Motion Prediction of a Ping Pong Ball

from
Eileen Salhofer and Felix Warmer

Implementation and execution notes

The application was implemented for Python 3.6. Additional package requirements are NumPy (implemented with Version 1.13.3) and OpenCV (implemented with Version 3.3.0.10).
Minimum call for the application is “python main.py -f <Video>. Use -h to see available parameter options.

Goal and general idea

The Goal of the application is a continuous motion prediction of a ping-pong ball in a ping-pong game. To be more precise, after taking a video from a ping-pong game with a static camera position, we try to visualize the future path of the ping-pong ball.

Application setup and workflow

The Application is currently split into the four files main.py, ball.py, table.py and helper.py, where:

- main.py is the starting point of the application and contains the main program flow logic.
- ball.py contains the Ball class and ball related functions.
- table.py contains the Table class and table related functions.
- helper.py contains functions that are general useful but don’t fit into the Ball or Table classes.

The workflow is as follows:

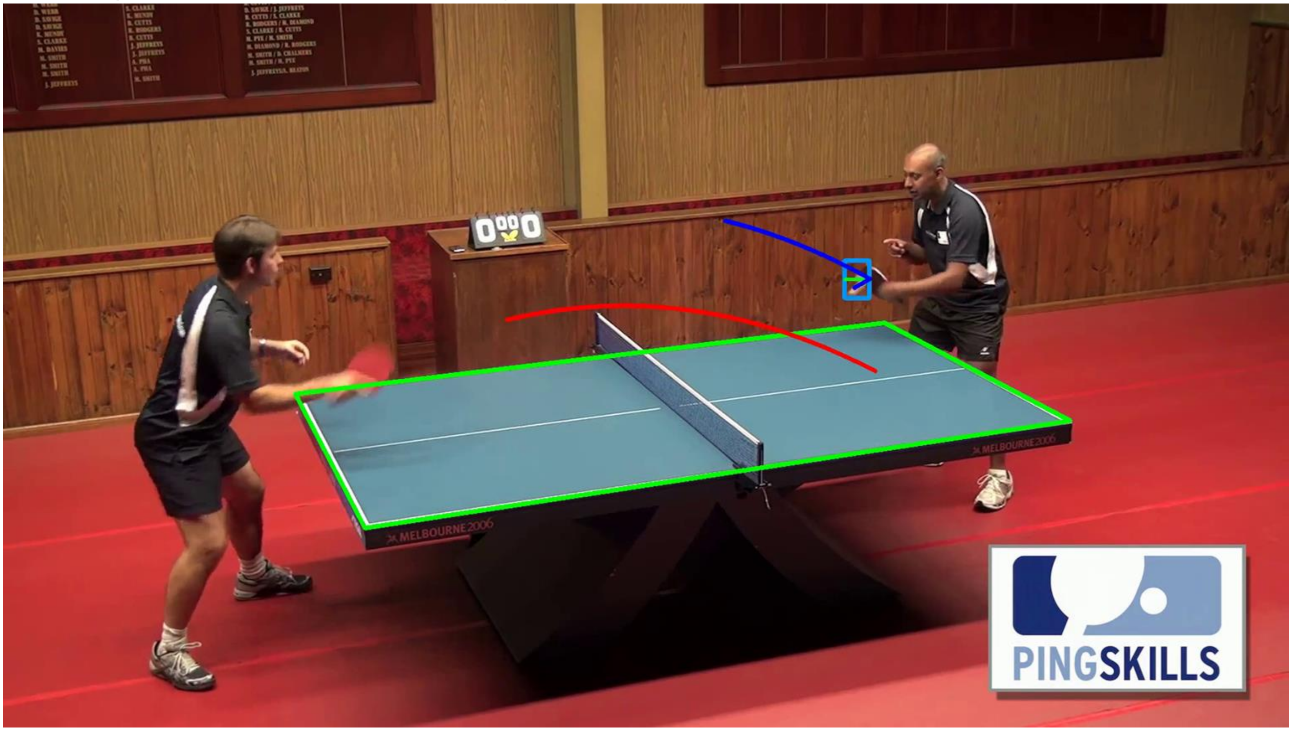
1. Finding the table (implemented in get_table function of the Table Class):
    - Use HSV color thresholding with the color of the table. Assumption is that the table is the biggest object with a specific color.
    - Use HSV color thresholding with white to get the white stripe on the outside of the table top.
    - Use corner detection to find the four corners of the table.
    - The corner of the table define the border of our region of interest for searching the ball
2. Finding the ball (implemented in get_ball_position function of the Ball Class):
    - Use Frame-subtraction to the moving objects on top of the table. Assumption is that the camera is static.
    - Use HSV color threshold with white to find the contours of the ball.
    - Use current position to narrow down future search of the ball.
    - Update the position of the ball with each frame
3. General moving direction of the ball (implemented in calculate_direction function of the Ball Class):
    - After ball is found twice successfully, a general direction (left, right, up, down) is calculated through the difference of the coordinates.
    - Update direction with each frame.
4. Predict future direction of the ball (implemented in draw_parable function of the Ball Class):
    - When a direction change is detected from left to right, right to left or downwards to upwards and the changed direction continuous for two frames, a guess is made of the future flying path of the ball through the current state of the ball.
    - The prediction is not updated with each frame. Instead the prediction is kept till a direction change is detect that fulfills the condition for a new prediction.

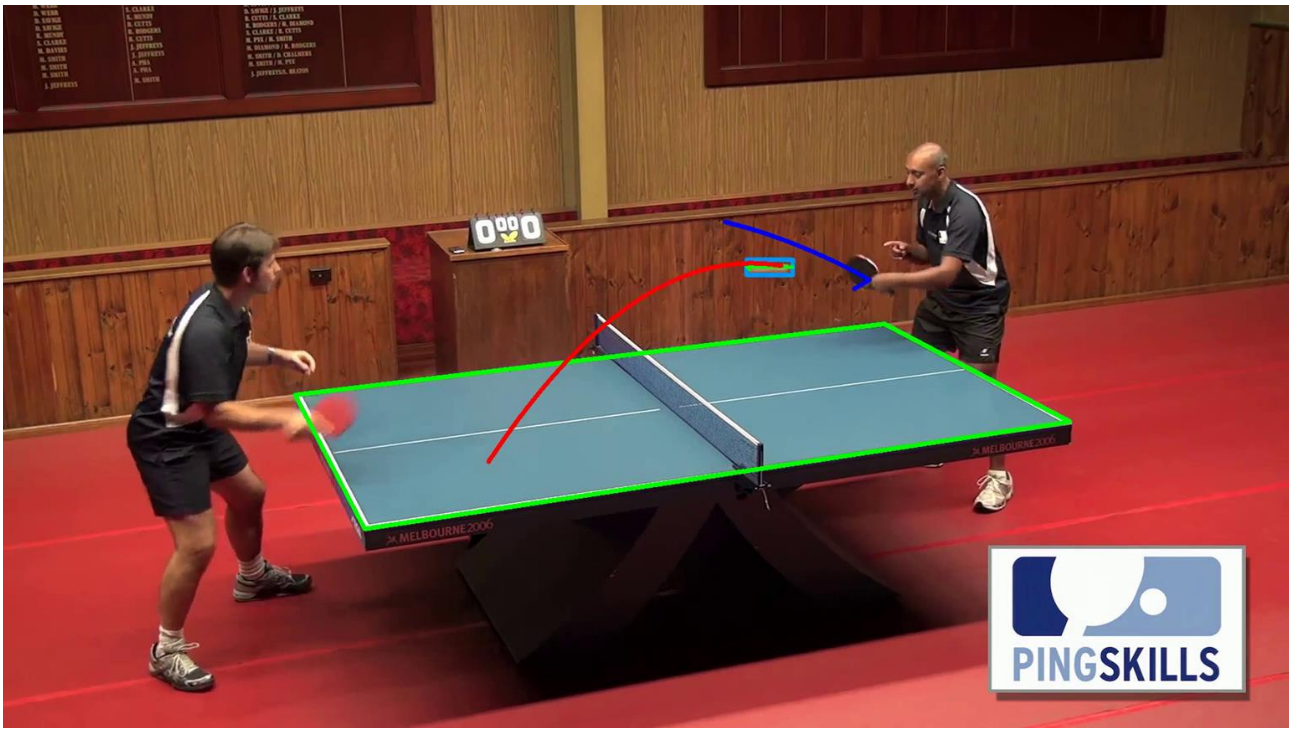
Input video and Result

The result of the application strongly depends on the video. For a successful usage its recommended to use a video where:

- The camera is static.
- The table takes up a good portion of the image.
- The players don’t block the view on the table.
- There is no background moment.
- The view more on top of the table.

See example result below for a well working video:


PINGSKILLS


PINGSKILLS