Posture control Documentation

version

2025, -

■■■■ 25, 2025

Contents

Posture control documentation

Add your content using `reStructuredText` syntax. See the reStructuredText documentation for details.

posture_control

Core package

Subpackages

Core.sensors package

Submodules

Core.sensors.Camera module

class `Core.sensors.Camera.`**`Camera`**(`camera_index=0`)

Bases: **`Sensor`**

Camera sensor class for posture analysis using MediaPipe.

This class handles video capture, pose estimation, and posture analysis. Implements the base Sensor interface for compatibility with the posture monitoring system.

Attributes: cap (cv2.VideoCapture): OpenCV VideoCapture object for accessing the camera. pose (mp.solutions.Pose): MediaPipe Pose solution instance.

`get_data`()

Capture and process a frame for posture analysis.

Returns:

Tuple containing:

- frame (np.ndarray | None): Captured video frame (BGR format),
  or None if capture fails.
- points (Points | None): Detected landmark points, or None if
  not detected. - posture_status (str | None): Analysis result, or None if no landmarks are present.

`start`()

Start the camera capture.

Opens the camera if it’s not already open.

`stop`()

Stop the camera capture and release resources.

Core.sensors.Sensor module

class `Core.sensors.Sensor.`**`Sensor`**

Bases: **`ABC`**

Abstract base class for sensor interfaces.

Defines the mandatory interface that all sensor classes must implement. This ensures consistent behavior across different sensor types.

abstract **`get_data`**()

Get data from the sensor.

Returns:

Any: Sensor data in a format specific to the implementation.

Raises:

NotImplementedError: If the method is not overridden in a subclass.

abstract **`start`** **()**
Start the sensor operation.
This method must be implemented to initialize and begin sensor data acquisition.

abstract **`stop`** **()**
Stop the sensor operation.
This method must be implemented to properly shutdown and release sensor resources.

Module contents

Submodules

Core.Points module

class `Core.Points.`**`Points`**`(landmarks)`
Bases: **`object`**
A wrapper class for landmark points to enable indexed access.
This class provides a simple interface to access individual landmark points from a collection of landmarks using index notation.

Core.nose_shoulder_ratio_func module

`Core.nose_shoulder_ratio_func.`**`nose_shoulder_ratio`**`(points)`
Calculate the ratio of nose-to-shoulder distance to shoulder width.

Args:

points (list): List of landmark points (expected to contain shoulder

and nose points).

Returns:

float or None: Ratio of vertical nose-to-shoulder distance to shoulder

width,or None if calculation fails due to missing landmarks or invalid input.

Core.posture_analyzer module

`Core.posture_analyzer.`**`analyze_posture`**`(points, w, h, nose_to_shoulder_ratio=0.33)`
Analyze posture based on shoulder landmarks and nose-to-shoulder ratio.

Args:

points (list): List of landmark points (expected to contain shoulder points). w (int): Width of the frame (used for normalization). h (int): Height of the frame (used for normalization). nose_to_shoulder_ratio (float, optional): Expected ratio for upright posture.

Defaults to 0.33 if not provided.

Returns:

dict: Dictionary with posture analysis result. Contains:

- “status” (str): “good”, “bad”, or “error”.
- “message” (str): Detailed description of the posture status.

Module contents

main module

class main.**CameraApp**(root, camera)
Bases: **object**

capture_reference()
Captures and stores the reference posture position.
This method is typically called during calibration. It captures the current body landmarks (e.g., nose and shoulders) and computes a reference ratio used for future posture analysis.

check_posture_problems(result, no_person_detected)
Monitors for posture issues and triggers notifications

Args:

result (dict): Dictionary containing posture analysis results no_person_detected (bool): Boolean indicating if person is detected

clear_notification()
Clears any active notification messages

show_notification(message)
Displays a notification message with auto-clear

Args:

message (text): Text message to display

start_analysis()
Begins continuous posture analysis

start_calibration()
Initiates the posture calibration sequence.
This method starts the calibration process, during which the user is expected to sit in a correct, upright posture. The system captures data to establish a reference ratio for future posture analysis.

stop_camera()
Stops camera capture and resets application state

update_frame()
Main video processing loop.

Index

Python Module Index

c

m