```
from unittest.mock import MagicMock, patch

import pytest

from swarm_models.zeroscope import ZeroscopeTTV



@patch("swarms.models.zeroscope.DiffusionPipeline")
@patch("swarms.models.zeroscope.DPMSolverMultistepScheduler")
def test_zeroscope_ttv_init(mock_scheduler, mock_pipeline):
    zeroscope = ZeroscopeTTV()
    mock_pipeline.from_pretrained.assert_called_once()
    mock_scheduler.assert_called_once()
    assert zeroscope.model_name == "cerspense/zeroscope_v2_576w"
    assert zeroscope.chunk_size == 1
    assert zeroscope.dim == 1
    assert zeroscope.num_inference_steps == 40
    assert zeroscope.height == 320
    assert zeroscope.width == 576
    assert zeroscope.num_frames == 36



@patch("swarms.models.zeroscope.DiffusionPipeline")
@patch("swarms.models.zeroscope.DPMSolverMultistepScheduler")
def test_zeroscope_ttv_forward(mock_scheduler, mock_pipeline):
    zeroscope = ZeroscopeTTV()
```

```
    mock_pipeline_instance = MagicMock()
    mock_pipeline.from_pretrained.return_value = (
        mock_pipeline_instance
    )
    mock_pipeline_instance.return_value = MagicMock(
        frames="Generated frames"
    )
    mock_pipeline_instance.enable_vae_slicing.assert_called_once()
    mock_pipeline_instance.enable_forward_chunking.assert_called_once_with(
        chunk_size=1, dim=1
    )
    result = zeroscope.forward("Test task")
    assert result == "Generated frames"
    mock_pipeline_instance.assert_called_once_with(
        "Test task",
        num_inference_steps=40,
        height=320,
        width=576,
        num_frames=36,
    )



@patch("swarms.models.zeroscope.DiffusionPipeline")
@patch("swarms.models.zeroscope.DPMSolverMultistepScheduler")
def test_zeroscope_ttv_forward_error(mock_scheduler, mock_pipeline):
    zeroscope = ZeroscopeTTV()
```

```
    mock_pipeline_instance = MagicMock()
    mock_pipeline.from_pretrained.return_value = (
        mock_pipeline_instance
    )
    mock_pipeline_instance.return_value = MagicMock(
        frames="Generated frames"
    )
    mock_pipeline_instance.side_effect = Exception("Test error")
    with pytest.raises(Exception, match="Test error"):
        zeroscope.forward("Test task")


@patch("swarms.models.zeroscope.DiffusionPipeline")
@patch("swarms.models.zeroscope.DPMSolverMultistepScheduler")
def test_zeroscope_ttv_call(mock_scheduler, mock_pipeline):
    zeroscope = ZeroscopeTTV()
    mock_pipeline_instance = MagicMock()
    mock_pipeline.from_pretrained.return_value = (
        mock_pipeline_instance
    )
    mock_pipeline_instance.return_value = MagicMock(
        frames="Generated frames"
    )
    result = zeroscope.__call__("Test task")
    assert result == "Generated frames"
    mock_pipeline_instance.assert_called_once_with(
```

```
        "Test task",
        num_inference_steps=40,
        height=320,
        width=576,
        num_frames=36,
    )


@patch("swarms.models.zeroscope.DiffusionPipeline")
@patch("swarms.models.zeroscope.DPMSolverMultistepScheduler")
def test_zeroscope_ttv_call_error(mock_scheduler, mock_pipeline):
    zeroscope = ZeroscopeTTV()
    mock_pipeline_instance = MagicMock()
    mock_pipeline.from_pretrained.return_value = (
        mock_pipeline_instance
    )
    mock_pipeline_instance.return_value = MagicMock(
        frames="Generated frames"
    )
    mock_pipeline_instance.side_effect = Exception("Test error")
    with pytest.raises(Exception, match="Test error"):
        zeroscope.__call__("Test task")


@patch("swarms.models.zeroscope.DiffusionPipeline")
@patch("swarms.models.zeroscope.DPMSolverMultistepScheduler")
```

```
def test_zeroscope_ttv_save_video_path(mock_scheduler, mock_pipeline):
    zeroscope = ZeroscopeTTV()
    mock_pipeline_instance = MagicMock()
    mock_pipeline.from_pretrained.return_value = (
        mock_pipeline_instance
    )
    mock_pipeline_instance.return_value = MagicMock(
        frames="Generated frames"
    )
    result = zeroscope.save_video_path("Test video path")
    assert result == "Test video path"
    mock_pipeline_instance.assert_called_once_with(
        "Test video path",
        num_inference_steps=40,
        height=320,
        width=576,
        num_frames=36,
    )
```