Intrusion Detection Project

Luca Burroni
Computer Vision and Image Processing Course
Academic Year 2022/2023
University of Bologna, Department of Computer Science and Engineering
luca.burroni@studio.unibo.it

Abstract

In this project, an intrusion detection system is developed. The system can detect objects (intruders) that do not belong to a static reference scene (background) and classify them. Given a video sequence the goal is reached through automatic video analysis and computer vision techniques.

1 Change Detection Algorithm

The selected approach consists in performing **background subtraction**.
Considering that the video sequence is made up of greyscale images (one channel), the subtraction is performed by **properly thresholding** the **absolute difference** between the frame and the background.
There exists no time interval showing the background only, so a proper **background initialization** is done.
Also **background updating** is applied since lighting changes slowly (darkening).

1.1 Background Initialization

Since there exists no time interval showing the background only, we can infer a background model using the first n frames, as reported in "Change Detection" lecture.
Background initialization can be "blind" or "selective":

- The **blind** approach consists in calculating, for each pixel, a central tendency measure (median, mean or mode) of its values across all n frames unconditionally.
- The **selective** one, otherwise, does not consider pixel values across all n frames unconditionally. In fact for each pixel, only the values corresponding to the frames in which the pixel has been classified as belonging to the background (using temporal frame difference + morphological operations) are selectively considered.

In this project a **selective background initialization** is used, since this way we can have a background model using less initial frames as requested.
For completeness also a blind background initialization is implemented using the same number of initial frames used for the selective approach. This way a comparison is provided and is shown the reason why selective method is preferred.
The **median** is used as central tendency measure because is the one that showed the better results.

Preprint. Under review.

1.1.1 Selective Method

To classify pixels of each of the *n* initial frames as belonging to the background or not a **temporal two-frame difference method** has been applied followed by **morphological operations**.
In Figure 1 is shown the temporal two-frame difference mask between a frame and the previous one.
In Figure 2 is shown the mask after morphological operations, in particular:

- **Opening** to remove small-size blobs. A 3x3 structuring element made of ones is used.
- **Closing** to fill holes in the figure. A 5x5 structuring element representing an ellipse made of ones is used with 12 iterations.
- **Opening** to remove other eventually remaining blobs which represent noise. A 9x9 structuring element made of ones is used.

Figure 1: Temporal two-frame difference mask

Figure 2: Temporal two-frame difference mask + morph ops

The final mask is used to classify pixels for each frame. This way for each pixel are selectively considered just values corresponding to the frames in which the pixel has been classified as belonging to the background, as explained before.
The *n* number of initial frames considered for background initialization is **automatically detected**. The method stops processing and computes background (median) once that for each pixel there are at least *n_values* (default 5) selected.
In this case it uses 20 initial frame and produces the background model shown in Figure 3

1.1.2 Blind Method

For comparison purpose in Figure 4 is shown the background gotten by using blind method (median) with the same number of initial frames (20).

Figure 3: Background model with selective method

Figure 4: Background model with blind method

1.2 Background Updating

Lighting changes slowly (darkening), so a **selective alpha-blending** procedure is applied for background updating.
The method uses the mask gotten by background subtraction from frame, and then **morphologically improved**, to detect background pixels of the frame and update the relative intensity in background model.
The formula is the following:

$$B_{t+1}(i,j) = \begin{cases} \alpha \cdot F_t(i,j) + (1-\alpha) \cdot B_t(i,j) & \text{if } C_t(i,j) = 0 \\ B_t(i,j) & \text{otherwise} \end{cases}$$

Alpha (c in the code) is the adaptation rate, in this project is set to 0.4 as default (value chosen after trial and error). *B* represents the background, *F* the frame and *C* the mask.

2 Foreground mask

For each frame of the video sequence (except those used for background initialization) **background subtraction** is performed using the **updated background** (updated each frame in the process).
This way a foreground mask is obtained to show objects that do not belong to background. Of course also in this case we perform **morphological operations** to improve such mask, in particular:

- **Opening** to remove small-size blobs. A 3x3 structuring element made of ones is used.
- **Closing** to fill holes in the figure. A 17x17 structuring element representing an ellipse made of ones is used.
- **Opening** to remove other eventually remaining blobs which represent noise. A 9x9 structuring element made of ones is used.

Figure 5: Foreground mask by background subtraction

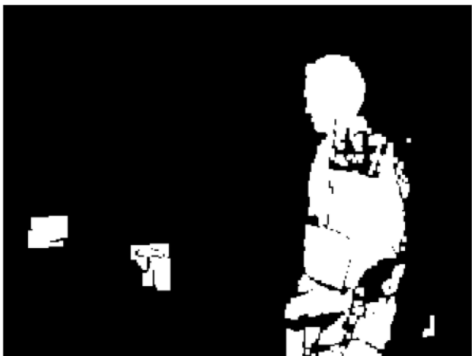

Figure 6: Mask post opening step (3x3 of ones)

Figure 7: Mask post closing step (17x17 ellipse of ones)

Figure 8: Mask post opening step (9x9 of ones)

3 Connected Components Labeling

The foreground mask gotten after all previous steps is then used for connected components (blobs) labeling and classification.
A proper **OpenCV function** is applied (with **8-way connectivity**) to obtain components.
Then an **RGB mask** is generated. To distinguish the different blobs each one is coloured differently, as shown in Figure 9.

Figure 9: Labeled and coloured mask

3.1 Blobs Features Extraction

For each blob of those identified some features are extracted.
In particular **area**, **perimeter** and **baricenter**. For perimeter calculation we used the **blob contours**, obtained with a specific OpenCV functions, the other two features are returned as outputs by the OpenCV function used for Connected Components Labeling.

3.2 Blobs Classification

Blobs are also classified as "**Person**" or "**Other**".
In case of "Other" they are then distinguished in "**True**" or "**False**".

3.2.1 Person Or Other

To classify a blob as "Person" or "Other" the **area** of the blob is used.
In fact, if the area exceeds a certain **threshold** the component is classified as "Person", otherwise as "Other".

3.2.2 True Or False (second task)

To distinguish a blob classified as "Other" in "True" or "False" we considered (as suggested) the **blob contours**.
Those contours are compared to the **current frame edges**, obtained applying **Canny's Edge Detector**.
A **bit-wise logical AND** between the two is computed. If the overlap pixels exceed a **threshold** the object is classified as "True", since it is **present on the current frame**. Otherwise as "False".
Before applying the *cv2.Canny* function on the frame (used to apply edges detection), we reduced the noise using the **Gaussian smoothing** (5x5 kernel).
The lower threshold and the upper threshold are **automatically calculated** for each frame as follow.
$L_t = max\{0, (1 - \sigma \cdot M)\}$
$L_t = min\{255, (1 + \sigma \cdot M)\}$
M is the median of the frame pixels values, sigma is an arbitrary parameter (0.33 by default in this case).
In Figure 10 and Figure 12 are reported the edges detected by the process discussed above for a certain frame, in Figure 11 are shown contours of a "False" object in that frame, in Figure 13 contours of a "True" one.

Figure 10: Canny's edges detector output

Figure 11: "False" object contours

Figure 12: Canny's edges detector output

Figure 13: "True" object contours

4 Outputs

As requested two outputs are provided:

- **Graphical**: Video that shows for each frame (but those used for background initialization) the labeled/coloured mask.
- **Textual**: File reporting, for each frame, the number of detected objects, and for each blob (object) the relative features and classifications. Figure 14 shows an example.

```
502     3
1       582.53  14090   [202.19, 152.46]        person  -
2       89.9    537     [30.06, 154.5]          other   false
3       106.97  671     [102.11, 176.97]        other   true
```

Figure 14: Textual output

5 Link to external resources:

- The project's Git-Hub repository.