’s modulus.

25. \*\*`calculate\_velocity\_after\_time(self, time\_interval)`\*\*

- \*\*Parameters\*\*: `time\_interval` (float) – The time interval over which to calculate velocity.

- \*\*Returns\*\*: `velocity` (tuple of floats) –

The updated velocity.

- \*\*Raises\*\*: `ValueError` if the mass is zero.

- \*\*Description\*\*: Calculates the component’s velocity after a given time interval based on net force and mass.

26. \*\*`calculate\_stiffness(self)`\*\*

- \*\*Returns\*\*: `stiffness` (float) – The stiffness of the component.

- \*\*Raises\*\*: `ValueError` if material or volume is not set.

- \*\*Description\*\*: Calculates the stiffness of the component based on material properties, surface area, and length.

27. \*\*`calculate\_kinetic\_energy(self)`\*\*

- \*\*Returns\*\*: `kinetic\_energy` (float) – The kinetic energy of the component.

- \*\*Description\*\*: Calculates the kinetic energy based on mass and velocity.

28. \*\*`calculate\_potential\_energy(self, gravity, reference\_height)`\*\*

- \*\*Parameters\*\*:

- `gravity` (float): Gravitational acceleration.

- `reference\_height` (float): The reference height.

- \*\*Returns\*\*: `potential\_energy` (float) – The potential energy of the component.

- \*\*Description\*\*: Calculates the potential energy based on mass, gravity, and position.

29. \*\*`derive\_target\_properties(self)`\*\*

- \*\*Returns\*\*: `target\_properties` (dict) – Derived target properties for material optimization.

- \*\*Description\*\*: Derives target material properties based on the component’s dynamic attributes, including density, resistance, thermal expansion, wear, and fatigue limit.

30. \*\*`optimize\_material\_ga(self, weight\_factors)`\*\*

- \*\*Parameters\*\*: `weight\_factors` (dict) – Weight factors for material optimization.

- \*\*Returns\*\*: `optimized\_material` (dict) – The optimized material properties.

- \*\*Description\*\*: Uses a genetic algorithm to optimize the material properties of the component based on target properties.

31. \*\*`update\_temperature\_sensor(self)`\*\*

- \*\*Description\*\*: Updates the temperature sensor data based on simulated friction-induced heat and random noise.

32. \*\*`update\_pressure\_sensor(self)`\*\*

- \*\*Description\*\*: Updates the pressure sensor data based on net force and surface area.

33. \*\*`update\_vibration\_sensor(self)`\*\*

- \*\*Description\*\*: Updates the vibration sensor data based on simulated vibration response and noise.

34. \*\*`simulate(self, time\_units)`\*\*

- \*\*Parameters\*\*: `time\_units` (int) – Number of time units to simulate.

- \*\*Description\*\*: Simulates the component’s behavior over a specified number of time units, updating forces, velocity, position, and sensor data in real-time.

---

### \*\*Material Library (`material\_lib`)\*\*

* \*\*Description\*\*: A dictionary containing material properties such as density, resistance, wear, and friction for various materials (e.g., steel, copper, aluminium).

---

This documentation provides an overview of the `MachineComponent` class and its methods, facilitating easier understanding and use of the module for simulation and material optimization in mechanical design.