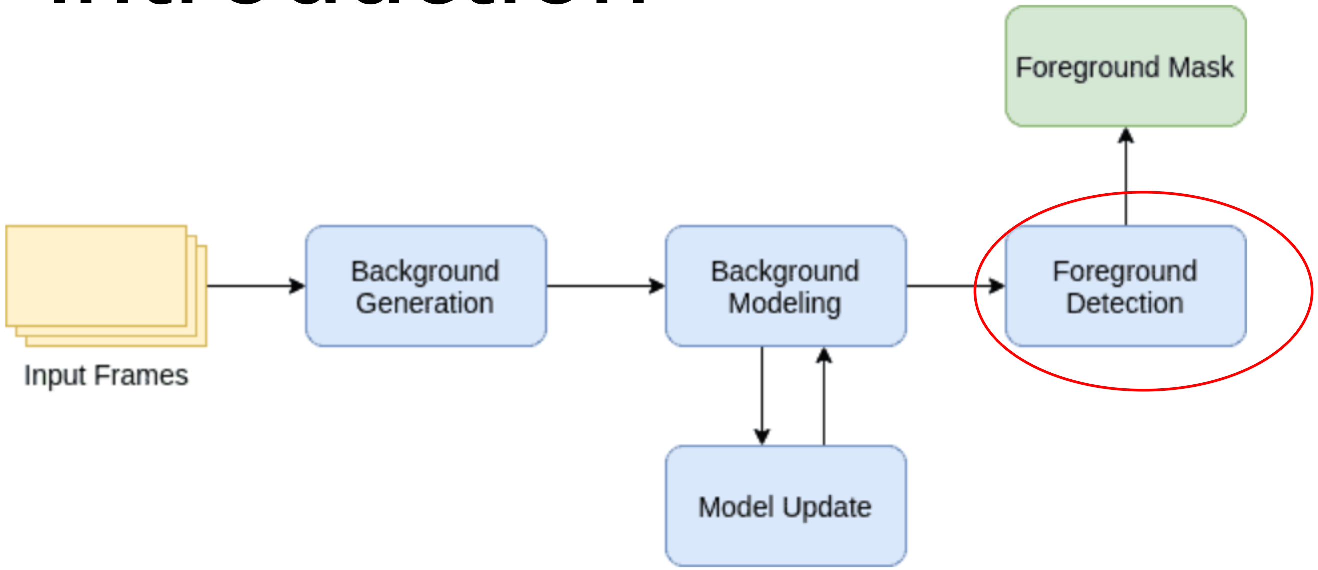
Background Subtraction of Videos by Mixtures of Gaussians

COMPSCI 302 (Spring 2023)

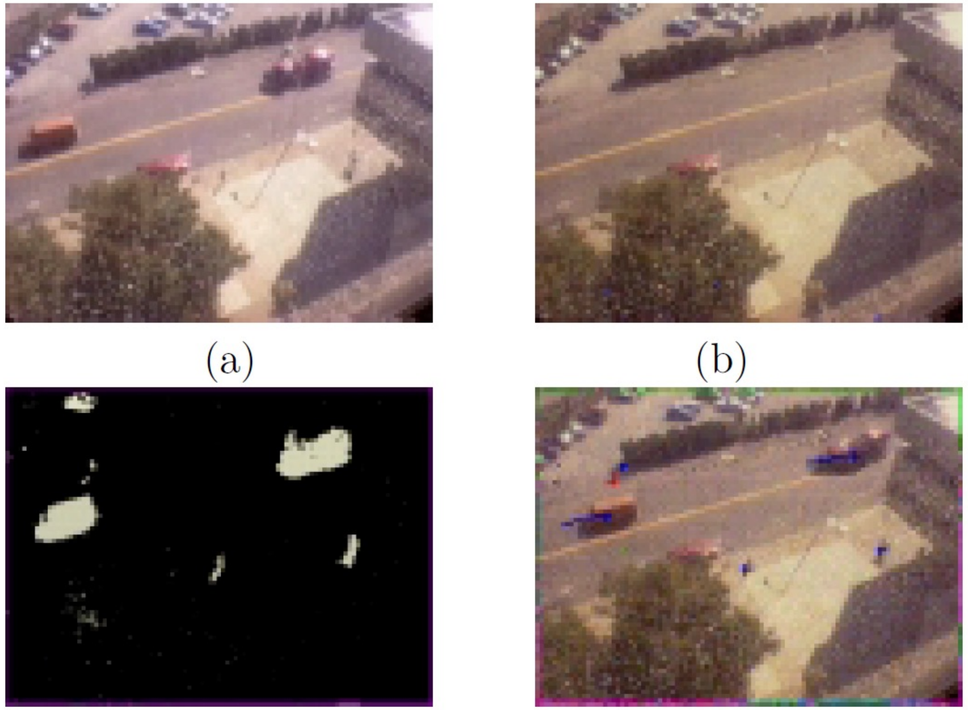
Jiayun Huang & Yike Guo

Introduction



Methods:
Running Gaussian Average
Temporal Median Filter
Mixture of Gaussians
Kernel Density Estimation
...

(Stauffer & Grimson, 1999)



(a) (b) (c) (d)

Pixel Process

- It is the collection of the values of a particular pixel location (x_0, y_0) over time t: $\{X_1, X_2, \dots, X_t\} = \{I(x_0, y_0, i): 1 < i < t\}$
- I values are scalars for BW images, and vectors for RGB images.

Running Gaussian Average

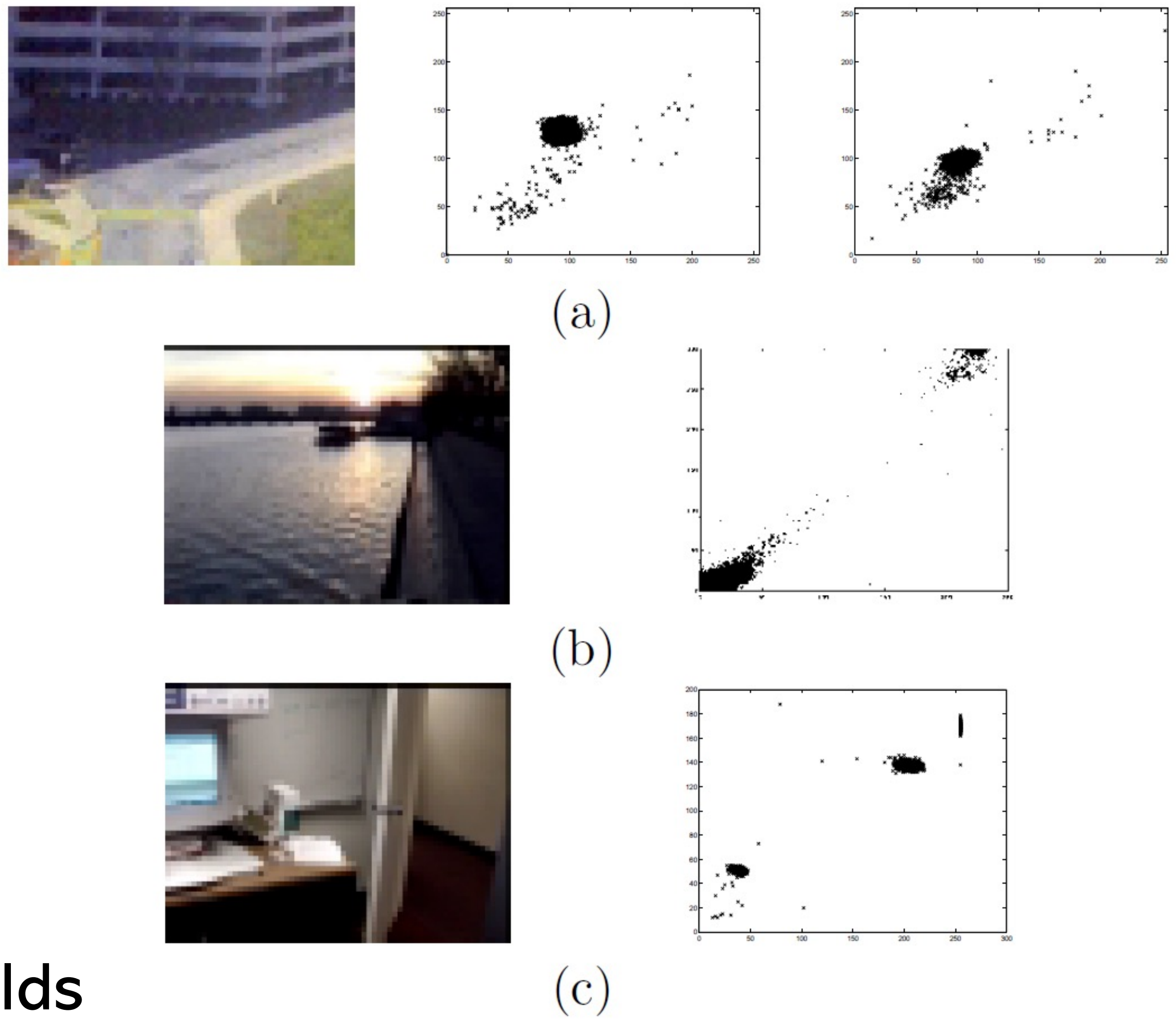
- The pixel process (last n values) at each pixel location is modeled with a Gaussian probability density function (pdf).
- At each new frame time t, a running average is calculated:
$$\mu_t = \alpha X_t + (1 - \alpha)\mu_{t-1}$$
- Threshold for classifying background values: $|X_t - \mu_t| \leq k\sigma_t$

Pixel process of a single pixel in R & G channels.
(Stauffer & Grimson, 1999)

Problems

- (a): luminance changing in background (the two process are 2 min apart)
- (b) & (c): bimodal distribution of the pixel process due to flickering

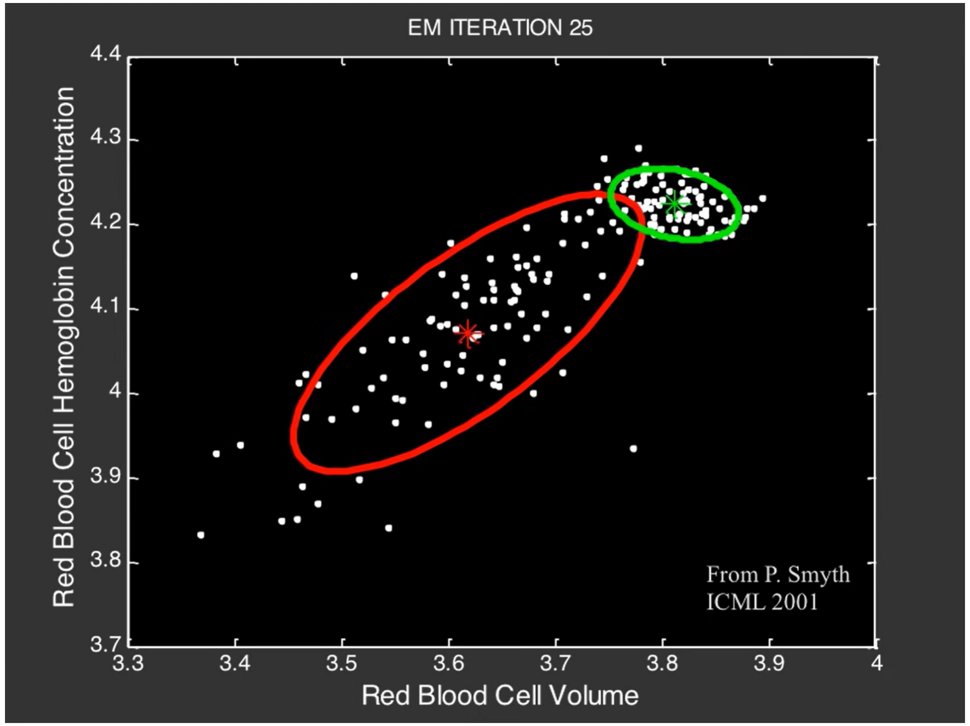
Calls for adaptive multimodal systems with automatic thresholds



Mixture of Gaussians

- The pixel process at each pixel location $\{X_1, X_2, \dots, X_t\}$ is modeled with a mixture of Gaussians: $P(X_t) = \sum_{i=1}^{K} \omega_{i,t} * \eta\left(X_t, \mu_{i,t}, \Sigma_{i,t}\right)$
- K: number of distributions (3-5)
- $\mu_{i,t}$: mean value of the ith Gaussian at time t
- $\omega_{i,t}$: the weight of the ith Gaussian at time t
- $\Sigma_{i,t}$: covariance matrix of the ith Gaussian at time t ($\approx \sigma_k^2 \boldsymbol{I}$)
- η: Gaussian probability density function (pdf)

A 2-D example of modeling data using mixtures of Gaussians. (Alexander Ihler, 2015)



Updating the model

- Standard way: expectation maximization (too costly)

→ K-means approximation

1. Every new X_t is checked if it matches any of the K existing distribution (threshold: $\left|X_t - \mu_{k,t}\right| \leq 2.5 * \sigma_{k,t}$)
2. Update parameters:
   - $\omega_{k,t} = (1-\alpha)\omega_{k,t-1} + \alpha\left(M_{k,t}\right)$, where $M_{k,t} = 1$ for matches and 0 for unmatched
   - If matched, $\mu_t = (1-\rho)\ \mu_{t-1} + \rho X_t$, $\sigma_t^2 = (1-\rho)\ \sigma_{t-1}^2 + \rho(X_t - \mu_t)^{\ T}(X_t - \mu_t)$, where $\rho = \alpha\eta(X_t|\mu_k, \sigma_k)$
3. If none of the K distributions match the current pixel value, the least probable distribution is replaced with a distribution with the current value as its mean value

Background Model Estimation

- 1. The Gaussians are ordered by $\frac{\omega}{\sigma}$
- 2. The first B distributions are chosen as the background model, where $B = argmin_2\left(\sum_{k=1}^{b} \omega_k > T\right)$
- T: a measure of the minimum portion of the data that should be accounted for the background.
- Larger T allows multi-modal distribution by repetitive background motions (e.g. leaves shivering)

Code Implementation

- Steps:
  - Background Initialization
  - Background Update.

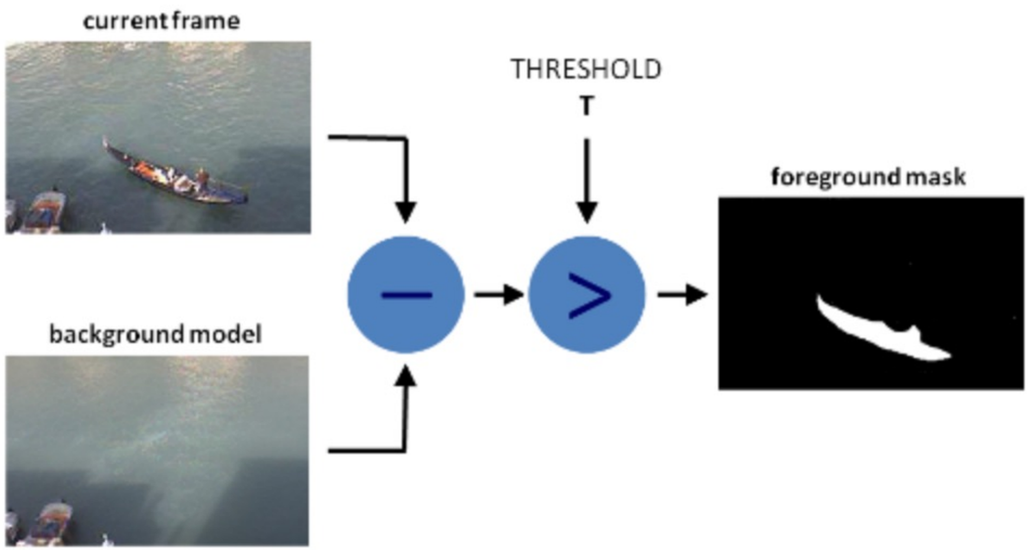


Image from https://docs.opencv.org/4.x/d1/dc5/tutorial_background_subtraction.html

Video from https://pythonprogramming.net/mog-background-reduction-python-opencv-tutorial/

Code Implementation

- Tutorial:
  - Read data from videos or image sequences using cv2.VideoCapture
  - Create and update the background model using cv2.BackgroundSubtractorMOG2()
  - Get and show the foreground mask using cv2.imshow()

Code Implementation

- API in OpenCV

```
cv2.createBackgroundSubtractorMOG2()
```

  - A Gaussian Mixture-based Background/Foreground Segmentation Algorithm
  - Based on two papers:
    - "Improved adaptive Gaussian mixture model for background subtraction" in 2004
    - "Efficient Adaptive Density Estimation per Image Pixel for the Task of Background Subtraction" in 2006
  - Feature of the algorithm: selects the appropriate number of gaussian distribution for each pixel

Video from https://pythonprogramming.net/mog-background-reduction-python-opencv-tutorial/

Code Implementation

- API in OpenCV

  cv.BackgroundSubtractorMOG2 (history = 500, varThreshold = 16, detectShadows = true)

  - Parameters:
    - History: length of history
    - varThreshold: Threshold on the squared distance between the pixel and the sample to decide whether a pixel is close to that sample
    - detectShadows: If true, the algorithm will detect shadows and mark them. It decreases the speed a bit, so if you do not need this feature, set the parameter to false
  - Return:
    - An instance
  - Generate foreground mask using 'Apply'

Reference: https://docs.opencv.org/3.4/de/df4/tutorial_js_bg_subtraction.html

Code Implementation

- API in OpenCV

```
cv.BackgroundSubtractorMOG2 (history = 500, varThreshold = 16, detectShadows = true)
```

  - Return:
    - An instance
  - Generate foreground mask using ‘Apply’

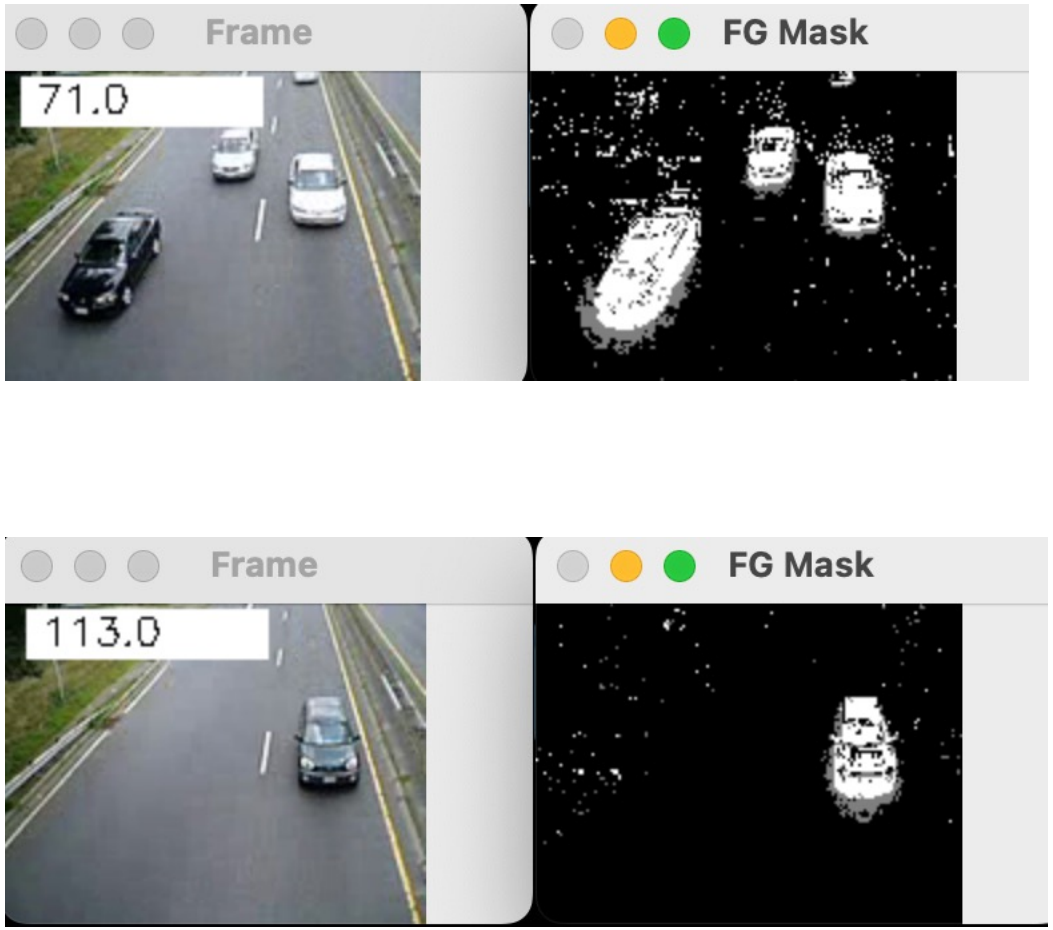
```
apply (image, fgmask, learningRate = -1)
```

    - Parameters:
      - Image: Next video frame. Floating point frame will be used without scaling and should be in range [0,255]
      - fgmask: The output foreground mask as an 8-bit binary image
      - learningRate: The value between 0 and 1 that indicates how fast the background model is learnt. Negative parameter value makes the algorithm to use some automatically chosen learning rate. 0 means that the background model is not updated at all, 1 means that the background model is completely reinitialized from the last frame

Reference: https://docs.opencv.org/3.4/de/df4/tutorial_js_bg_subtraction.html

Code Implementation

background_sub.ipynb



References

- Alexander Ihler (Director). (2015, March 7). *Clustering (4): Gaussian Mixture Models and EM.* https://www.youtube.com/watch?v=qMTuMa86NzU
- OpenCV. Background Subtraction Available online: https://docs.opencv.org/3.4/de/df4/tutorial_js_bg_subtraction.html
- OpenCV. How to Use Background Subtraction Methods. Available online: https://docs.opencv.org/4.x/d1/dc5/tutorial_background_subtraction.html
- Santoyo-Morales, J. E., & Hasimoto-Beltran, R. (2014). Video Background Subtraction in Complex Environments. *Journal of Applied Research and Technology. JART*, *12*(3), 527–537. https://doi.org/10.1016/S1665-6423(14)71632-3
- Stauffer, C., & Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, *2*, 246-252 Vol. 2. https://doi.org/10.1109/CVPR.1999.784637