Watershed algorithm for image segmentation and analysis of parameters with Object Motion Detection

Digital Image processing course(E9 241 IISC Bangalore)

Instructors: Prof . Rajiv Soundararajan, Prof . Soma Biswas

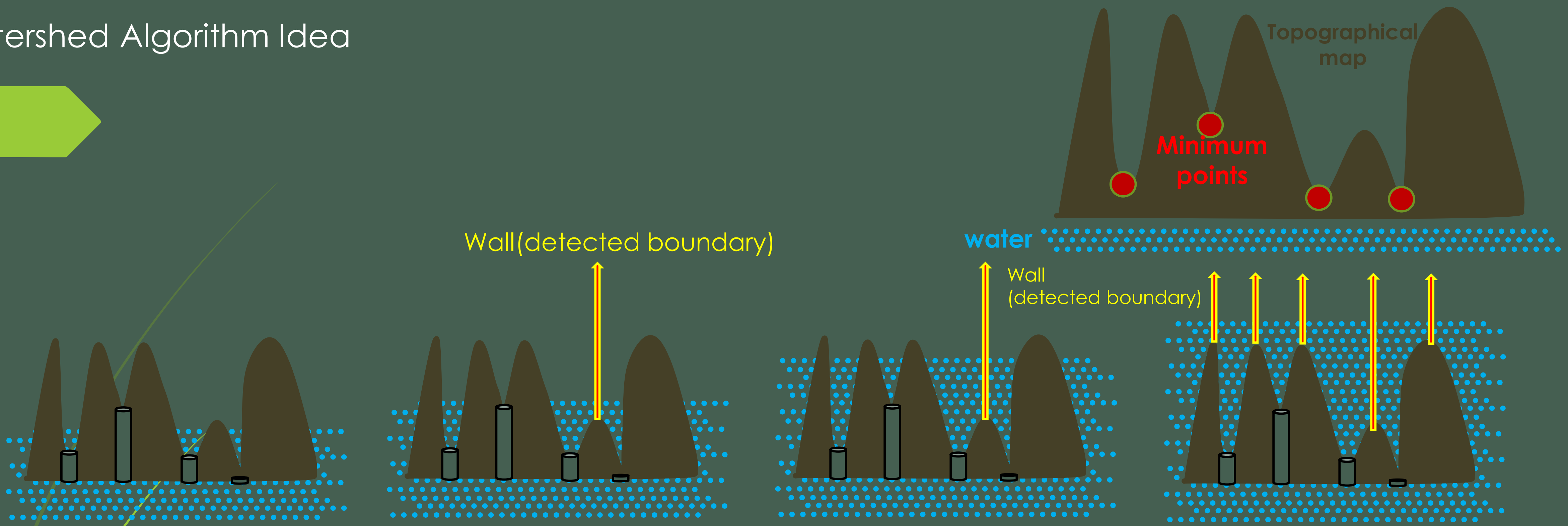
Vijaya Kumar . T (18191) vijayakumart@iisc.ac.in

Sowjanya . P (180066) sowjanyap@iisc.ac.in

Watershed : Draw boundary where two connected components connects

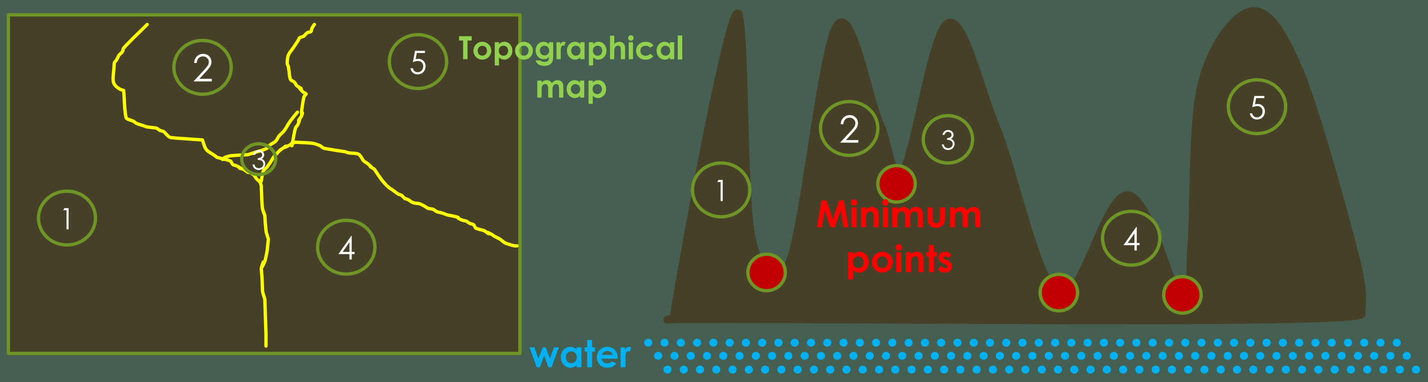
Note : Everything in this ppt and in Project Code implemented on our own.

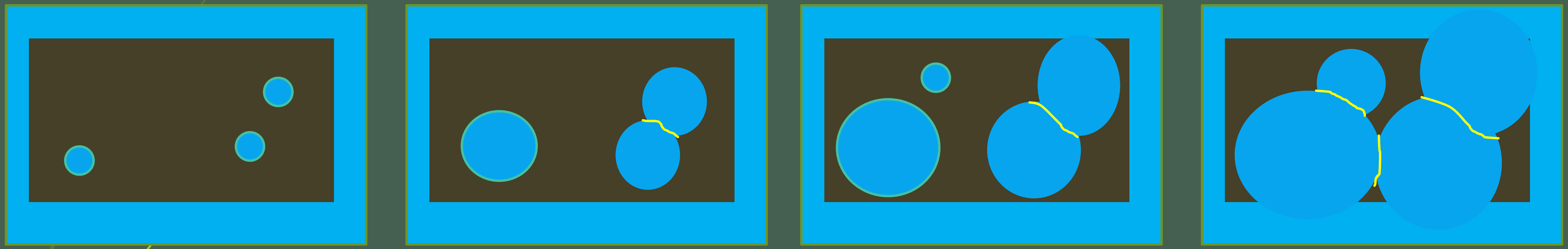
Watershed Algorithm Idea



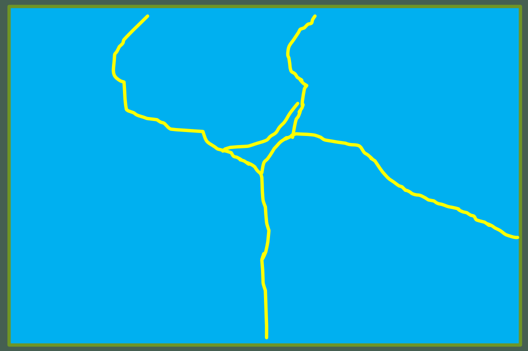
- watershed
  - When water flows through each hole from minima everything starts as separate basins.
  - When two separate basins merge together we built there a wall(dam).
  - We make more boundaries by increasing flow more.
  - At last we get all boundaries around each of the topographical regions.

Watershed Algorithm Idea on Images





- Algorithm in simple words
  1. Take image and identify the minima(n) assuming intensities as topographical Hight.
  2. We convert image into binary using ‘n’ as threshold
  3. Now using binary image find connected components
  4. Increase n -> n+1
  5. Compare previous connected components to new if two old connected components merging together record that common pixels as boundaries.
  6. Repeat 2 to 5 process.

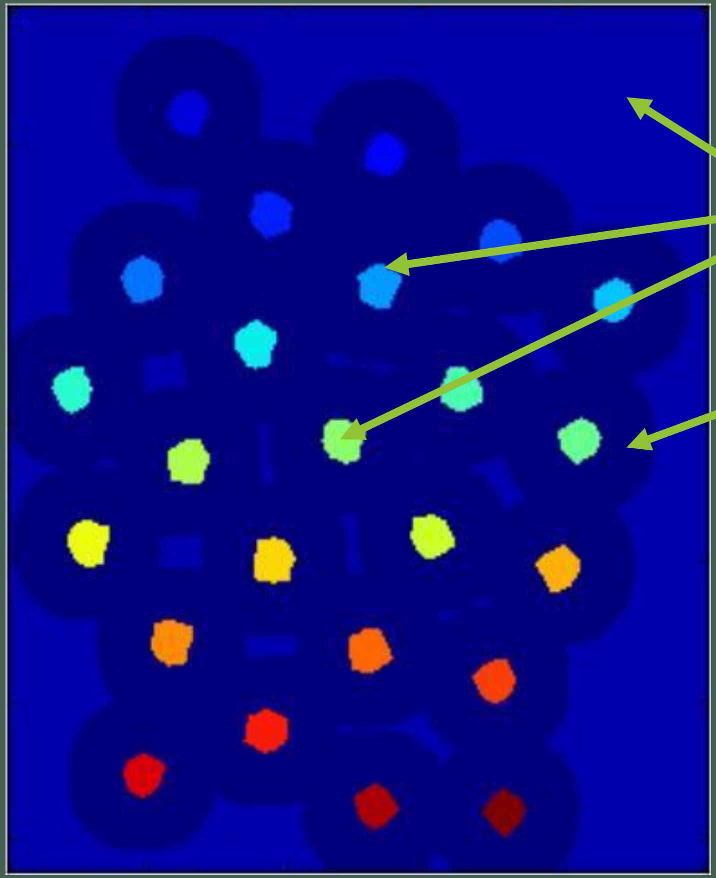


What are markers ?

- Usually watershed segment using minima of image applying directly watershed considering all local minima as basins we will get over segmentation.
- To avoid that problem we will predefine minima as markers by labelling known regions as minima.

How to obtain markers?

- For an image its markers is also a image, which is labeled with non zero intensity for known region and labelled 0 for unknown region.
- To obtain such markers in our motion we take difference of frames which signifies motion of object assigning labels by connected components we will get predefined location of object.
- Taking gradient and labelling edges with 0 and appending above markers will lead us the image of known and unknown regions specified.
- So that we can apply watershed to predefined markers.



Known Data

Unknown Data

Image source
https://docs.opencv.org/master/d3/db4/tutorial_py_watershed.html

Project Work

By understanding of watershed, in order to detect moving object we have focused more on making markers.

Our entire work flow can be divided as follows

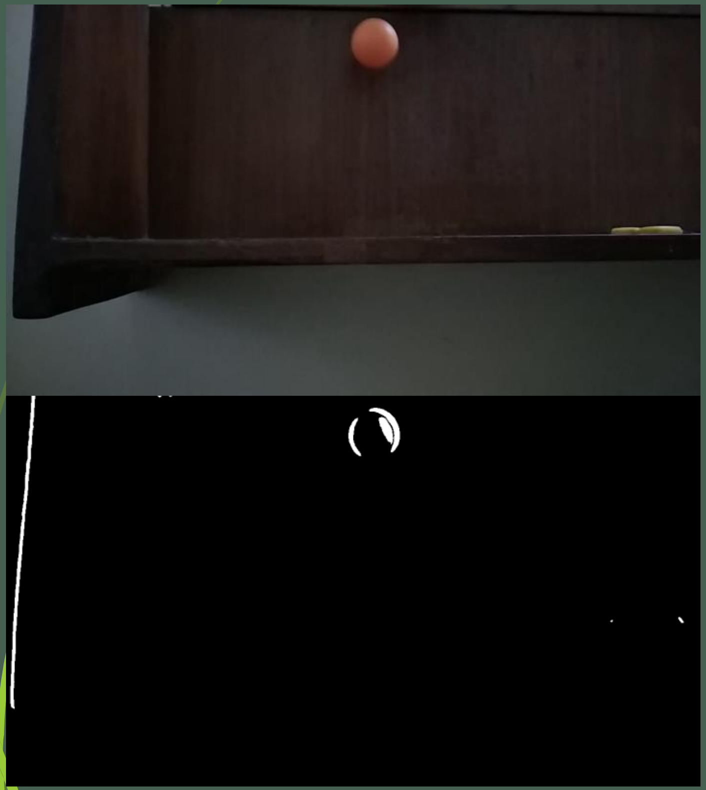
- **Stage1**
  - Created data set, Analyzed video data with python libraries.
- **Stage2**
  - Experimented watershed with parameters such as markers using consecutive frames.
  - Obtained promising two methods to video.
- **Stage3**
  - Tried first promising method of stage 2 but results are not satisfactory.
- **Stage4**
  - Tried second promising method obtained promising results.
- **Stage5**
  - Compared python Contours inbuilt function segmentation with our results.

Each one of stage explained in next slides with results

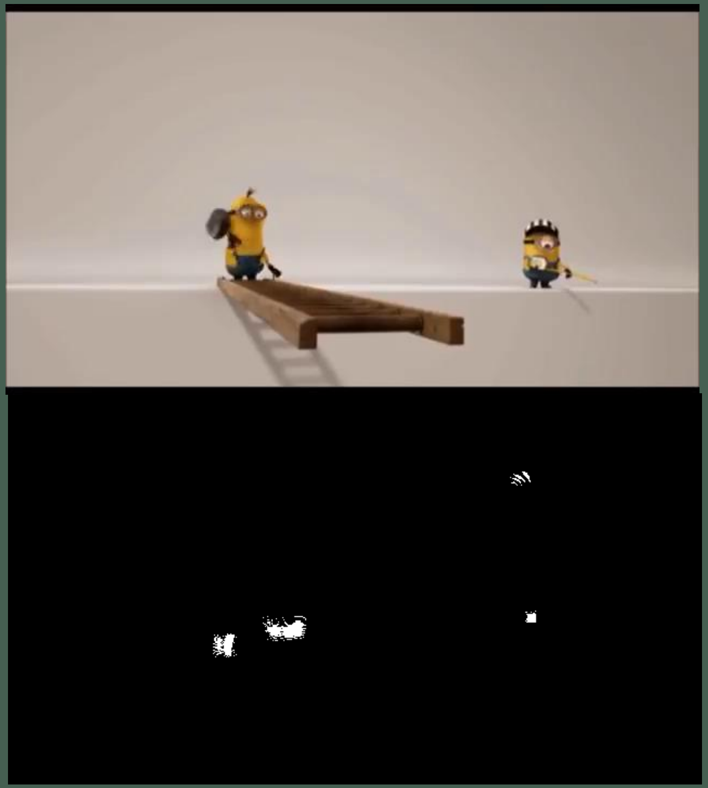
Stage 1

We took 5 videos as datasets (3 are recorded on own, 2 are from web)

Low noise
(low background movement)



Medium noise
(Medium background movement)



High noise
(High background movement)

Stage 2

- Methods tried
  - Method 1.1: Took difference between 2 consecutive frames and using that as markers applied watershed to gradient of the frame.
  - Method 1.2 : Took difference between 3 consecutive frames using that as markers applied watershed to gradient of the frame with preprocessing.
  - Above methods failed to decrease noise without losing markers
  - Method 3 : Took difference between two consecutive frames and gradient of the frame. Using both, formed markers and applied watershed to original frame.
- Common steps
  - In each method to decrease noise we have applied gaussian blur.
  - Converted into binary image by Otsu's Threshold.
  - Applied Morphological dilate and erode operation to decrease noise.
  - Using Connected components converted binary image into labelled image to make markers.

Stage 4 : Methodology

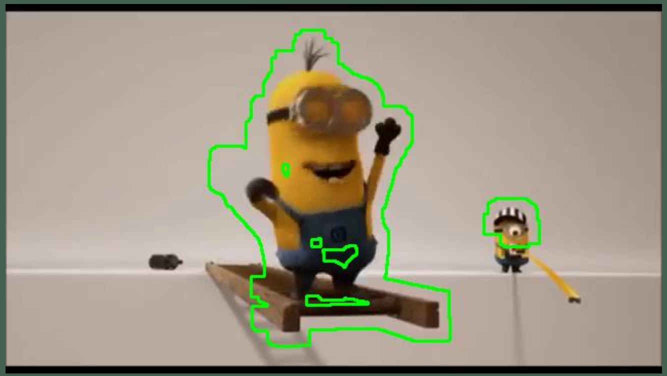
Old image $\xrightarrow[\text{process}]{}$ new image

1. Took two consecutive gray frames(a0,b0), applied gaussian blur to image so that noise decreases while taking difference.{(a0,b0) → (a1,b1)}
2. Found difference of filtered images. {c0 → |a1 – b1|}
3. Applied Otsu's binarization to obtain binary image. {c0 → c1}
4. With morphological operation first dilated that binary image to fill gaps then erosion to eliminate noise. {c1 → c2}
5. Applied connected components to label the object for markers. {c2 →c3}
6. Found gradient of gray frame(a0) and done binarization. { a0 → d0 }
7. Using erosion to gradient extracted the boundaries of object and labelled as unknown region to apply watershed. { d0 → d1 }
8. Made markers by adding images from step 5,6. { m → d1 + c3 }
9. Applied watershed algorithm to initial frame using markers. { op → watershed(a0, m) }
10. Obtained output Is binary boundary appended them to original frame as detected boundaries.

Stage 5 : Comparisons

Compared watershed result with the built-in function code which is contour

Contours



Contours

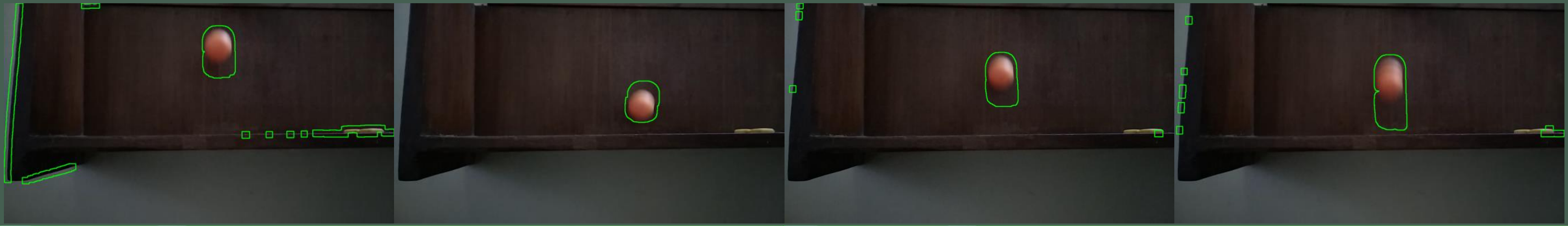
Contours

Watershed

Watershed

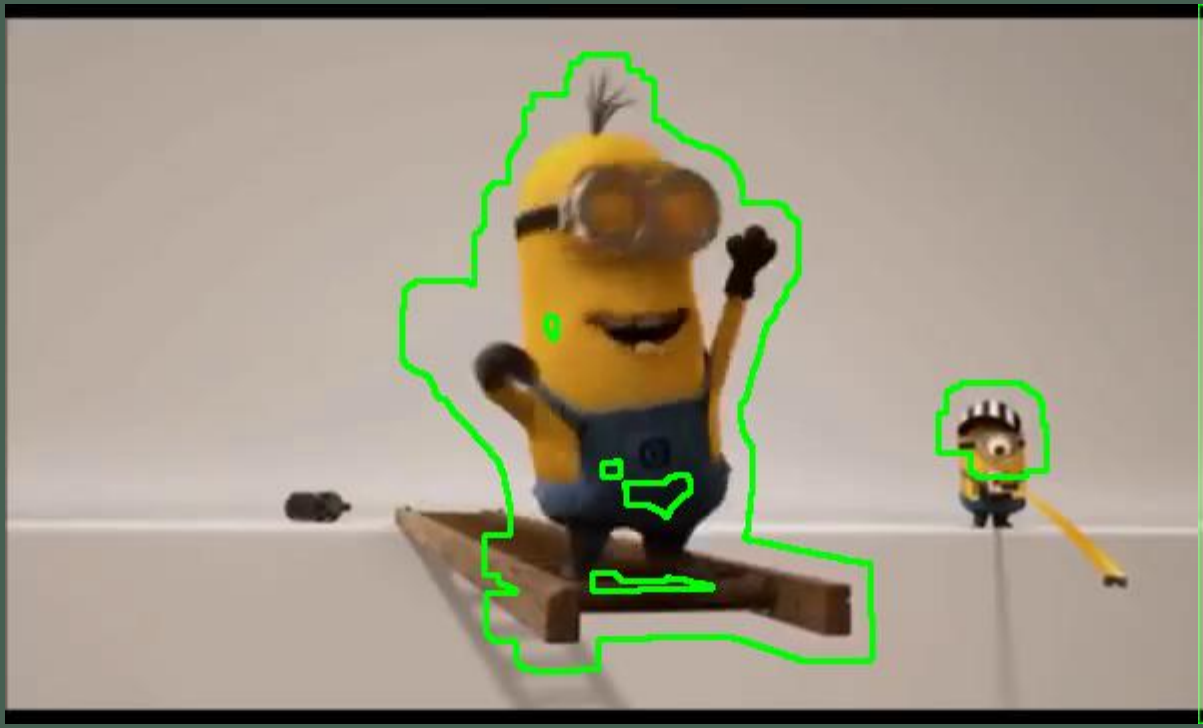
Watershed

Contours in built function



Watershed applied result

Contours in built function



Watershed applied result

Parameters that can change in our code

- No of iterations of erosion and dilation of morphological image to extent known and unknown regions.
- We can try other thresholding value instead of Otsu.
- We can implement with 3 consecutive frame for efficiency.

What we have learned

- Working principles of watershed algorithm.
- Reading and manipulation of video using python
- Effective use of morphological operations.
- Last but not least implementing own code is challenging.

Submitted Project files Details

- Code folder
  - It contains each data set code with two python files
  - Each code runs using OpenCV python library
- Resources folder
  - It contains the material we referred to understand the algorithm
  - Along with it Our main source is Gonzalez Textbook.
- Results folder
  - It contains all data set results which we compare, that is images of both contours and watershed.