Beyond Sliding Windows

C.Lampert, M.Blaschko, T.Hofmann

Presented by Arie Meir

Agenda

- Object localization problem
- Why sliding window isn't good enough
- Efficient Sub-window Search
- Applications

Object Localization flavors

Contour

Center

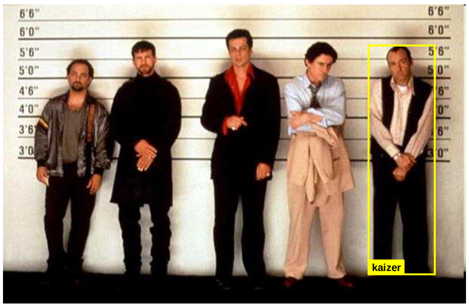
Object Localization flavors

Contour

Center

Center + parts

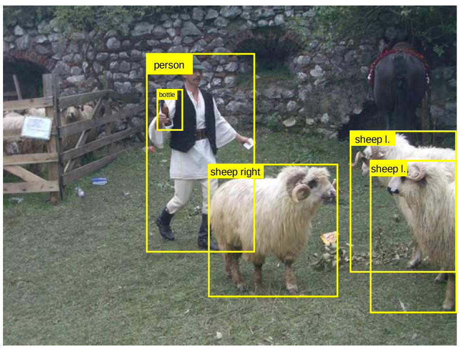
Bounding Box Localization



Intuitive, easy to get ground truth data

Example: identify all objects

A pastoral example: identify all objects



Approaches to bounding box localization

Target/quality function, ***score***, ***grade***: **all refer to a function f : window → real**

f is an order relation, reflects the likelihood of the object to be found in the given window

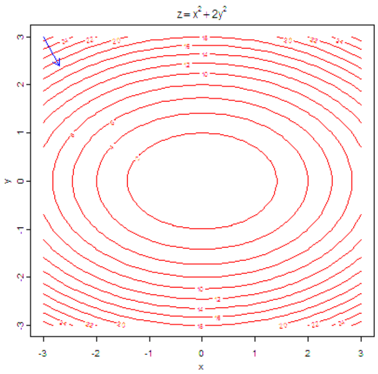
Gradient ascent approach

- **flaw: Finds local maxima**

Sliding window algorithms

- **flaw: slow, O(n^4) windows to check**

Gradient ascent example



Sliding Window: example

f(w) = +0.1

f(w) = -0.2

f(w) = -0.1

f(w) = +0.1

f(w) = +1.5

After a while….

f(w) = +0.5

f(w) = +0.4

f(w) = +0.3

$f(w_1 \ldots w_n)$ = 0.1,-0.2,-0.1,0.1 … 1.5 … 0.5,0.4,0.3

Sliding Window Approach

Performance issues:

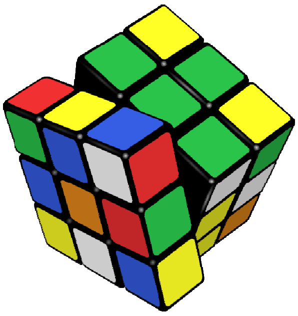
For an image ($n \times n$) : $O(n^4)$ windows

Evaluate a subset of windows

- Scale
- Aspect ratio
- Grid Size

Might Miss Solutions

We need a different paradigm !



Alternative: Exhaustive but smart !

<u>Intuition</u>: images with only the target object get the highest classification score

Find the window with $Score_{max}$ and you have found the object !

Bet on the winners early !
Don’t waste time on losers with low score !

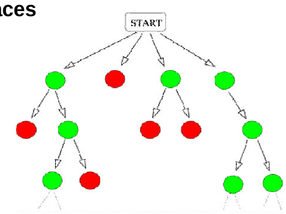
Branch and Bound

Branch and Bound

Linear Programming Idea from the 60s

Branching: Dividing a space of candidate rectangles into subspaces

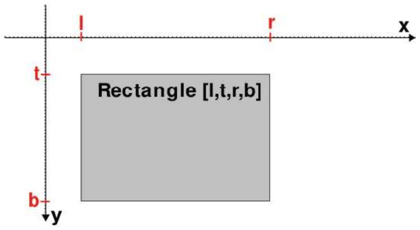
Bounding: Pruning subspaces with a highest possible score lower than some guaranteed score in other subspaces



B&B Design Steps

1. Parameterization of Search Space
2. How to split regions of Search Space
3. Bound for selection of most promising regions

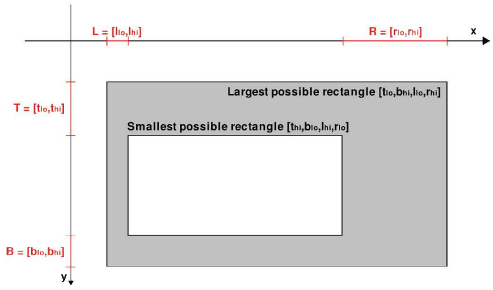
Parameterization of the Search Space



Four numbers : single rectangle

Four intervals : a rectangle sub-space

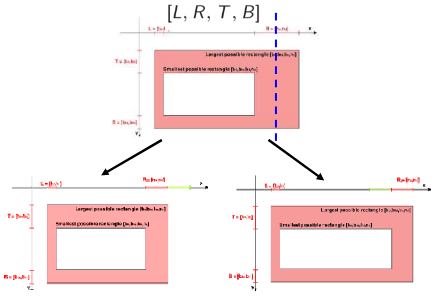
Parameterization of the Search Space



Subspace defined by (T,B,L,R) where each is interval (T_{low},T_{high})

Splitting: largest interval

$[L, R, T, B]$



$[L, R_1, T, B]$ $\qquad$ $[L, R_2, T, B]$

$R_1 := [r_{lo}, \lfloor \frac{r_{lo}+r_{hi}}{2} \rfloor]$ $\qquad$ $R_2 := [\lfloor \frac{r_{lo}+r_{hi}}{2} \rfloor + 1, r_{hi}]$

Bounding function

To use branch-and-bound for given quality function *f*, we need to define upper bound function:

$$i) \quad \hat{f}(\mathcal{R}) \geq \max_{R \in \mathcal{R}} f(R),$$

$$ii) \quad \hat{f}(\mathcal{R}) = f(R), \quad \text{if } R \text{ is the only element in } \mathcal{R}.$$

ESS: Effective Sub-window Search

I : Image (n×m)

$\hat{f}$: Bounding function

ESS

Window R: global maximum of f(R)

Rectangle R(t_{max},b_{max},l_{max},r_{max}) such that f(R) is maximum over all R ⊂ I

Good Bound is a challenge

Choosing a good Bound function isn't trivial

Bound fast to calculate but tight enough

$\hat{f}(\mathcal{R}) \geq \max_{R \in \mathcal{R}} f(R)$

Large Constant

Exact Equality

Fast evaluation of bound, but useless : need to go over all the rectangles

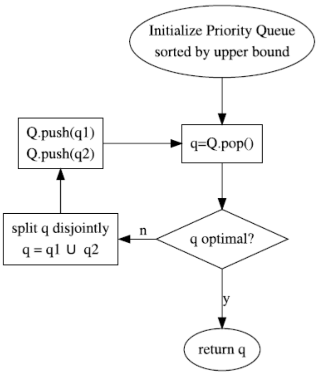
Fast convergence, but calculating the bound is equal to the original problem

THE ESS ALGORITHM

ESS outline

Require: image $I \in \mathbb{R}^{n \times m}$
Require: quality bounding function $\hat{f}$ (see text)
Ensure: $(t_{\max}, b_{\max}, l_{\max}, r_{\max}) = \operatorname{argmax}_{R \subset I} f(R)$
initialize P as empty priority queue
set $[T, B, L, R] = [0, n] \times [0, n] \times [0, m] \times [0, m]$
repeat
split $[T, B, L, R] \to [T_1, B_1, L_1, R_1] \,\dot\cup\, [T_2, B_2, L_2, R_2]$
push $([T_1, B_1, L_1, R_1], \hat{f}([T_1, B_1, L_1, R_1]))$ into P
push $([T_2, B_2, L_2, R_2], \hat{f}([T_2, B_2, L_2, R_2]))$ into P
retrieve top state $[T, B, L, R]$ from P
until $[T, B, L, R]$ consists of only one rectangle
set $(t_{\max}, b_{\max}, l_{\max}, r_{\max}) = [T, B, L, R]$

ESS Flow

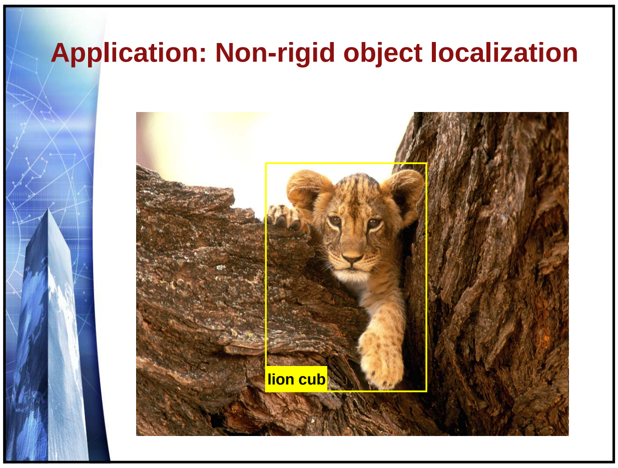


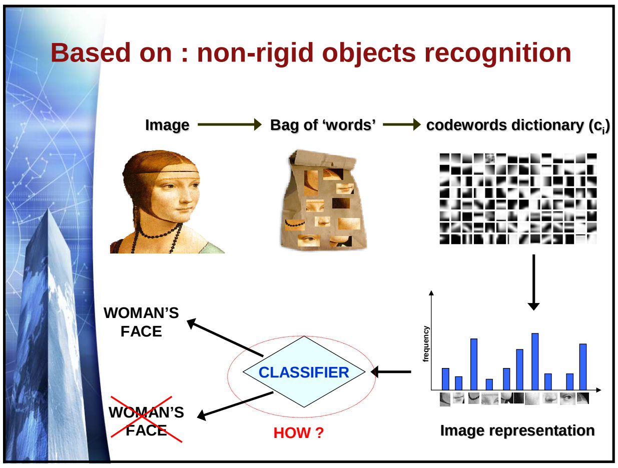
ESS AUTHORS' FANTASY

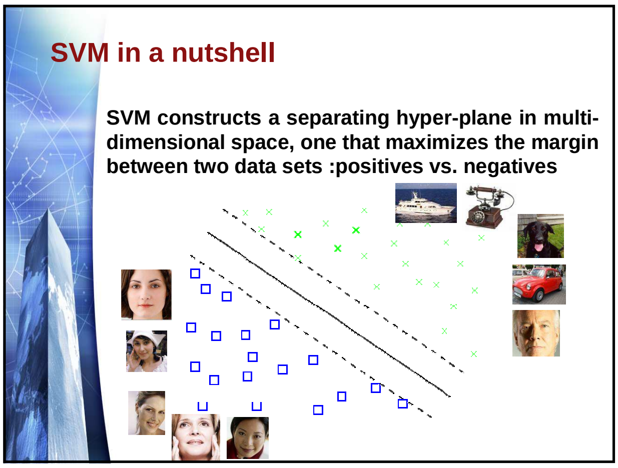




ESS
APPLICATION

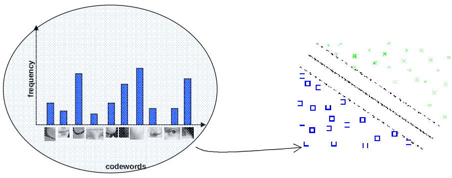

Application: Non-rigid object localization
lion cub


Based on : non-rigid objects recognition
Image
Bag of 'words'
codewords dictionary (c_i)
frequency
WOMAN'S FACE
CLASSIFIER
WOMAN'S FACE
HOW ?
Image representation


SVM in a nutshell
SVM constructs a separating hyper-plane in multi-dimensional space, one that maximizes the margin between two data sets :positives vs. negatives

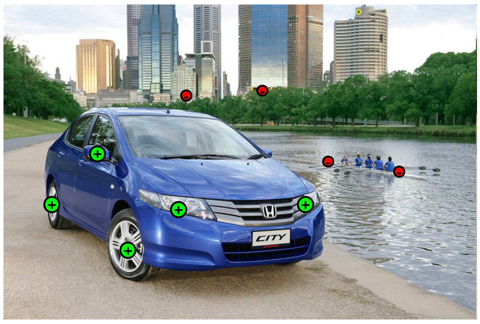
SVM in this example

Image → histogram → a single point



Given a new image, the task is to decide:
what side of the hyper-plane is it on ?
Positive match: SVM gives large values (+)
Negative match: SVM gives small values (-)

SVM Discriminative Nature – per point



Discriminating (Car-unique) points get large (positive) weight
Non-Discriminating points get small (negative) weight

Applying ESS

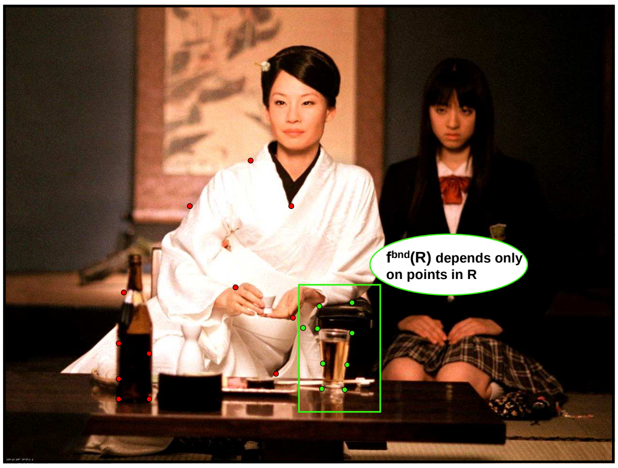
We need a relatively tight, easy to compute bound

Use SVM decision function as the base:

Given a query image, the match result given by SVM Decision function: (h: query, h^i: training set)

$$f(I) = \beta + \sum_i \alpha_i \langle h, h^i \rangle$$

Separate into single key-point contribution for fast computation



God is in the details

SVM Decision function: (h: query, h^i: training set)

$$f(I) = \beta + \sum_i \alpha_i \langle h, h^i \rangle$$

h is a histogram ➔ We can express f(I) as a sum of per-point contributions with weights

$$w_j = \sum_i \alpha_i h_j^i$$

$$f(I) = \beta + \sum_{j=1}^{n} w_{c_j}$$

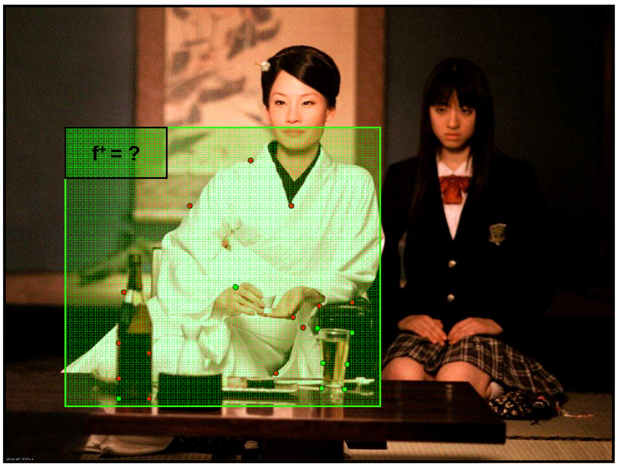
c_j – codeword of point j

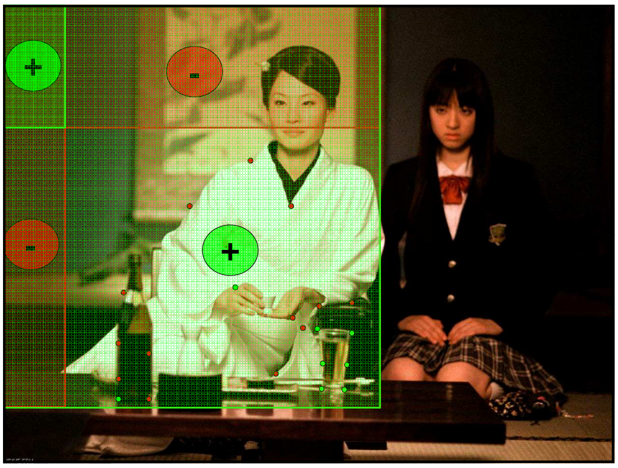
Denote for subspace R(T,B,L,R)

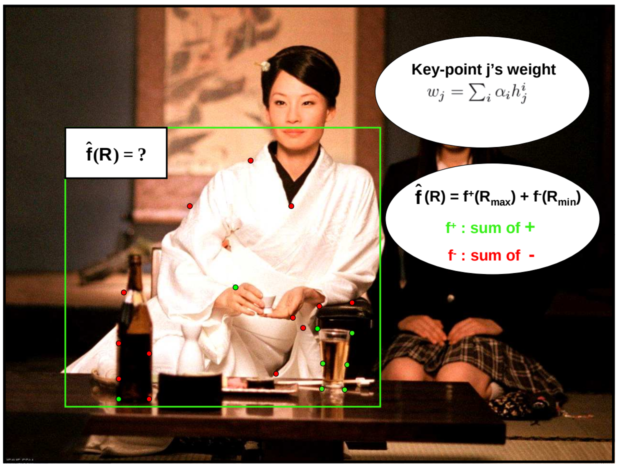
R_{max} = largest rectangle, R_{min} smallest rectangle

$$\hat{f}(\mathcal{R}) := f^+(R_{\max}) + f^-(R_{\min})$$

By using Integral Images, f^+,f^- are calculated in O(1)

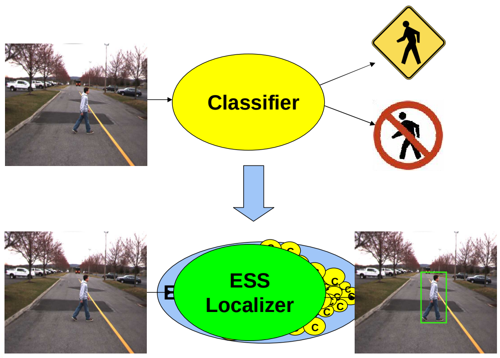






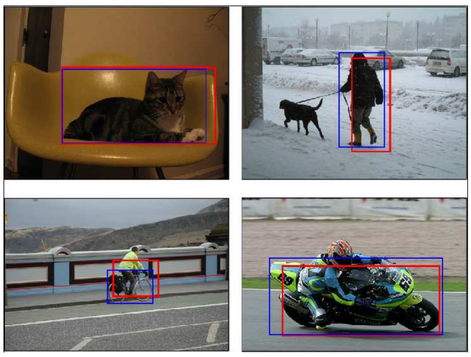
TO SUM UP:

Remember the main idea



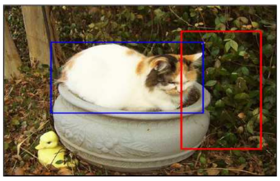
EXPERIMENTS

Experimental Results

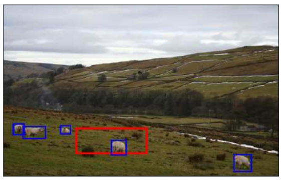


Blue :Ground truth Red: ESS

Can we learn from failures ?

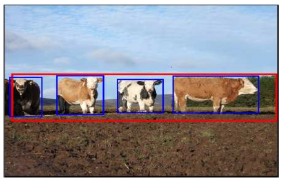


Problem:
Insufficient feature data

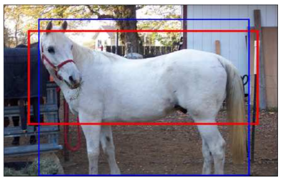


Possible solution:
More/different features

Another class of problems

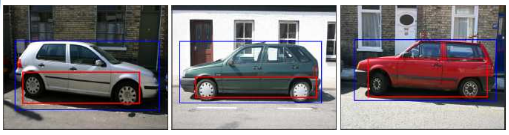


Problem:
Close objects/
Small extended parts



Possible solution:
Geometric regularization

Problem on wheels



The problem:

Bounding box are smaller, include wheels. Why ?

SVM looks for discriminating features, W_{wheels} is high !

A possible solution:

Post processing step, regression of the true bounding box based on the maximum score box

Conclusions

640x480 image has tens of billions of windows

Sliding window trades off runtime vs. accuracy

ESS Finds global maximum

BYOB = Bring Your Own Bound Function

Bring Your Own

Sliding Microsoft Windows xp

A Microsoft conspiracy?

References

- Beyond Sliding Window by C.Lampert, M.Blaschko, T.Hofman
- Recognizing Object Categories course by Li Fei-Fei
- Shawshank redemption, Kill Bill, Usual Suspects, Rocky

Thank You !

Arie Meir