KIPPI : KInetic Polygonal Partitioning of Images

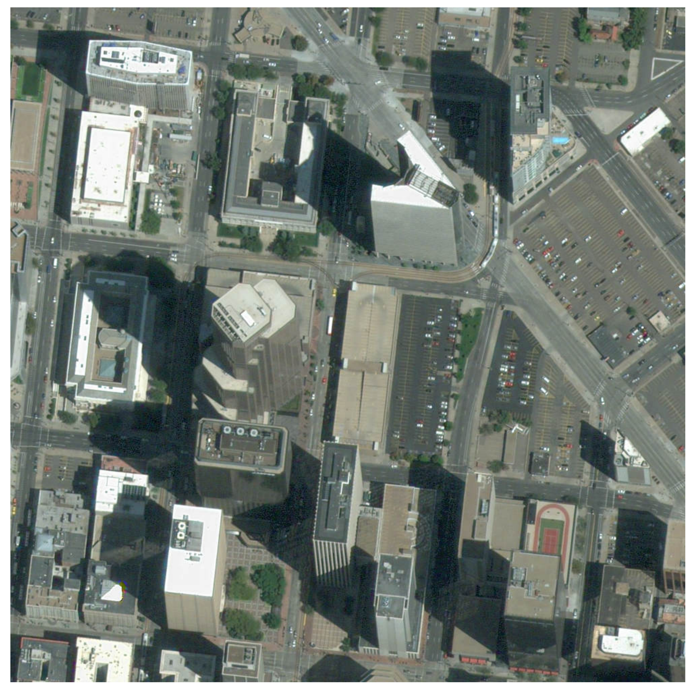
Jean-Philippe Bauchet and Florent Lafarge

Inria – France

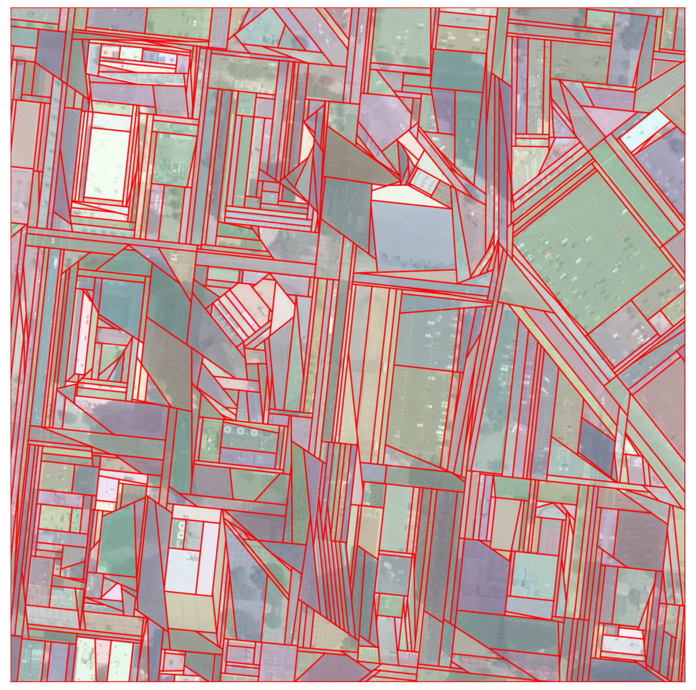
Problem statement

Goal

To partition an image into polygons that capture the geometric structures contained in man-made environments



Input image



Output partition of floating polygons

Motivations

- To offer **more scalability and computational efficiency** than traditional superpixel methods, e.g. [3], by reasoning at the scale of geometric shapes instead of pixels
- To offer **more flexibility** on the shapes and the sizes of the polygons than current polygonal partitioning approaches [4,5], which return homogeneously-sized cells

Algorithm overview

1. Detection of line-segments [1]
2. Regularization of line-segments
3. Kinetic propagation

References

[1] Von Gioi R. G., Jakubowicz J., et al. LSD: A fast line segment detector with a false detection control. PAMI 32(4), 2010

[2] Guibas L. Kinetic data structures. Handbook of data structures and applications, 2004

[3] Liu M.-Y., Tuzel O. et al. Entropy rate superpixel segmentation. CVPR 2011

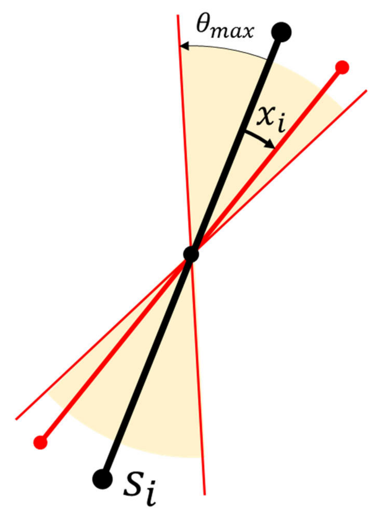
[4] Achanta R., Süsstrunk S. Superpixels and polygons using simple non-iterative clustering. CVPR 2017

[5] Duan L., Lafarge F. Image partitioning into convex polygons. CVPR 2015

[6] KIPPI executable: http://www-sop.inria.fr/members/Florent.Lafarge/codes.html

Our method

2 Regularization



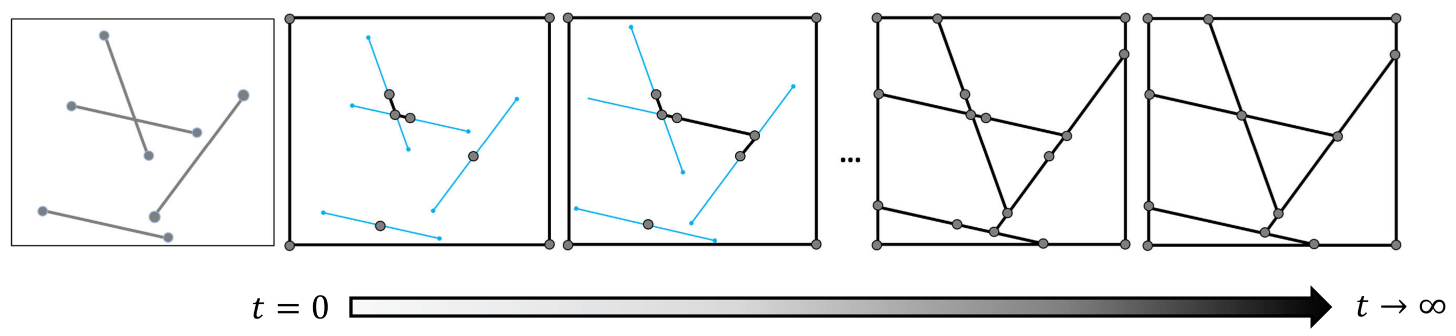
- We apply a set of local rotations $X = (x_1, x_2, \dots x_n)$ to the line-segments returned by [1] with respect to their centers. The vector X is obtained by minimizing the energy

$$E(X) = (1-\lambda)\, D(X) + \lambda\, V(X)$$

with $D(X) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{x_i}{\theta_{max}}\right)^2$, $\quad V(X) = \frac{1}{\sum_{i=1}^{n}\sum_{j>i}\mu_{ij}}\sum_{i=1}^{n}\sum_{j>i}\mu_{ij}\frac{|\theta_{ij} - x_i + x_j|}{4\theta_{max}}$

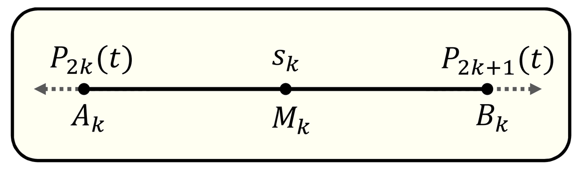
- $V(X)$ encourages pairs of adjacent **near-parallel** or **near-orthogonal** segments to be **exactly parallel** or **orthogonal** while $D(X)$ discourages high deviations from the original orientation of the line-segments
- Minimizing $E(X)$ is a quadratic optimization problem with linear constraints
- We then use an analogous formulation to translate adjacent parallel segments along their orthogonal axis, so that **near-collinear** segments get **exactly collinear**

3 Kinetic propagation



A kinetic data structure [2] consists of:

- A set of geometric **primitives** whose coordinates are continuous functions of time,



- A set of **certificates**, which are properties that a kinetic data structure must satisfy at every time t of the simulation, here a dynamic planar graph,

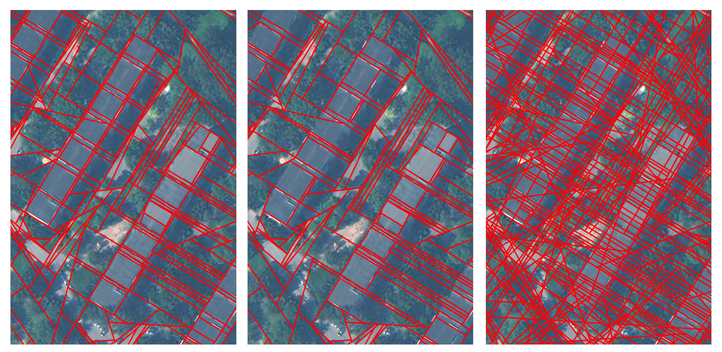
$$C_i(t) = \prod_{j \neq i} Pr_{i,j}(t)$$

$$Pr_{i,j}(t) = \begin{cases} 1 & \text{if } d\left(P_i(t), s_j(t)\right) = 0 \\ 0 & \text{otherwise} \end{cases}$$

- A queue of **events**, defined as moments when such certificates get invalid and imply an update of the structure,

$$e_i(t) = \{t \;/\; C_i(t) = 0\}$$

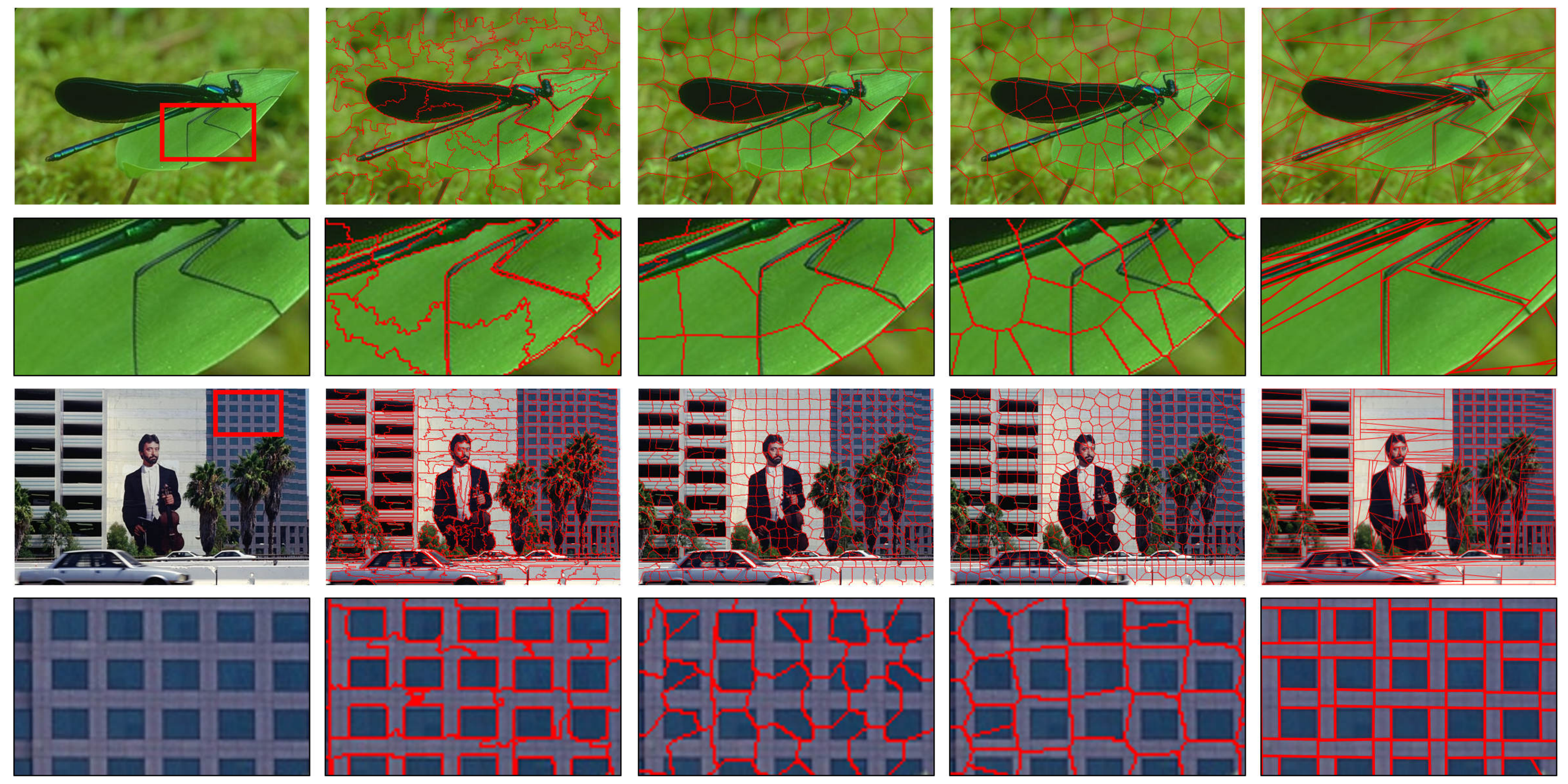
- A **stopping condition,** which tests whether the primitive should keep propagating after an event. The condition can be either gradient-based or a maximal number of intersections *K*.



Gradient-based | K = 1 | K >> 1

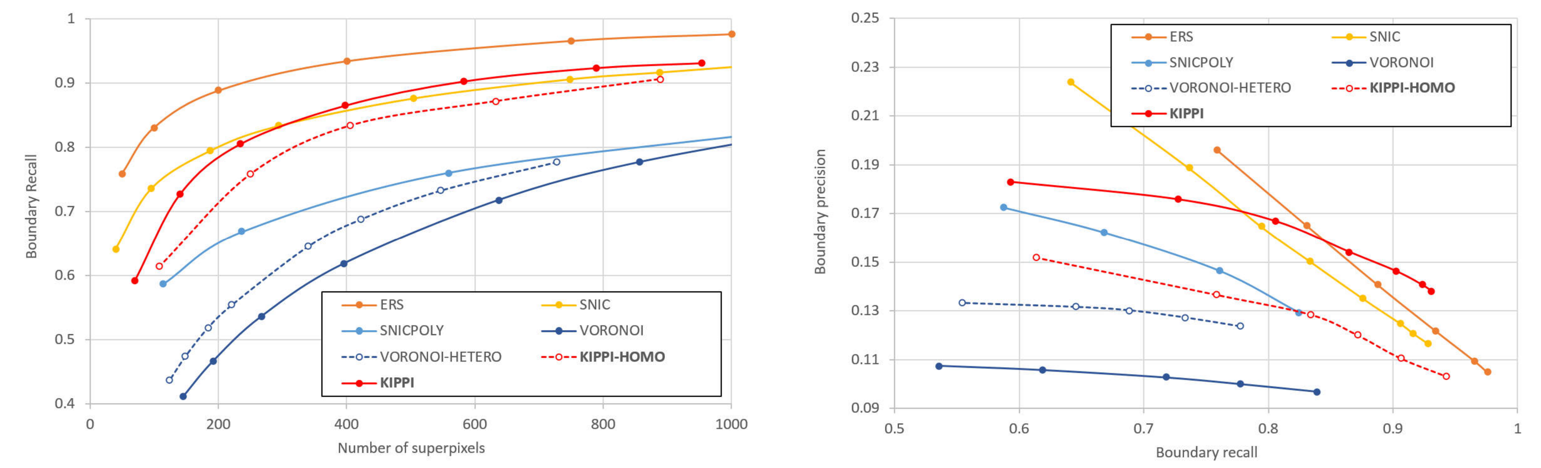
Experiments

Qualitative comparisons with polygonal partitioning methods



Input | ERS [3] | SNICPOLY [4] | VORONOI [5] | Ours

Quantitative evaluation on Berkeley300 dataset

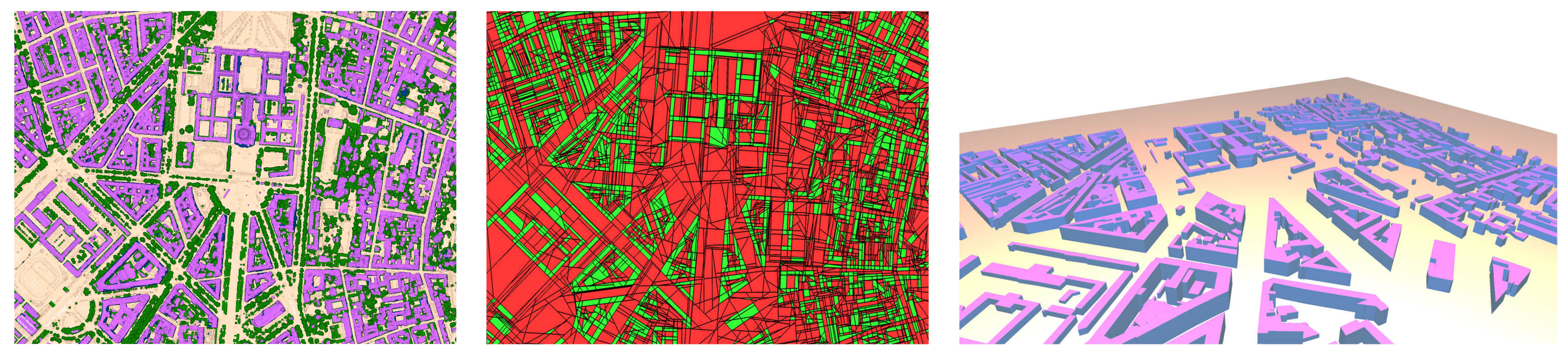


Running time of kinetic propagation: 51 msec, 0.23 sec and 45 sec for images of 154K, 2M and 106M pixels respectively. Executable available online at [6].

Applications

- **Object contouring.** Each polygon of the partition is labeled as foreground or background, using a standard graph-cut model and a set of user strokes.

- **Building reconstruction from airborne data.** Line-segments are fitted to the projection of "facade" points onto the horizontal plane. Our algorithm then decomposes the scene into polygons labeled as building or non-building.



Classified input point cloud | Labeled polygonal partition | Output 3D model