Object/Speed Detection

Numerical Programming - 2024/11/09

Erekle Khomasuridze

Algorithm Explanation

Step 1: Setup and Initialization

- The necessary libraries are imported, including OpenCV for video processing, NumPy for numerical operations, distance from SciPy for calculating distances, and DBSCAN from scikit-learn for clustering.
- The video file is loaded using `cv2.VideoCapture`, and key properties such as frame rate, frame dimensions, and total number of frames are retrieved.
- A `VideoWriter` object is initialized to save the processed video with the specified codec and properties.
- Variables and dictionaries are set up to keep track of the bacteria, including unique IDs, maximum allowed disappearance frames, and area thresholds for object filtering.

Step 2: Background Subtraction and Contrast Enhancement

- The code uses a background subtractor (MOG2) to isolate moving bacteria from the background. It also applies CLAHE (Contrast Limited Adaptive Histogram Equalization) to enhance the contrast of the grayscale images, making bacteria more distinguishable.
- The frame is converted to grayscale, enhanced using CLAHE, and combined with the background mask to further highlight the bacteria.

Step 3: Noise Reduction and Thresholding

- Morphological operations, such as closing, are used to reduce noise and fill small gaps in the detected objects.
- Otsu's thresholding is applied to convert the enhanced image into a binary image, where bacteria appear as distinct blobs.

Step 4: Connected Components and Filtering

- Connected components analysis is performed on the binary image to label individual blobs. Each blob's area is calculated, and blobs that are too small or too large (based on predefined area thresholds) are filtered out.
- The centroids of the remaining blobs are computed and stored for further processing.

Step 5: Clustering Using DBSCAN

- The centroids of the detected blobs are clustered using the DBSCAN algorithm. This helps group nearby centroids and remove noise (isolated points).
- The average centroid of each cluster is calculated and used as the final position of the detected bacteria in the current frame.

Step 6: Tracking Bacteria Across Frames

- If no bacteria have been tracked yet, all detected centroids are registered as new bacteria and assigned unique IDs. Each bacterium’s position, speed, and lifetime are initialized.
- If bacteria have already been tracked in previous frames, the code matches the current centroids to previously tracked bacteria using a distance matrix. This matrix calculates the Euclidean distance between the previous and current centroids.
- Bacteria are matched based on the minimum distance, and their positions and lifetimes are updated. Speed is calculated based on the change in position and the frame rate.
- If a bacterium is not detected in the current frame, its “disappeared” count is incremented. If it disappears for too many consecutive frames, it is removed from tracking.
- New bacteria are registered for any unmatched centroids in the current frame.

Step 7: Speed and Average Calculations

- The average speed of all bacteria that have existed for a minimum number of frames is calculated. The fastest bacterium is identified based on average speed.
- The number of tracked bacteria and the overall average speed are displayed on the video frame.

Step 8: Display and Save the Processed Video

- The processed frame, with information such as the number of bacteria, average speed, and progress percentage, is displayed in a window and saved to the output video file.
- The code waits for a short delay between frames to control the playback speed. Pressing the "ESC" key exits the loop early.
- Once all frames have been processed, the video capture and writer objects are released, all OpenCV windows are closed, and the location of the saved video file is printed.

Code explanation in more detail can be seen in the following link:

My GitHub Repository

Videos and Test Results

Bacteria_1 Detection Success

- **Video 1**: This video featured approximately 30-50 bacteria, all of which were clearly in focus under the microscope. These bacteria were of relatively similar size, shape, and translucency, making them ideal candidates for this algorithm.
  - **Detection Success**: The background subtraction, thresholding, and clustering steps worked effectively to isolate and count each bacterium, as there was little interference from noise or overlapping objects. The CLAHE enhancement allowed the bacteria to stand out sharply against the background, making it easier for the morphological operations to fill any small gaps and create distinct blobs.
  - **Tracking and Speed Measurement**: Due to the clarity of the video and the uniformity of the bacteria, the tracking mechanism (using DBSCAN clustering and Euclidean distance matching) was able to maintain a high level of accuracy when detecting movements across frames. This allowed for consistent speed measurements and fewer discrepancies in the detection count. This video demonstrated the algorithm's ideal working conditions, as the bacteria were isolated, clearly focused, and homogeneously distributed across the frame.

Bacteria_2 Detection Partial Success

- **Video 2**: This video presented a challenge due to out-of-focus bacteria, which appeared blurred, translucent, and difficult to distinguish from the background. These bacteria varied in shape and size, further complicating detection.
  - **Challenges in Detection**: The blurred and translucent bacteria had lower contrast with the background, making it difficult for CLAHE and background subtraction to isolate them effectively. As a result, some bacteria were missed entirely during the connected components and thresholding steps. Additionally, the translucent nature of these bacteria sometimes caused them to merge with noise or appear faint, making their detection inconsistent.
  - **Partial Success in Tracking**: Although some bacteria were missed, the overall count was lower than the other videos, which allowed the algorithm to achieve partial success. The few detected bacteria were tracked successfully, but their intermittent visibility led to inaccurate speed measurements. This video highlighted the algorithm's limitations with lower-contrast, irregularly shaped objects that blend into the background, underscoring the need for more robust edge enhancement techniques or alternative filtering mechanisms.

Blood Cells Detection Fail

- **Video 3**: This video aimed to detect and measure the speed of densely clustered blood cells. The cells were very close to each other, and in some cases, they were touching or overlapping, which proved to be a significant limitation for this algorithm.
  - **Detection Limitations**: Since the blood cells were tightly packed and often touching, the connected components analysis frequently grouped multiple cells into a single blob. This clustering error led to an underestimation of the total cell count and inaccurate speed tracking, as multiple cells were mistaken for one.
  - **Clustering and Tracking Challenges**: The DBSCAN clustering and Euclidean distance calculations struggled with these close, overlapping cells, as the minimum separation distance between distinct objects was often smaller than the algorithm's distance threshold. Consequently, clusters frequently included multiple blood cells, resulting in false positives or merged detections. The algorithm's performance on this video indicated that it is better suited for detecting well-separated objects and may require tuning or alternative clustering approaches for densely packed objects like blood cells.

Conclusion and Improvements

Reflection and Analysis

- **Algorithm Strengths**: The algorithm performed well under ideal conditions with distinct, focused, and separated bacteria. Under such circumstances, it demonstrated reliable object detection, tracking, and speed measurements.
- **Limitations and Areas for Improvement**:
  - **Low Contrast and Translucency**: Blurred, translucent objects proved challenging due to their low contrast with the background. Potential improvements could involve incorporating more advanced contrast enhancement or adaptive thresholding methods.
  - **Clustered Objects**: The close proximity of objects, like blood cells, caused issues in distinguishing individual objects. A possible enhancement could include a modified clustering technique or an algorithm that can better handle tightly packed objects.
- **Future Directions**: Incorporating edge detection, modifying the DBSCAN parameters dynamically based on object density, or integrating deep learning-based segmentation models could improve accuracy in cases with low contrast or clustered objects.

References

[Bacteria1] https://www.istockphoto.com/video/euglena-is-a-genus-of-single-celled-flagellate-eukaryotes-under-microscopic-view-for-gm1178310901-329281738

[Bacteria2] https://www.istockphoto.com/video/colony-of-ciliates-coleps-under-a-microscope-gm 1063517902-284330906

[BloodCells] https://www.istockphoto.com/video/fresh-human-blood-in-microscope-gm1147229910-309370293