Report: Play Tic-Tac-Toe Using Your Face as a Game Controller

A Trace of Moves in that Game

```
| | | |
| | | |
| | | |
Turn: X
Player X took position (0, 2).
| | |X|
| | | |
| | | |
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
Emotion detected as neutral (row 0). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to
continue.
reference:
col 0 is neutral.
col 1 is happy.
col 2 is surprise.
Emotion detected as neutral (col 0). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to
continue.
Player O took position (0, 0).
|O| |X|
| | | |
| | | |
Turn: X
Player X took position (0, 1).
|O|X|X|
| | | |
| | | |
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
Emotion detected as surprise (row 2). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to
continue.
reference:
col 0 is neutral.
col 1 is happy.
col 2 is surprise.
Emotion detected as surprise (col 2). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to
continue.
Player O took position (2, 2).
```

```
|O|X|X|
| | | |
| | |O|
Turn: X
Player X took position (2, 1).
|O|X|X|
| | | |
| |X|O|
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
Emotion detected as neutral (row 0). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to
continue.
reference:
col 0 is neutral.
col 1 is happy.
col 2 is surprise.
Emotion detected as neutral (col 0). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to
continue.
Position (0, 0) is already taken.
|O|X|X|
| | | |
| |X|O|
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
Emotion detected as neutral (row 0). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to
continue.
reference:
col 0 is neutral.
col 1 is happy.
col 2 is surprise.
Emotion detected as surprise (col 2). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to
continue.
Position (0, 2) is already taken.
|O|X|X|
| | | |
| |X|O|
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
Emotion detected as happy (row 1). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to continue.
reference:
col 0 is neutral.
col 1 is happy.
col 2 is surprise.
Emotion detected as happy (col 1). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to continue.
Player O took position (1, 1).
|O|X|X|
| |O| |
| |X|O|
Player O has won!
```

How well did your interface work?
In our opinion, the interface seemed to work relatively well, and had working and accurate functionality in the following areas:

1. Recognition Accuracy:
    a. It successfully detected and classified all three facial expressions (neutral, happy, surprise)
    b. It provided reliable mapping between expressions and board positions.
    c. There was no need to use 'text' for the text input, since the code appeared to correctly understand and implement the intended expression

2. Technical Performance:
    a. The camera integration worked fine so there were no technical issues
    b. The image processing and emotion detection had fast response times, but could be a little quicker with regard to capturing the image, understanding and implementing the position, and allowing the user to move onto the next
    c. The error handling functioned as intended when invalid positions were selected

Did it recognize your facial expressions with the same accuracy as it achieved against the test set? If not, why not?
The real-world performance of using our facial expressions did appear better than the 73.62% test accuracy because the usage scenario was more controlled and constrained than the conditions represented in the test set, but it was also somewhat dependent on the user since one of the partners had an easier time than the other.

Real-World Facial Expression Performance:

- The model successfully recognized facial expressions in every attempt
- There were no apparent misclassifications that occurred during the game

The relatively strong performance during gameplay could be attributed to several factors that made real-world recognition potentially easier than test set conditions, in the case of the specific game that was traced and a few other similar ones. This includes that there was a controlled environment (user was using somewhat more exaggerated expressions and positioning, and there was good lighting); there was immediate feedback from the interface since it recognized emotions immediately, and there was a 'text' input option if recognition failed (which was fortunately not needed); and there was a limited expression set with the user consciously making one of the three specific emotions.

The Lines of Code Added in the _get_emotion Function to Use the Model for Emotion Detection

```
def _get_emotion(self, img) -> int:
    # Resize input image to match model's expected size
    resized_img = cv2.resize(img, image_size)

    # Convert grayscale to RGB
    rgb_img = cv2.cvtColor(resized_img, cv2.COLOR_GRAY2RGB)

    # Add batch dimension and normalize pixel values
    input_img = np.expand_dims(rgb_img, axis=0)

    # Load trained model if not already loaded
    if not hasattr(self, 'model'):
        self.model = load_model('part_six_basic_model.keras')

    # Make prediction
    prediction = self.model.predict(input_img, verbose=0)

    # Get the index of the highest probability class (0 for neutral, 1 for happy, 2 for surprise)
    emotion_index = int(np.argmax(prediction[0]))

    return emotion_index
```