Pedestrain Tracking

Zhengrong Wang, Hsienyu Meng, Liuyang Zhan

June 9, 2015

1 Introduction

2 Detection
- HOG + Adaboost
- Background Subtraction
- Results

3 Tracking
- Particle Filter
- Online Boosting Classifier
- Single Target Tracking
- Multiple Targets Tracking
- Energy Minimization

4 Summary

HOG + Adaboost

- Offline trained Adaboost classifier with 20 layers.
- Each layer contains variable number of weak classifiers.
- Each weak classifier calculates the HOG feature of a small patch in the ROI.
- (2 × 2 subregions) × (9 bins for HOG) = feature vector of 36 dimension.
- Sliding windows and non-maximum suppression to detect pedestrain.

Background Subtraction

- Mean-shift to construct background model.
- Subtracting background from the current model.
- Find the connected components.
    - Label as pedestrain if it satisfys some criterions.
    - Use Adaboost classifier for further detection.

Results

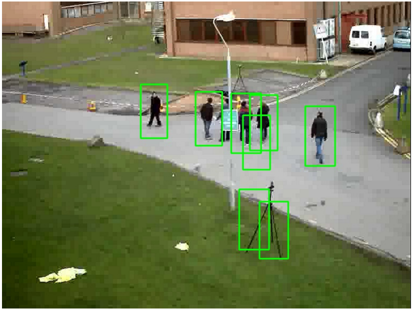


Figure: HOG Detection Results

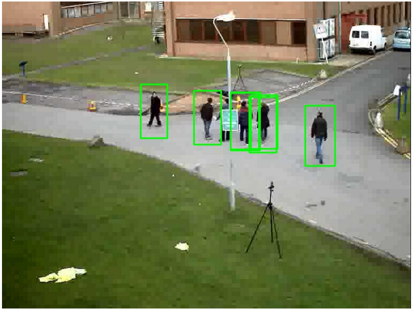


Figure: BKG Subtraction Results

Particle Fitler

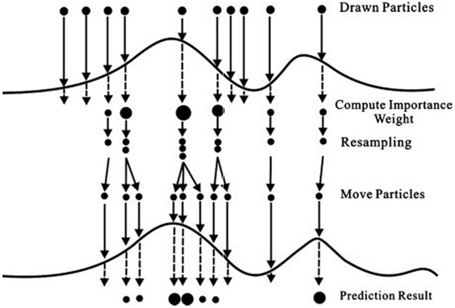


Figure: Particle Filters

- Propagate.
- Observe.
- Resample.

Particle Filter with Constant Velocity

- Propagate.

$$x_t = x_{t-1} + v_{t-1} + N(0, \sigma_x) \tag{1}$$
$$v_t = v_{t-1} + N(0, \sigma_v) \tag{2}$$

- Observe.
  - Online boosting classifier.
  - Matched detection.
- Resample.
  Resample with the weights calculated in previous stage.

Online Boosting Classifier

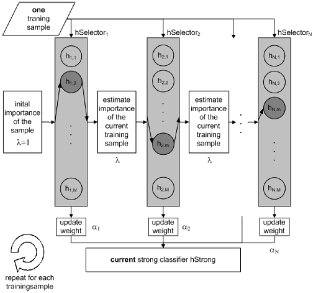


Figure: Online Boosting

- Weak classifier pool.
- Selectors.
- Strong classifier.

Weak Classifier

- Each weak classifier evaluates a feature.
- Uses Kalman filter to estimate the Gaussian distribution of positive and negative features.

$$K_t = \frac{P_{t-1}}{P_{t-1} + R} \tag{3}$$

$$\mu_t = K_t f(\mathbf{x}) + (1 - K_t)\mu_{t-1} \tag{4}$$

$$\sigma_t^2 = K_t(f(\mathbf{x}) - \mu_t)^2 + (1 - K_t)\sigma_{t-1}^2 \tag{5}$$

$$P_t = (1 - K_t)P_{t-1} \tag{6}$$

- A simple Eculidean distance threshold is enough to generate the hyposis.

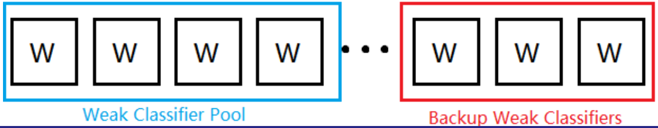
$$h(\mathbf{x}) = \min(D(f(\mathbf{x}), \mu_+), D(f(\mathbf{x}), \mu_-)) \tag{7}$$

Selector

- Evaluates the error rate of a weak classifier by:

$$err \approx \frac{\lambda_{wrong}}{\lambda_{wrong} + \lambda_{correct}} \tag{8}$$

- For each training, selects the best weak classifier, and replaces the worst classifier with a newly initialized one.
- A cycle queue for backup of weak classifiers.



Strong Classifier

- Cascade of selectors is a strong classifier.
- The final output is given by:

$$h^{strong}(\mathbf{x}) = \mathrm{sign}(\sum_{i=1}^{N} \alpha_i \cdot h_i^{selector}(\mathbf{x})) \tag{9}$$

- α is the weight of each selector given by:

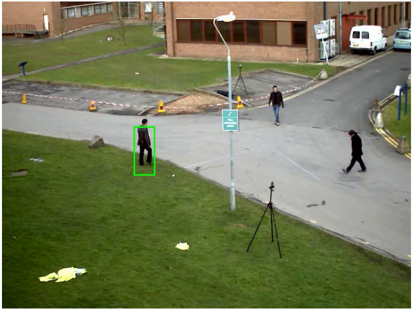
$$\alpha_i = \ln(\frac{1 - err_i}{err_i}) \tag{10}$$

Single Target Tracking

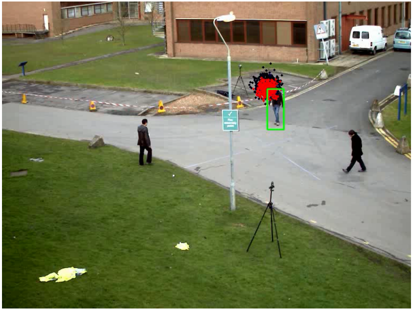
1. Initialize the target.
2. Propagate the particles.
3. Make observation.
4. Resample the particles, and find the target.
5. Sample around the new position and train the online boosting classifier. Go back to Step 2.

Results

Grayscale Haar-like feature.



RedGreenIntensity feature.



Multiple Targets Tracking

- Data Association Problem
  - Match matrix with score:
  - Greedy algorithm to find the match.
- Occlusion
  - Depth field.
  - Occlusion reasoning.

Results

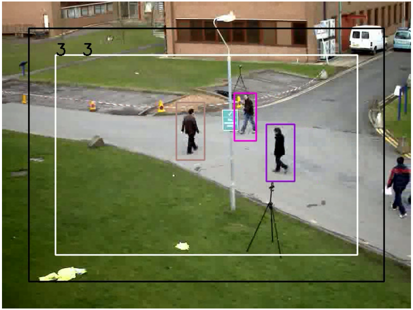


Figure: Multiple Tracker Initialized

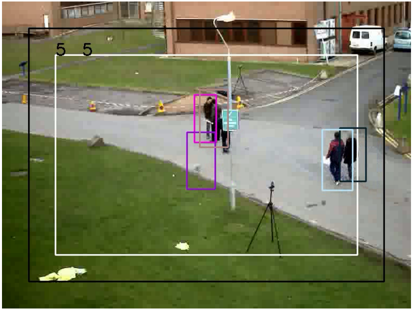


Figure: Multiple Tracker Lost

Problems and Solutions

- Match score should be more robustic and less dependent on the distance term.
- If there is occlusion, the detector may fail for many frames, The particle filter will almost certainly lose its target.
- With depth field this may be solved.

Energy Minimization

- Use Kalman filter or extended Kalman filter to get a initial solution.
- Optimized on a energy function within a temporal window.
- The result is quite good, however this method is not causal. It needs information in the future.

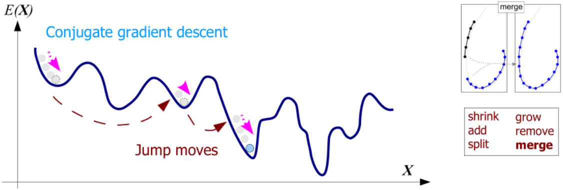


Figure: Energy Minimization

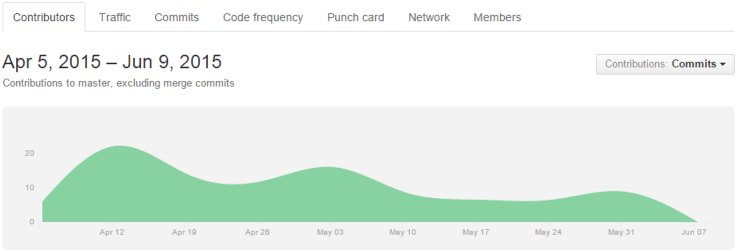
Results

Summary

- Multiple targets tracking is still a very challenging problem in computer vision.
- Real time? With GPU and multiple threads, this is possible.

Thank you!

- Project Website: https://zerowong.github.io/PedestrainCounting/
- GitHub Repo: https://github.com/zerowong/PedestrainCounting/



Apr 5, 2015 – Jun 9, 2015

Contributions to master, excluding merge commits