1 Petri Dish Image Processing: Main Execution Block

1.1 Import and Initialization

```
def process_series(series_prefix, series, min_radius, min_count):
    date, scanner, experiment, files, mask, dish_coords, dish_angles, first_frame,
    ↪ last_frame = series
    for i, (x, y, r) in enumerate(dish_coords):
        print(f'output/{series_prefix}_{i+1:02}_d03.jpg')

    if len(dish_coords) < 6:
        print(f'WARNING: PREPROCESSED/{series_prefix}_%d_%03d has {len(dish_coords)}
        ↪ dishes, expected 6')
```

Explanation:

- Initializes the `process_series` function with parameters `series_prefix`, `series`, `min_radius`, and `min_count`.
- Prints output file names based on `series_prefix` and dish coordinates.
- Issues a warning if fewer than 6 dishes (number of petridishes in the initial raw image) are found.

1.2 Image Handling and Initial Frame Extraction

```
    last_frame = {}
    for filepath in files:
        try:
            frame = cv2.imread(filepath)
            masked_frame = cv2.bitwise_and(frame, mask)
            for i, (x, y, r) in enumerate(dish_coords):
                x, y, r = max(x, min_radius), max(y, min_radius), min(min_radius, min(x,
                ↪ y, masked_frame.shape[1] - x, masked_frame.shape[0] - y))
                dish = masked_frame[y - min_radius:y + min_radius, x - min_radius:x +
                ↪ min_radius]
                dish = apply_curve_transformation(dish, piecewise_linear_transformation,
                ↪ 0.0, 0.85)
                last_frame[i] = dish
        except:
            continue
```

Explanation:

- Initializes `last_frame` to store images.
- Iterates over file paths to read and mask images.
- Extracts Petri dish regions from each image, applies curve transformation, and updates `last_frame`.

1.3 Frame Processing Loop

```
    this_frame_index = 1
    frame_index = 0
    for filepath in files:
        frame = cv2.imread(filepath)
        frame_index = int(filepath.split('_')[-1].split('.')[0])
        if frame is None:
            continue

        while this_frame_index < frame_index:
            for i, (x, y, r) in enumerate(dish_coords):
                this_last_frame = last_frame[i].copy()
                cv2.putText(this_last_frame, f'f{this_frame_index:03}', (105, 105),
                ↪  cv2.FONT_HERSHEY_SIMPLEX, 3, (255, 255, 255), 2)
                cv2.circle(this_last_frame, (50, 50), 40, (50, 50, 250), -1)

                ↪  cv2.imwrite(f'PREPROCESSED/{series_prefix}_{i+1:02}_{this_frame_index:03}.jpg',
                ↪  this_last_frame)
            this_frame_index += 1
```

Explanation:

- Initializes `this_frame_index` and `frame_index` to track frames.
- Processes each file, skipping empty frames.
- Saves missing frames by creating copies from `last_frame` with timestamps (provision inclusion in case of missing timeframes in experimental data collection.)

1.4 Mask Application and Rotation

```
        if dish_coords is not None and mask is not None:
            masked_frame = cv2.bitwise_and(frame, mask)
            for i, (x, y, r) in enumerate(dish_coords):
                x, y, r = max(x, min_radius), max(y, min_radius), min(min_radius, min(x,
                ↪  y, masked_frame.shape[1] - x, masked_frame.shape[0] - y))
                dish = masked_frame[y - min_radius:y + min_radius, x - min_radius:x +
                ↪  min_radius]
                if dish_angles[i] != 0:
                    rotation_matrix = cv2.getRotationMatrix2D((int(min_radius),
                    ↪  int(min_radius)), dish_angles[i]*180/np.pi, 1)
                    dish = cv2.warpAffine(dish, rotation_matrix, (2*int(min_radius),
                    ↪  2*int(min_radius)))
                if dish.size > 0 and dish.shape[0] > 0 and dish.shape[1] > 0:
                    dish = apply_curve_transformation(dish,
                    ↪  piecewise_linear_transformation, 0.0, 0.85)
                    last_frame[i] = dish.copy()
```

Explanation:

- Applies mask to each frame and extracts Petri dish regions.
- Rotates dishes if necessary based on `dish_angles`.
- Applies curve transformation to the dishes and updates `last_frame`.

1.5 Output Saving

```
# Save preprocessed images
cv2.imwrite(f'PREPROCESSED/{series_prefix}_{i+1:02}_{frame_index:03}.jpg',
↪  dish)
```

Explanation:

- Saves preprocessed Petri dish images to the ‘PREPROCESSED‘ directory with appropriate filenames.

Explanation Summary

The `process_series` function processes a series of Petri dish images by extracting and transforming regions of interest, applying masks, and rotating images as needed. It handles missing frames by generating them based on the previous frames and saves all processed images to the specified output directory. The function ensures that the images are properly masked and transformed, adhering to specified parameters for dish coordinates and rotation angles.