Proof[theorem]

Cone Formation from Circle Folding: A Comprehensive Analysis

October 20, 2024

Abstract

This paper explores the mathematical details behind the geometric transformation of folding a circle into a cone. It includes detailed theorems, proofs, and Python implementations for visualization, providing a thorough analysis of the transformation process.

1 Introduction

Folding a circle into a cone involves removing a sector with angle θ from a circle of radius r. This transformation is not only of interest in geometry but also in practical applications like material science and design.

2 Theoretical Framework

Theorem 1. *When a sector of angle θ is removed from a circle of radius r, and the remaining shape is folded into a cone, the cone's base radius r_1 and height η are given by:*

$$r_1 = r - \frac{r\theta}{2\pi}$$

$$\eta = \sqrt{r^2 - r_1^2}$$

Proof. Starting from the circumference relationship:

$$\theta r = 2\pi r - 2\pi r_1$$

Solving for r_1:

$$r_1 = r - \frac{r\theta}{2\pi}$$

Using Pythagorean theorem for height η:

$$\eta = \sqrt{r^2 - r_1^2}$$

Given $r_1 = r \sin \beta$, where β is the angle formed by the slant height and the base of the cone:

$$\eta = r \sin \beta$$

□

Lemma 1. *The height η of the cone in terms of r and θ:*

$$\eta = \frac{\sqrt{4\pi r^2\theta - r^2\theta^2}}{2\pi}$$

Proof. From the equation for η:

$$\eta = \sqrt{r^2 - \left(r - \frac{r\theta}{2\pi}\right)^2}$$

Simplifying inside the square root:

$$\eta = \frac{\sqrt{4\pi r^2\theta - r^2\theta^2}}{2\pi}$$

□

Lemma 2. *θ can be solved for in terms of r and η:*

$$\theta = \frac{2\pi(r^2 \pm \sqrt{r^4 - r^2\eta^2})}{r^2}$$

Proof. Starting from the height equation in terms of θ and solving for θ:

$$\eta = \frac{\sqrt{4\pi r^2\theta - r^2\theta^2}}{2\pi}$$

Squaring both sides, rearranging terms, and solving the quadratic in θ gives:

$$\theta = \frac{2\pi(r^2 \pm \sqrt{r^4 - r^2\eta^2})}{r^2}$$

□

3 Python Implementation for Visualization

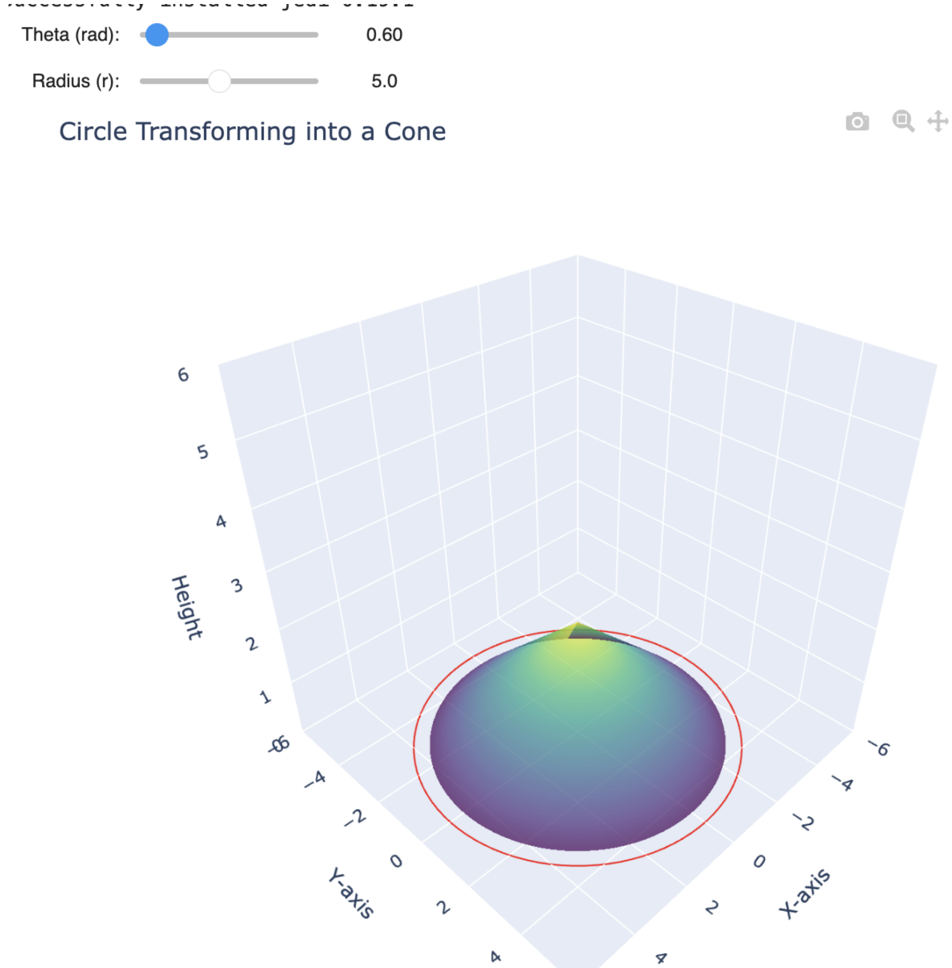
Here's the Python code implementing the cone visualization:

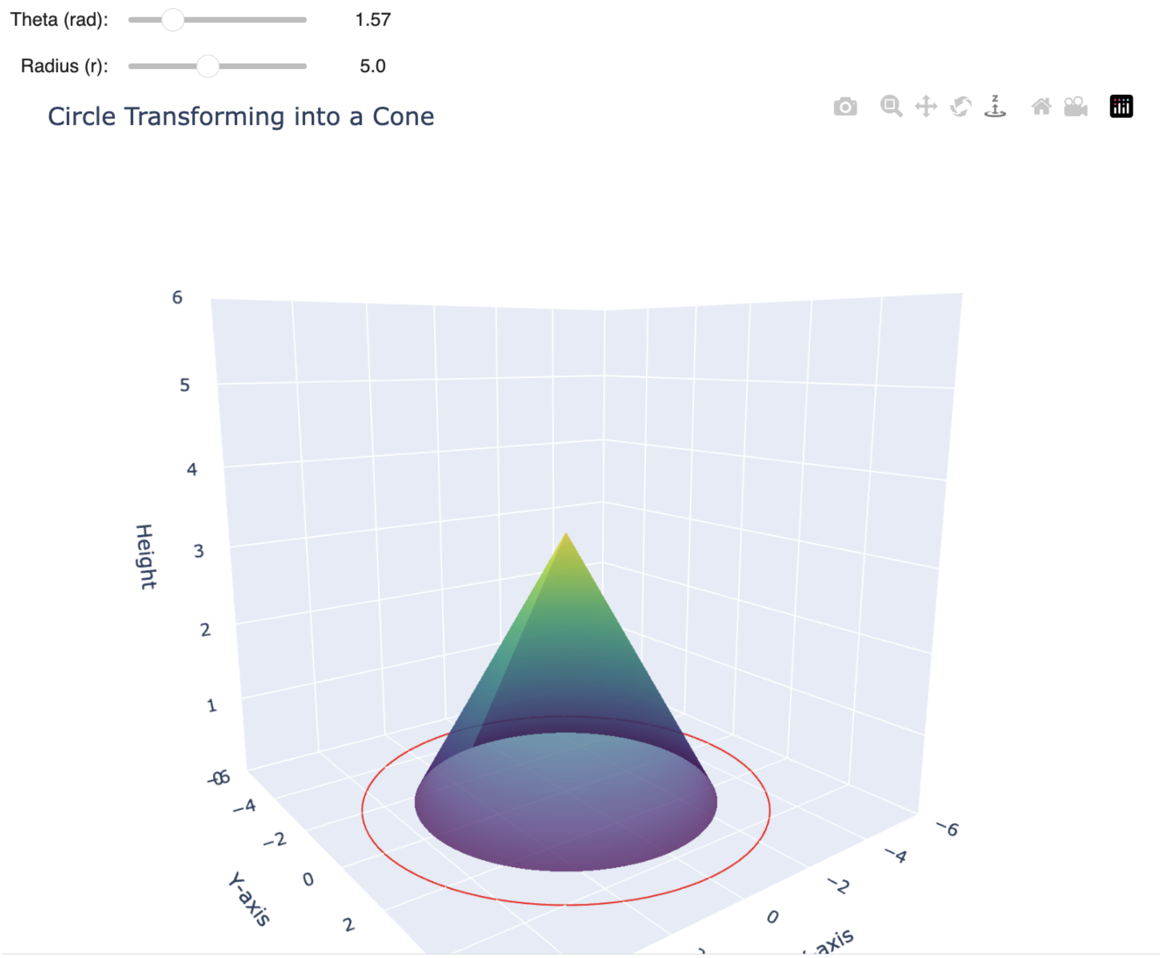
```
!pip install plotly ipywidgets

# Enable the custom widget manager for Google Colab
from google.colab import output
output.enable_custom_widget_manager()

```

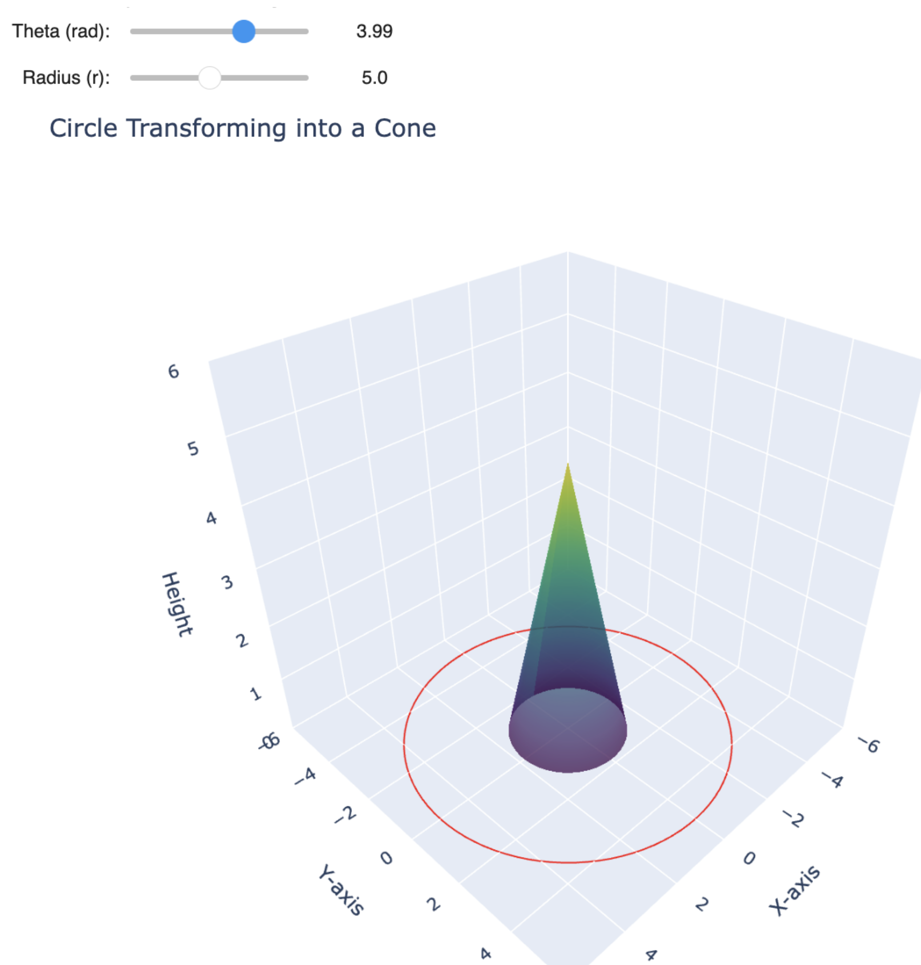
```python
import numpy as np
import plotly.graph_objects as go
import ipywidgets as widgets
from ipywidgets import interact, fixed

def generate_cone_and_circle(theta, r):
    # Calculate parameters using your specified equation
    r1 = r - (r * theta) / (2 * np.pi)
    eta = np.sqrt(4 * np.pi * r**2 * theta - r**2 * theta**2) / (2 * np.pi)

    # Handle near-zero or negative eta values
    if eta <= 0:
        eta = 1e-6  # Small value to prevent issues

    # Generate data for the cone
    n_theta = 100
    n_z = 50
    theta_cone = np.linspace(0, 2 * np.pi, n_theta)
    z = np.linspace(0, eta, n_z)
    theta_grid, z_grid = np.meshgrid(theta_cone, z)
    r_grid = r1 * (eta - z_grid) / eta
    X_cone = r_grid * np.cos(theta_grid)
    Y_cone = r_grid * np.sin(theta_grid)
    Z_cone = z_grid

    # Generate data for the original circle
    theta_circle = np.linspace(0, 2 * np.pi, 100)
    X_circle = r * np.cos(theta_circle)
    Y_circle = r * np.sin(theta_circle)
    Z_circle = np.zeros_like(X_circle)  # Circle lies in x-y plane at z=0

    return X_cone, Y_cone, Z_cone, X_circle, Y_circle, Z_circle

def update_plot(theta, r):
    X_cone, Y_cone, Z_cone, X_circle, Y_circle, Z_circle = generate_cone_and_circle(theta, r)

    # Create the cone surface
    cone_surface = go.Surface(
        x=X_cone, y=Y_cone, z=Z_cone,
        colorscale='Viridis',
        opacity=0.7,
```

```
        showscale=False,
        name='Cone'
    )

    # Create the circle trace
    circle_trace = go.Scatter3d(
        x=X_circle, y=Y_circle, z=Z_circle,
        mode='lines',
        line=dict(color='red', width=2),
        name='Original Circle'
    )

    # Layout for the scene
    layout = go.Layout(
        title='Circle Transforming into a Cone',
        scene=dict(
            xaxis=dict(title='X-axis', range=[-r-1, r+1]),
            yaxis=dict(title='Y-axis', range=[-r-1, r+1]),
            zaxis=dict(title='Height', range=[0, r+1]),
            aspectmode='cube'
        ),
        autosize=False,
        width=800,
        height=600,
        margin=dict(l=0, r=0, t=50, b=0),
    )

    # Create figure with both the circle and the cone
    fig = go.Figure(data=[cone_surface, circle_trace], layout=layout)
    fig.show()

# Sliders for interactive control
theta_slider = widgets.FloatSlider(
    value=np.pi/2,
    min=0.01,
    max=2 * np.pi - 0.01,
    step=np.pi / 180,
    description='Theta (rad):',
    continuous_update=False,
    readout_format='.2f',
)
```

```
radius_slider = widgets.FloatSlider(
    value=5,
    min=1,
    max=10,
    step=0.1,
    description='Radius␣(r):',
    continuous_update=False,
    readout_format='.1f',
)

# Interactive visualization
interact(update_plot, theta=theta_slider, r=
   radius_slider);
```




Theta (rad):
1.57
Radius (r):
5.0
Circle Transforming into a Cone
Height
Y-axis
6
5
4
3
2
1
−4
−2
0
2
−6
−4
−2
0



4 Conclusion

This document has detailed the mathematical intricacies involved in transforming a circle into a cone through folding. Through theorems, lemmas, and Python visualizations, we've explored how the dimensions of the cone relate to the original circle's properties. This exploration not only enhances our understanding of geometric transformations but also provides tools for practical applications in design and education.

References

[1] Parker Emmerson, *The Cone of Perception*, Zenodo, 2023. https://doi.org/10.5281/zenodo.7710313

[2] Parker Emmerson, *The Sphere of Realization: The Mathematical Path of Harmonious Balance*, Zenodo, 2023. https://doi.org/10.5281/zenodo.10202331

[3] Parker Emmerson, *Phenomenological Velocity*, Zenodo, 2023. https://doi.org/10.5281/zenodo.8433050

[4] Parker Emmerson, *Folding a Circle into A Cone - Parameterization Mathematica Notebook*, Zenodo, 2023. https://doi.org/10.5281/zenodo.10005143

[5] Dharmandar, *[Title Not Provided]*, Zenodo, 2023. https://doi.org/10.5281/zenodo.10005143