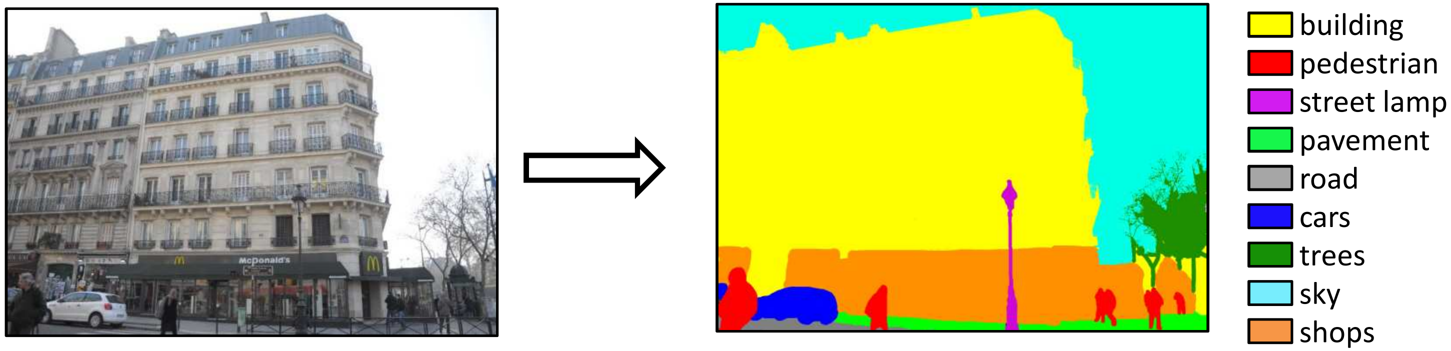
Parsing Image Facades with Reinforcement Learning

Symmetry Detection from Real Word Images Workshop, CVPR 2011

Olivier Teboul, **Iasonas Kokkinos**, Panagiotis Katsourakis, Loic Simon and Nikos Paragios

Semantic Segmentation of Urban Scenes

- Image Parsing



- Cropped and Rectified Building Images: `Facade Parsing’

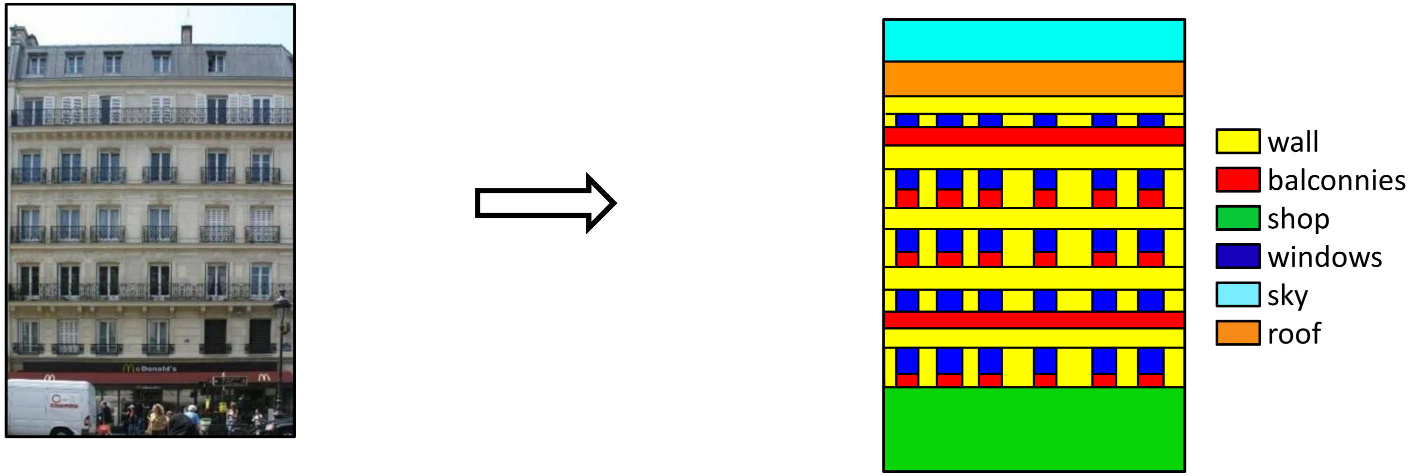


Image-based Procedural 3D Models

- Based on 2D parsing + simple extrude and insertion rules turn 2D to 3D…

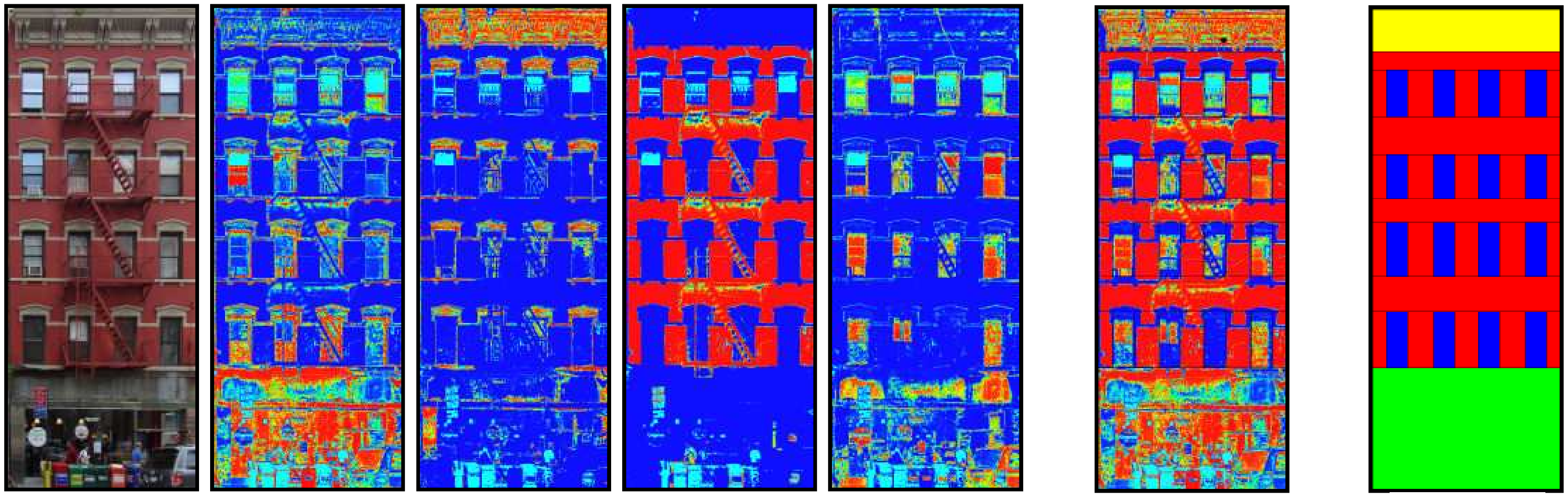
Problem Statement

- Input: image $I(x, y)$
- Output: labelling $l(x, y)$
- Pixel-level classification function: $m(c, x, y) = p(c|I(x, y))$
- Objective:

$$C(l) = \sum_{x,y} m(l(x, y), x, y)$$

- Wanted:

$$l^* = \arg \max_{l:\text{building}} C(l)$$



original $m(shop)$ $m(roof)$ $m(wall)$ $m(window)$ $m(l(x, y), x, y)$ $l(x, y)$

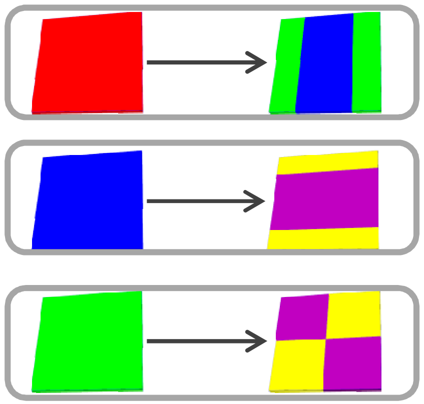
Shape Grammars: Recursive Derivation of Labelling

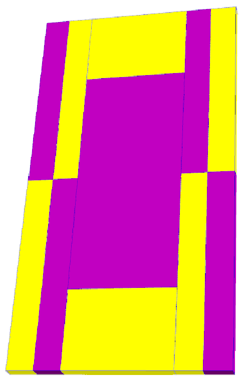
– Top level: axiom

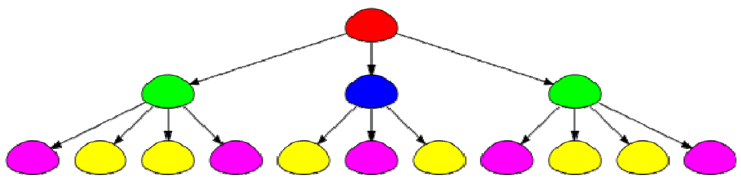
– Recursive application of shape operators

  • Partition domain and assign label to each part

– Terminals: semantic labels (e.g. window, door etc)

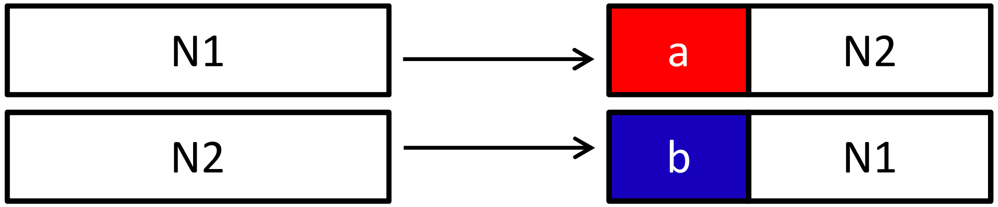




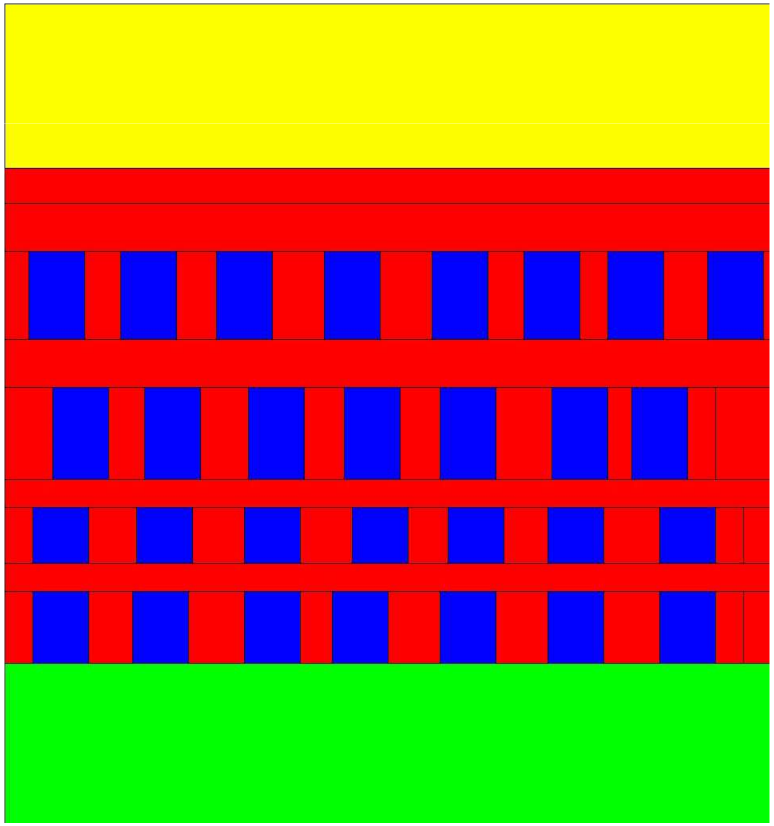


Binary Split Grammars

– Binary:

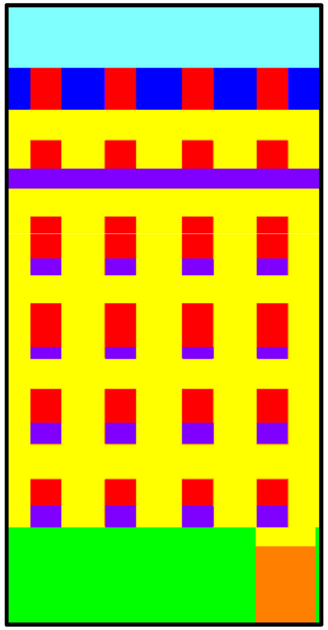


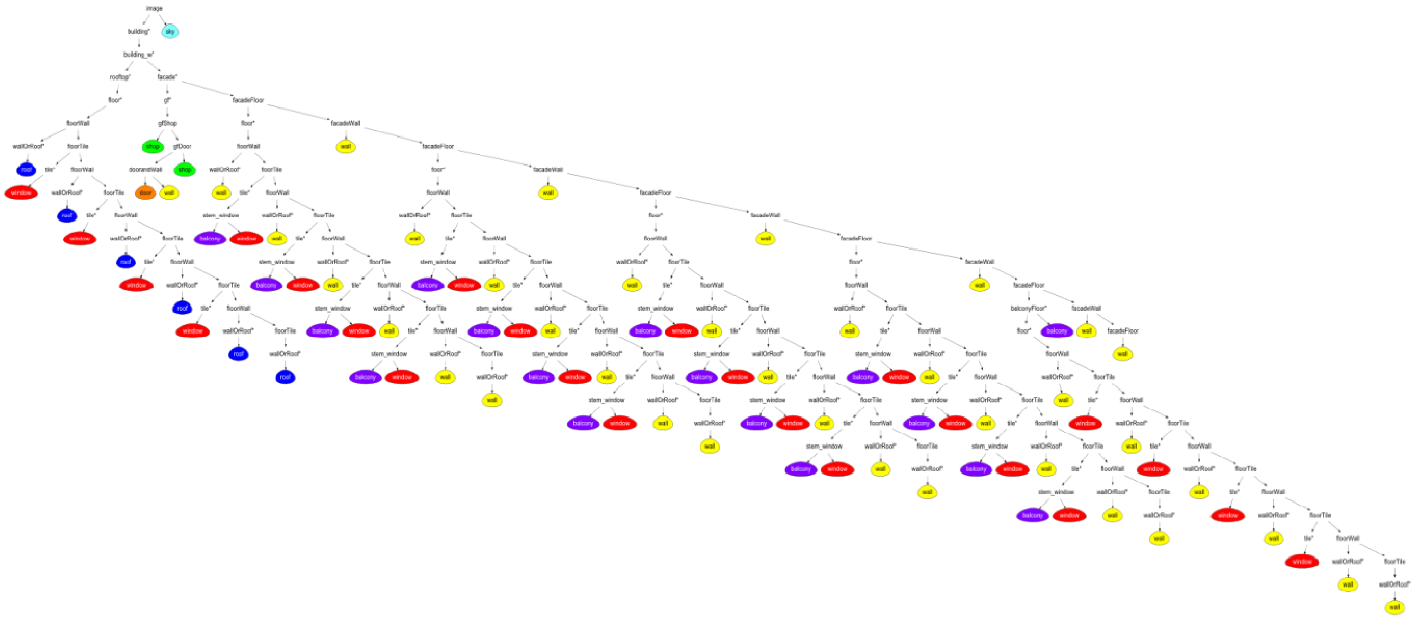
– Split: one dimension at a time



Challenges

- Joint optimization: topology + geometry
- Enforce the result to be in the language of the grammar: $C \in L(G)$
- High and unknown dimensionality: $card\big(L(G)\big)$ up to 1 gogol! (10^{100})

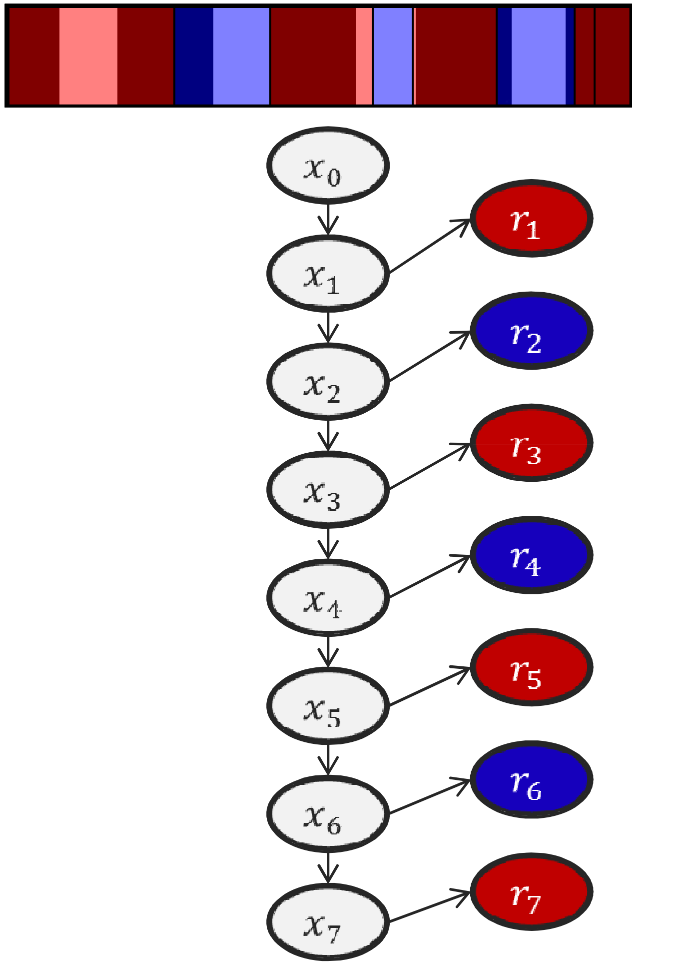




REINFORCEMENT LEARNING FOR SHAPE GRAMMAR PARSING

The 1D case

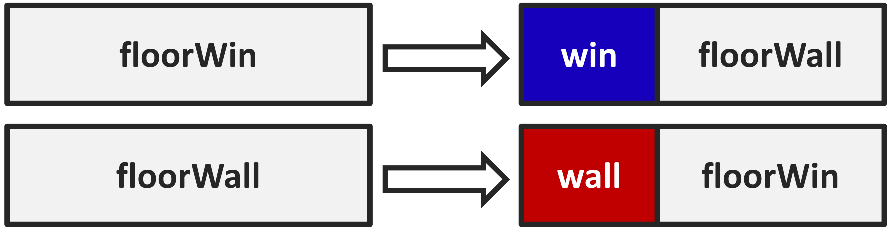
Task: horizontal split(s) of image slice



$$R_0 = \sum r_k$$

Binary Split Grammar

- 2 rules



- Recursive segmentation

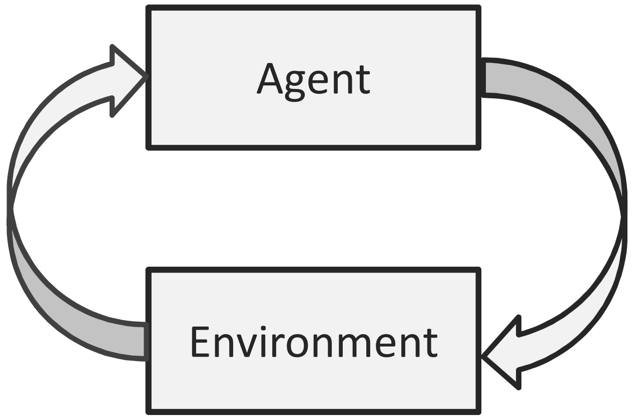
Markov Decision Process (MDP) formulation

- Agent iteratively interacting with environment
  - Agent takes action, lands in new state

$$s_t \xrightarrow{\alpha_t} s_{t+1}$$

  - Environment yields reward

$$r_t = r(s_t, \alpha_t)$$



  - Potentially stochastic state transition and reward functions
- Goal: maximize cumulative reward

$$(N, \alpha_1, \ldots, \alpha_N) = \arg\max \left(\sum_{t=0}^{N} r_t \right)$$

- Markov assumption $P(s_{t+1}, r_t | s_t, a_t, \ldots, s_0, a_0) = P(s_{t+1}, r_t | s_t, a_t)$

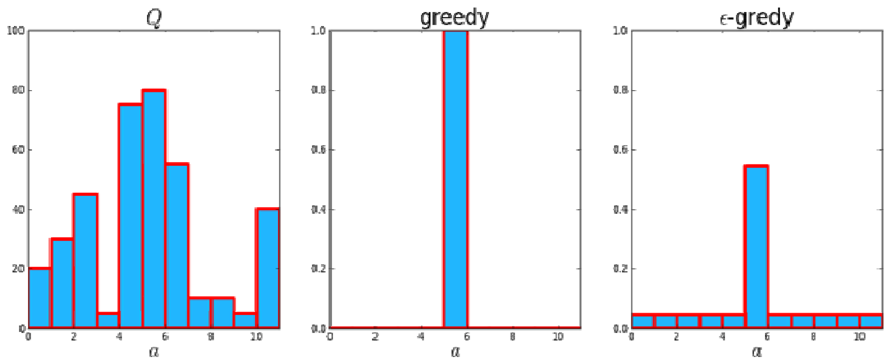
MDP & Policy functions

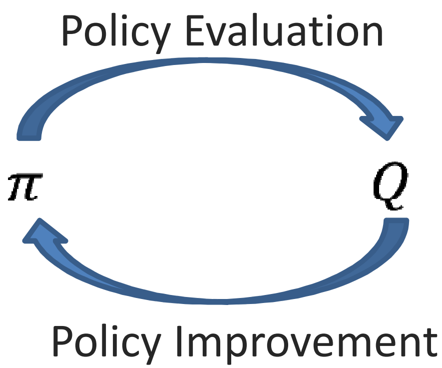
- Policy function adopted by agent: $\pi(s,\alpha) = p(\alpha_t = \alpha | s_t = s)$
- Merit function

$$Q(s,\alpha) = E_\pi \left[\sum_{t'>t} r_{t'} | s_t = s, \alpha_t = \alpha \right]$$

  – Expected reward-to-go if at s we perform α, and then follow π

- ε-greedy policy: $\pi(s,\alpha) = (1-\epsilon)\delta(\alpha,\alpha^*) + \epsilon u(\alpha)$





- Reinforcement learning (Q-learning)

$$Q(s_t,\alpha_t) \leftarrow Q(s_t,\alpha_t) + \alpha \left[\left(r_t + \max_{\alpha_{t+1}} Q(s_{t+1},\alpha_{t+1}) \right) - Q(s_t,\alpha_t) \right]$$

Reinforcement Learning Algorithm

$\forall s, a \; Q(s,a) = 0$

$\forall \; s, \pi(s,.) \leftarrow Uniform$

Loop

$s \leftarrow (\omega, 0)$

repeat

$a \leftarrow$ choose an applicable rule according to $\pi(s,a)$

Apply rule a, observe s', r

$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \max'_a Q(s',a') - Q(s,a)]$

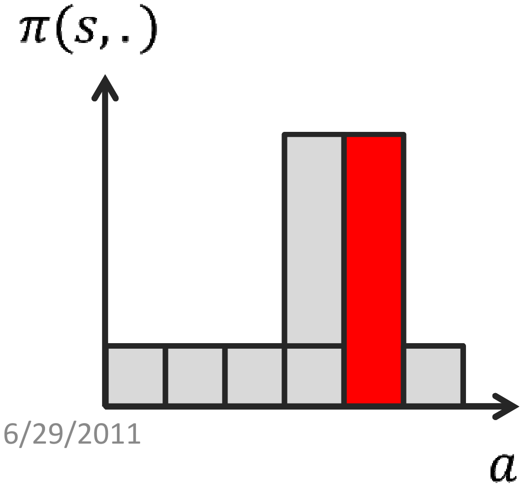
$\pi(s,.) \leftarrow \epsilon$-greedy w.r.t $Q(s,.)$

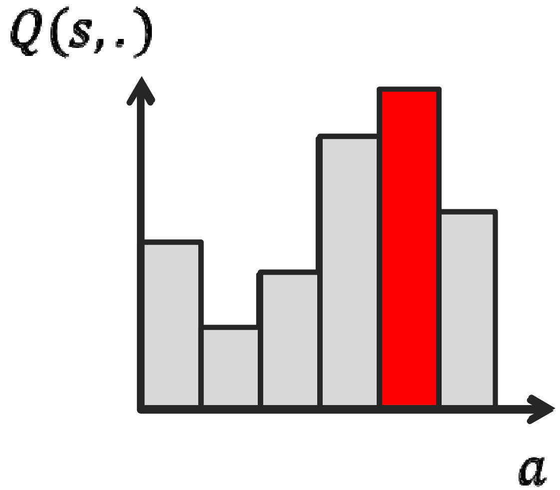
$s \leftarrow s'$

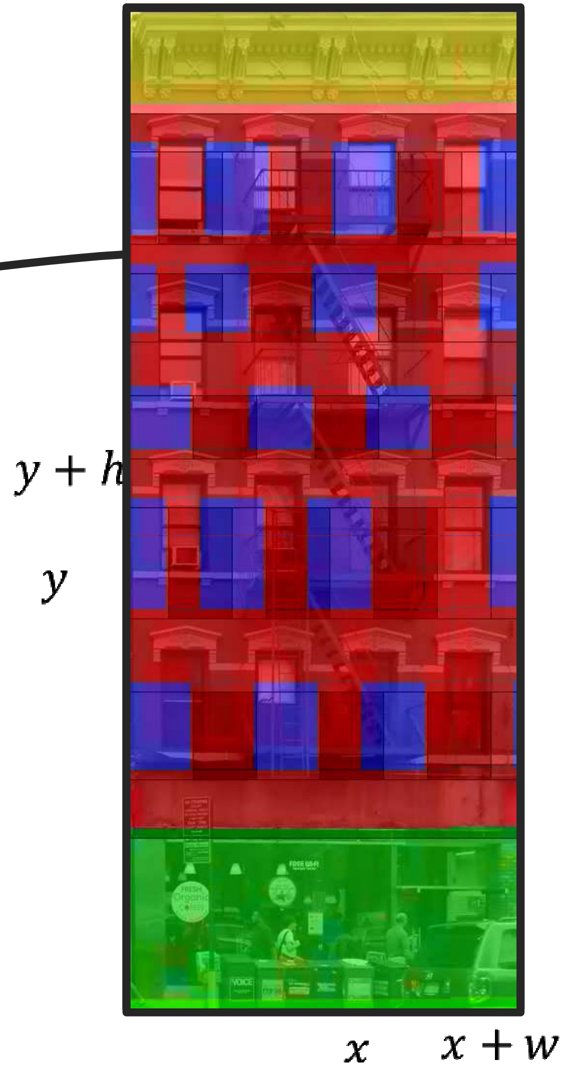
update α

update ϵ

until end of the episode







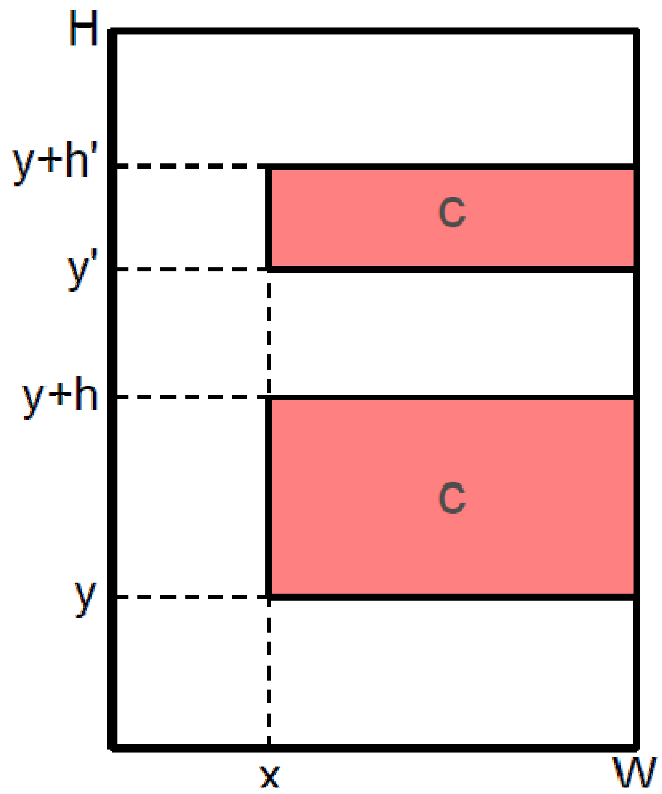
Enforcing Symmetry

- Straighforward extension: 2D state, 2D action
  - Large state: Slow convergence
  - Impossible to enforce floor symmetry

- Can we use single policy for all floors?
  - DP: ?
  - RL: Yes, with **state aggregation**

$$s = (x, y, y + h) \rightarrow \tilde{s} = (x)$$

$$s' = (x, y', h') \rightarrow \tilde{s} = (x)$$



Data-Driven Exploration

- Bottom-up cues:
  - Line detection, window detection,...
- How can we exploit them in model fitting?
  - Modify ε-greedy exploration strategy

$$\pi(s, \alpha) = (1-\epsilon)\delta(\alpha, \alpha^*) + \epsilon u(\alpha)$$

  - Accumulate gradients:

$$h(x) = \sum_y |\nabla_{\pi/2} I(x, y)|$$

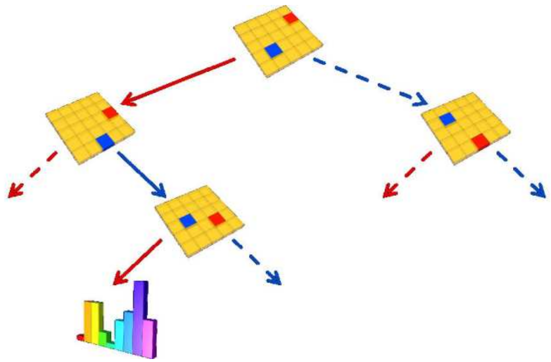
  - Use to `propose' actions:

$$\pi(s, \alpha) = (1-\epsilon)\delta(\alpha, \alpha^*) + \epsilon \frac{\exp(h(s+\alpha))}{\sum_{\alpha'} \exp(h(s+\alpha'))}$$

EXPERIMENTAL VALIDATION

Randomized Forest

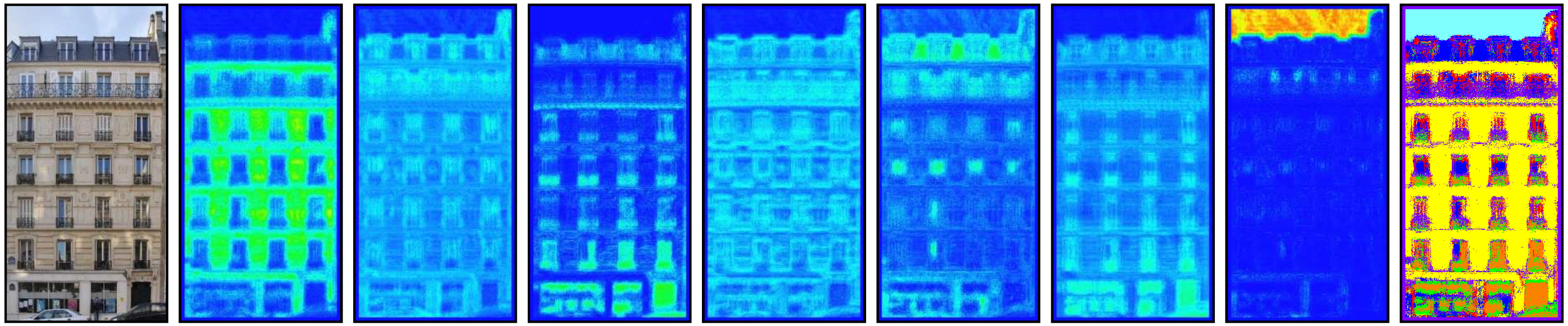
- Multivariate Classifier based on decision trees



- For each class c and each pixel x in image I, it provides $\boldsymbol{p(c|x,I)}$

$$\boldsymbol{m(x,c) = p(c|x,I)}$$

- Feature vectors = 13x13 RGB patches $\in \mathbb{R}^{507}$
- Well suited to very repetitive architectural styles



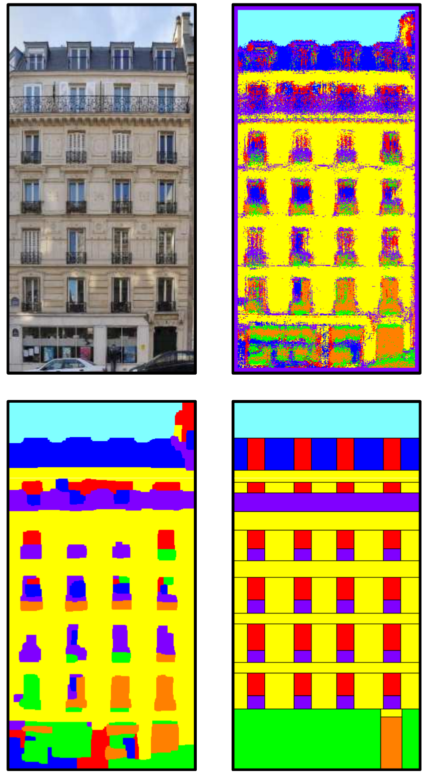
Quantitative Validation: Benchmark 2010

- 20 images for training 10 images for testing

$$\underbrace{\begin{pmatrix} \mathbf{29} & 13 & 13 & 11 & 22 & 6 & 6 \\ 4 & \mathbf{63} & 11 & 8 & 4 & 2 & 8 \\ 10 & 11 & \mathbf{42} & 13 & 12 & 1 & 11 \\ 2 & 2 & 1 & \mathbf{90} & 0 & 0 & 5 \\ 8 & 12 & 5 & 0 & \mathbf{62} & 12 & 0 \\ 1 & 0 & 0 & 0 & 4 & \mathbf{95} & 0 \\ 6 & 7 & 9 & 43 & 8 & 1 & \mathbf{26} \end{pmatrix}}_{\text{MAP}} \underbrace{\begin{pmatrix} \mathbf{29} & 16 & 12 & 11 & 23 & 3 & 6 \\ 2 & \mathbf{72} & 8 & 7 & 2 & 1 & 8 \\ 6 & 11 & \mathbf{60} & 10 & 4 & 0 & 9 \\ 0 & 1 & 1 & \mathbf{96} & 0 & 0 & 2 \\ 5 & 12 & 1 & 0 & \mathbf{71} & 11 & 0 \\ 1 & 0 & 0 & 0 & 3 & \mathbf{96} & 0 \\ 6 & 5 & 6 & 46 & 7 & 1 & \mathbf{29} \end{pmatrix}}_{\text{Potts}, \lambda = 1} \begin{array}{l} \textit{window} \\ \textit{wall} \\ \textit{balcony} \\ \textit{door} \\ \textit{roof} \\ \textit{sky} \\ \textit{shop} \end{array}$$

$$\underbrace{\begin{pmatrix} \mathbf{81} & 9 & 6 & 0 & 4 & 0 & 0 \\ 5 & \mathbf{83} & 8 & 1 & 0 & 0 & 3 \\ 13 & 13 & \mathbf{72} & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & \mathbf{71} & 0 & 0 & 29 \\ 8 & 12 & 0 & 0 & \mathbf{80} & 0 & 0 \\ 2 & 0 & 0 & 0 & 4 & \mathbf{94} & 0 \\ 0 & 0 & 0 & 5 & 0 & 0 & \mathbf{95} \end{pmatrix}}_{\textbf{[Simon 2011]}} \underbrace{\begin{pmatrix} 81 & 11 & 3 & 0 & 5 & 0 & 0 \\ 5 & 84 & 7 & 1 & 1 & 0 & 2 \\ 10 & 26 & 63 & 0 & 0 & 0 & 1 \\ 0 & 2 & 0 & 84 & 0 & 0 & 14 \\ 10 & 4 & 0 & 0 & 86 & 0 & 0 \\ 2 & 0 & 0 & 0 & 4 & 94 & 0 \\ 0 & 1 & 0 & 2 & 0 & 0 & 97 \end{pmatrix}}_{\textbf{RL Parsing}} \begin{array}{l} \textit{window} \\ \textit{wall} \\ \textit{balcony} \\ \textit{door} \\ \textit{roof} \\ \textit{sky} \\ \textit{shop} \end{array}$$

original MAP



Potts RL

	#generated buildings	Time(sec)
[Simon 2011]	10^6	~600
RLParsing	3.10^3	~30

Quantitative Validation: Benchmark 2011

- Complete Benchmark:
  - 104 annotated images
  - Manual parsing

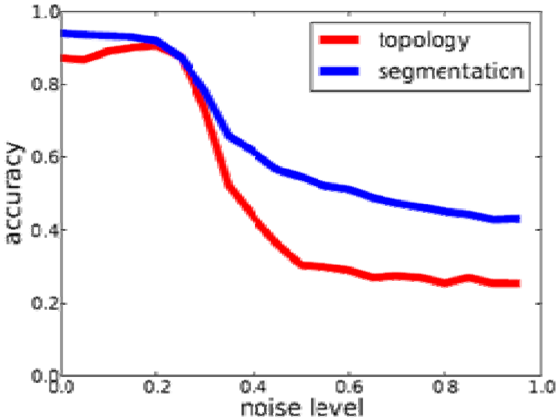
$$\underbrace{\begin{pmatrix} \mathbf{27} & 15 & 15 & 13 & 19 & 4 & 8 \\ 5 & \mathbf{63} & 11 & 9 & 4 & 2 & 7 \\ 11 & 17 & \mathbf{34} & 13 & 12 & 2 & 11 \\ 2 & 2 & 1 & \mathbf{81} & 3 & 0 & 10 \\ 10 & 6 & 11 & 4 & \mathbf{54} & 10 & 4 \\ 4 & 3 & 3 & 1 & 14 & \mathbf{75} & 1 \\ 6 & 12 & 10 & 42 & 7 & 0 & \mathbf{22} \end{pmatrix}}_{\text{MAP}} \underbrace{\begin{pmatrix} \mathbf{68} & 23 & 4 & 0 & 4 & 2 & 0 \\ 3 & \mathbf{87} & 7 & 0 & 1 & 0 & 1 \\ 9 & 24 & \mathbf{64} & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & \mathbf{53} & 0 & 0 & 46 \\ 6 & 3 & 0 & 0 & \mathbf{83} & 8 & 0 \\ 1 & 0 & 0 & 0 & 3 & \mathbf{96} & 0 \\ 0 & 6 & 1 & 6 & 0 & 0 & \mathbf{88} \end{pmatrix}}_{\text{RL Parsing}} \begin{array}{c} window \\ wall \\ balcony \\ door \\ roof \\ sky \\ shop \end{array}$$

	Mean	**Std**
Topology	0.93	0.09
Appearance	0.81	0.07

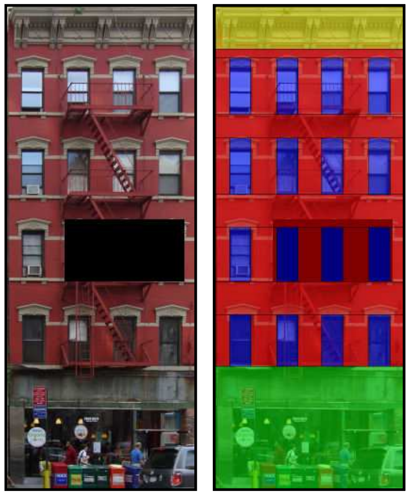
Qualitative Validation

Robustness to Artificial Noise and Occlusions

- Salt-and-pepper noise from 0 to 100% (GMM learnt on noise-free image)



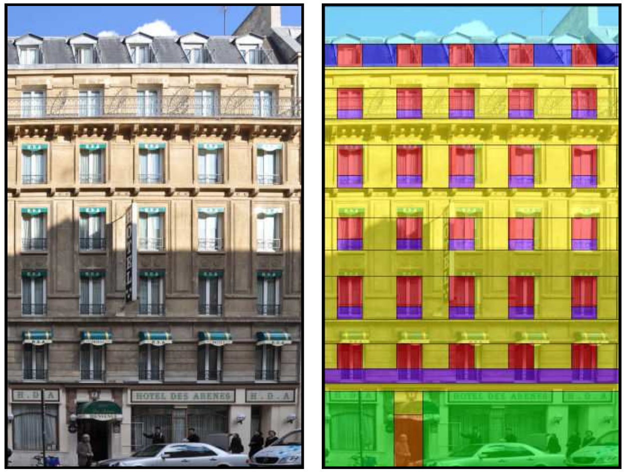
- Artificial occlusions added on images



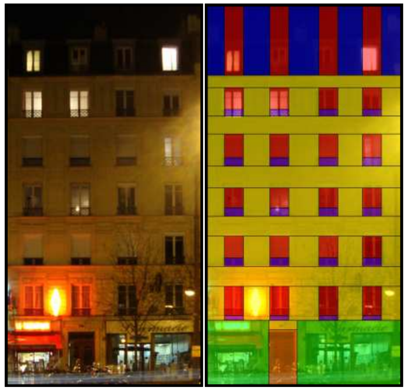
Robustness to Real Occlusions and Illuminations

- Natural Occlusions

- Cast Shadows



- Night Lights



CONCLUSION

Theoretical contributions

Theoretical contributions

Binary Split Grammars: natural fit for façade modeling

Reinforcement learning: flexible techniques for shape parsing

- Enforcing symmetry via state aggregation
- Data-driven exploration
- Efficient exploration of state-action space

State-of-the-art results on many grammars

Practical contributions

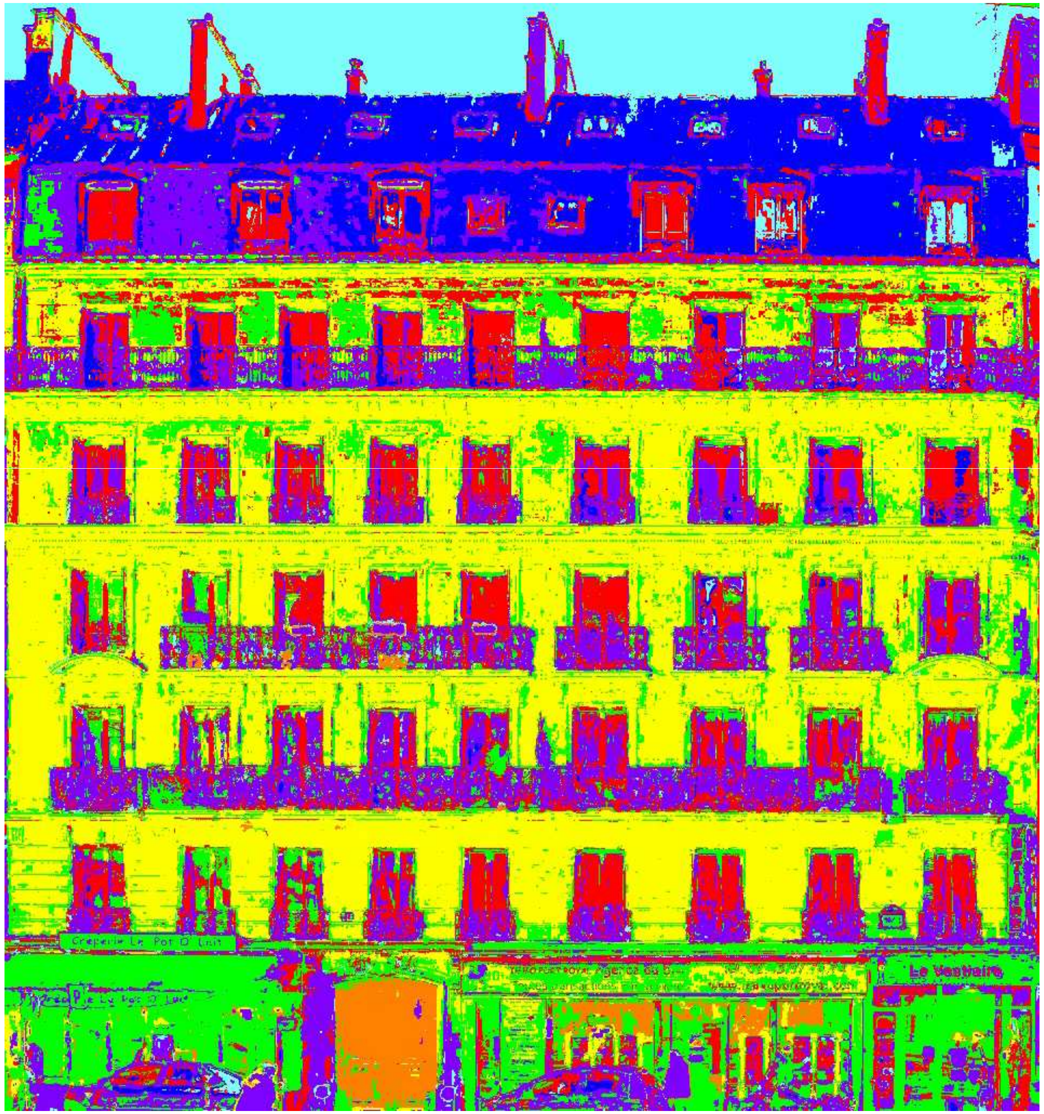
Annotated benchmark for façade parsing

Rflib: Open Source Libraries for Randomized Forests

grapes: software for Facade Parsing with Shape Grammar

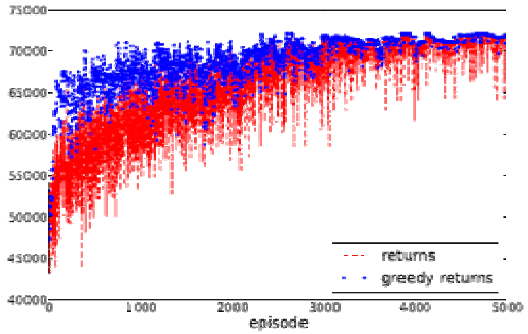
Q&A

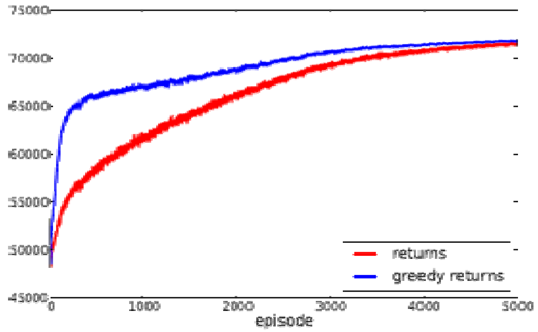
- Thank you!



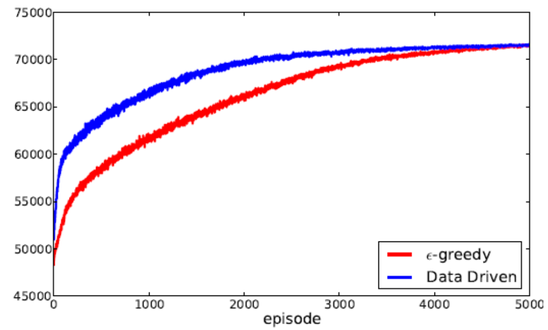
Parsing Algorithm Convergence

Artificial Data



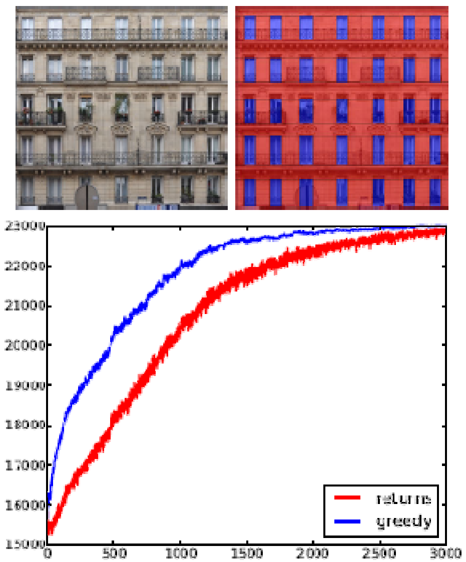


ϵ-greedy

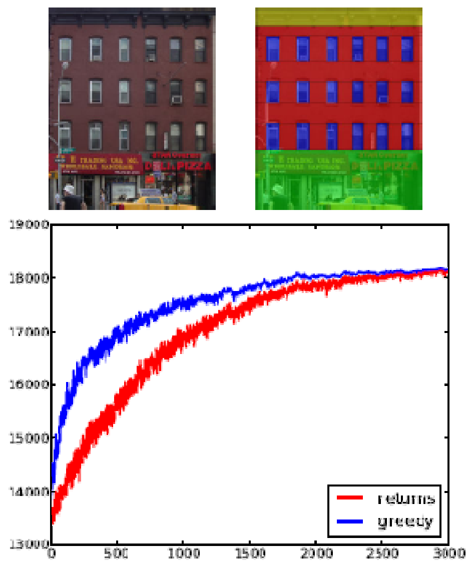


ϵ-greedy / data driven

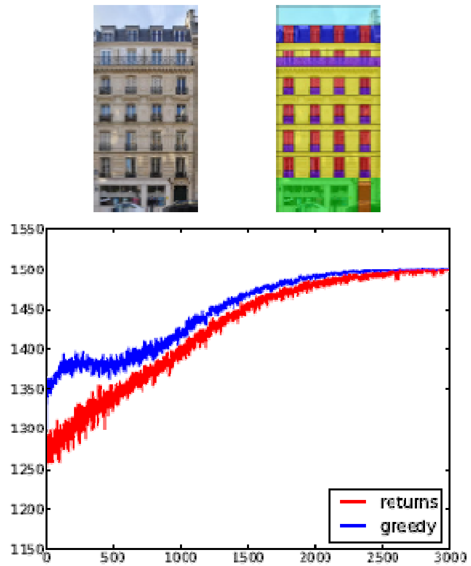
Real Data



(a) Binary - Hue



(b) 4-color - GMM



(c) Haussmannian - RF

Contributions

Theory

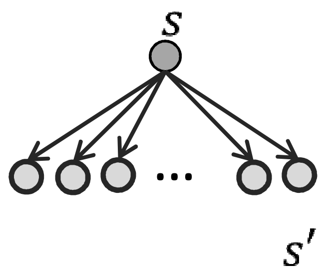
- Binary Shape Grammar(BSG): generic mutually recursive grammars well-suited for façade modeling and optimization.
- Reformulation of the Parsing problem in the Reinforcement Learning framework
- Generic reinforcement learning algorithm for suitable for any BSG
- State aggregation for fast and consistent parsing
- Data-driven exploration to boost the convergence
- State-of-the-art quantitative and qualitative results on many grammars

Practice

- Annotated benchmark for façade parsing
- Rflib: Open Source Libraries for Randomized Forests
- grapes: software for Facade Parsing with Shape Grammar

3 classes of solutions

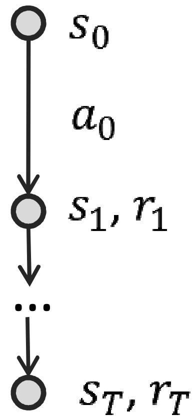
- Dynamic Programming
  - At each state s, consider all actions α.
  - Obtain merit of state s, backpropagate.



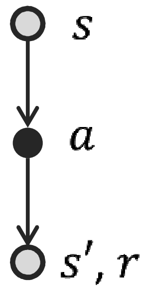
  Needs to consider all state-action combinations

- Monte Carlo
  - Fix first action, α
  - Probabilistically sample subsequent actions.

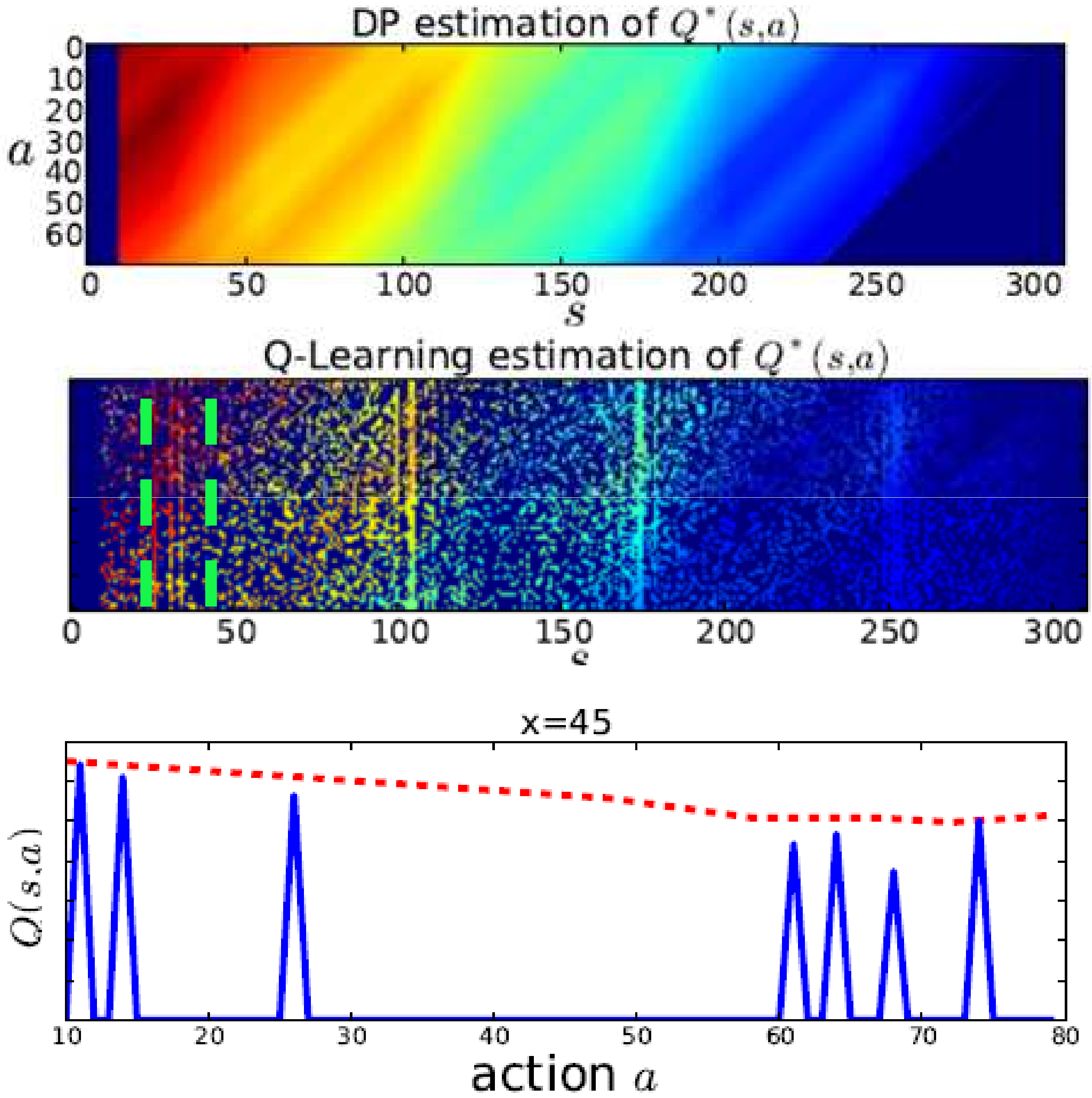
  Needs to consider full episode



- Reinforcement Learning
  - At each state pick single action
  - Back-propagate locally

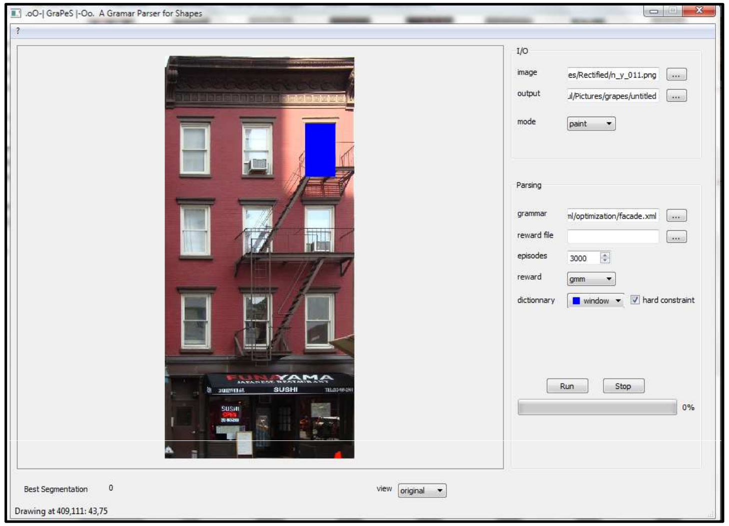


Dynamic Programming vs. Reinforcement Learning



User-defined Constraints

- The user selects a region (x, y, w, h) and a semantic c



- Idea: Reward more the agent when he creates a labeled rectangle that coincides with the constraint.
- If $m(x, c) \in [0,1]$, the reward obtained while the constraint is met is:

$$r = 2wh$$

- Constraint is not *hard*. No guarantee.

MDP & Policy functions

- Policy function adopted by agent: $\pi(s, \alpha) = p(\alpha_t = \alpha | s_t = s)$
- Merit function

$$Q(s, \alpha) = E_\pi \left[\sum_{t' > t} r_{t'} | s_t = s, \alpha_t = \alpha \right]$$

  – Expected reward-to-go if at s we perform α, and then follow π

- Bellman's recursion:

$$Q^\pi(s_t, \alpha_t) = \sum_{s_{t+1}} P(s_{t+1} | s_t, \alpha_t) \left[r(s_t, a_t) + \sum_{\alpha_{t+1}} P(s_{t+1}, \alpha_{t+1}) Q^\pi(s_{t+1}, \alpha_{t+1}) \right]$$

- Bellman's recursion for optimal policy, $\pi^*(s, a)$:

$$Q^*(s_t, \alpha_t) = \sum_{s_{t+1}} P(s_{t+1} | s_t, \alpha_t) \left[r(s_t, a_t) + \max_{\alpha_{t+1}} Q^*(s_{t+1}, \alpha_{t+1}) \right]$$

Q-Learning Algorithm

- Watkins 1989

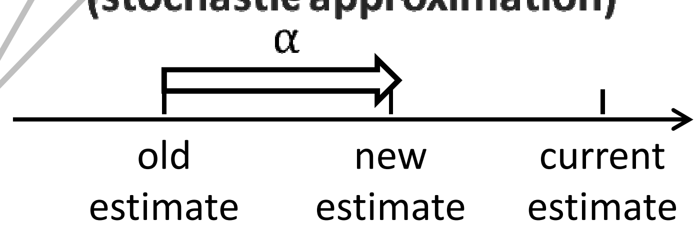
$\forall s, a \; Q(s,a) = 0$
$\forall \; s, \pi(s,.) \leftarrow Uniform$
Loop
  $s \leftarrow$ first state of the episode
  repeat
    $a \leftarrow$ sample from $\pi(s,a)$
    Take action a, observe s', r
    $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max'_a Q(s',a') - Q(s,a)]$
    $\pi(s,.) \leftarrow \epsilon$-greedy w.r.t $Q(s,.)$
    $s \leftarrow s'$
  until end of the episode

Policy Evaluation

Policy Improvement

Learning rate α decreases with the iterations (stochastic approximation)

Sampling (MC)



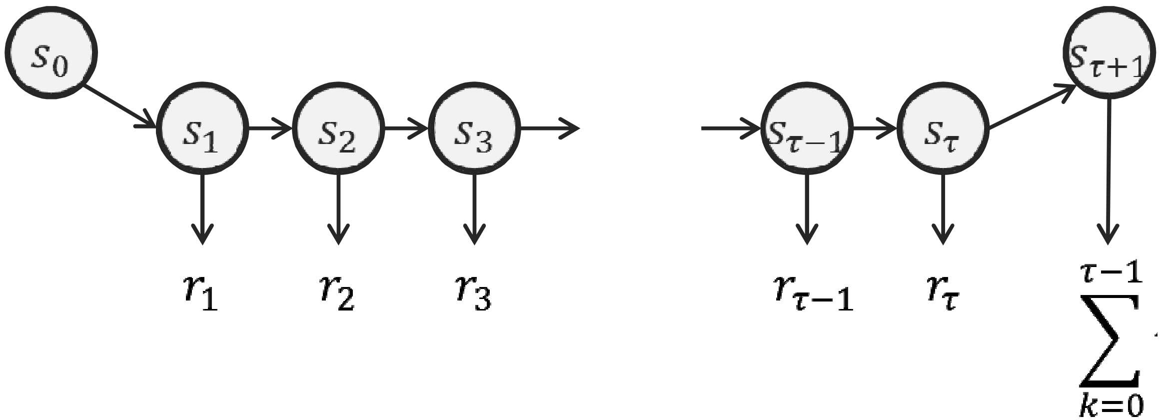
Estimate of $Q(s,a)$ based on :
- **observed reward r**
- **the estimates of $Q(s',a')$**

ϵ decreases to 0 (GLIE)
→ Exploration/exploitation trade-off

→ Converges towards Q^*

Semi Markov Decision Processes

- Some decisions may take more time than others
- Introduction of delayed rewards and a waiting-time τ (random variable)



- Well-suited to model hierarchies
- Natural extension of Q-learning:

$$\mathrm{Q}(s,a) \leftarrow Q(s,a) + \alpha \left[\sum_{k=0}^{\tau-1} \gamma^k r_{k+1} + \gamma \max_a' Q(s',a') - Q(s,a) \right]$$

- Existence of specific algorithms (MAXQ)

Other RL-friendly Techniques

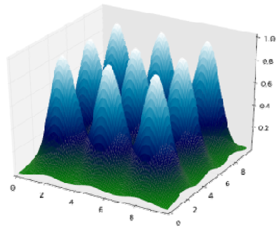
- <u>Model selection:</u>
  - Design several compact grammars rather that a single very generic one
  - The choice of the grammar becomes the first decision of the process
- <u>Function Approximation:</u>
  - Idea: update several state-action pairs at a time (Q is continuous)
  - Linear Approximation with basis ϕ_i → find the M weights w_i

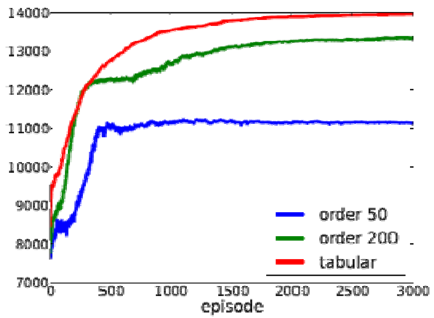
$$Q(s,a) = \sum_{i}^{M} w_i \phi_i(s,a) = w^T \phi(s,a)$$

  - Stochastic Gradient descent to update the estimate of w (therefore of Q)

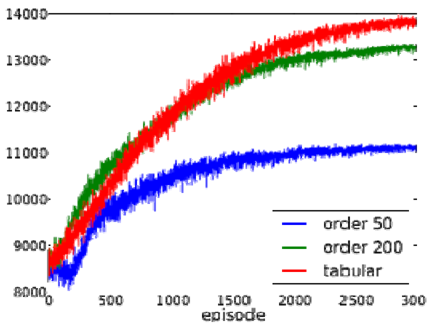
$$w_{t+1} = w_t + \alpha \left[r_{t+1} + \max_{a'} Q_t(s',a') \; - Q_t(s,a) \right] \phi$$

  - Consistent with tabular Q-learning
  - Choice of a basis of functions: failed with Radial-basis functions





(a) Greedy Returns



(b) Returns

Gaussian Mixture Models

- For each class c, a set of inputs $\{y_i = (r_i, g_i, b_i) \in \mathbb{R}^3\}_{i \leq N}$ (brush strokes)
- Observations are explained by a mixture of K Gaussians

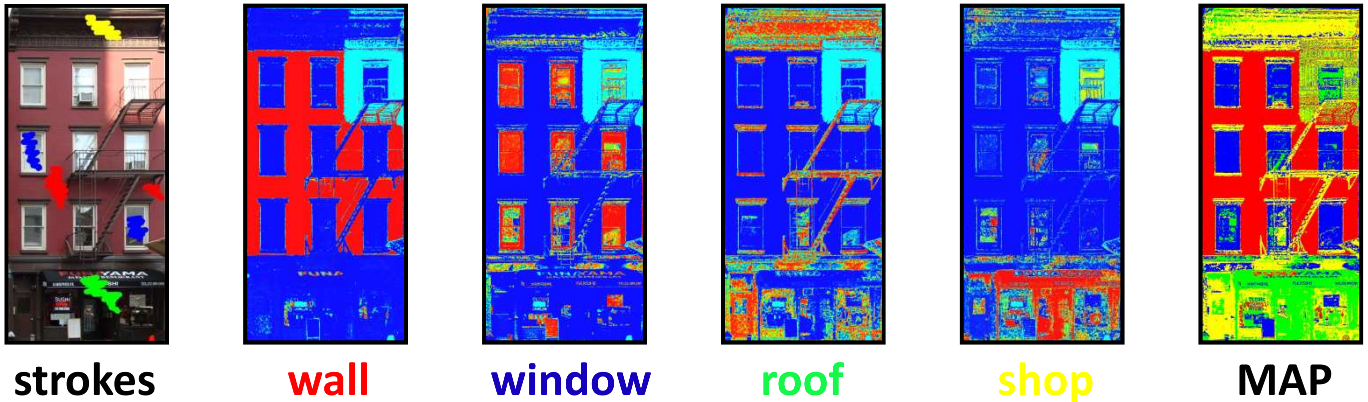
$$p(y|c) = \sum_{k=1}^{K} \pi_k \, \mathcal{N}(y|\mu_k, \Sigma_k)$$

where $$\mathcal{N}(y|\mu, \Sigma) = \frac{1}{\sqrt{2\pi}^3 \sqrt{|\det(\Sigma)|}} \exp\left(-\frac{1}{2}(y-\mu)^T \Sigma^{-1} (y-\mu)\right)$$

- Posterior probability comes from Bayes rules:

$$\boldsymbol{m}(\boldsymbol{x}, \boldsymbol{c}) = p(c|x) = \frac{p(x|c)p(c)}{\sum_{c'} p(x|c')p(c')}$$

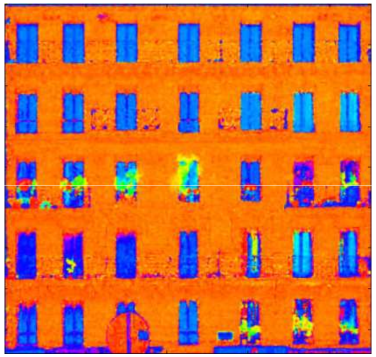
- Optimization using Expectation-Maximization (EM)



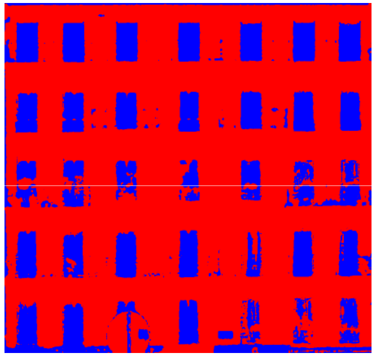
Hue

- RF and GMM are based on *supervised learning*
- The hue reward is based on *unsupervised learning+heuristic*
- Heuristic: the façade shows 2 kinds of elements with 2 different colors
- Idea: Cluster the two classes in the Hue space (K-Means or EM)
- Catch: the Hue is an angle → circular geometry → compute everything in $\mathbb{C}$

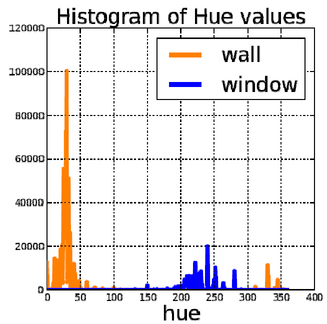
original



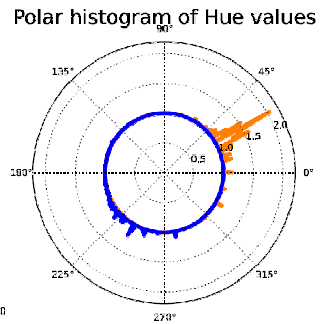
hue



MAP



histogram



polar histogram

$$m(x, c) = p(c|x, I)$$